

## TABLE OF CONTENTS:

<b>Change Log for version 1806261326</b> .....	4
<b>1. COMMUNICATION PROTOCOL</b> .....	<b>5</b>
1.1. Message format from the software application to the FPR: .....	5
1.2. Message format from the FPR to the software application: .....	6
1.2.1. ACK response: .....	6
1.2.2. Message response .....	7
1.3. Short messages for testing the status of the FPR .....	7
<b>2. DESCRIPTION OF THE COMMANDS</b> .....	<b>8</b>
2.1. Format and presentation of commands .....	8
2.2. General commands .....	9
2.2.1. Command: 20h / SP - Status .....	9
2.2.2. Command: 21h / ! - Version .....	10
2.2.3. Command: 22h / " - Diagnostics .....	10
2.2.4. Command: 24h / # - Clear display .....	10
2.2.5. Command: 25h / % - Display text line 1 .....	11
2.2.6. Command: 26h / & - Display text line 2 .....	11
2.2.7. Command: 27h / ' - Display text lines 1 and 2 .....	11
2.2.8. Command: 28h / ( - Display date and time .....	11
2.2.9. Command: 2Ah / * - Cash drawer opening .....	11
2.2.10. Command: 2Bh / + - Paper feeding .....	12
2.2.11. Command: 51h / Q - Print barcode 'QP' .....	12
2.3. Fiscal commands .....	13
2.3.1. Command: 40h / @ - SET Manufacturing number .....	13
2.3.1.1. Command: 41h / A (1) - SET VAT and fiscal number .....	13
2.3.1.2. Command: 41h / A (2) - Confirm VAT and fiscal numbers .....	13
2.3.2. Command: 42h / B - VAT rate change .....	14
2.3.3. Command: 43h / C - Change of decimal point position .....	14
2.3.4. Command: 57h / W (0) - Restore previous header .....	14
2.3.5. Command: 57h / W (1) - Print current header .....	14
2.3.6. Command: 57h / W (2) - Store current header into fiscal memory .....	15
2.4. Programming commands .....	16
2.4.1. Command: 44h / D - Programming of payment types .....	16
2.4.2. Command: 45h / E - Programming of parameters .....	17
2.4.3. Command: 46h / F - Programming the external display .....	17
2.4.4. Command: 47h / G - Department programming .....	19
2.4.5. Command: 48h / H - Setting the date and time .....	19
2.4.6. Command: 49h / I - Programming of Display Greeting message .....	20
2.4.7. Command: 49h / I - Programming of header lines .....	20
2.4.8. Command: 49h / I - Programming of footer line .....	20
2.4.9. Command: 49h / I - Programming of CIF name message .....	21
2.4.10. Command: 49h / I - Programming of storno name message .....	21
2.4.11. Command: 49h / I - Programming of VAT number .....	21
2.4.12. Command: 49h / I - Programming of customer receipt name .....	21
2.4.13. Command: 4Ah / J - Programming of operator name and password .....	22
2.4.14. Command: 4Bh / K - option 1, Programming of PLU .....	22
2.4.15. Command: 4Bh / K - option 2, Programming of quantity in stock .....	22
2.4.16. Command: 4Bh / K - option 3, Programming of PLU barcode .....	23
2.4.17. Command: 4Bh / K - option 4, Programming of PLU price .....	23
2.4.18. Command: 4Bh / K - option 5, Programming of PLU Category .....	23
2.4.19. Command: 4Bh / K - option \$, Erase all PLU data base .....	24
2.4.20. Command: 4Ch / L - Programming of Logo without setting a number (default number 0) .....	24
2.4.21. Command: 4Dh / M- Programming of logo with setting a number .....	24
2.4.22. Command: 23h / #- Set active logo file number .....	25
2.4.23. Command: 52h / R - option P, Programming of Customer data .....	25
2.4.24. Command: 53h / S - option A, Programming of Other Charges (Alte taxe) name .....	25
2.4.25. Command: 53h / S - option L, option 1, Programming of service contract date .....	26
2.4.26. Command: 53h / S - option L, option 2, Disable the service contract function .....	26
2.4.27. Command: 53h / S - option L, option 3, Programming of service contract warning messages .....	26
2.4.28. Command: 4Fh / O - Program the duplicates number of the invoice receipt .....	27
2.5. Data reading commands .....	28
2.5.1. Command: 60h / ` - Reading the FPR numbers .....	28
2.5.2. Command: 61h / a - Reading the VAT number .....	28
2.5.3. Command: 62h / b - Reading the VAT rates .....	28
2.5.4. Command: 63h / c - Reading the decimal point .....	29
2.5.5. Command: 64h / d - Reading the types of payment .....	29
2.5.6. Command: 65h / e - Reading of parameters .....	29
2.5.7. Command: 67h / g - Read of department registers .....	30
2.5.8. Command: 68h / h - Reading the date and time .....	30
2.5.9. Command: 69h / i - Reading the display greeting message .....	31

2.5.10. Command: 69h / i – Reading the header lines.....	31
2.5.11. Command: 69h / i – Reading the footer line .....	32
2.5.12. Command: 69h / i – Reading CIF name message .....	32
2.5.13. Command: 69h / i – Reading storno name message .....	32
2.5.14. Command: 69h / i – Reading VAT number .....	32
2.5.15. Command: 6Ah / j – Reading the operator's name and password .....	33
2.5.16. Command: 6Bh / k – option “ (all), Reading of article registers .....	33
2.5.17. Command: 6Bh / k – option 1 (general), Reading of article registers .....	34
2.5.18. Command: 6Bh / k – option 2 (QTY), Reading of article registers .....	35
2.5.19. Command: 6Bh / k – option 3 (barcode), Reading of article registers .....	35
2.5.20. Command: 6Bh / k – option 4 (price), Reading of article registers .....	35
2.5.21. Command: 6Bh / k – option 5 (Category), Reading of article registers .....	36
2.5.22. Command: 6Ch / l – Logo printing .....	36
2.5.23. Command: 52h / R – option R, Reading of Customer data .....	36
2.5.24. Command: 53h / S – option L, option 1, Reading of service contract date .....	37
2.5.25. Command: 53h / S – option L, option 3, Reading service contract warning messages. ....	37
2.5.26. Command: 53h / S – option A, Reading of Other Charges (Alte taxe) name .....	38
2.5.27. Command: 58h / X – option O - Reset Odometer function .....	38
2.5.28. Command: 58h / X – option O - Read Odometer function .....	38
2.5.29. Command: 4Fh / O – Reading the duplicates number of the invoice receipt .....	39
2.6. Receipt operations commands .....	40
2.6.1. Command: 2Eh / . – Non-fiscal receipt opening .....	40
2.6.2. Command: 2Eh / . – Non-fiscal receipt opening with postponed printing .....	40
2.6.3. Command: 2Fh / / – Non-fiscal receipt closure .....	40
2.6.4. Command: 30h / 0 – Standard fiscal receipt opening .....	40
2.6.5. Command: 30h / 0 – Standard fiscal receipt opening with postponed printing .....	40
2.6.6. Command: 30h / 0 – Standard fiscal receipt opening with buffered printing .....	41
2.6.7. Command: 30h / 0 – Special Customer fiscal receipt document opening .....	41
2.6.8. Command: 30h / 0 – Special Customer fiscal receipt document opening with postponed printing .....	42
2.6.9. Command: 30h / 0 – Special Customer fiscal receipt document opening with buffered printing .....	43
2.6.10. Command: 31h / 1 – Sale/correction of article with VAT class definition .....	43
2.6.11. Command: 31h / 1 – Sale/correction of article belonging to departament .....	44
2.6.12. Command: 31h / 1 – Sale/correction of article with specified VAT belonging to departament .....	45
2.6.13. Command: 32h / 2 – sale/correction of article from FPR database .....	46
2.6.14. Command: 33h / 3 – Subtotal .....	46
2.6.15. Command: 33h / 3 – Subtotal with value discount with specified VAT definition .....	47
2.6.16. Command: 34h / 4 – Sale/correction of article with department definition .....	49
2.6.17. Command: 34h / 4 – Sale/correction of article with specified department belonging to VAT class .....	50
2.6.18. Command: 35h / 5 – Payment .....	51
2.6.19. Command: 35h / 5 – Pay exact sum .....	52
2.6.20. Command: 36h / 6 – Automatic receipt closure .....	52
2.6.21. Command: 37h / 7 – Free text printing .....	52
2.6.22. Command: 38h / 8 – Fiscal receipt closure .....	53
2.6.23. Command: 39h / 9 – Fiscal receipt cancel .....	53
2.6.24. Command: 3Ah / : – Print a copy of the last document .....	53
2.6.25. Command: 3Bh / ; – Non-fiscal RA and PO amounts .....	53
2.6.26. Command: 3Ch / < – Storno function of article with VAT class definition .....	54
2.6.27. Command: 3Ch / < – Storno function of article belonging to departament .....	55
2.6.28. Command: 3Ch / < – Storno function of article with specified VAT belonging to departament .....	56
2.6.29. Command: 3Dh / '=' – Sell / Correction of article from with fractional quantity belonging to VAT class definition .....	57
2.6.30. Command: 3Dh / '=' – Sell / Correction of article with fractional quantity belonging to departament .....	58
2.6.31. Command: 3Dh / '=' – Sell / Correction of article with fractional quantity with specified VAT belonging to departament .....	59
2.6.32. Command: 3Eh / > – Discount/ addition – FP Only .....	60
2.7. Commands for reading the data in FPR's registers .....	61
2.7.1. Command: 6Dh / m – Reading of amounts by VAT Groups .....	61
2.7.2. Command: 6Eh / n – Reading of registers – 0 (on hand) .....	61
2.7.3. Command: 6Eh / n – Reading of registers – 1 (general) .....	62

2.7.4. Command: 6Eh / n – Reading of registers – 2 (RA).....	62
2.7.5. Command: 6Eh / n – Reading of registers – 3 (PO) .....	63
2.7.6. Command: 6Eh / n – Reading of registers – 4 (received) .....	63
2.7.7. Command: 6Eh / n – Reading of registers – 5 (counters) .....	64
2.7.8. Command: 6Fh / o – Reading of operator's report – '1' (general) .....	64
2.7.9. Command: 6Fh / o – Reading of operator's report – 2 (RA) .....	65
2.7.10. Command: 6Fh / o – Reading of operator's report – 3 (PO) .....	65
2.7.11. Command: 6Fh / o – Reading of operator's report – 4 (received) .....	66
2.7.12. Command: 6Fh / o – Reading of operator's report – 5 (counters) .....	66
2.7.13. Command: 6Fh / o – Reading of operator's report – 6 (returned) .....	67
2.7.14. Command: 71h / q – Reading of receipt number .....	67
2.7.15. Command: 72h / r – Reading information about the current opened receipt .....	67
2.7.16. Command: 73h / s – Reading the last date of a daily report .....	68
2.7.17. Command: 74h / t – Reading of free FM reporting records .....	68
2.7.18. Command: 75h / u – Reading of FM content .....	69
2.8. Reports printing commands .....	70
2.8.1. Command: 76h / v – Report by department .....	70
2.8.2. Command: 77h / w – Special events FM report .....	70
2.8.3. Command: 77h / w – Special events FM report by number of reports .....	70
2.8.4. Command: 77h / w – Special events FM report by date .....	70
2.8.5. Command: 78h / x – Detailed FM report by number of reports .....	71
2.8.6. Command: 78h / x – Detailed Payment FM report by number of reports .....	71
2.8.7. Command: 78h / x – Storage detailed FM report by number of reports .....	71
2.8.8. Command: 79h / y – Brief FM report by number of reports .....	71
2.8.9. Command: 79h / y – Brief payment FM report by number of reports .....	72
2.8.10. Command: 79h / y – Store Brief FM report by number of reports .....	72
2.8.11. Command: 7Ah / z – Detailed FM report by date .....	72
2.8.12. Command: 7Ah / z – Detailed Payment FM report by date .....	72
2.8.13. Command: 7Ah / z – Storage detailed FM report by date .....	73
2.8.14. Command: 7Bh / { – Brief FM report by date .....	73
2.8.15. Command: 7Bh / { – Brief payment FM report by date .....	73
2.8.16. Command: 7Bh / { – Store Brief FM report by date .....	73
2.8.17. Command: 7Ch /   – Daily fiscal report X or Z .....	74
2.8.18. Command: 7Ch /   – Printing Electronic Journal report from date do date .....	74
2.8.19. Command: 7Ch /   – Printing Electronic Journal report from receipt number to receipt number .....	74
2.8.20. Command: 7Ch /   – Printing Electronic Journal report from number Z report to number Z report .....	75
2.8.21. Command: 7Ch /   – Printing Electronic Journal report from beginning to end .....	75
2.8.22. Command: 7Ch /   – Receipts xml export from number Z report to number Z report .....	75
2.8.23. Command: 7Dh / } – Operator's report .....	76
2.8.24. Command: 7Eh / ~ – Article report .....	76
2.8.25. Command: 7Fh / █ – Extended daily report .....	76
2.8.26. Command: 52h / R – option X, Print Customer X or Z report .....	76
2.8.27. Command: 59h / Y – option H - Print hourly report .....	77
2.8.28. Command: 5Ah / Z – option 1, Reading Z report number from date .....	77
2.8.29. Command: 5Ah / Z – option 2, Reading Z report number from date .....	77
2.9. Reports READING commands .....	78
2.9.1. Command: 7Ch /   – Reading Electronic Journal report from date do date .....	78
2.9.2. Command: 7Ch /   – Reading Electronic Journal report from receipt number to receipt number .....	78
2.9.3. Command: 7Ch /   – Reading Electronic Journal report from number Z report to number Z report .....	78
2.9.4. Command: 7Ch /   – Reading Electronic Journal report from beginning to end .....	79
<b>3. SOFTWARE APPLICATION REQUIREMENTS .....</b>	<b>81</b>
3.1. Rules for using the commands .....	81
3.2. Sample sale transaction of FPR .....	81
<b>4. AUXILARY GS PROTOCOL (COMMANDS 1Dh) .....</b>	<b>82</b>

## CHANGE LOG FOR VERSION 1807101103

Command 71h, [ReadLastReceiptNo\(\)](#)

- Result field *NoLastIsRec* is renamed to *NoLastIsRcp*
- Result field *NoLastIsRcp* is with changed length from 4 to 1..4 in format ####
- Added new result field *NoTotalRcp* for total receipt count

Command 47h, [ProgDep\(...\)](#)

- Input field *OptionDep* is renamed to *OptionDepPrice*
- Input field *Category* is with changed delimiter from '+' to ';' ,

Command 67h, [ReadDep\(DepNo\)](#)

- Result field *OptionDep* is renamed to *OptionDepPrice*

Command 6Bh

- Changed zfpdef from [zfpdef: ReadPLU4\(PLUNo\)](#) to [zfpdef: ReadPLU5\(PLUNo\)](#)

## 1. COMMUNICATION PROTOCOL

The type of the protocol is Master / Slave. The communication session is always initiated by the Application Software. FPR carries out the commands sent by the software application and provides a feedback depending on the result. FPR sends back an „ACK response” or „message response”. All messages of the protocol are either packed or single-byte. FPR supports communication standard RS232 using the TxD, RxD and Gnd signals.

Serial port adjustment parameters:

Speed: 115200 bit/s (or 19200, 38400, 57600 and 9600 if such is set for the FPR)

8 bit word

No parity

1 stop bit

### 1.1. MESSAGE FORMAT FROM THE SOFTWARE APPLICATION TO THE FPR:

All messages except those described in 3.4.3., sent to the FPR by the PC have the following structure:

**<STX><LEN><NBL><CMD><DATA...DATA><CS><CS><ETX>**

The table below contains description of the field enclosed between the symbols < and >:

Field	No. of bytes	Value
STX	1	<b>Message start</b> – always <b>02h</b>
LEN	1	<b>Message length</b> (number of bytes including LEN, NBL, CMD, DATA) increased by <b>20h</b> i.e. a number in the 20h - 9Fh range
NBL	1	<b>Message number</b> increased by <b>20h</b> i.e. a number in the 20h - 9Fh range
CMD	1	<b>Command</b> - a number in the 20h - 7Fh range(see the description of commands)
DATA.. DATA	0 - 3902	<b>Additional data</b> – a group of data fields separated with the symbols ';', giving additional information needed for execution of the command (see the description of commands)
CS CS	2	<b>Checksum</b> , compiled as follows: 1) Operation XOR of all bytes from LEN to DATA inclusive = 0 .. FFh 2) Conversion of 2 bytes by adding 30h, for example: B5h -> 3Bh 35h
ETX	1	<b>End of message</b> – always <b>0Ah</b> (LF)

The text data of the message is sent as ASCII text with code table cp1251 (Windows 1251).

## 1.2. MESSAGE FORMAT FROM THE FPR TO THE SOFTWARE APPLICATION:

There are several types of response depending on the message received.

### 1.2.1. ACK RESPONSE:

**Positive ACK** – when package format is correct. It is sent when the command is acknowledged as well as when it is rejected (errors in the data sent (field <DATA...DATA>) or the command cannot be executed or the command is illegal depending on the current status of the FPR indicated by the two status bytes). It is a package message with the following format:

**<ACK><NBL><STE><STE><CS><CS><ETX>**

Fields description:

Field	No. of bytes	Value
ACK	1	06h
NBL	1	<b>No. of message</b> = NBL of message related to receipt
STE STE	2	<b>2 error status-bytes</b> . A two-digit ASCII number. (see Table Errors)
CS CS	2	<b>Checksum</b> , compiled as follows: 1) Operation XOR on NBL STE и STE = 00h .. FFh 2) Conversion of 2 bytes by adding 30h, for example: B5h -> 3Bh 35h
ETX	1	0Ah (LF)

The two status-bytes are a two-digit ASCII number, in which the first digit provides information about the error in the FPR, and the second one – about a command error.

**Table Errors:**

Byte value	FPR errors	Byte value	Command errors
30h	OK	30h	OK
31h	Out of paper, printer failure	31h	Invalid command
32h	Registers overflow	32h	Illegal command
33h	Clock failure or incorrect date&time	33h	Z daily report is not zero
34h	Opened fiscal receipt	34h	Syntax error
35h	Payment residue account	35h	Input registers overflow
36h	Opened non-fiscal receipt	36h	Zero input registers
37h	Registered payment but receipt is not closed	37h	Unavailable transaction for correction
38h	Fiscal memory failure	38h	Insufficient amount on hand
39h	Incorrect password	39h	Not used
3ah	Missing external display	3Ah	No access
3bh	24hours block – missing Z report		
3ch	Overheated printer thermal head.		
3dh	Interrupt power supply in fiscal receipt (one time until status is read)		
3eh	Overflow EJ		
3fh	Insufficient conditions		
41h	Certificates are not loaded		

A two-digit number is compiled depending on the type of error.

Example: Error 32 – Illegal command due to clock failure

**Negative ACK** – It is sent when the package format is incorrect. It is 1 byte **NACK = 15h** without checksum.

**Repetition request** – It is sent when the FPR is busy executing the preceding command. It is 1 byte **RETRY = 0Eh** without checksum.

### 1.2.2. MESSAGE RESPONSE

It has the format of the packed message sent by the SA to the FPR but is returned by the FPR to the SA and contains information – response to the query (see description of commands).

### 1.3. SHORT MESSAGES FOR TESTING THE STATUS OF THE FPR

The exchange protocol includes two unpacked single-byte codes for testing the status of the FPR, which can quickly determine the status of the device. The two codes and their meaning are shown in the table below:

Query SA	Response FPR		Meaning
04	04		FPR is on
09	40	Bit.0	FPR is READY
	41		FPR is busy
	42	Bit.1	FPR out of paper
	43		FPR out of paper and busy
	44	Bit.2	FPR printer is overheated
	45		FPR printer is overheated and busy
	48	Bit.3	FPR missing external display
	49		FPR missing external display and busy
	50	FPR is waiting for password (LAN or WiFi connection only)	
	60	FPR is already busy with another connection (LAN or WiFi connection only)	
	70	Wrong password (LAN or WiFi connection only)	

The format of the commands is described in art. 2. **DESCRIPTION OF THE COMMANDS OF FISCAL PRINTER**

## 2. DESCRIPTION OF THE COMMANDS

### 2.1. FORMAT AND PRESENTATION OF COMMANDS

All commands are described and presented using the following terms and symbols:

#### Key terms:

**Command** – the value of the CMD field of the message sent by the software application and in the message response of the the FPR.

**input** – structure of the fields included in the DATA field of the message sent by the software application.

**output** – for each command it may be one of the following:

- ACK response
- Structure of the fields included in the DATA field of the message response sent by the FPR

**Input data** – description of the contents of the “input” fields.

**Output data** – description of the contents of the “output” fields.

#### Key symbols:

- ' ' – compulsory symbol
- < > – compulsory data field
- <;> – field separator
- [] – field length
- { } – non-compulsory data field

#### General rules:

Format of the price/value field – from 1 to 10 symbols, a floating decimal point number, preceded by +, - or SPACE.

Examples:                -12.34        +56.7    8

Format of the quantity field – from 1 to 10 symbols, a floating decimal point number, up to three digits after the decimal point.

Examples:                1.234        56.78    9

Format of the rate (percentage) field – from 2 to 7 symbols, a floating decimal point number, up to two digits after the decimal point, preceded by the percent symbol - %.

Examples:                -12.34%    +5.67%                8.9%                10%

Payment No. 0 corresponds to the main payment – IN CASH, payment No. 4 corresponds to the special payment - VAT account, payments No. 1, 2 and 3 are programmable.



## 2.2. GENERAL COMMANDS

These are commands for the general functions of the FPR, related to obtaining diagnostic information and to direct access to some of the functions of the device (paper feeding, paper cutting, and display visualization).

### 2.2.1. Command: 20h / SP - Status

**input:** n. a.

**output:** <StatusBytes[6]>

**FPR operation:** Provides detailed 6-byte information about the current status of the fiscal printer.

**Input data :** n. a.

**Output data :**

<i>N byte</i>	<i>N bit</i>	<i>status flag</i>
ST0	0	FM Read only ( ST3.0 or ST3.1 or ST3.2 )
	1	Power down in opened fiscal receipt
	2	Printer not ready or overheated
	3	Incorrect time
	4	Incorrect date
	5	RAM reset
	6	Date and time hardware error
	7	Reserved

ST1	0	Printer not ready or no paper
	1	Reports registers overflow
	2	Blocking after 24 hours
	3	Non-zero daily report
	4	Non-zero article report
	5	Non-zero operator report
	6	Non-printed copy
	7	Reserve

ST2	0	Opened Non-fiscal Receipt
	1	Opened Fiscal Receipt
	2	Standard Cash Receipt
	3	VAT included in the receipt
	4	Reserved
	5	EJ near full
	6	EJ full
	7	Reserved

ST3	0	No FM module
	1	FM error
	2	FM full

	3	FM near full
	4	Decimal point (1=fract, 0=whole)
	5	FM fiscalized
	6	FM produced
	7	Reserved

ST4	0	Printer: automatic cutting
	1	External Display Management
	2	Reserved
	3	Reserved
	4	Drawer: automatic opening
	5	Customer logo included in the receipt
	6	Service jumper
	7	Reserved

ST5	0	No Sec.IC
	1	No certificates
	2	Reserved
	3	Reserved
	4	No SD card response
	5	Wrong SD card
	6	Reserved
	7	Reserved

**zfpdef:**ReadStatus()

### 2.2.2. Command: 21h / ! - Version

**input:** n. a.

**output:** <Model[100]>

**FPR operation:** Provides information about the device model and firmware version.

**Input data :** n. a.

**Output data :**

Model                      Model information

**zfpdef:** ReadVersion()

### 2.2.3. Command: 22h / " - Diagnostics

**input:** n. a.

**output:** ACK

**FPR operation:** Prints out a diagnostic receipt.

**Input data :** n. a.

**Output data :** n. a.

**zfpdef:**PrintDiagnostics()

### 2.2.4. Command: 24h / # - Clear display

**input:** n. a.

**output:** ACK

**FPR operation:** Clears the display.

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** ClearDisplay()

### 2.2.5. Command: 25h / % - Display text line 1

**input:** <Text[20]>

**output:** ACK

**FPR operation:** Shows a 20-symbol text in the upper display line.

**Input data :**

Text                      20 symbols text

**Output data: n. a.**

**zfpdef:** DisplayText1(Text)

### 2.2.6. Command: 26h / & - Display text line 2

**input:** <Text[20]>

**output:** ACK

**FPR operation:** Shows a 20-symbol text in the lower display line.

**Input data :**

Text                      20 symbols text

**Output data: n. a.**

**zfpdef:** DisplayText2(Text)

### 2.2.7. Command: 27h / ' - Display text lines 1 and 2

**input:** <Text[40]>

**output:** ACK

**FPR operation:** Shows a 40-symbol text in the two display lines.

**Input data :**

Text                      40 symbols text

**Output data: ACK**

**zfpdef:** DisplayText1and2(Text)

### 2.2.8. Command: 28h / ( - Display date and time

**input:** n. a.

**output:** ACK

**FPR operation:** Shows the current date and time on the display.

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** DisplayDateTime()

### 2.2.9. Command: 2Ah / \* - Cash drawer opening

**input:** n. a.

**output:** ACK

**FPR operation:** Opens the cash drawer

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** CashDrawerOpen()

## 2.2.10. Command: 2Bh / + - Paper feeding

input: n. a.

output: ACK

FPR operation: Feeds 1 line of paper.

Input data : n. a.

Output data : n. a.

zfpdef: PaperFeed()

## 2.2.11. Command: 51h / Q – Print barcode 'QP'

input: <'P'> <;> <CodeType[1]> <;> <CodeLen[1..2]> <;> <CodeData[100]>

output: ACK

FPR Operation: Prints barcode from type stated by CodeType and CodeLen and with data stated in CodeData field.

Input data:

'P' 1 character 'P'

CodeType 1 symbol with possible values:

- '0' – UPC A

- '1' – UPC E

- '2' – EAN 13

- '3' – EAN 8

- '4' – CODE 39

- '5' – ITF

- '6' – CODABAR

- 'H' – CODE 93

- 'I' – CODE 128

CodeLen 1..2 bytes for number of bytes according to the table

CodeData From 0 to 255 bytes data in range according to the table

Output data: n.a.

Table:

Barcode type	<CodeType>	<CodeLen>	Range of <CodeData>
UPC-A	'0' or 'A'	11 or 12	Digits from '0' to '9'
UPC-E	'1' or 'B'	11 or 12	Digits from '0' to '9'
JAN13 (EAN13)	'2' or 'C'	12 or 13	Digits from '0' to '9'
JAN8 (EAN8)	'3' or 'D'	7 or 8	Digits from '0' to '9'
CODE 39	'4' or 'E'	from 1 to 10	Characters: 'SP' '\$' '%' '+' '-' '.' '/' Digits from '0' to '9' letters from 'A' to 'Z'
ITF	'5' or 'F'	from 2 to 18 (evens only)	Digits from '0' to '9'
CODABAR	'6' or 'G'	From 1 to 15	Characters: '\$' '+' '-' '/' digits from '0' to '9' letters from 'A' to 'D'
CODE 93	'H'	From 1 to 14	Bytes from 0 to 7F
CODE 128	'I'	From 1 to 12	Bytes from 0 to 7F

The length restriction for some of the barcode types is because of the the print area not because of the barcode standard. If more data is sent the printed barcode may not be read correctly.

zfpdef: PrintBarcode(CodeType, CodeLen, CodeData)

## 2.3. FISCAL COMMANDS

These are commands requiring data recording in the fiscal memory of the device. Password access is required.

### 2.3.1. Command: 40h / @ – SET Manufacturing number

**input:** <Password[6]> <;> <SerialNo[8]>

**output:** ACK

**FPR operation:** Stores the Manufacturing number into the operative memory.

#### **Input data :**

Password 6-symbol string

SerialNo (Serial No) 8 symbol Manufacturing number

#### **Output data: n. a.**

**zfpdef:** SetSerialNo(Password, SerialNo)

### 2.3.1.1. Command: 41h / A (1) – SET VAT and fiscal number

**input:** <Password[6]> <;> <'1'> <;> <VATNo[15]><;> <FMNo[10]><;>  
<TypeVATregistration[1]>

**output:** ACK

**FPR operation:** Stores the VAT registration and Fiscal Memory number into the operative memory.

#### **Input data :**

Password 6-symbol string

'1' One symbol is compulsory 1

VATNo (VAT No) 15 symbols VAT registration number

FMNo (FM No) 10 symbols Fiscal Memory serial number

TypeVATregistration 1 symbol for type of owner's VAT registration:

- '1' – Yes

- '0' – No

#### **Output data: n. a.**

**zfpdef:** SetFiscalNumbers(Password, VATNo, FMNo, TypeVATregistration )

### 2.3.1.2. Command: 41h / A (2) – Confirm VAT and fiscal numbers

**input:** <Password[6]> <;> <'2'> <;>

**output:** ACK

**FPR operation:** Store VAT and Fiscal Memory numbers into the Fiscal memory.

#### **Input data :**

Password 6-symbols string

'2' One symbol for option

#### **Output data: n. a.**

**zfpdef:** ConfirmFiscalization(Password)

### 2.3.2. Command: 42h / B – VAT rate change

**input:** <Password[6]> <;> <VATrateA[2..6]> <;> <VATrateB[2..6]> <;>  
<VATrateC[2..6]> <;> <VATrateD[2..6]> <;> <VATrateE['00.00']>

**output:** ACK

**FPR operation:** Stores a block containing the values of the tax rates into the fiscal memory. Prints the values on the printer.

**Input data :**

Password	6-symbols string
VATrateA	Value of VAT rate A from 2 to 6 symbols with format ##.##
VATrateB	Value of VAT rate B from 2 to 6 symbols with format ##.##
VATrateC	Value of VAT rate C from 2 to 6 symbols with format ##.##
VATrateD	Value of VAT rate D from 2 to 6 symbols with format ##.##
VATrateE	5 symbols with value '00.00'

**Output data: n. a.**

**zfpdef:** ProgVATrates(Password, VATRateA, VATRateB, VATRateC, VATRateD)

### 2.3.3. Command: 43h / C – Change of decimal point position

**input:** < Password [6]> <;> <DecimalPointPosition[1]>

**output:** ACK

**FPR operation:** Stores a block containing the number format into the fiscal memory. Prints the current status on the printer.

**Input data :**

Password	6-symbol string
DecimalPointPosition	1 symbol with values: - '0' - whole numbers - '2' - fractions

**Output data: n. a.**

**zfpdef:** ProgDecPointPosition(Password, OptionDecimalPoint)

### 2.3.4. Command: 57h / W (0) – Restore previous header

**input:** <'0'>

**output:** ACK

**FPR operation:** Restore previous header if current header is not saved into fiscal memory.

**Input data :**

'0'	One symbol for option 0
-----	-------------------------

**Output data: n. a.**

**Zfpdef:** RestorePreviousHeader()

### 2.3.5. Command: 57h / W (1) – Print current header

**input:** <'1'>

**output:** ACK

**FPR operation:** Print operative header buffer and FM header buffer.

**Input data :**

'1'	One symbol for option 1
-----	-------------------------

**Output data: n. a.**

**Zfpdef:** PrintCurrentHeader()

### 2.3.6. Command: 57h / W (2) – Store current header into fiscal memory

input: <'2'><;><Password[6]>

output: ACK

FPR operation: Store the header into fiscal memory.

#### **Input data :**

'2'                      One symbol for option 2

Password                6-symbols string

#### **Output data: n. a.**

**Zfpdef:** *StoreInFMcurrentHeader(Password)*

## 2.4. PROGRAMMING COMMANDS

Set of commands, for programming the FPR configuration according to the POS requirements and the user's needs.

### 2.4.1. Command: 44h / D – Programming of payment types

**input:** <OptionPaymentType[1]><;> <Name[10]> {<;> <Rate[10]>}

**output:** ACK

**FPR operation:** Programs the name of the type of payment.

**Input data :**

*OptionPaymentType* (Payment Types) 1 symbol for payment type:

- '0' – Payment 0
- '1' – Payment 1
- '2' – Payment 2
- '3' – Payment 3
- '4' – Payment 4
- '5' – Payment 5
- '6' – Payment 6
- '7' – Payment 7
- '8' – Payment 8
- '9' – Payment 9

*Name* (Payment Name) 10 symbols for payment type name

*Rate* (Exchange Rate) 10 symbols for exchange rate in format: #####.#####  
of the 9th payment type, maximal value 0420.00000

**Output data: n. a.**

**zfpdef:** *ProgPayment(OptionPaymentType, Name, Rate)*



## 2.4.2. Command: 45h / E - Programming of parameters

**input:** <NoPOS[4]> <;> <PrintLogo[1]> <;> <AutoOpenDrawer[1]> <;> <AutoCut[1]>  
<;> <ExternalDispManagement[1]> <;> <reserved['1']> <;> <EnableCurrency[1]>  
<;> <reserved['1']> <;> <USBHost[1]>

**output:** ACK

**FPR operation:** Programs the number of POS, printing of logo, Cash drawer opening, display mode, cutting permission. Changes in USBHost parameter will power off the printer.

### **Input data :**

NoPOS	(POS No) 4 symbols for number of POS in format ####
PrintLogo	(Print Logo) 1 symbol of value: - '1' - Yes - '0' - No
AutoOpenDrawer	(Auto Open Drawer) 1 symbol of value: - '1' - Yes - '0' - No
AutoCut	(Auto Cut) 1 symbol of value: - '1' - Yes - '0' - No
ExternalDispManagement	(External Display Management) 1 symbol of value: - '1' - Manuel - '0' - Auto
reserved	1 symbols reserved with value '1'
EnableCurrency	(Enable Currency) 1 symbol of value: - '1' - Yes - '0' - No
reserved	1 byte reserved with value '1'
USBHost	(USB in host mode) 1 symbol, FP Only, with value: - '1' - Yes - '0' - No

**Output data:** n. a.

### **Notes:**

The logo is a graphical file in **BMP** format which is printed at the head of every receipt  
“Transparent display” is a mode, in which the FPR does not send information to the display except when executing the 25h, 26h and 27h commands. When this mode is off the FPR “uses” the display to show data during sales, at receipt finalization, etc.

**zfpdef:** ProgParam(NoPOS, PrintLogo, CashDrawer, AutoCut, ExternalDispManagment, ShortEJ, EnableCurrency)

## 2.4.3. Command: 46h / F – Programming the external display

**input:** <Password[6]> <NoBytesCom1line[1]> <Com1line[8]>  
<NoBytesCom2line[1]> <Com2Line[8]> < NoBytesClrDis[1]> <ComClrDis[8]>  
<NobytesXtrCom[1]> <ComXtrCom[1]> <FlagPrecod[1]> {<PrecodTabl[64]>}

**output:** ACK

**FPR operation:** Preprograms the external display.

### **Input data :**

Password	A 6-symbol string
NoBytesCom1line	Number of bytes (X = 1..8), for Command: show on line 1 of the display – 1 byte
Com1line	Command string show on line 1 of the display – 8 bytes, the first X bytes are command

<i>NoBytesCom2line</i>	Number of bytes (Y = 1..8), for Command: show on line 2 of the display – 1 byte
<i>Com2line</i>	String for Command show on line 2 of the display – 8 bytes, the first Y bytes are command
<i>NoBytesClrDis</i>	Number of bytes (Z = 1..8) – 1 byte
<i>ComClrDis</i>	String for Command clear display – 8 bytes, the first Z bytes are command
<i>NoBytesXtrCom</i>	Number of bytes (U = 0..8, 0 if there is no such command), for screensaver mode command – 1 byte, for hello message use line 0 of the template
<i>ComXtrCom</i>	String for Command for screensaver mode – 8 bytes, the first U bytes are command
<i>FlagPrecod</i>	Flag for precoding of the codetable for display cyrillization (0 – w/o precoding, 1 – with precoding) length 1 byte
<i>PrecodTabl</i>	Precoding table with the codes of the Cyrillic alphabet, capital and small letters

**Output data: n. a.**

#### **Notes:**

N command symbols should be specified for the number of bytes command. Then specify 8 bytes of control symbols, the first N of which are the command and the rest will be ignored. However, the symbols must be 8 in order to keep the format. If the display supports animation suitable for *screen-saver* - follow the above steps, otherwise set the <NoBytesXtrCom> as a 0. <FlagShift> is either 0 or 1 depending on whether a Cyrillic precoding is to be done or not. If precoding should be done input the code table.

**zfpdef:** ProgExtDisplay(Password, NoBytesCom1line, Com1line, NoBytesCom2line, Com2Line, NobytesClrDis, ComClrDis, NobytesXtrCom, ComXtrCom, FlagPrecod, PrecodTabl)

#### 2.4.4. Command: 47h / G – Department programming

**input:** <Number[2]> <;><Name[20]> <;> <OptionVATClass[1]> {<;> <Price[1..10]>  
<;> <OptionDepPrice[1]><;> <AdditionalName[14]> <;> <Category[1..7]>}

**output:** ACK

**FPR operation:** Set data for the stated department number from the internal FPR database. Parameters Price, OptionDepPrice, AdditionalName and Category are not obligatory and require the previous not obligatory parameter.

##### **Input data:**

Number	(Department No) 2 symbols department number in format: ##
Name	(Department Name) 20 characters department name
OptionVATClass	1 symbol for article's VAT class with optional values:" - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	Up 10 symbols for department price
OptionDepPrice	1 symbol for Department flags with next value: - '0' - Free price disabled - '1' - Free price enabled - '2' - Limited price - '4' - Free price disabled for single transaction - '5' - Free price enabled for single transaction - '6' - Limited price for single transaction
AdditionalName	14 characters additional department name
Category	From 1 to 7 symbols for categoryin format: #####.##

##### **Output data : n. a.**

###### **Note:**

When changing the VAT class attachment of department must actualize the VAT class of all articles attached to this department. Otherwise they won't be accessible for sale

**zfpdef:** ProgDep(Number, Name, Option VATClass, Price, OptionDepPrice, AdditionalName, Category)

#### 2.4.5. Command: 48h / H – Setting the date and time

**input:** <DateTime “DD-MM-YY HH:MM:SS”>

**output:** ACK

**FPR operation:** Sets the date and time and prints the current values using the RECEIPT printer.

##### **Input data :**

DateTime      Date Time parameter in format: DD-MM-YY [Space] HH:MM:SS

##### **Output data : n. a.**

**zfpdef:** SetDateTime(DateTime)

## 2.4.6. Command: 49H / I – Programming of Display Greeting message

input:<'D'><;><DisplayGreetingText[20]>

output: ACK

FPR operation: Program the contents of a Display Greeting message.

### Input data :

'D' 1 symbol with value 'D'

DisplayGreetingText 20 symbols for display greeting message

Output data: n. a.

[zfpdef:ProgDisplayGreeting\(DisplayGreetingText\)](#)

## 2.4.7. Command: 49h / I – Programming of header lines

input:<'H'><;><OptionHeaderLine[1]><;><HeaderText[TextLength]>

output: ACK

FPR operation: Program the contents of a header lines.

### Input data :

'H' 1 symbol with value 'H'

OptionHeaderLine (Line Number) 1 symbol with value:

- '1' – Header 1

- '2' – Header 2

- '3' – Header 3

- '4' – Header 4

- '5' – Header 5

- '6' – Header 6

- '7' – Header 7

- '8' – Header 8

HeaderText TextLength symbols for header lines

Output data: n. a.

[zfpdef:ProgHeader\(OptionHeaderLine, Text\)](#)

## 2.4.8. Command: 49h / I – Programming of footer line

input:<'F'><;><OptionFooterLine[1]><;><FooterText[TextLength]>

output: ACK

FPR operation: Program the contents of a footer lines.

### Input data :

'F' 1 symbol 'F'

OptionFooterLine (Line Number) 1 symbol for option footer line:

- '1' – Footer 1

- '2' – Footer 2

- '3' – Footer 3

FooterText TextLength symbols for footer line

Output data: n. a.

[zfpdef:ProgFooter\(OptionFooterLine, FooterText\)](#)

#### 2.4.9. Command: 49h / I – Programming of CIF name message

input:<'C'> <;> <Password[6]> <;> <CIFName[8]>

output: ACK

FPR operation: Program the contents of a CIF name message.

##### **Input data :**

'C'	1 symbol with value 'C'
Password	6-symbol string
CIFName	8 symbols for CIF name

**Output data: n. a.**

**zfpdef:**[\*ProgCIFNameMessage\(Password,CIFName\)\*](#)

#### 2.4.10. Command: 49h / I – Programming of storno name message

input:<'S'> <;> <StornoName[TextLength]>

output: ACK

FPR operation: Program the contents of storno name message.

##### **Input data :**

'S'	1 symbol with value 'S'
StornoName	TextLength symbols for storno name

**Output data: n. a.**

**zfpdef:**[\*ProgStornoNameMessage\(StornoName\)\*](#)

#### 2.4.11. Command: 49h / I – Programming of VAT number

input:<'V'> <;><Password[6]> <;> <OwnerVATNo[15]><;> <TypeVATregistration[1]>

output: ACK

FPR operation: Program the owner's VAT number.

##### **Input data :**

'V'	1 symbol with value 'V'
Password	6-symbol string
OwnerVATNo	15 symbols for VAT number
TypeVATregistration	1 symbol for type of owner's VAT registration: - '1' – Yes - '0' – No

**Output data: n. a.**

**zfpdef:**[\*ProgCustomerVATNo\(Password,OwnerVATNo, TypeVATregistration\)\*](#)

#### 2.4.12. Command: 49h / I – Programming of customer receipt name

input:<'I'> <;> <CustRecName[TextLength]>

output: ACK

FPR operation: Program the contents of customer receipt name message.

##### **Input data :**

'I'	1 symbol with value 'I'
CustRecName	TextLength symbols for customer receipt name

**Output data: n. a.**

**zfpdef:**[\*ProgCustomerRcpNameMessage\(CustRecName\)\*](#)

### 2.4.13. Command: 4Ah / J – Programming of operator name and password

**input:** <Number[2]> <;> <Name[20]> <;> <Password[4]>

**output:** ACK

**FPR operation:** Programs the operator's name and password.

#### **Input data :**

*Number* (Operator No) 2 Symbol from 1 to 20 with leading zeroes in format: ## corresponding to the operator's number  
*Name* (Operator Name) 20 symbols for operator's name  
*Password* (Operator Password) 4 symbols for operator's password

#### **Output data : n. a.**

**zfpdef:** ProgOperator(Number, Name, Password)

### 2.4.14. Command: 4Bh / K – option 1, Programming of PLU

**input:** <PLUNo[5]> <;> <Option['1']> <;> <Name[34]> <;> <Price[1..10]> <;> <OptionPrice[1]> <;> <OptionVATClass[1]> <;> <BelongToDepNo[1..2]> <;> <AlteTaxNum[1..11]> <;> <AlteTaxValue[1..11]> {<;> <OptionTransaction[1]>}

**output:** ACK

**FPR operation:** Programs internal database items.

#### **Input data :**

*PLUNo* (PLU No) 5 symbols for article number in format: #####  
*Option* '1'  
*Name* 34 symbols for article name; Symbol for LF=7Ch - '|'; separator for MU=80h or 60h followed up to 3 symbols for unit.  
*Price* 1 to 10 symbols for article price  
*OptionPrice* 1 byte for Price flag with next value:  
- '0'- Free price is disable /valid only programmed price/  
- '1'- Free price is enable  
- '2'- Limited price  
*OptionVATClass* 1 symbol for article VAT class:  
- 'A' – VAT class A  
- 'B' – VAT class B  
- 'C' – VAT class C  
- 'D' – VAT class D  
- 'E' – VAT class E  
- 'F' – Alte taxe  
*BelongToDepNo* BelongToDepNo + 80h. 1 symbol for article department attachment, formed in the following manner: example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h  
*AlteTaxNum* 1..11 symbol for Alte Tax number  
*AlteTaxValue* 1..11 symbols for Alte tax value  
*OptionTransaction* 1 symbol with value:  
- '1' - Active Single transaction in receipt  
- '0' – Inactive /default value/

Note: this parameter is not obligatory

#### **Output data: n. a.**

**zfpdef:** ProgPLU1(PLUNo, Name, Price, OptionPrice, OptionVATClass, BelongToDepNo, AlteTaxNum, AlteTaxValue, OptionTransaction)

### 2.4.15. Command: 4Bh / K – option 2, Programming of quantity in stock

**input:** <PLUNo[5]> <;> <Option['2']> <;> <AvailQTY[1..11]> <;> <OptionQtyType[1]>

**output:** ACK

**FPR operation:** Program the available quantity for a certain article (item) from the internal database.

**Input data :**

PLUNo (PLU No) 5 symbols for article number in format: #####  
Option '2'  
AvailQTY (Available Quantity) 1..11 symbols for quantity in stock  
OptionQtyType 1 byte for Quantity flag with next value:  
- '0'- for availability of PLU stock is not monitored  
- '1'- for disable negative quantity  
- '2'- for enable negative quantity

**Output data: n. a.**

**zfpdef:** *ProgPLU2(PLUNo, AvailQty, OptionQtyType)*

## 2.4.16. Command: 4Bh / K – option 3, Programming of PLU barcode

**input:** <PLUNo[5]><;><Option['3']><;><Barcode[13]>

**output:** ACK

**FPR operation:** Program the Barcode number for a certain article (item) from the internal database.

**Input data :**

PLUNo (PLU No) 5 symbols for article number in format: #####  
Option '3'  
Barcode 13 symbols for barcode

**Output data: n. a.**

**zfpdef:** *ProgPLU3(PLUNo, Barcode)*

## 2.4.17. Command: 4Bh / K – option 4, Programming of PLU price

**input:** <PLUNo[5]><;><Option['4']><;><Price[1..10]><;><OptionPrice[1]><;><AlteTaxNum[1..11]><;> <AlteTaxValue[1..11]>

**output:** ACK

**FPR operation:** Program the price for a certain article (item) from the internal database.

**Input data :**

PLUNo (PLU No) 5 symbols for article number in format: #####  
Option '4'  
Price 1 to 10 symbols for article price  
OptionPrice 1 byte for Price flag with next value:  
- '0'- Free price is disable /valid only programmed price/  
- '1'- Free price is enable  
- '2'- Limited price  
AlteTaxNum 1..11 symbol for Alte Tax number  
AlteTaxValue 1..11 symbols for Alte tax value

**Output data: n. a.**

**zfpdef:** *ProgPLU4(PLUNo, Price, OptionPrice, AlteTaxNum, AlteTaxValue)*

## 2.4.18. Command: 4Bh / K – option 5, Programming of PLU Category

**input:** <PLUNo[5]><;><Option['5']><;><Category[1..7]>

**output:** ACK

**FPR operation:** Programs the PLU Category for a certain article (item) from the internal database.



**Input data :**

PLUNo (PLU No) 5 symbols for article number in format: #####  
 Option '5'  
 Category Up to 7 symbols for PLU Category code in format ####.##

**Output data: n. a.**

**zfpdef:** *ProgPLU5(PLUNo, Category)*

**2.4.19. Command: 4Bh / K – option \$, Erase all PLU data base**

**input:** <PLUNo["00000"]><;><Option["\$"]><;><Password[6]>

**output:** ACK

**FPR operation:** Programs the PLU Category for a certain article (item) from the internal database.

**Input data :**

PLUNo 5 symbols '00000'  
 Option '\$'  
 Password 6 symbols for password

**Output data: n. a.**

**zfpdef:** *EraseAllPLUs(PLUNo, Password)*

**2.4.20. Command: 4Ch / L – Programming of Logo without setting a number (default number 0)**

**input:** <BMPfile[9022]>

**output:** ACK

**FPR Operation:** Stores in the memory the graphic file under number 0. Prints information about loaded in the printer graphic files.

**Input data:**

BMPfile \*BMP file with fixed size 9022 bytes

**Output data: n.a.****Notes:**

FP has the ability to store up to 10 different BMP files for logo with numbers from 0 to 9, as one of them is „active“ and is printed as receipt's logo. If there is no file loaded under the number, stated as „active“, FP will work as set for work without logo.

**zfpdef:** *ProgLogo(BMPfile)*

**2.4.21. Command: 4Dh / M- Programming of logo with setting a number**

**input:** <LogoNumber[1]> <BMPfile[9022]>

**output:** ACK

**FPR Operation:** Stores in the memory the graphic file under stated number. Prints information about loaded in the printer graphic files.

**Input data:**

LogoNumber 1 character value from '0' to '9' setting the number where the logo will be saved.

BMPfile \*BMP file with fixed size 9022 bytes

**Output data: n.a.**

**zfpdef:** *ProgLogoNo(LogoNumber, BMPfile)*



## 2.4.22. Command: 23h / #- Set active logo file number

**Input:** <LogoFileName[1]>

**output:** ACK

**FPR Operation:** Sets the number of logo file, which is active and will be printed as logo in the receipt header. Prints Information about active number.

### **Input data:**

LogoFileName      1 character value from '0' to '9' or '?'. The number sets the active file, and the '?' invokes only printing of information

**Output data: n. a.**

**zfpdef:** SetActiveLogoNo(LogoFileName)

## 2.4.23. Command: 52h / R – option P, Programming of Customer data

**input:** <Option['P']> <;><Number[3]> <;> <VatNo[15]> <;> <Name[30]> <;>  
<Addr[30]> <;><FreeLine1[20]> <;> <FreeLine2[20]> <;> <FreeLine3[20]> <;>  
<FreeLine4[20]>

**output:** ACK

**FPR operation:** Programs the customer DB for special customer receipt issuing.

### **Input data :**

Option	'P' – for programming customer data
Number	3 symbols for customer number in order in format ###
VatNo	15 symbols for customer VAT registration number
Name	30 symbols for customer name
Addr	30 symbols for customer address
FreeLine1	20 ASCII symbols for customer data
FreeLine2	20 ASCII symbols for customer data
FreeLine3	20 ASCII symbols for customer data
FreeLine4	20 ASCII symbols for customer data

**Output data: n. a.**

**zfpdef:** ProgCustomerData(Option, VatNo, Name, Addr, FreeLine1, FreeLine2, FreeLine3, FreeLine4)

## 2.4.24. Command: 53h / S – option A, Programming of Other Charges (Alte tax) name

**input:** <Option['A']> <;><Option['P']> <;><Number[1]> <;> <Name[12]>

**output:** ACK

**FPR operation:** Program the other charges (Alte tax) name.

### **Input data :**

Option	'A' – for alte tax data
Option	'P' – for programming
Number	1 symbol for number in order up to 7
Name	12 symbols for Alte tax name

**Output data: n. a.**

**zfpdef:** ProgAlteTaxe(Number, Name)

#### 2.4.25. Command: 53h / S – option L, option 1, Programming of service contract date

**input:** <Option['L']> <;><'P'[1]> <;><Password[6]> <;> <'1'[1]><;>  
<ExpiryDate "DD-MM-YYYY">

**output:** ACK

**FPR operation:** Program the service contract expiry date.

##### **Input data :**

Option	'L' – for service contract
'P'	'P' – for programming
Password	6 symbols for service password
'1'	'1' – for date
ExpiryDate	10 symbols for expiry date of service contract

**Output data: n. a.**

**zfpdef:** ProgServiceContractDate(Option, Password, ExpiryDate)

#### 2.4.26. Command: 53h / S – option L, option 2, Disable the service contract function

**input:** <Option['L']> <;><'P'[1]> <;><Password[6]> <;> <'2'[1]>

**output:** ACK

**FPR operation:** Set service contract disable

##### **Input data :**

Option	'L' – for service contract
'P'	'P' – for programming
Password	6 symbols for service password
'2'	'2' – for disable service contract function

**Output data: n. a.**

**zfpdef:** DisableContractFunction(Option, Password)

#### 2.4.27. Command: 53h / S – option L, option 3, Programming of service contract warning messages

**input:** <Option['L']> <;><'P'[1]> <;><Password[6]> <;> <'3'[1]><;>  
<Line1[TextLength]> <;><Line2[TextLength]> <;> <Line3[TextLength]>

**output:** ACK

**FPR operation:** Program the service contract expired warning message text.

##### **Input data :**

Option	'L' – for service contract
'P'	'P' – for programming
Password	6 symbols for service password
'3'	'3' – for warning message
Line1	TextLength symbols for warning message of line 1
Line2	TextLength symbols for warning message of line 2
Line3	TextLength symbols for warning message of line 3

**Output data: n. a.**

**zfpdef:** ProgContractWarningMessage(Option, Password, Line1, Line2, Line3)

## 2.4.28. Command: 4Fh / O – Program the duplicates number of the invoice receipt

input: <Option1['D']> <;><Option2['W']><;> <DuplicatesNumber[1]>

output: ACK

**FPR operation:** Program the the number of the duplicates which can be printed after invoice receipt.

### **Input data :**

Option1	1 symbol with value 'D'
Option2	1 symbol with value 'W'
DuplicatesNumber	1 symbol for number of duplicates which can be print. Possible values from '0' to '5'.

**Output data: n. a.**

**zfpdef:** ProgDuplicatesNumber(DuplicatesNumber)

## 2.5. DATA READING COMMANDS

Set of commands for receiving information from the FPR about programmed values as well as additional information.

### 2.5.1. Command: 60h / ` – Reading the FPR numbers

**input:** n. a.

**output:** <SerialNo[10]>

**FPR operation:** Provides information about the manufacturing number of the fiscal device.

**Input data : n. a.**

**Output data :**

SerialNo            10 symbols for individual number of the fiscal device

**Zfpdef:** *ReadSerialNo()*

### 2.5.2. Command: 61h / a – Reading the VAT number

**input:** n. a.

**output:** <VATNo[15]><;><FMnumber[10]><;><TypeVATregistration[1]>

**FPR operation:** Read the VAT registration and Fiscal Memory numbers.

**Input data : n. a.**

**Output data :**

VATNo                      15 symbols for owner's VAT registration number

FMnumber                10 symbols for FM serial number

TypeVATregistration    1 symbol for type of owner's VAT registration:

- '1' – Yes

- '0' – No

**Zfpdef:** *ReadVATNumber()*

### 2.5.3. Command: 62h / b – Reading the VAT rates

**input:** n. a.

**output:** < VATRateA[1..6]> <;> < VATRateB[1..6]> <;> < VATRateC[1..6]> <;>  
<VATRateD[1..6]> <;> <VATRateE[1..6]> <;> <AlteTaxeF[1..6]>

**FPR operation:** Provides information about the current tax rates (the last values stored into the FM).

**Input data : n. a.**

**Output data : n. a.**

VATRateA            6 symbols for VAT rates of VAT class A in format ##.##%

VATRateB            6 symbols for VAT rates of VAT class B in format ##.##%

VATRateC            6 symbols for VAT rates of VAT class C in format ##.##%

VATRateD            6 symbols for VAT rates of VAT class D in format ##.##%

VATRateE            6 symbols for VAT rates of VAT class E in format ##.##%

AlteTaxeF            6 symbols for VAT rates of Alte Taxe F in format ##.##%

**Zfpdef:** *ReadVATrates()*

#### 2.5.4. Command: 63h / c – Reading the decimal point

**input:** n. a.

**output:** <DecimalPointPosition[1]>

**FPR operation:** Provides information about the current (the last value stored into the FM) decimal point format.

**Input data : n. a.**

**Output data :**

*DecimalPointPosition* 1 symbol with values:  
- '0' - whole numbers  
- '2' - fractions

**Zfpdef:** *ReadDecPoint()*

#### 2.5.5. Command: 64h / d – Reading the types of payment

**input:** n. a.

**output:** <NamePaym0[10]> <;> <NamePaym1[10]> <;> <NamePaym2[10]> <;>  
<NamePaym3[10]> <;> <NamePaym4[10]> <;> <NamePaym5[10]> <;>  
<NamePaym6[10]> <;> <NamePaym7[10]> <;> <NamePaym8[10]> <;>  
<NamePaym9[10]> <;> <ExRate[10]>

**FPR operation:** Provides information about all programmed types of payment.

**Input data : n. a.**

**Output data :**

*NamePaym0* 10 symbols for type 0 of payment name  
*NamePaym1* 10 symbols for type 1 of payment name  
*NamePaym2* 10 symbols for type 2 of payment name  
*NamePaym3* 10 symbols for type 3 of payment name  
*NamePaym4* 10 symbols for type 4 of payment name  
*NamePaym5* 10 symbols for type 5 of payment name  
*NamePaym6* 10 symbols for type 6 of payment name  
*NamePaym7* 10 symbols for type 7 of payment name  
*NamePaym8* 10 symbols for type 8 of payment name  
*NamePaym9* 10 symbols for type 9 of payment name  
*ExRate* 10 symbols for exchange rate of payment type 9 in format: #####.#####

**Zfpdef:** *ReadPayments()*

#### 2.5.6. Command: 65h / e – Reading of parameters

**input:** n. a.

**output:** <NoPOS[4]> <;> <PrintLogo[1]> <;> <AutoOpenDrawer[1]> <;>  
<AutoCut[1]> <;> <ExternalDispManagement[1]> <;> <reserved['1']>  
<;> <EnableCurrency[1]> <;> <reserved['1']> <;> <USBHost[1]>

**FPR operation:** Provides information about the programmed number of POS and the current values of the logo and Cash drawer options.

**Input data : n. a.**

**Output data :**

*NoPOS* (POS No) 4 symbols for number of POS in format ####  
*PrintLogo* (Print Logo) 1 symbol of value:  
- '1' – Yes  
- '0' - No  
*AutoOpenDrawer* (Auto Open Drawer) 1 symbol of value:  
- '1' - Yes  
- '0' - No

<i>AutoCut</i>	(Auto Cut) 1 symbol of value: - '1' - Yes - '0' - No
<i>ExternalDispManagement</i>	(External Display Management) 1 symbol of value: - '1' - Manuel - '0' - Auto
<i>reserved</i>	1 symbol reserved with value '1'
<i>EnableCurrency</i>	(Enable Currency) 1 symbol of value: - '1' – Yes - '0' – No
<i>reserved</i>	1 byte reserved with value '1'
<i>USBHost</i>	(USB in host mode)1 symbol with value: - '1' – Yes - '0' – No

**zfpdef:** *ReadParam()*

### 2.5.7. Command: 67h / g – Read of department registers

**input:** <Number[2]>

**output:** <Number[2]> <;><Name[34]> <;> <OptionVATClass[1]> <;>  
 <Turnover[1..11]> <;> <QtySold[1..11]> <;><LastZrepNum[5]> <;>  
 <LastZrepDate “DD-MM-YYYY HH:MM”> <;><Price[1..10]> <;> <OptionDepPrice[1]> <;>  
 <Category[1..7]>

**FPR operation:** Provides information for the programmed data, the turnover from the stated department number

#### **Input data :**

Number (Department No) 2 symbols for department number in format: ##

#### **Output data :**

Number 2 symbols for department number in format ##

Name 34 symbols for department name

*OptionVATClass* 1 symbol for article's VAT class with optional values:"

- 'A' – VAT class A

- 'B' – VAT class B

- 'C' – VAT class C

- 'D' – VAT class D

- 'E' – VAT class E

- 'F' – Alte taxe

Turnover 1..11 symbols for accumulated turnover of the department

QtySold 1..11 symbols for sold quantity of the department

LastZrepNum 5 symbols for the number of last Z report in format #####

LastZrepDate 16 symbols for the date and hour in last Z report

Price 1 to 10 symbols for department price

*OptionDepPrice* 1 symbol for Department flags with next value:

- '0' - Free price disabled

- '1' - Free price enabled

- '2' - Limited price

- '4' - Free price disabled for single transaction

- '5' - Free price enabled for single transaction

- '6' - Limited price for single transaction

Category Up to 7 symbols for PLU Category code in format #####.##

**zfpdef:** *ReadDep(DepNo)*

### 2.5.8. Command: 68h / h – Reading the date and time

**input:** n. a.

**output:** <DateTime “DD-MM-YYYY HH:MM”>

**FPR operation:** Provides information about the current date and time.

**Input data : n. a.**

**Output data :**

*DateTime* Date Time parameter in format: DD-MM-YY [Space] hh:mm:ss

**Zfpdef:** *ReadDateTime()*

### 2.5.9. Command: 69h / i – Reading the display greening message

**input:** <'D'>

**output:** <'D'> <;> <DisplayGreetingText[20]>

**FPR operation:** Provide information about the display greeting message.

**Input data :**

'D' 1 symbol with value 'D'

**Output data:**

'D' 1 symbol with value 'D'

*DisplayGreetingText* 20 symbols for greeting message

**Zfpdef:** *ReadDisplayGreeting()*

### 2.5.10. Command: 69h / i – Reading the header lines

**input:** <'H'><;><OptionHeaderLine[1]>

**output:** <'H'><;><OptionHeaderLine[1]> <;><HeaderText[TextLength]>

**FPR operation:** Provide information about the contents of the line.

**Input data :**

*OptionHeaderLine* (Line Number)1 byte with value:  
- '1' – Header 1  
- '2' – Header 2  
- '3' – Header 3  
- '4' – Header 4  
- '5' – Header 5  
- '6' – Header 6  
- '7' – Header 7  
- '8' – Header 8

**Output data:**

'H' 1 symbol with value 'H'

*OptionHeaderLine* (Line Number)1 byte with value:  
- '1' – Header 1  
- '2' – Header 2  
- '3' – Header 3  
- '4' – Header 4  
- '5' – Header 5  
- '6' – Header 6  
- '7' – Header 7  
- '8' – Header 8

*HeaderText* LineLength symbols

**Zfpdef:** *ReadHeader(OptionHeaderLine)*

### 2.5.11. Command: 69h / i – Reading the footer line

**input:** <'F'><;><OptionFooterLine[1]>

**output:** <'F'><;><OptionFooterLine[1]> <;> <FooterText[TextLength]>

**FPR operation:** Provide information about the contents of the footer line.

#### **Input data :**

'F' 1 symbol with value 'F'  
OptionFooterLine (Line Number) 1 symbol with value:  
- '1' – Footer 1  
- '2' – Footer 2  
- '3' – Footer 3

#### **Output data:**

'F' 1 symbol with value 'F'  
OptionFooterLine (Line Number) 1 symbol with value:  
- '1' – Footer 1  
- '2' – Footer 2  
- '3' – Footer 3  
FooterText LineLength symbols for footer line

**Zfpdef:** [ReadFooter\(OptionFooterLine\)](#)

### 2.5.12. Command: 69h / i – Reading CIF name message

**input:** <'C'>

**output:** <'C'> <;> <CIFName[8]>

**FPR operation:** Provide information about the CIF name message.

#### **Input data :**

'C' 1 symbol with value 'C'

#### **Output data:**

'C' 1 symbol with value 'C'  
CIFName 8 symbols for CIF name message

**Zfpdef:** [ReadCIFNameMessage\(\)](#)

### 2.5.13. Command: 69h / i – Reading storno name message

**input:** <'S'>

**output:** <'S'> <;> <StornoName[TextLength]>

**FPR operation:** Provide information about the Storno Name Message.

#### **Input data :**

'S' 1 symbol with value 'S'

#### **Output data:**

'S' 1 symbol with value 'S'  
StornoName TextLength symbols for storno name

**Zfpdef:** [ReadStornoNameMessage\(\)](#)

### 2.5.14. Command: 69h / i – Reading VAT number

**input:** <'V'>

**output:** <'V'> <;> <CustomerVATNo[15]><;> <TypeVATregistration[1]>

**FPR operation:** Provide information about the VAT number.

#### **Input data :**

'V' 1 symbol with value 'V'



**Output data:**

'V'	1 symbol with value 'V'
CustomerVATNo	15 symbols for VAT number
TypeVATregistration	1 symbol for type of owner's VAT registration: - '1' – Yes - '0' – No

**Zfpdef:** *ReadCustomerVATNo()*

## 2.5.15. Command: 6Ah / j – Reading the operator's name and password

**input:** <Number[1..2]>

**output:** <Number[1..2]> <;> <Name[20]> <;> <Password[4]>

**FPR operation:** Provides information about an operator's name and password.

**Input data :**

Number	(Operator No) Symbol from 1 to 20 corresponding to the number of operator
--------	---

**Output data:**

Number	Symbol from 1 to 20 corresponding to the number of operator
Name	20 symbols for operator's name
Password	4 symbols for operator's password

**zfpdef:** *ReadOperatorNamePassword(Number)*

## 2.5.16. Command: 6Bh / k – option “ (all), Reading of article registers

**input:** <PLUNo[1..5]><;><Option[""]>

**output:** <PLUNo[1..5]><;><Option[""]><;><PLUName[34]><;><Price[1..10]><;>  
<FlagsPricePLU[1]><;><OptionVATClass[1]><;> <BelongToDepNo[1]> <;>  
<AlteTaxNum[1..11]><;> <AlteTaxValue[1..11]><;> < TurnoverAmount [1..10]> <;>  
<SoldQuantity[1..11]><;>< LastZrepNum [1..5]><;>  
<LastZreportDate “DD-MM-YYYY HH:MM”> <;> <AvailableQTY[1..11]><;>  
<Barcode[13]><;> <AlteTaxAmount[1..11]> <;><Category[1..7]>

**FPR operation:** Provides information about the all registers of the specified article.

**Input data :**

PLUNo	(PLU No) 1..5 symbols for article number with leading zeroes in format: #####
Option	One symbol with value ""

**Output data :**

PLUNo	1..5 symbols for article number with leading zeroes in format: #####
Option	One symbol with value ""
PLUName	34 symbols for article name (LF=7Ch, MU separator = 80h or 60h followed up to 3 symbols for unit)
Price	1..10 symbols for article price
FlagsPricePLU	1 symbol for flags = 0x80 + FlagSinglTr + FlagQTY + OptionPrice Where OptionPrice: 0x00 - for free price is disable /valid only programmed price/ 0x01 - for free price is enable 0x02 - for limited price FlagQTY: 0x00 - for availability of PLU stock is not monitored 0x04 - for disable negative quantity 0x08 - for enable negative quantity FlagSingleTr: 0x00 – no single transaction 0x10 – single transaction is active
OptionVATClass	1 symbol for article's VAT class with optional values:"

- 'A' – VAT class A
- 'B' – VAT class B
- 'C' – VAT class C
- 'D' – VAT class D
- 'E' – VAT class E
- 'F' – Alte taxe

<i>BelongToDepNo</i>	<i>BelongToDepNo</i> + 80h, 1 symbol for PLU department = 0x80 ... 0x93
<i>AlteTaxNum</i>	1..11 symbol for Alte Tax number
<i>AlteTaxValue</i>	1..11 symbols for Alte tax value
<i>TurnoverAmount</i>	1..11 symbols for PLU accumulated turnover
<i>SoldQuantity</i>	1..11 symbols for Sales quantity of the article
<i>LastZrepNum</i>	1..5 symbols for the number of the last article report with zeroing
<i>LastZreportDate</i>	16 symbols for the date and time of the last article report with zeroing
<i>AvailableQTY</i>	(Available Quantity) - 1..11 symbols for quantity in stock
<i>Barcode</i>	13 symbols for article barcode
<i>AlteTaxAmount</i>	1..11 symbols for Alte tax amount
<i>Category</i>	Up to 7 symbols for PLU Category code in format #####.##
<i>zfpdf: ReadPLU0(PLUNo)</i>	

## 2.5.17. Command: 6Bh / k – option 1 (general), Reading of article registers

**input:** <PLUNo[5]><;><Option['1']>

**output:** <PLUNo[5]><;><Option['1']><;><PLUName[34]><;><Price[1.. 10]><;>  
 <OptionPrice[1]><;> <OptionVATClass[1]> <;> <BelongToDepNo[1]><;>  
 <AlteTaxNum[1.. 11]><;> <AlteTaxValue[1.. 11]> <;> <TurnoverAmount[1.. 10]> <;>  
 <SoldQuantity[1.. 11]><;> <NoLastZ[1.. 5]><;> <LastZreportDate "DD-MM-YYYY  
 HH:MM"><;> <SingleTr[1]> <;> <AlteTaxTurnover[1.. 11]>

**FPR operation:** Provides information about the registers of the specified article.

### Input data :

<i>PLUNo</i>	(PLU No) 5 symbols for article number with leading zeroes in format: #####
<i>Option</i>	One symbol with value '1'

### Output data :

<i>PLUNo</i>	5 symbols for article number with leading zeroes in format #####
<i>Option</i>	One symbol with value '1'
<i>PLUName</i>	34 symbols for article name /LF=7Ch, MU separator = 80h or 60h followed up to 3 symbols for unit/
<i>Price</i>	1..10 symbols for article price
<i>OptionPrice</i>	1 symbol for price flag with next value: - '0'- Free price is disable /valid only programmed price/ - '1'- Free price is enable - '2'- Limited price
<i>OptionVATClass</i>	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
<i>BelongToDepNo</i>	<i>BelongToDepNo</i> + 80h, 1 symbol for PLU department = 0x80 ... 0x93
<i>AlteTaxNum</i>	1..11 symbol for Alte Tax number
<i>AlteTaxValue</i>	1..11 symbols for Alte tax value
<i>TurnoverAmount</i>	1..11 symbols for PLU accumulated turnover
<i>SoldQuantity</i>	1..11 symbols for Sales quantity of the article
<i>NoLastZ</i>	5 symbols for the number of the last article report with zeroing in format #####
<i>LastZreportDate</i>	16 symbols for the date and time of the last article report with zeroing

*SingleTr* 1 symbol with value:  
 - '1' - Active Single transaction in receipt  
 - '0' - Inactive /default value/  
*AlteTaxTurnover* 1.11 symbols for Alte Tax Turnover  
*zfpdef: ReadPLU1(PLUNo)*

## 2.5.18. Command: 6Bh / k – option 2 (QTY), Reading of article registers

**input:** <PLUNo[5]><;><Option['2']>

**output:** <PLUNo[5]><;><Option['2']><;><AvailQTY[1..11]><;><OptionQTY[1]>

**FPR operation:** Provides information about the registers of the specified article.

### **Input data :**

*PLUNo* (PLU No) 5 symbols for article number with leading zeroes in format: #####

*Option* One symbol with value 2

### **Output data :**

*PLUNo* 5 symbols for article number with leading zeroes in format #####

*Option* One symbol with value 2

*AvailQTY* (Available Quantity) - 1..11 symbols for quantity in stock

*OptionQTY* 1 byte for Quantity option with next value:  
 - '0'- Availability of PLU stock is not monitored  
 - '1'- Disable Negative Quantity  
 - '2'- Enable Negative Quantity

*zfpdef: ReadPLU2(PLUNo)*

## 2.5.19. Command: 6Bh / k – option 3 (barcode), Reading of article registers

**input:** <PLUNo[5]><;><Option['3']>

**output:** <PLUNo[5]><;><Option['3']><;><Barcode[13]>

**FPR operation:** Provides information about the registers of the specified article.

### **Input data :**

*PLUNo* (PLU No) 5 symbols for article number with leading zeroes in format: #####

*Option* One symbol with value 3

### **Output data :**

*PLUNo* 5 symbols for article number with leading zeroes in format #####

*Option* One symbol with value 3

*Barcode* 13 symbols for article barcode

*zfpdef: ReadPLU3(PLUNo)*

## 2.5.20. Command: 6Bh / k – option 4 (price), Reading of article registers

**input:** <PLUNo[5]><;><Option['4']>

**output:** <PLUNo[5]><;><Option['4']><;><Price[1..10]><;><OptionPrice[1]><;>

<AlteTaxNum[1]><;> <AlteTaxValue[1..11]>

**FPR operation:** Provides information about the registers of the specified article.

### **Input data :**

*PLUNo* (PLU No) 5 symbols for article number with leading zeroes in format: #####

*Option* One symbol with value 4

### **Output data :**

*PLUNo* 5 symbols for article number with leading zeroes in format #####

*Option* One symbol with value 4

*Price* 1..10 symbols for article price

OptionPrice 1 byte for Price flag with next value:  
 - '0'- Free price is disable /valid only programmed price/  
 - '1'- Free price is enable  
 - '2'- Limited price  
 AlteTaxNum 1 symbol for Alte Tax number  
 AlteTaxValue 1..11 symbols for Alte tax value  
**zfpdef:** ReadPLU4(PLUNo)

### 2.5.21. Command: 6Bh / k – option 5 (Category), Reading of article registers

**input:** <PLUNo[5]><;><Option['5']>  
**output:** <PLUNo[5]><;><Option['5']><;><Category[1..7]>  
**FPR operation:** Read the PLU Category code.

#### **Input data :**

PLUNo (PLU No) 5 symbols for article number with leading zeroes in format: #####  
 Option One symbol with value 5

#### **Output data :**

PLUNo 5 symbols for article number with leading zeroes in format #####  
 Option One symbol with value 5  
 Category Up to 7 symbols for PLU Category code in format #####.##

**zfpdef:** ReadPLU5(PLUNo)

### 2.5.22. Command: 6Ch / l – Logo printing

**input:** <Number[1..2]>  
**output:** ACK  
**FPR operation:** Prints the programmed graphical logo with the stated number.

#### **Input data :**

Number Number of logo to be printed. If missing prints logo with number 0

#### **Output data : n. a.**

**zfpdef:**PrintLogo(Number)

### 2.5.23. Command: 52h / R – option R, Reading of Customer data

**input:** <Option['R']> <;><CustomerNum[3]>  
**output:** <Option['R']><;><CustomerNum[3]> <;> <CustomerVatN[15]> <;>  
 <CustomerName[30]> <;> <CustomerAddr[30]> <;><FreeLine1[20]> <;> <FreeLine2[20]>  
 <;> <FreeLine3[20]> <;> <FreeLine4[20]><;> <CustTurn[1..11]>  
**FPR operation:** Reads the customer DB

#### **Input data :**

Option 'R' – for reading customer data  
 CustomerNum 3 symbols for customer number in order in format ###

#### **Output data:**

Option 'R' – for reading customer data  
 CustomerNum 3 symbols for customer number in order in format ###  
 CustomerVatN 15 symbols for customer VAT registration number  
 CustomerName 30 symbols for customer name  
 CustomerAddr 30 symbols for customer address  
 FreeLine1 20 ASCII symbols for customer data  
 FreeLine2 20 ASCII symbols for customer data  
 FreeLine3 20 ASCII symbols for customer data

FreeLine4            20 ASCII symbols for customer data  
 CustTurn            1..11 symbols for accumulated turnover of the customer  
**zfpdef:** [ReadClientData\(ClientNo\)](#)

## 2.5.24. Command: 53h / S – option L, option 1, Reading of service contract date

**input:** <Option['L']> <;><'R'[1]><;><Password[6]> <;> <'1'[1]>  
**output:** <Option['L']> <;><'R'[1]><;> <'1'[1]><;><ExpiryDate "DD-MM-YYYY">  
**FPR operation:** Read the the service contract expiry date.

### **Input data :**

Option            'L' – for alte taxe data  
 'R'               'R' – for reading  
 Password        6 symbols for service passowrd  
 '1'               '1' – for date

### **Output data: n. a.**

Option            'L' – for alte taxe data  
 'R'               'R' – for reading  
 '1'               '1' – for date  
 ExpiryDate      10 symbols for expiry date of service contract

**zfpdef:** [ReadServiceContractDate\(Option, Password\)](#)

## 2.5.25. Command: 53h / S – option L, option 3, Reading service contract warning messages.

**input:** <Option['L']> <;><'R'[1]><;><Password[6]> <;> <'3'[1]>  
**output:** <Option['L']> <;><'R'[1]><;> <'3'[1]><;><Line1[TextLength]>  
 <;><Line2[TextLength]> <;><Line3[TextLength]>  
**FPR operation:** Read the service contract expired warning message text.

### **Input data :**

Option            'L' – for alte taxe data  
 'R'               'R' – for reading  
 Password        6 symbols for service passowrd  
 '3'               '3' – for warning message

### **Output data: n. a.**

Option            'L' – for alte taxe data  
 'R'               'R' – for reading  
 '3'               '3' – for warning message  
 Line1            TextLength symbols for warning message for line 1  
 Line2            TextLength symbols for warning message for line 2  
 Line3            TextLength symbols for warning message for line 3

**zfpdef:** [ReadServiceWarningMessage\(Password\)](#)

### 2.5.26. Command: 53h / S – option A, Reading of Other Charges (Alte taxe) name

**input:** <Option['A']> <;><Option['R']> <;><Number[1]>

**output:** <Option['A']> <;><Option['R']> <;><Number[1]> <;> <Name[12]>

**FPR operation:** Read the other charges (Alte taxe) name.

#### **Input data :**

Option                    'A' – for alte taxe data  
Option                    'R' – for reading  
Number                   1 symbol for number in order up to 7

#### **Output data: n. a.**

Option                    'A' – for alte taxe data  
Option                    'R' – for reading  
Number                   1 symbol for number in order  
Name                      12 symbols for Alte taxe name

**zfpdef:** [ReadAlteTaxe\(Number\)](#)

### 2.5.27. Command: 58h / X – option O - Reset Odometer function

**Input:** <Option['O']><;><Option['Z']><;><Password[6]>

**output:** **ACK**

**FPR Operation:** Reset odometer function

*\*the command is valid for ADPOS model only.*

#### **Input data:**

Option                   1 symbol 'O'  
Option                   1 symbol 'Z'  
Password                6 symbols for access password

#### **Output data: n.a.**

**zfpdef:** [ResetOdometer\(Password\)](#)

### 2.5.28. Command: 58h / X – option O - Read Odometer function

**Input:** <Option['O']><;><Option['R']>

**output:** <Option['O']><;> <OdomResult[1..19]>

**FPR Operation:** Read odometer result.

*\*the command is valid for ADPOS model only.*

#### **Input data:**

Option                   1 symbol 'O'  
Option                   1 symbol 'R'

#### **Output data:**

Option                   1 symbol 'O'  
OdomResult            1..19 symbols for odometer result: used paper length in mm

**zfpdef:** [ReadOdometer\(\)](#)

## 2.5.29. Command: 4Fh / O – Reading the duplicates number of the invoice receipt

**input:** <Option1['D']> <;><Option2['R']>

**output:** <Option1['D']> <;><Option2['R']> <;> <DuplicatesNumber[1]>

**FPR operation:** Read the the number of the duplicates which can be printed after invoice receipt.

### **Input data :**

Option1 1 symbol with value 'D'

Option2 1 symbol with value 'R'

### **Output data:**

Option1 1 symbol with value 'D'

Option2 1 symbol with value 'R'

DuplicatesNumber 1 symbol for number of duplicates which can be print. Possible values from '0' to '5'.

**zfpdef:** ReadDuplicatesNumber()



## 2.6. RECEIPT OPERATIONS COMMANDS

These commands are used mainly for FPR sales registration. The group also includes some auxiliary commands providing information for the current receipt as well as commands for RA and PO amounts.

### 2.6.1. Command: 2Eh / . – Non-fiscal receipt opening

input: <Number[1..2]> <;> <Password[4]>

output: ACK

FPR operation: Opens a non-fiscal receipt assigned to the specified operator

**Input data :**

Number (Operator No) Symbol from 1 to 20 corresponding to operator's number

Password (Operator Password) 4 symbols for operator's password

**Output data: n. a.**

**zfpdef:** [OpenNonFiscReceipt\(Number, Password\)](#)

### 2.6.2. Command: 2Eh / . – Non-fiscal receipt opening with postponed printing

input: <Number[1..2]> <;> <Password[4]><;><Reserved['0']><;><Reserved['1']>

output: ACK

FPR operation: Opens a non-fiscal receipt assigned to the specified operator

**Input data :**

Number (Operator No) Symbol from 1 to 20 corresponding to operator's number

Password (Operator Password) 4 symbols for operator's password

Reserved 1 symbol reserved with value "0"

Reserved 1 symbol reserved with value "1"

**Output data: n. a.**

**zfpdef:** [OpenNonFiscReceiptPostponedPrn\(Number, Password\)](#)

### 2.6.3. Command: 2Fh / / – Non-fiscal receipt closure

input: n. a.

output: ACK

FPR operation: Closes the non-fiscal receipt.

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** [CloseNonFiscReceipt\(\)](#)

### 2.6.4. Command: 30h / 0 – Standard fiscal receipt opening

input: <OperNum[1..2]> <;> <OperPass[4]>

output: ACK

FPR operation: Opens a fiscal receipt assigned to the specified operator

**Input data :**

OperNum (Operator Number) Symbols from 1 to 20 corresponding to operator's number

OperPass (Operator Password) 4 symbols for operator's password

**Output data: n. a.**

**zfpdef:** [OpenFiscReceipt\(OperNum, OperPass\)](#)

### 2.6.5. Command: 30h / 0 – Standard fiscal receipt opening with postponed printing

input: <OperNum[1..2]> <;> <OperPass[4]> <;><Reserved['0']> <;> <Reserved['0']>  
<;><'2'>



**output: ACK**

**FPR operation:** Opens a fiscal receipt assigned to the specified operator with postponed printing.

**Input data :**

OperNum	(Operator Number) Symbols from 1 to 20 corresponding to operator's number
OperPass	(Operator Password) 4 symbols for operator's password
Reserved	1 symbol with value '0'
Reserved	1 symbol with value '0'
'2'	1 symbol with value '2' for postponed printing (all lines are printed after receipt close command)

**Output data: n. a.**

**Note:** After opened fiscal receipt, all the next commands will be executed but won't be printed immediately. The data for all receipt are stored to be printed up to AS sent information for receipt closure. If up to 5 sec timeout no command receipt closing the receipt will be canceled.

**zfpdef:** [OpenFiscReceiptPostponedPrn\(OperNum, OperPass\)](#)

## 2.6.6. Command: 30h / 0 – Standard fiscal receipt opening with buffered printing

**input:** <OperNum[1..2]> <;> <OperPass[4]> <;><Reserved['0']> <;> <Reserved['0']> <;><'4'>

**output: ACK**

**FPR operation:** Opens a fiscal receipt assigned to the specified operator with buffered printing

**Input data :**

OperNum	(Operator Number) Symbols from 1 to 20 corresponding to operator's number
OperPass	(Operator Password) 4 symbols for operator's password
Reserved	1 symbol with value '0'
Reserved	1 symbol with value '0'
'4'	1 symbol with value '4' for buffered printing

**Output data: n. a.**

**Note:**

After opens fiscal receipt, all the next commands will be executed but won't be printed immediately. The data is stored to be printed with a little delay after first 4 transactions are done.

**zfpdef:** [OpenFiscReceiptBufferedPrn\(OperNum, OperPass\)](#)

## 2.6.7. Command: 30h / 0 – Special Customer fiscal receipt document opening

**input:** <Number[1..2]> <;> <Password[4]><;> <Reserved['0']> <;> <Reserved['0']> <;><'1'> <;> <CustomerVATNo[15]> <;> <InvRecipient[30]> <;> <InvFree1[20]> <;> <InvFree2[20]> <;> <InvFree3[20]> <;> <InvFree4[20]> <;> <Address[30]>

**output: ACK**

**FPR operation:** Opens a special Customer fiscal receipt document assigned to the specified operator and Customer data.

**Input data :**

Number	(Operator No) Symbol from 1 to 20 corresponding to operator's number
Password	(Operator Password) 4 symbols for operator's password
Reserved	1 symbol with value '0'

<i>Reserved</i>	1 symbol with value '0'
<i>'1'</i>	1 symbol with value '1' for immediately printing step by step
<i>CustomerVATNo</i>	15 ASCII symbols text for Customer VAT number
<i>InvRecipient</i>	30 ASCII symbols for Recipient name
<i>InvFree1</i>	20 ASCII symbols for free text line
<i>InvFree2</i>	20 ASCII symbols for free text line
<i>InvFree3</i>	20 ASCII symbols for free text line
<i>InvFree4</i>	20 ASCII symbols for free text line
<i>Address</i>	30 ASCII symbols for customer address

**Output data: n. a.**

**zfpdef:** *OpenSpecialFiscDocument(Number, Password, CustomerVATNo, InvRecipient, InvFree1, InvFree2, InvFree3, InvFree4, InvAddr)*

## 2.6.8. Command: 30h / 0 – Special Customer fiscal receipt document opening with postponed printing

**input:** <OperNum[1..2]> <;> <OperPass[4]><;> <Reserved['0']> <;>  
 <Reserved['0']> <;> <'3'> <;> <CustomerVATNo[15]> <;> <InvRecipient[30]> <;>  
 <InvFree1[20]> <;> <InvFree2[20]> <;> <InvFree3[20]> <;> <InvFree4[20]> <;>  
 <Address[30]>

**output: ACK**

**FPR operation:** Opens a special Customer fiscal receipt document assigned to the specified operator and Customer data with postponed printing.

**Input data :**

<i>OperNum</i>	(Operator Number) Symbol from 1 to 20corresponding to operator's number
<i>OperPass</i>	(Operator Password) 4 symbols for operator's password
<i>Reserved</i>	1 symbol with value '0'
<i>Reserved</i>	1 symbol with value '0'
<i>'3'</i>	1 symbol with value '3' for postponed printing (all lines are printed after receipt close command)
<i>CustomerVATNo</i>	15 ASCII symbols text for Customer VAT number
<i>InvRecipient</i>	30 ASCII symbols for Recipient name
<i>InvFree1</i>	20 ASCII symbols for free text line
<i>InvFree2</i>	20 ASCII symbols for free text line
<i>InvFree3</i>	20 ASCII symbols for free text line
<i>InvFree4</i>	20 ASCII symbols for free text line
<i>Address</i>	(Address) 30 symbols for Address

**Output data: n. a.**

**Note:** After opened fiscal receipt, all the next commands will be executed but won't be printed immediately. The data for all receipt are stored to be printed up to AS sent information for receipt closure. If up to 5 sec timeout no command receipt closing the receipt will be canceled.

**zfpdef:** *OpenSpecialFiscDocumentPostponedPrn(OperNum, OperPass, CustomerVATNo, InvRecipient, InvFree1, InvFree2, InvFree3, InvFree4, Address)*

### 2.6.9. Command: 30h / 0 – Special Customer fiscal receipt document opening with buffered printing

**input:** <OperNum[1..2]> <;> <OperPass[4]><;> <Reserved['0']> <;>  
<Reserved['0']> <;><'5'> <;> <CustomerVATNo[15]> <;> <InvRecipient[30]> <;>  
<InvFree1[20]> <;> <InvFree2[20]> <;> <InvFree3[20]> <;> <InvFree4[20]> <;>  
<Address[30]>

**output: ACK**

**FPR operation:** Opens a special Customer fiscal receipt document assigned to the specified operator and Customer data with buffered printing.

**Input data :**

OperNum	(Operator No) Symbol from 1 to 20 corresponding to operator's number
OperPass	(Operator Password) 4 symbols for operator's password
Reserved	1 symbol with value '0'
Reserved	1 symbol with value '0'
'5'	1 symbol with value '5' for buffered printing
CustomerVATNo	15 ASCII symbols text for Customer VAT number
InvRecipient	30 ASCII symbols for Recipient name
InvFree1	20 ASCII symbols for free text line
InvFree2	20 ASCII symbols for free text line
InvFree3	20 ASCII symbols for free text line
InvFree4	20 ASCII symbols for free text line
Address	(Address) 30 symbols for Address

**Output data: n. a.**

**Note:** After opens fiscal receipt, all the next commands will be executed but won't be printed immediately. The data is stored to be printed with a little delay after first 4 transactions are done.

**zfpdef:** *OpenSpecialFiscDocumentBufferedPrn(OperNum, OperPass, CustomerVATNo, InvRecipient, InvFree1, InvFree2, InvFree3, InvFree4, Address)*

### 2.6.10. Command: 31h / 1 – Sale/correction of article with VAT class definition

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]>  
{<'\*> <Quantity[1..10]>} {<','> <DiscAddP[2..7]>} {<':'> <DiscAddV[2..8]>}  
{<'@> <DiscNamed[2..8]>} {<'+'> <Category[1..7]>} {<'!'> <NamePLUextension[12]>}  
{<';> <AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

**Input data :**

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: <ul style="list-style-type: none"><li>- 'A' – VAT class A</li><li>- 'B' – VAT class B</li><li>- 'C' – VAT class C</li><li>- 'D' – VAT class D</li><li>- 'E' – VAT class E</li><li>- 'F' – Alte taxe</li></ul>
Price	1 to 10 symbols for article's price

'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity) 1 to 10 symbols for quantity
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name
<b>Output data : n. a.</b>	

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** SaleOrCorrectionPLUwithVATdefinition\_(NamePLU, OptionVATClass, Price, Quantity, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

## 2.6.11. Command: 31h / 1 – Sale/correction of article belonging to departament

**input:** <NamePLU[36]> <;> <Reserved[' ']> <;> <Price[1..10]>  
 {<'\*> <Quantity[1..10]>} {<'&> <DepNo[1..2]>} {<'> <DiscAddP[2..7]>}  
 {<'> <DiscAddV[2..8]>} {<'@> <DiscNamed[2..8]>} {<'> <Category[1..7]>}  
 {<'!> <NamePLUextension[12]>} {<;> <AdditionalNamePLU[108]> }

**output:** ACK

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name) 36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
Reserved	1 symbol with value "space"
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity) 1 to 10 symbols for quantity
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; <i>example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h</i>
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount

'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name
<b>Output data : n. a.</b>	

#### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** SaleOrCorrectionPLUBelongingToDEP\_1 (NamePLU, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

### 2.6.12. Command: 31h / 1 – Sale/correction of article with specified VAT belonging to department

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]>  
 {<'\*> <Quantity[1..10]> }> {<'&><DepNo[1..2]>} {<'> <DiscAddP[2..7]>}  
 {<'><DiscAddV[2..8]>} {<'@><DiscNamed[2..8]>} {<'><Category[1..7]>}  
 {<'!><NamePLUextension[12]>} {<;><AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

#### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)1 to 10 symbols for quantity
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; <i>example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h</i>
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %.) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only

*AdditionalNamePLU* (Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

**Notes:**

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** *SaleOrCorrectionPLUwithVATdefinitionBelongingToDEP\_1* (NamePLU, OptionVATClass, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

### 2.6.13. Command: 32h / 2 – sale/correction of article from FPR database

**input:** <OptionSign[1]> <NoPLU[5]> {<'\*> <Quantity[1..10]>} {<','> <DiscAddP[2..7]>} {<':'><DiscAddV[2..8]>} {<'@><DiscNamed[2..8]>}

**output:** ACK

**FPR operation:** Registers the sale or correction of a specified quantity of an article of the internal database of the FPR.

**Input data :**

<i>OptionSign</i>	(Sale/Correction)1 symbol with optional value: - '+' - Sale - '-' - Correction
<i>NoPLU</i>	(PLU No)5 symbols for number of article of FPR's db in format: #####
<i>'*</i>	1 symbol '*' indicating the presence of quantity field
<i>Quantity</i>	(Quantity)1 to 10 symbols for article's quantity sold
<i>','</i>	1 symbol ',' indicating the presence of discount/addition field
<i>DiscAddP</i>	(Discount/Addition %)2 to 7 for percentage of discount/addition
<i>':'</i>	1 symbol ':' indicating the presence of value discount/addition field
<i>DiscAddV</i>	(Discount/Addition Value)2 to 8 symbols for value of discount/addition
<i>'@'</i>	1 symbol '@' indicating the presence value of named discount
<i>DiscNamed</i>	(Named Discount) 2 to 8 symbols for value of named discount

**Output data : n. a.**

**Notes:**

The FPR will perform a correction operation only if the same quantity of the article has already been sold.

**If the selected article has programmed department attachment, the FPR will execute the command only if the article and the certain department are in same VAT class.**

**zfpdef:** *SaleOrCorrectionFPRArticle(OptionSign, NoPLU, Price, Quantity, DiscAddP, DiscAddV, DiscNamed)*

### 2.6.14. Command: 33h / 3 – Subtotal

**input:** <OptionPrint[1]> <;> <OptionDisplay[1]> {<':'> <DiscAddV[1..10]>} {<','> <DiscAddP[2..7]>}

**output:** <SubtotalValue[1..10]>

**FPR operation:** Calculates the subtotal amount with printing and display visualization options. Provides information about values of the calculated amounts. If a percent or value discount/addition has been specified the subtotal and the discount/addition value will be printed regardless the parameter for printing.



**Input data :**

<i>OptionPrint</i>	(Print)1 symbol with value: - '1' – Yes - '0' – No
<i>OptionDisplay</i>	(Display)1 symbol with value: - '1' – Yes - '0' – No
<i>':'</i>	1 symbol ':' indicating the presence of value discount/addition field
<i>DiscAddV</i>	(Discount/Addition Value)1 to 10 symbols for the value of the discount/addition
<i>','</i>	1 symbol ',' indicating the presence of percent discount/addition field
<i>DiscAddP</i>	(Discount/Addition %)2 to 7 symbols for the percentage value of the discount/addition

**Output data:**

<i>SubtotalValue</i>	1..10 symbols for the value of the subtotal amount
----------------------	--

**Notes:**

The discount/addition may be either values or percentages.

When the discount/addition is a percentage the amount is distributed proportionally over the turnover items and is automatically transferred to the turnovers of the corresponding VAT classes.

A value discount/addition may be specified only if all sales are of articles (items) belonging to one and the same VAT class.

**zfpdef:** *Subtotal*(*OptionPrint*, *OptionDisplay*, *DiscAddV*, *DiscAddP*)

## 2.6.15. Command: 33h / 3 – Subtotal with value discount with specified VAT definition

**input:** <*OptionPrint*[1]><;><*OptionDisplay*[1]>{<':'><*DiscAddV*[1..10]>}{<;><*OptionVATClass*[1]>}

**output:** <*SubtotalValue*[1..10]>

**FPR operation:** Calculates the subtotal amount with printing and display visualization options. Provides information about values of the calculated amounts. If a percent or value discount/addition has been specified the subtotal and the discount/addition value will be printed regardless the parameter for printing.

**Input data :**

<i>OptionPrint</i>	(Print)1 symbol with value: - '1' – Yes - '0' – No
<i>OptionDisplay</i>	(Display)1 symbol with value: - '1' – Yes - '0' – No
<i>':'</i>	1 symbol ':' indicating the presence of value discount/addition field
<i>DiscAddV</i>	(Discount/Addition Value)1 to 10 symbols for the value of the discount/addition
<i>OptionVATClass</i>	1 symbol for article's VAT class with optional values:" - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe

**Output data:**

<i>SubtotalValue</i>	1..10 symbols for the value of the subtotal amount
----------------------	--

**Notes:**

The discount/addition may be either values or percentages.

When the discount/addition is a percentage the amount is distributed proportionally over the turnover items and is automatically transferred to the turnovers of the corresponding VAT classes.

A value discount/addition may be specified only if all sales are of articles (items) belonging to one and the same VAT class.

***zfpdef:**SubtotalWithSpecifiedVAT\_(OptionPrint, OptionDisplay, DiscAddV, OptionVATClass)*



## 2.6.16. Command: 34h / 4 – Sale/correction of article with department definition

**input:** <NamePLU[36]> <;> <DepNo[1..2]> <;> <Price[1..10]> {<'\*>  
<Quantity[1..10]>} {<'&> <OptionVATClass[1]>} {<'> <DiscAddP[2..7]>}  
{<'> <DiscAddV[2..8]>} {<'> <Category[1..7]>} {<'@> <DiscNamed[2..8]>} {<'!>  
<NamePLUextension[12]>} {<;> <AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction. The VAT Class field is not obligatory. If it is not present the sale is associated to VAT Class of department belonging.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; <i>example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h</i>
Price	1 to 10 symbols for article's price
'*	1 symbol '*' indicating the presence of quantity field
Quantity	1 to 10 symbols for quantity
'&	1 symbol '&' indicating the VAT class field
OptionVATClass	1 symbol for article's VAT class with optional values:" - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition Percentage) 2 to 7 symbols for percentage of discount/addition
':'	1 symbol ':' indicating the of value discount/addition field
DiscAddV	(Discount/Addition Value) 2..8 for value of discount/addition
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	1..7 symbols for PLU Category code in format #####.##
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

### Output data : n. a.

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** SaleOrCorrectionPLUwithDEPdefinitionBelongingToVAT\_4(NamePLU, DepNo, Price, Quantity, OptionVATClass, DiscAddP, DiscAddV, Category, DiscNamed, NamePLUextension, AdditionalNamePLU)

## 2.6.17. Command: 34h / 4 – Sale/correction of article with specified department belonging to VAT class

**input:** <NamePLU[36]> <;> <DepNo[1..2]> <;> <Price[1..10]> {<'\*>  
<Quantity[1..10]>} {<','> <DiscAddP[2..7]>} {<':'><DiscAddV[2..8]>} {<'+><Category[1..7]>}  
{<'@><DiscNamed[2..8]>} {<'!'> <NamePLUextension[12]>} {<;>  
<AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction. The VAT Class field is not obligatory. If it is not present the sale is associated to VAT Class of department belonging.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; <i>example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h</i>
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	1 to 10 symbols for quantity
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition Percentage) 2 to 7 symbols for percentage of discount/addition
':'	1 symbol ':' indicating the of value discount/addition field
DiscAddV	(Discount/Addition Value) 2..8 for value of discount/addition
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	1..7 symbols for PLU Category code in format #####.##
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

### Output data : n. a.

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** [SaleOrCorrectionPLUwithDEPdefenition\\_4\(NamePLU, DepNo, Price, Quantity, DiscAddP, DiscAddV, Category, DiscNamed, NamePLUextension, AdditionalNamePLU\)](#)

## 2.6.18. Command: 35h / 5 – Payment

**input:** <OptionPaymentType[1]> <;> <reserved['0']> <;> <Amount[1..10]>  
<;><reserved['1']>

**output: ACK**

**FPR operation:** Registers the payment in the receipt with specified type of payment and amount received (if the payment type is 1-9 the amount of change due is not obligatory.)

### **Input data :**

*OptionPaymentType* (Payment Type Options)1 symbol for payment type:

- '0' – Payment 0
- '1' – Payment 1
- '2' – Payment 2
- '3' – Payment 3
- '4' – Payment 4
- '5' – Payment 5
- '6' – Payment 6
- '7' – Payment 7
- '8' – Payment 8
- '9' – Payment 9

*reserved* 1 symbol with value '0'

*Amount* 1 to 10 characters for received amount

*reserved* 1 symbols with value '1'

**Output data:** n.a.

### **Notes:**

By executing this command the FPR enters the payment mode. No further sales and/or corrections are allowed.

The receipt can be finalized only when the last payment transfer is sufficient to cover the whole amount due (the grand total amount), i.e. the payment procedure has been finalized.

**zfpdef:** *Payment*(*OptionPaymentType*, *Amount*)

### 2.6.19. Command: 35h / 5 – Pay exact sum

**input:** <OptionPaymentType[1]> <;> <reserved['0']> <;> <Amount[""]>  
<;><reserved['1']>

**output:** ACK

**FPR operation:** Registers the payment in the receipt with specified type of payment and exact amount (if the payment type is 1-9 the amount of change due is not obligatory.)

**Input data :**

OptionPaymentType (Payment Type) 1 symbol for payment type:

- '0' – Payment 0
- '1' – Payment 1
- '2' – Payment 2
- '3' – Payment 3
- '4' – Payment 4
- '5' – Payment 5
- '6' – Payment 6
- '7' – Payment 7
- '8' – Payment 8
- '9' – Payment 9

reserved 1 symbol with value '0'

Amount 1 symbol ' ' - /quotation mark/ for pay with exact sum

reserved 1 symbol with value '1'

**Output data:** n.a.

**zfpdef:** *PayExactSum*(OptionPaymentType)

### 2.6.20. Command: 36h / 6 – Automatic receipt closure

**input:** n. a.

**output:** ACK

**FPR operation:** Close the fiscal receipt, paying the due amount in cash

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** *CashPayCloseReceipt*()

### 2.6.21. Command: 37h / 7 – Free text printing

**input:** <Text[TextLength]>

**output:** ACK

**FPR operation:** Prints a free text.

**Input data :**

Text Free text - TextLength symbols

**Output data:** ACK

**zfpdef:** *PrintText*(Text)

## 2.6.22. Command: 38h / 8 – Fiscal receipt closure

input: n. a.

output: ACK

FPR operation: Closes the opened fiscal receipt.

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** *CloseFiscReceipt()*

## 2.6.23. Command: 39h / 9 – Fiscal receipt cancel

input: n. a.

output: ACK

FPR operation: Cancel the opened fiscal receipt.

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** *CancelFiscReceipt()*

## 2.6.24. Command: 3Ah / : – Print a copy of the last document

input: n. a.

output: ACK

FPR operation: Print a copy of the last receipt document issued

**Input data : n. a.**

**Output data : n. a.**

**zfpdef:** *PrintDuplicate()*

## 2.6.25. Command: 3Bh / ; – Non-fiscal RA and PO amounts

input: <Number[1..2]> <;> <Password[4]> <;> <OptionPayment['0']> <;>  
<Amount[1..10]>{<'@> <Text[TextLength-2]>}

output: ACK

FPR operation: Lodges/withdraws the stated amount in the specified type of payment from the registers of the specified operator (the '-' symbol preceding the amount means withdrawal).

### **Input data :**

Number	(Operator No) Symbol from 1 to 20 corresponding to the operator's number
Password	(Operator Password) 4 symbols for operator's password
OptionPayment	1 symbol with value 0
Amount	1 to 10 symbols for the amount lodged/withdrawn
'@'	Symbol @
Text	Text - TextLength-2 symbols length

**Output data: n. a.**

**zfpdef:** *RA\_PO(Number, Password, OptionPaymentType, Amount, Text)*

## 2.6.26 Command: 3Ch / < – Storno function of article with VAT class definition

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]> {<'> <Quantity[1..10]> }>{<','> <DiscAddP[2..7]> } {<':'><DiscAddV[2..8]>} {<'@><DiscNamed[2..8]>} {<'+'><Category[1..7]>} {<'!'><NamePLUextension[12]>} {<';><AdditionalNamePLU[108]> }

**output:** ACK

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)1 to 10 symbols for quantity
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the ':' symbol.

**zfpdef:** StornoPLUwithVATdefenition\_(NamePLU, OptionVATClass, Price, Quantity, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

## 2.6.27. Command: 3Ch / < – Storno function of article belonging to departament

**input:** <NamePLU[36]> <;> <Reserved[' ']> <;> <Price[1..10]>  
 {<'\*> <Quantity[1..10]>} ]>}{<'&><DepNo[1..2]>}{<','> <DiscAddP[2..7]>}  
 {<':'><DiscAddV[2..8]>} {<'@><DiscNamed[2..8]>} {<'+'><Category[1..7]>}  
 {<'!'><NamePLUextension[12]>} {<;><AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
Reserved	1 symbol with value "space"
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)1 to 10 symbols for quantity
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; <i>example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h</i>
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** StornoPLUBelongingToDEP\_1(NamePLU, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)



## 2.6.28 Command: 3Ch / < – Storno function of article with specified VAT belonging to departament

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]> {<'> <Quantity[1..10]> }> {<'&'><DepNo[1..2]>} {<'> <DiscAddP[2..7]>} {<'> <DiscAddV[2..8]>} {<'@> <DiscNamed[2..8]>} {<'> <Category[1..7]>} {<'!'> <NamePLUextension[12]>} {<;> <AdditionalNamePLU[108]> }

**output:** ACK

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)1 to 10 symbols for quantity
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h
' '	1 symbol ' ' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
'.'	1 symbol '.' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** StornoPLUwithVATdefinitionBelongingToDEP\_1(NamePLU, OptionVATClass, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)



## 2.6.29. Command: 3Dh / '=' – Sell / Correction of article from with fractional quantity belonging to VAT class definition

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]> {<'\*> <Quantity[10]> }>{<','> <DiscAddP[2..7]> } {<':'><DiscAddV[2..8]> } {<'@><DiscNamed[2..8]> } {<'+'><Category[1..7]> } {<'!'><NamePLUextension[12]> } {<';><AdditionalNamePLU[108]> }

**output:** ACK

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### **Input data :**

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)3 to 10 symbols for quantity in format fractional format (e.g. 1/3)
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
':'	1 symbol ':' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### **Notes:**

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the ':' symbol.

**zfpdef:** [SaleOrCorrectionFractionalQuantityPLUwithVATdefenition\\_\(NamePLU, OptionVATClass, Price, Quantity, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU\)](#)

## 2.6.30. Command: 3Dh / '=' – Sell / Correction of article with fractional quantity belonging to departament

**input:** <NamePLU[36]> <;> <Reserved[' ']> <;> <Price[1..10]>  
 {<'\*> <Quantity[10]>} ]> } {<'&'> <DepNo[1..2]>} {<'> <DiscAddP[2..7]>}  
 {<'> <DiscAddV[2..8]>} {<'@'> <DiscNamed[2..8]>} {<'+'> <Category[1..7]>}  
 {<'!'> <NamePLUextension[12]>} {<;> <AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
Reserved	1 symbol with value "space"
Price	1 to 10 symbols for article's price
'*'	1 symbol '*' indicating the presence of quantity field
Quantity	(Quantity)3 to 10 symbols for quantity in format fractional format (e.g. 1/3)
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h
','	1 symbol ',' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
'.'	1 symbol '.' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'+'	1 symbol '+' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format ####.##
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '-' symbol.

**zfpdef:** SaleOrCorrectionFractionalQuantityPLUBelongingToDEP\_1(NamePLU, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

## 2.6.31. Command: 3Dh / '=' – Sell / Correction of article with fractional quantity with specified VAT belonging to departament

**input:** <NamePLU[36]> <;> <OptionVATClass[1]> <;> <Price[1..10]> {<'> <Quantity[10]> }> {<'&'><DepNo[1..2]>} {<'> <DiscAddP[2..7]>} {<'> <DiscAddV[2..8]>} {<'@><DiscNamed[2..8]>} {<'> <Category[1..7]>} {<'!'><NamePLUextension[12]>} {<;><AdditionalNamePLU[108]> }

**output: ACK**

**FPR operation:** Registers the sale or correction of article with specified name, price, quantity, VAT class and/or discount/addition on the transaction.

### Input data :

NamePLU	(PLU Name)36 symbols for article's name included separator for MU=80h or 60h followed up to 3 symbols for unit
OptionVATClass	1 symbol for article's VAT class with optional values: - 'A' – VAT class A - 'B' – VAT class B - 'C' – VAT class C - 'D' – VAT class D - 'E' – VAT class E - 'F' – Alte taxe
Price	1 to 10 symbols for article's price
'&'	1 symbol '&' indicating the presence of quantity field
Quantity	(Quantity)3 to 10 symbols for quantity in format fractional format (e.g. 1/3)
'&'	1 symbol '&' indicating the presence of departament
DepNo	DepNo + 80h (Department No) 1 symbol for article department attachment, formed in the following manner; example: Dep01 = 81h, Dep02 = 82h ... Dep19 = 93h
'>'	1 symbol '>' indicating the presence of discount/addition field
DiscAddP	(Discount/Addition %) 2 to 7 for percentage of discount/addition
'>'	1 symbol '>' indicating the presence of value discount/addition field
DiscAddV	(Discount/Addition Value) 2 to 8 for value of discount/addition
'@'	1 symbol '@' indicating the presence value of named discount
DiscNamed	(Named Discount) 2 to 8 symbols for value of named discount
'>'	1 symbol '>' indicating the presence of value PLU Category code
Category	Up to 7 symbols for PLU Category code in format #####.##
'!'	1 symbol with value '!'
NamePLUextension	(PLU Name Extension) 12 symbols for extension of the PLU Name: FP Only
AdditionalNamePLU	(Additional PLU name) 108 symbols for additional PLU name

**Output data : n. a.**

### Notes:

If the price field is preceded by a '-' the command is executed by the FPR as a correction/void (only if the amount of the corresponding VAT class of the receipt is sufficient).

The quantity fields are not obligatory. If no value is stated for them the FPR executed the command for a default quantity of 1.000

The discount/addition fields are not obligatory. The discount/addition must be in percents and is determined from the presence or absence of the '>' symbol.

**zfpdef:** SaleOrCorrectionFractionalQuantityPLUwithVATdefinitionBelongingToDEP\_1

(NamePLU, OptionVATClass, Price, Quantity, DepNo, DiscAddP, DiscAddV, DiscNamed, Category, NamePLUextension, AdditionalNamePLU)

### 2.6.32 Command: 3Eh / > – Discount/ addition – FP Only

**input:** <OptionType[1]><;><OptionDisplay[1]>{<':'><DiscAddV[1..10]>}  
{<','><DiscAddP[2..7]>}

**output:** <DiscAddValue[1..10]>

**FPR operation:** Percent or value discount/addition over sum of transaction or over subtotal sum depended from byte “type”.

**Input data :**

*OptionType* 1 symbol with value  
- '2' - Defined from the device  
- '1' - Over subtotal  
- '0' - Over transaction sum

*OptionDisplay* (Display) 1 symbol with value:  
- '1' – Yes  
- '0' – No

*':'* 1 symbol ':' indicating the presence of value discount/addition field

*DiscAddV* (Discount/Addition Value) 1 to 10 symbols for the value of the discount/addition.  
Use minus sign '-' for discount

*','* 1 symbol ',' indicating the presence of percent discount/addition field

*DiscAddP* (Discount/Addition %) 2 to 7 symbols for the percentage value of the discount/addition. Use minus sign '-' for discount

**Output data :**

*DiscAddValue* 1 to 10 symbols for the value of the applied discount/addition.

**zfpdef:** DiscountAddition(*Option Type*, *OptionDisplay*, *DiscAddV*, *DiscAddP*)

## 2.7. COMMANDS FOR READING THE DATA IN FPR'S REGISTERS

This set of commands provides information about the status of FPR's registers without causing a device activity, i.e. the information is obtained through the communication interface without printing or display visualization.

### 2.7.1. Command: 6Dh / m – Reading of amounts by VAT Groups

**input:** n. a.

**output:** <AmntTaxGrA[1..11]> <;> <AmntTaxGrB[1..11]> <;> <AmntTaxGrC[1..11]> <;> <AmntTaxGrD[1..11]> <;> <AmntTaxGrE[1..11]> <;> <AmntTaxGrF[1..11]>

**FPR operation:** Provides information about the accumulated amount by VAT class.

**Input data : n. a.**

**Output data :**

*AmntTaxGrA* 1..11 symbols for the amount accumulated in the VAT group A

*AmntTaxGrB* 1..11 symbols for the amount accumulated in the VAT group B

*AmntTaxGrC* 1..11 symbols for the amount accumulated in the VAT group C

*AmntTaxGrD* 1..11 symbols for the amount accumulated in the VAT group D

*AmntTaxGrE* 1..11 symbols for the amount accumulated in the VAT group E

*AmntTaxGrF* 1..11 symbols for the amount accumulated in the VAT group F

**zfpdef:** [ReadVATClassAmounts\(\)](#)

### 2.7.2. Command: 6Eh / n – Reading of registers – 0 (on hand)

**input:** <'0'>

**output:** <'0'> <;> <AmntPmnt0[1..11]> <;> <AmntPmnt1[1..11]> <;> <AmntPmnt2[1..11]> <;> <AmntPmnt3[1..11]> <;> <AmntPmnt4[1..11]> <;> <AmntPmnt5[1..11]> <;> <AmntPmnt6[1..11]> <;> <AmntPmnt7[1..11]> <;> <AmntPmnt8[1..11]> <;> <AmntPmnt9[1..11]>

**FPR operation:** Provides information about the amounts on hand by type of payment.

**Input data :**

<'0'> 1 symbol obligatory '0'

**Output data:**

<'0'> 1 symbol obligatory '0'

*AmntPmnt0* 1..11 symbols for the accumulated amount by payment type 0

*AmntPmnt1* 1..11 symbols for the accumulated amount by payment type 1

*AmntPmnt2* 1..11 symbols for the accumulated amount by payment type 2

*AmntPmnt3* 1..11 symbols for the accumulated amount by payment type 3

*AmntPmnt4* 1..11 symbols for the accumulated amount by payment type 4

*AmntPmnt5* 1..11 symbols for the accumulated amount by payment type 5

*AmntPmnt6* 1..11 symbols for the accumulated amount by payment type 6

*AmntPmnt7* 1..11 symbols for the accumulated amount by payment type 7

*AmntPmnt8* 1..11 symbols for the accumulated amount by payment type 8

*AmntPmnt9* 1..11 symbols for the accumulated amount by payment type 9

**zfpdef:** [ReadVatGrAmounts\(\)](#)

### 2.7.3. Command: 6Eh / n – Reading of registers – 1 (general)

**input:** <'1'>

**output:** <'1'> <;> <NoCust[1..5]> <;> <NoDisc[1..5]> <;> <AmntDisc[1..11]> <;>  
<NoAdd[1..5]> <;> <AmntAdd[1..11]> <;> <NoVoid[1..5]> <;> <AmntVoid[1..11]>

**FPR operation:** Provides information about the number of customers (number of fiscal receipt issued), number of discounts, additions and corrections made and the accumulated amounts.

#### **Input data :**

<'1'> 1 symbol obligatory '1'

#### **Output data:**

<'1'> 1 symbol obligatory '1'

NoCust 1..5 symbols for number of customers

NoDisc 1..5 symbols for number of discounts

AmntDisc 1..11 symbols for accumulated amount of discounts

NoAdd 1..5 symbols for number of additions

AmntAdd 1..11 symbols for accumulated amount of additions

NoVoid 1..5 symbols for number of corrections

AmntVoid 1..11 symbols for accumulated amount of corrections

[\*zfpdef:ReadGeneralDaily\(\)\*](#)

### 2.7.4. Command: 6Eh / n – Reading of registers – 2 (RA)

**input:** <'2'>

**output:** <'2'> <;> <AmntPmnt0[1..11]> <;> <AmntPmnt1[1..11]> <;>  
<AmntPmnt2[1..11]> <;> <AmntPmnt3[1..11]> <;> <AmntPmnt4[1..11]> <;>  
<AmntPmnt5[1..11]> <;> <AmntPmnt6[1..11]> <;> <AmntPmnt7[1..11]> <;>  
<AmntPmnt8[1..11]> <;> <AmntPmnt9[1..11]> <;> <NoRA[1..5]> <;>

**FPR operation:** Provides information about the RA amounts by type of payment and the total number of operations.

#### **Input data :**

<'2'> 1 symbol obligatory '2'

#### **Output data:**

<'2'> 1 symbol obligatory '2'

AmntPmnt0 1..11 symbols for the accumulated amount by payment type 0

AmntPmnt1 1..11 symbols for the accumulated amount by payment type 1

AmntPmnt2 1..11 symbols for the accumulated amount by payment type 2

AmntPmnt3 1..11 symbols for the accumulated amount by payment type 3

AmntPmnt4 1..11 symbols for the accumulated amount by payment type 4

AmntPmnt5 1..11 symbols for the accumulated amount by payment type 5

AmntPmnt6 1..11 symbols for the accumulated amount by payment type 6

AmntPmnt7 1..11 symbols for the accumulated amount by payment type 7

AmntPmnt8 1..11 symbols for the accumulated amount by payment type 8

AmntPmnt9 1..11 symbols for the accumulated amount by payment type 9

NoRA 1..5 symbols for the total number of operations

[\*zfpdef: ReadDailyRA\(\)\*](#)



## 2.7.5. Command: 6Eh / n – Reading of registers – 3 (PO)

**input:** <'3'>

**output:** <'3'> <;> <AmntPmnt0[1..11]> <;> <AmntPmnt1[1..11]> <;>  
<AmntPmnt2[1..11]> <;> <AmntPmnt3[1..11]> <;> <AmntPmnt4[1..11]> <;>  
<AmntPmnt5[1..11]> <;> <AmntPmnt6[1..11]> <;> <AmntPmnt7[1..11]> <;>  
<AmntPmnt8[1..11]> <;> <AmntPmnt9[1..11]> <;> <NoPO[1..5]> <;>

**FPR operation:** Provides information about the PO amounts by type of payment and the total number of operations.

### **Input data :**

<'3'> 1 symbol obligatory '3'

### **Output data:**

<'3'> 1 symbol obligatory '3'

AmntPmnt0 1..11 symbols for the accumulated amount by payment type 0

AmntPmnt1 1..11 symbols for the accumulated amount by payment type 1

AmntPmnt2 1..11 symbols for the accumulated amount by payment type 2

AmntPmnt3 1..11 symbols for the accumulated amount by payment type 3

AmntPmnt4 1..11 symbols for the accumulated amount by payment type 4

AmntPmnt5 1..11 symbols for the accumulated amount by payment type 5

AmntPmnt6 1..11 symbols for the accumulated amount by payment type 6

AmntPmnt7 1..11 symbols for the accumulated amount by payment type 7

AmntPmnt8 1..11 symbols for the accumulated amount by payment type 8

AmntPmnt9 1..11 symbols for the accumulated amount by payment type 9

NoPO 1..5 symbols for the total number of operations

**zfpdef:** [ReadDailyPO\(\)](#)

## 2.7.6. Command: 6Eh / n – Reading of registers – 4 (received)

**input:** <'4'>

**output:** <'4'> <;> <AmntPmnt0[1..11]> <;> <AmntPmnt1[1..11]> <;>  
<AmntPmnt2[1..11]> <;> <AmntPmnt3[1..11]> <;> <AmntPmnt4[1..11]> <;>  
<AmntPmnt5[1..11]> <;> <AmntPmnt6[1..11]> <;> <AmntPmnt7[1..11]> <;>  
<AmntPmnt8[1..11]> <;> <AmntPmnt9[1..11]> <;>

**FPR operation:** Provides information about the amounts received from sales by type of payment.

### **Input data :**

<'4'> 1 symbol obligatory '4'

### **Output data:**

<'4'> 1 symbol obligatory '4'

AmntPmnt0 1..11 symbols for the accumulated amount by payment type 0

AmntPmnt1 1..11 symbols for the accumulated amount by payment type 1

AmntPmnt2 1..11 symbols for the accumulated amount by payment type 2

AmntPmnt3 1..11 symbols for the accumulated amount by payment type 3

AmntPmnt4 1..11 symbols for the accumulated amount by payment type 4

AmntPmnt5 1..11 symbols for the accumulated amount by payment type 5

AmntPmnt6 1..11 symbols for the accumulated amount by payment type 6

AmntPmnt7 1..11 symbols for the accumulated amount by payment type 7

AmntPmnt8 1..11 symbols for the accumulated amount by payment type 8

AmntPmnt9 1..11 symbols for the accumulated amount by payment type 9

**zfpdef:** [ReadDailyRecAmounts\(\)](#)

### 2.7.7. Command: 6Eh / n – Reading of registers – 5 (counters)

**input:** <'5'>

**output:** <'5'> <;> <NoRep[1..5]> <;> <NoLastFMBlock[1..5]> <;> <NoEJ[1..5]> <;>  
<DateTime "DD-MM-YYYY HH:MM">

**FPR operation:** Provides information about the current reading of the daily-report-with-zeroing counter, the number of the last block stored in FM, the number of EJ and the date and time of the last block storage in the FM.

#### **Input data :**

<'5'> 1 symbol obligatory '5'

#### **Output data:**

<'5'> 1 symbol obligatory '5'

NoRep Up to 5 symbols for number of the last report from reset

NoLastFMBlock Up to 5 symbols for number of the last FM report

NoEJ Up to 5 symbols for number of EJ

DateTime 16 symbols for date and time of the last block storage in FM

**zfpdef:** [ReadDailyCounters\(\)](#)

### 2.7.8. Command: 6Fh / o – Reading of operator's report – '1' (general)

**input:** <'1'><;><OpNo[1..2]>

**output:** <'1'><;><OpNo[1..2]><;><NoCustomers[1..5]><;><NoDiscounts[1..5]><;>  
<AmountDiscounts[1..11]><;><NoAdditions[1..5]><;><AmountAdditions[1..11]><;>  
<NoVoids[1..5]> <;><AmountVoids[1..11]><;>

**FPR operation:** Read the total number of customers, discounts, additions, corrections and accumulated amounts by specified operator.

#### **Input data :**

'1' 1 symbol obligatory '1'

OpNo (Operator Number) Symbol from 1 to 20 in format corresponding to operator's number

#### **Output data:**

'1' 1 symbol obligatory '1'

OpNo Symbol from 1 to 20 corresponding to operator's number

NoCustomers Up to 5 symbols for number of customers

NoDiscounts Up to 5 symbols for number of discounts

AmountDiscounts Up to 11 symbols for accumulated amount of discounts

NoAdditions Up to 5 symbols for number of additions

AmountAdditions Up to 11 symbols for accumulated amount of additions

NoVoids Up to 5 symbols for number of corrections

AmountVoids Up to 11 symbols for accumulated amount of corrections

**zfpdef:** [ReadGeneralOper\(OpNo\)](#)



## 2.7.9. Command: 6Fh / o – Reading of operator's report – 2 (RA)

**input:** <'2'> <;> <OpNo[1..2]>

**output:** <'2'> <;> <OpNo[1..2]> <;> <AmountPayment0[1..11]> <;>

<AmountPayment1[1..11]> <;> <AmountPayment2[1..11]> <;> <AmountPayment3[1..11]> <;>  
<AmountPayment4[1..11]> <;> <AmountPayment5[1..11]> <;> <AmountPayment6[1..11]> <;>  
<AmountPayment7[1..11]> <;> <AmountPayment8[1..11]> <;> <AmountPayment9[1..11]> <;>  
<NoRA[1..5]> <;>

**FPR operation:** Provides information about the RA by type of payment as well as the total number of operations by specified operator.

### **Input data :**

<'2'> 1 symbol obligatory '2'  
OpNo (Operator No) Symbol from 1 to 20 corresponding to operator's number

### **Output data:**

<'2'> 1 symbol obligatory '2'  
OpNo Symbol from 1 to 20 corresponding to operator's number  
AmountPayment0 1..11 symbols for the RA by type of payment 0  
AmountPayment1 1..11 symbols for the RA by type of payment 1  
AmountPayment2 1..11 symbols for the RA by type of payment 2  
AmountPayment3 1..11 symbols for the RA by type of payment 3  
AmountPayment4 1..11 symbols for the RA by type of payment 4  
AmountPayment5 1..11 symbols for the RA by type of payment 5  
AmountPayment6 1..11 symbols for the RA by type of payment 6  
AmountPayment7 1..11 symbols for the RA by type of payment 7  
AmountPayment8 1..11 symbols for the RA by type of payment 8  
AmountPayment9 1..11 symbols for the RA by type of payment 9  
NoRA 1..5 symbols for the total number of operations

**zfpdef:** ReadOperRA(OpNo)

## 2.7.10. Command: 6Fh / o – Reading of operator's report – 3 (PO)

**input:** <'3'> <;> <OpNo[1..2]>

**output:** <'3'> <;> <OpNo[1..2]> <;> <AmountPayment0[1..11]> <;>

<AmountPayment1[1..11]> <;> <AmountPayment2[1..11]> <;> <AmountPayment3[1..11]> <;>  
<AmountPayment4[1..11]> <;> <AmountPayment5[1..11]> <;> <AmountPayment6[1..11]> <;>  
<AmountPayment7[1..11]> <;> <AmountPayment8[1..11]> <;> <AmountPayment9[1..11]> <;>  
<NoPO[1..5]> <;>

**FPR operation:** Provides information about the PO by type of payment as well as the total number of operations by specified operator

### **Input data :**

<'3'> 1 symbol obligatory '3'  
OpNo (Operator №) Symbol from 1 to 20 corresponding to operator's number

### **Output data:**

<'3'> 1 symbol obligatory '3'  
OpNo Symbol from 1 to 20 corresponding to operator's number  
AmountPayment0 1..11 symbols for the RA by type of payment 0  
AmountPayment1 1..11 symbols for the RA by type of payment 1  
AmountPayment2 1..11 symbols for the RA by type of payment 2  
AmountPayment3 1..11 symbols for the RA by type of payment 3  
AmountPayment4 1..11 symbols for the RA by type of payment 4  
AmountPayment5 1..11 symbols for the RA by type of payment 5  
AmountPayment6 1..11 symbols for the RA by type of payment 6  
AmountPayment7 1..11 symbols for the RA by type of payment 7  
AmountPayment8 1..11 symbols for the RA by type of payment 8  
AmountPayment9 1..11 symbols for the RA by type of payment 9

NoPO 1..5 symbols for the total number of operations  
**zfpdef:** *ReadOperPO(OpNo)*

### 2.7.11. Command: 6Fh / o – Reading of operator's report – 4 (received)

**input:** <'4'> <;> < OpNo[1..2]>

**output:** <'4'> <;> <OpNo[1..2]> <;> <AmountPayment0[1..11]> <;>

<AmountPayment1[1..11]> <;> <AmountPayment2[1..11]> <;> <AmountPayment3[1..11]> <;>

<AmountPayment4[1..11]> <;> <AmountPayment5[1..11]> <;> <AmountPayment6[1..11]> <;>

<AmountPayment7[1..11]> <;> <AmountPayment8[1..11]> <;> <AmountPayment9[1..11]>

**FPR operation:** Provides information about the amounts received from sales by type of payment and specified operator.

#### **Input data :**

<'4'> 1 symbol obligatory '4'

OpNo (Operator №) Symbol from 1 to 20 corresponding to operator's number

#### **Output data:**

<'4'> 1 symbol obligatory '4'

OpNo Symbol from 1 to 20 corresponding to operator's number

AmountPayment0 1..11 symbols for the RA by type of payment 0

AmountPayment1 1..11 symbols for the RA by type of payment 1

AmountPayment2 1..11 symbols for the RA by type of payment 2

AmountPayment3 1..11 symbols for the RA by type of payment 3

AmountPayment4 1..11 symbols for the RA by type of payment 4

AmountPayment5 1..11 symbols for the RA by type of payment 5

AmountPayment6 1..11 symbols for the RA by type of payment 6

AmountPayment7 1..11 symbols for the RA by type of payment 7

AmountPayment8 1..11 symbols for the RA by type of payment 8

AmountPayment9 1..11 symbols for the RA by type of payment 9

**zfpdef:** *ReadOperRecAmounts(OpNo)*

### 2.7.12. Command: 6Fh / o – Reading of operator's report – 5 (counters)

**input:** <'5'> <;> < OpNo[1..2]>

**output:** <'5'> <;> < OpNo[1..2]> <;> <NoRep[1..5]> <;>

<DateTime "DD-MM-YYYY HH:MM"> <;>

**FPR operation:** Provides information about the number of the last operator's report and its date and time.

#### **Input data :**

<'5'> 1 symbol obligatory '5'

OpNo (Operator №) Symbol from 1 to 20 corresponding to operator's number

#### **Output data:**

<'5'> 1 symbol obligatory '5'

OpNo Symbol from 1 to 20 corresponding to operator's number

NoRep 5 symbols for number of the last report

DateTime 16 symbols for date and time of the last operator's report

**zfpdef:** *ReadOperCounters(OpNo)*

### 2.7.13. Command: 6Fh / o – Reading of operator's report – 6 (returned)

**input:** <'6'> <;> <OpNo[1..2]>

**output:** <'6'> <;> <OpNo[1..2]> <;> <AmountPayment0[1..11]> <;>

<AmountPayment1[1..11]> <;> <AmountPayment2[1..11]> <;> <AmountPayment3[1..11]> <;>

<AmountPayment4[1..11]> <;> <AmountPayment5[1..11]> <;> <AmountPayment6[1..11]> <;>

<AmountPayment7[1..11]> <;> <AmountPayment8[1..11]> <;> <AmountPayment9[1..11]>

**FPR operation:** Provides information about the amounts returned as change by different payment types for the specified operator.

#### **Input data :**

<'6'> 1 symbol obligatory '6'

OpNo (Operator №) Symbol from 1 to 20 corresponding to operator's number

#### **Output data:**

<'6'> 1 symbol obligatory '6'

OpNo Symbol from 1 to 20 corresponding to operator's number

AmountPayment0 1..11 symbols for the RA by type of payment 0

AmountPayment1 1..11 symbols for the RA by type of payment 1

AmountPayment2 1..11 symbols for the RA by type of payment 2

AmountPayment3 1..11 symbols for the RA by type of payment 3

AmountPayment4 1..11 symbols for the RA by type of payment 4

AmountPayment5 1..11 symbols for the RA by type of payment 5

AmountPayment6 1..11 symbols for the RA by type of payment 6

AmountPayment7 1..11 symbols for the RA by type of payment 7

AmountPayment8 1..11 symbols for the RA by type of payment 8

AmountPayment9 1..11 symbols for the RA by type of payment 9

**zfpdef:** ReadOpCounters(OpNo)

### 2.7.14. Command: 71h / q – Reading of receipt number

**input:** n. a.

**output:** <NoLastIssRcp[1..4]> <;> <NoTotalRcp[1..7]>

**FPR operation:** Provides information about the number of the last issued receipt.

#### **Input data : n. a.**

#### **Output data :**

NoLastIssRcp 1..4 symbols for the number of the last issued receipt by FPR in format ####

NoTotalRcp 1..7 symbols for the number of the total issued receipts by FPR  
in format #####

**zfpdef:** ReadLastReceiptNo()

### 2.7.15. Command: 72h / r – Reading information about the current opened receipt

**input:** n. a.

**output:** <ParamOpenRec[1]> <;> <NoSales[3]> <;> <SubtotalAmountVATGA[1..11]>

<;> <SubtotalAmountVATGB[1..11]> <;> <SubtotalAmountVATGC[1..11]> <;>

<SubtotalAmountVATGD[1..11]> <;> <SubtotalAmountVATGE[1..11]> <;>

<ParamForbiddenVoid[1]> <;> <ParamVATinReceipt[1]> <;> <ParamDetailedReceipt[1]>

<;> <ParamInitiatedPayment[1]> <;> <ParamFinalizedPayment[1]> <;>

<FlagPowerDown[1]> <;> <ParamClientReceipt[1]> <;> <ChangeAmount[1..11]> <;>

<OptionChangeType[1]> <;> <AlteTaxeValue[1..11]>

**FPR operation:** Read the current status of the receipt.

#### **Input data : n. a.**

#### **Output data :**

ParamOpenRec 1 symbol with value '1' for initiated (opened) receipt

NoSales 3 symbols for number of sales

<i>SubtotalAmountVATGA</i>	1..11 symbols for subtotal by VAT group A
<i>SubtotalAmountVATGB</i>	1..11 symbols for subtotal by VAT group B
<i>SubtotalAmountVATGC</i>	1..11 symbols for subtotal by VAT group C
<i>SubtotalAmountVATGD</i>	1..11 symbols for subtotal by VAT group D
<i>SubtotalAmountVATGE</i>	1..11 symbols for subtotal by VAT group E
<i>ParamForbiddenVoid</i>	1 symbol with value: - '0' – allowed - '1' - forbidden
<i>ParamVATinReceipt</i>	1 symbol with value: - '0' – with printing - '1' - without printing
<i>ParamDetailedReceipt</i>	1 symbol with value: - '0' – brief - '1' - detailed format
<i>ParamInitiatedPayment</i>	1 symbol with value: - '0' – initiated payment - '1' - not initiated payment
<i>ParamFinalizedPayment</i>	1 symbol with value: - '0' – finalized payment - '1' - not finalized payment
<i>ParamClientReceipt</i>	1 symbol with value: - '0' - standard receipt - '1' - invoice (client) receipt
<i>FlagPowerDown</i>	1 symbol with value: - '0' - no power down - '1' - power down
<i>ChangeAmount</i>	1..11 symbols the amount of the due change in the stated payment type
<i>OptionChangeType</i>	1 symbols with value: - '0' - Change In Cash - '1' - Same As The payment - '2' – Change In Currency
<i>AlteTaxeValue</i>	1..11 symbols for alte tax amount

**zfpdef:** [ReadCurrentRecInfo\(\)](#)

## 2.7.16. Command: 73h / s– Reading the last date of a daily report

**input:** n. a.

**output:** <LastZDailyReportDate “DD-MM-YYYY”><;> <NoLastZDailyReport[1..4]><;><NoLastRAMReset[1..4]>

**FPR operation:** Read the date and number of the last Z-report and the last RAM reset event.

**Input data :** n. a.

**Output data :**

*LastZDailyReportDate* 10 symbols for last Z-report date in DD-MM-YYYY format  
*NoLastZDailyReport* 1..4 symbols for the number of the last daily report  
*NoLastRAMReset* 1..4 symbols for the number of the lastRAM reset

**zfpdef:** [ReadLastDailyRepDate\(\)](#)

## 2.7.17. Command: 74h / t – Reading of free FM reporting records

**input:** n. a.

**output:** <FreeFMrecords[4]><;>

**FPR operation:** Read the number of the remaining free records for Z-report in the Fiscal Memory.

**Input data : n. a.**

**Output data :**

*FreeFMrecords* 4 symbols for the number of free records for Z-report in the FM

**zfpdef:** [ReadFMfreeDailyRecords\(\)](#)

## 2.7.18. Command: 75h / u – Reading of FM content

**input: n. a.**

**output: ACK +**

*end number of packed messages for every block stored in FM:*

*<Nsegm[4]> <OptionCodeStored[1]> <DateStored[16]> <Status[1]> <ReadData [~]> +*

*a message for end of string: <Nsegm[4]><'@'>*

**FPR operation:** Provides consequently information about every single block stored in the FM starting with ACKs and ending with end message.

**Input data : n. a.**

**Output data :**

<i>Nsegm</i>	4 symbols for physical FM block number
<i>OptionCodeStored</i>	1 symbol stating the type of the stored block with the following values: 0 – factory number of FPR 1 – tax number, decimal point position and tax rate at fiscalization 4 – daily financial report 6 – change of VAT rates 7 – change of decimal point position
<i>DateStored</i>	16 symbols for the date and time of block storing
<i>Status</i>	1 symbol 0 or 1 resp. for correct/incorrect block checksum
<i>ReadData</i>	Total fields of data read
<i>&lt;'@'&gt;</i>	1 symbol obligatory '@' for end of string

**zfpdef:** [ReadFMcontent\(\)](#)

## 2.8. Reports printing commands

Set of commands for printing of reports generated by FPR.

### 2.8.1. Command: 76h / v – Report by department

**input:** <OptionZeroing[1]>

**output:** ACK

**FPR operation:** Prints departments report

**Input data :**

OptionZeroing 1 symbol (Parameter) with following values:

- 'Z' –Zeroing
- 'X' – Not zeroing

**Output data : n. a.**

*zfpdef: PrintDepReport(OptionZeroing)*

### 2.8.2. Command: 77h / w – Special events FM report

**input:** n. a.

**output:** ACK

**FPR operation:** Print all special FM events report.

**Input data : n. a.**

**Output data : n. a.**

*zfpdef: PrintSpecFMreport()*

### 2.8.3. Command: 77h / w – Special events FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]>

**output:** ACK

**FPR operation:** Print the special FM events report by initial and end FM report number.

**Input data :**

StartNo (Start) 4 symbols for the initial report number included in report, format #####

EndNo (End) 4 symbols for the final report number included in report, format #####

**Output data: n. a.**

*zfpdef: PrintSpecFMreportByReportNo (StartNo, EndNo)*

### 2.8.4. Command: 77h / w – Special events FM report by date

**input:** <StartDate “DDMMYY”><;><EndDate “DDMMYY”>

**output:** ACK

**FPR operation:** Prints the special FM events report by initial and end date.

**Input data :**

StartDate 6 symbols for initial date in the DDMMYY format

EndDate 6 symbols for final date in the DDMMYY format

**Output data: n. a.**

*zfpdef: PrintSpecFMreportByDate(StartDate, EndDate)*



### 2.8.5. Command: 78h / x – Detailed FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]>

**output:** ACK

**FPR operation:** Print a detailed FM report by initial and end FM report number.

**Input data :**

StartNo (Start) 4 symbols for the initial report number included in report, format #####

EndNo (End) 4 symbols for the final report number included in report, format #####

**Output data: n. a.**

**zfpdef:** *PrintDetFMdailyReportByZNo(StartNo, EndNo)*

### 2.8.6. Command: 78h / x – Detailed Payment FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]><;><OptionPayment['P']>

**output:** ACK

**FPR operation:** Print a detailed FM payment report by initial and end FM report number.

**Input data :**

StartNo (Start) 4 symbols for initial FM report number included in report, format #####

EndNo (End) 4 symbols for final FM report number included in report, format #####

OptionPayment 1 symbol 'P'

**Output data: n. a.**

**zfpdef:** *PrintDetFMdailyPaymReportByZNo(StartNo, EndNo)*

### 2.8.7. Command: 78h / x – Storage detailed FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]><;><OptionStorage[1]>

**output:** ACK

**FPR operation:** Storage a detailed FM report by initial and end FM report number.

**Input data :**

StartNo (Start) 4 symbols for the initial report number included in report, format: #####

EndNo (End) 4 symbols for the final report number included in report, format: #####

OptionStorage (Storage FM Report) 1 symbol for destination:  
- '2' – Storage in External USB Flash memory.  
- '4' – Storage in External SD card memory

**Output data: n. a.**

**zfpdef:** *StorageDetFMdailyReportByZNo(StartNo, EndNo, OptionStorage)*

### 2.8.8. Command: 79h / y – Brief FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]>

**output:** ACK

**FPR operation:** Print a brief FM report by initial and end FM report number.

**Input data :**

StartNo (Start) 4 symbols for the initial FM report number included in report, format #####

EndNo (End) 4 symbols for the final FM report number included in report, format #####

**Output data: n. a.**

**zfpdef:** *PrintBriefFMdailyReportByZNo(StartNo, EndNo)*

### 2.8.9. Command: 79h / y – Brief payment FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]><;><OptionPayment['P']>

**output:** ACK

**FPR operation:** Print a brief payment FM report by initial and end FM report number.

#### **Input data :**

*StartNo* (Start)4 symbols for the initial FM report number included in report, format #####

*EndNo* (End)4 symbols for the final FM report number included in report, format #####

*OptionPayment* 1 symbol 'P'

**Output data: n. a.**

*zfpdef:* [PrintBriefFMdailyPaymReportByZNo\(StartNo, EndNo\)](#)

### 2.8.10. Command: 79h / y – Store Brief FM report by number of reports

**input:** <StartNo[4]><;><EndNo[4]> {<;><OptionStorage[1]>}

**output:** ACK

**FPR operation:** Store a brief FM report by initial and end FM report number.

#### **Input data :**

*StartNo* (Start)4 symbols for the initial FM report number included in report, format #####

*EndNo* (End)4 symbols for the final FM report number included in report, format #####

*OptionStorage* (Storage FM Report) 1 symbol for destination:  
- '2' – Storage in External USB Flash memory.  
- '4' – Storage in External SD card memory

**Output data: n. a.**

*zfpdef:* [StoreBriefFMdailyReportByZNo\(StartNo, EndNo, OptionStorage\)](#)

### 2.8.11. Command: 7Ah / z – Detailed FM report by date

**input:** <StartDate "DDMMYY"><;><EndDate "DDMMYY">

**output:** ACK

**FPR operation:** Prints a detailed FM report by initial and end date.

#### **Input data :**

*StartDate* 6 symbols for initial date in the DDMMYY format

*EndDate* 6 symbols for final date in the DDMMYY format

**Output data: n. a.**

*zfpdef:* [PrintDetFMdailyReportByDate\(StartDate, EndDate\)](#)

### 2.8.12. Command: 7Ah / z – Detailed Payment FM report by date

**input:** <StartDate "DDMMYY"> <;> <EndDate "DDMMYY"> <;> <OptionPayment['P']>

**output:** ACK

**FPR operation:** Print a detailed payment FM report by initial and end date.

#### **Input data :**

*StartDate* 6 symbols for initial date in the DDMMYY format

*EndDate* 6 symbols for final date in the DDMMYY format

*OptionPayment* 1 symbol 'P'

**Output data: n. a.**

*zfpdef:* [PrintDetFMdailyPaymReportByDate\(StartDate, EndDate\)](#)



### 2.8.13. Command: 7Ah / z – Storage detailed FM report by date

**input:** <StartDate "DDMMYY"><;><EndDate "DDMMYY"> {<;><OptionStorage[1]>}

**output:** ACK

**FPR operation:** Storage a detailed FM report by initial and end date.

#### **Input data :**

StartDate            6 symbols for initial date in the DDMMYY format  
EndDate             6 symbols for final date in the DDMMYY format  
OptionStorage       (Storage FM Report) 1 symbol for destination:  
                      - '2' – Storage in External USB Flash memory.  
                      - '4' – Storage in External SD card memory

**Output data: n. a.**

**zfpdef:** *StorageDetFMdailyReportByDate(StartDate, EndDate, OptionStorage)*

### 2.8.14. Command: 7Bh / { – Brief FM report by date

**input:** <StartDate "DDMMYY"><;><EndDate "DDMMYY">

**output:** ACK

**FPR operation:** Print a brief FM report by initial and end date.

#### **Input data :**

StartDate            6 symbols for initial date in the DDMMYY format  
EndDate             6 symbols for final date in the DDMMYY format

**Output data: n. a.**

**zfpdef:** *PrintBriefFMdailyReportByDate(StartDate, EndDate)*

### 2.8.15. Command: 7Bh / { – Brief payment FM report by date

**input:** <StartDate "DDMMYY"> <;> <EndDate "DDMMYY"> <;> <OptionPayment["P"]>

**output:** ACK

**FPR operation:** Print a brief payment FM report by initial and end date.

#### **Input data :**

StartDate            6 symbols for initial date in the DDMMYY format  
EndDate             6 symbols for final date in the DDMMYY format  
OptionPayment       1 symbol 'P'

**Output data: n. a.**

**zfpdef:** *PrintBriefFMdailyPaymReportByDate(StartDate, EndDate)*

### 2.8.16. Command: 7Bh / { – Store Brief FM report by date

**input:** <StartDate "DDMMYY"><;><EndDate "DDMMYY"> {<;><OptionStorage[1]>}

**output:** ACK

**FPR operation:** Store a brief FM report by initial and end date.

#### **Input data :**

StartDate            6 symbols for initial date in the DDMMYY format  
EndDate             6 symbols for final date in the DDMMYY format  
OptionStorage       (Storage FM Report) 1 symbol for destination:  
                      - '2' – Storage in External USB Flash memory.  
                      - '4' – Storage in External SD card memory

**Output data: n. a.**

**zfpdef:** *StoreBriefFMdailyReportByDate(StartDate, EndDate, OptionStorage)*

### 2.8.17. Command: 7Ch / | – Daily fiscal report X or Z.

**Input:** <OptionZeroing[1]>

**Output:** ACK

**FPR Operation:** Depending on the parameter prints:

- daily fiscal report with zeroing and fiscal memory record, preceded by Electronic Journal report print ('Z');
- daily fiscal report without zeroing ('X');

**Input data:**

*OptionZeroing* 1 symbol (Parameter) with following values:  
- 'Z' –Zeroing  
- 'X' – Not zeroing

**Output data:** n.a.

**zfpdef:** *PrintDailyReport(OptionZeroing)*

### 2.8.18. Command: 7Ch / | – Printing Electronic Journal report from date do date

**input:** <OptionRepStorage[2]> <;> <'D'> <;> <StartDate“DDMMYY”> <;>  
<EndDate“DDMMYY”>

**output:** ACK

**FPR operation:** Printing Electronic Journal Report from Start Date to End Date.

**Input data:**

*OptionRepStorage* (Storage FM Report) 2 symbols for destination:  
- 'J1' – Printing  
- 'J2' – Storage in External USB Flash memory.  
- 'J4' – Storage in External SD card memory  
  
'D'  
1 symbol 'D'  
*StartDate* 6 symbols for initial date in the DDMMYY format  
*EndDate* 6 symbols for final date in the DDMMYY format

**Output data:** n. a.

**zfpdef:** *PrintEJreportByDate(OptionSReptorage, StartDate, EndDate)*

### 2.8.19. Command: 7Ch / | – Printing Electronic Journal report from receipt number to receipt number

**input:** <OptionRepStorage[2]><;><'N'><;><StartNo[5]><;><EndNo[5]>

**output:** ACK

**FPR operation:** Printing Electronic Journal Report from receipt number to receipt number.

**Input data:**

*OptionRepStorage* (Printing/Storage EJ Report) 2 symbols for destination:  
- 'J1' – Printing  
- 'J2' – Storage in External USB Flash memory.  
- 'J4' – Storage in External SD card memory  
  
'N'  
1 symbol 'N'  
*StartNo* (Start rec.No) 5 symbols for initial receipt number included in report, format #####  
*EndNo* (End rec.No) 5 symbols for final receipt number included in report, format #####

**Output data:** n. a.

**zfpdef:** *PrintEJreportByRecNo(OptionRepStorage, StartNo, EndNo)*

## 2.8.20. Command: 7Ch / | – Printing Electronic Journal report from number Z report to number Z report

**input:** <OptionRepStorage[2]><;><'Z'><;><StartNo[4]><;><EndNo[4]>

**output:** ACK

**FPR operation:** Printing Electronic Journal Report from report number to report number.

### **Input data:**

*OptionRepStorage* (Printing/Storage EJ Report) 2 symbols for destination:  
- 'J1' – Printing  
- 'J2' – Storage in External USB Flash memory.  
- 'J4' – Storage in External SD card memory  
  
'Z' 1 symbol 'Z'  
*StartNo* (Start Z No)4 symbols for initial number report in format ####  
*EndNo* (End Z No)4 symbols for final number report in format ####

**Output data: n. a.**

**zfpdef:** PrintEjreportZreportNo(*OptionRepStorage*, *StartNo*, *EndNo*)

## 2.8.21. Command: 7Ch / | – Printing Electronic Journal report from beginning to end

**input:** <OptionRepStorage[2]><;><'\*'>

**output:** ACK

**FPR operation:** Printing all Electronic Journal report.

### **Input data:**

*OptionRepStorage* (Printing/Storage EJ Report) 2 symbols for destination:  
- 'J1' – Printing  
- 'J2' – Storage in External USB Flash memory.  
- 'J4' – Storage in External SD card memory  
  
'\*' 1 symbol '\*'

**Output data: n. a.**

**zfpdef:** PrintEJwhole()

## 2.8.22. Command: 7Ch / | – Receipts xml export from number Z report to number Z report

**input:** <OptionRcpXmlStorage[2]><;><'Z'><;><StartNo[4]><;><EndNo[4]>

**output:** ACK

**FPR operation:** Storage of receipts xml files by Z-report number to USB Flash memory or SD card memory.

### **Input data:**

*OptionRcpXmlStorage* (Storage receipts xml) 2 symbols for destination:  
- 'Jx' – Storage in External USB Flash memory.  
- 'JX' – Storage in External SD card memory  
  
'Z' 1 symbol 'Z'  
*StartNo* (Start Z No)4 symbols for initial number report in format ####  
*EndNo* (End Z No)4 symbols for final number report in format ####

**Output data: n. a.**

**zfpdef:** ExportEjreportZreportNo(*OptionRcpXmlStorage*, *StartNo*, *EndNo*)

### 2.8.23. Command: 7Dh / } – Operator's report

**input:** <OptionZeroing[1]> <;> <Number[1..2]>

**output:** ACK

**FPR operation:** Prints an operator's report for a specified operator (0 = all operators) with or without zeroing ('Z' or 'X'). When a 'Z' value is specified the report should include all operators.

**Input data :**

*OptionZeroing* 1 symbol (Parameter) with following values:  
- 'Z' –Zeroing  
- 'X' – Not zeroing

*Number* (Operator No) Symbols from 0 to 20 corresponding to operator's number (0 = all operators)

**Output data: n. a.**

**zfpdef:** PrintOperReport(*OptionZeroing*, *Number*)

### 2.8.24. Command: 7Eh / ~ – Article report

**input:** <OptionZeroing[1]>

**output:** ACK

**FPR operation:** Prints an article report with or without zeroing ('Z' or 'X').

**Input data :**

*OptionZeroing* 1 symbol (Parameter) with following values:  
- 'Z' –Zeroing  
- 'X' – Not zeroing

**Output data: n. a.**

**zfpdef:** PrintArticleReport(*OptionZeroing*)

### 2.8.25. Command: 7Fh / ■ – Extended daily report

**input:** <OptionZeroing[1]>

**output:** ACK

**FPR operation:** Prints an extended daily financial report (an article report followed by a daily financial report) with or without zeroing ('Z' or 'X').

**Input data :**

*OptionZeroing* 1 symbol (Parameter) with following values:  
- 'Z' –Zeroing  
- 'X' – Not zeroing

**Output data: n. a.**

**zfpdef:** PrintExtendDailyRep(*OptionZeroing*)

### 2.8.26. Command: 52h / R – option X, Print Customer X or Z report

**input:** <OptionZeroing[1]>

**output:** ACK

**FPR Operation:** Print Customer X or Z report

**Input data:**

*OptionZeroing* 1 symbol (Parameter) with following values:  
- 'Z' –Zeroing  
- 'X' – Not zeroing

**Output data: n.a.**

**zfpdef:** PrintCustomerReport(*OptionZeroing*)

### 2.8.27. Command: 59h / Y – option H - Print hourly report

**Input:** <'H'>

**output:** *ACK*

**FPR Operation:** Print hourly report

*\*the command is valid for ADPOS model only.*

#### **Input data:**

'H'                      1 symbol 'H'

**Output data:** *n.a.*

**zfpdef:** *PrintHourlyReport()*

### 2.8.28. Command: 5Ah / Z – option 1, Reading Z report number from date

**input:** <Option['1']> <;> <Date "DDMMYY">

**output:** <ZreportNo[4]>

**FPR operation:** Provide information about first found Z report, having the same date or date after than the input date

#### **Input data :**

Option                      1 symbol with value '1'

Date                        6 symbols for specified date in format "DDMMYY"

**Output data:**

ZreportNo                4 symbols for the first found Z report number in format: #####

**zfpdef:** *ReadZreportNoFromDate1(Date)*

### 2.8.29. Command: 5Ah / Z – option 2, Reading Z report number from date

**input:** <Option['2']> <;> <Date "DDMMYY">

**output:** <ZreportNo[4]>

**FPR operation:** Provide information about last found Z report, having the same date or date before than the input date

#### **Input data :**

Option                      1 symbol with value '2'

Date                        6 symbols for specified date in format "DDMMYY"

**Output data:**

ZreportNo                4 symbols for the last found Z report number in format: #####

**zfpdef:** *ReadZreportNoFromDate2(Date)*

## 2.9. Reports READING commands

Set of commands for reading of reports generated by FPR.

### 2.9.1. Command: 7Ch / | – Reading Electronic Journal report from date do date

**input:** <'J0'><;><'D'><;><StartDate "DDMMYY"><;><EndDate "DDMMYY">

**output:** ACK+

**FPR operation:** Reading Electronic Journal Report from Start Date to End Date.

**Input data:**

'J0'	(Reading EJ Report) 1 character with value 'J0'
'D'	1 symbol 'D'
StartDate	6 symbols for initial date in the DDMMYY format
EndDate	6 symbols for final date in the DDMMYY format

**Output data: n. a.**

**zfpdef:** ReadEJreportByDate(StartDate, EndDate)

### 2.9.2. Command: 7Ch / | – Reading Electronic Journal report from receipt number to receipt number

**input:** <'J0'><;><'N'><;><StartNo[5]><;><EndNo[5]>

**output:** ACK+

**FPR operation:** Reading Electronic Journal Report from receipt number to receipt number.

**Input data:**

'J0'	(Reading EJ Report) 1 character with value 'J0'
'N'	1 symbol 'N'
StartNo	(Start rec.No) 5 symbols for initial receipt number included in report, format #####
EndNo	(End rec.No) 5 symbols for final receipt number included in report, format #####

**Output data: n. a.**

**zfpdef:** ReadEJreportByRecNo(StartNo, EndNo)

### 2.9.3. Command: 7Ch / | – Reading Electronic Journal report from number Z report to number Z report

**input:** <'J0'><;><'Z'><;><StartNo[4]><;><EndNo[4]>

**output:** ACK+

**FPR operation:** Reading Electronic Journal Report from report number to report number.

**Input data:**

'J0'	(Reading EJ Report) 1 character with value 'J0'
'Z'	1 symbol 'Z'
StartNo	(Start Z No)4 symbols for initial number report in format ####
EndNo	(End Z No)4 symbols for final number report in format ####

**Output data: n. a.**

**zfpdef:** ReadEJreportZreportNo(StartNo, EndNo)

#### 2.9.4. Command: 7Ch / | – Reading Electronic Journal report from beginning to end

input: <'J0'><;><'\*!'>

output: ACK+

FPR operation: Reading all Electronic Journal report from beginning to the end.

##### **Input data:**

'J0' (Reading EJ Report) 1 character with value 'J0'  
'\*' 1 symbol '\*'

##### **Output data: n. a.**

**zfpdef:** EJreportFromBeginningEnd()





### **3. SOFTWARE APPLICATION REQUIREMENTS**

#### **3.1. RULES FOR USING THE COMMANDS**

The commands should be used observing the following rules:

- Do not send a subsequent command prior to receiving a response of the preceding one.
- Observe the sequence of sent and received messages.
- The number of the message in every subsequent command should differ from that in the preceding one.
- Observe the two status bites of the Acknowledgment response.
- When the information received is insufficient request detailed status information – Command 20h.
- Use unpacked messages to check the standby status of the FPR.

#### **3.2. SAMPLE SALE TRANSACTION OF FPR**

The sale transaction controlled by a software application (SA) is a procedure, which consists of several commands, of which obligatory are: initiation of customer fiscal receipt (command 30h), sale registration (command 31h or 32h), payment (command 35h) and finalization of the fiscal receipt (command 38h ).

Sample command sequence for issuing a customer fiscal receipt:

- fiscal receipt opening (command 30h) – contains information about the operator's number and password, the type of receipt – detailed/brief, with/without VAT printing
- sale registration (command 31h) – contains information about the article's name, price and VAT class as well as non-compulsory information about the quantity sold and value/percent discount/addition;
- sale registration from the article database of FPR (32h) – contains the number of the article as well as non-compulsory information about the quantity sold and value/percent discount/addition;
- subtotal amount (command 33h) – contains non-compulsory parameters for printing, external display visualization and value/percent discount/addition of the amount accumulated;
- current receipt information (command 72h) – requires a response from the FPR, which contains the current parameters of the receipt, the number of sales, the accumulated amounts by VAT classes, information about initiated or finalized payments;
- calculation and payment of VAT on VAT account (command 36h) – performs automatic calculation of VAT in the receipt and its payment on VAT account;
- payment (command 35h) – contains information about the amount due and type of payment, which may cover partially or in full the grand total amount due as well as a parameter for calculation of change due;
- fiscal receipt closure (command 38h).

#### 4. AUXILARY GS PROTOCOL (COMMANDS 1Dh)

GS command	Message from the AS	FP Answer
Information	<1Dh><Info> where: <b>Info</b> – character 49h - 'I'	<'I'><Number of printable characters per line[2]> <;><PLU number[4]> <;><Departments number[1..2]> <;><Operators number[1..2]> <;><VAT classs number[1]> <;><header lines number[1..2]> <;><payments number[1..2]> <;><logo number[1..2]> <;><reserve> <;><receipt transaction number[3]> <;><clients base info> <;><reserve> <0Ah>
Model	<1Dh><Model> where: <b>Model</b> – character 77h - 'M'	<'M'><country[2]> <;><encoding[1..20]> <;><device type [2]> <;><build No[2..4]> <;><build date [10]> <;><interfaces[1..40]> <;><current interface[1..4]> <;><battery support[0..12]> <;><current battery suplay ('E' or 'B') [1]> <;><sleep mode [1] > <;>< Sd card journal [1]> <;><SD card format [1..2]> <;><SD card additional DB [1]> <;><server exchange[1]> <;>< Number of printable characters per line[2]> <;><number of printable free text[2]> <;><article name symbols in command[2]> <;><printable article name symbols[2]> <;><department name symbols in command[2]> <;><printable department name symbols[2]> <;><department first number[1]> <;><article last number[5]> <;><total number of headers> <;><number of fiscal headers only> <;><number of footers> <0Ah>