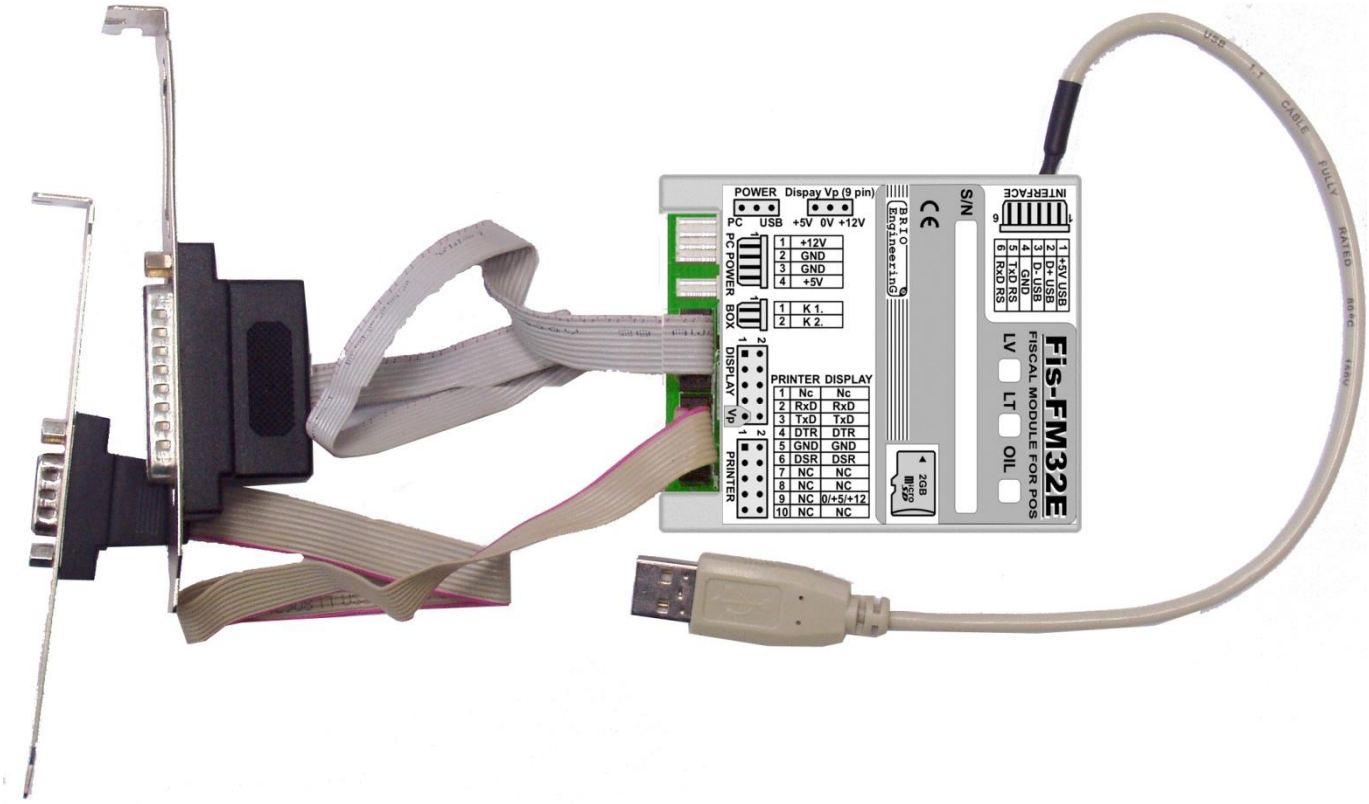
BRIO Fis-FM32/32E

FISCAL MODULE

SERVICE INSTRUCTION



BRIO EngineerinG®

O. Halatov

Fiscal module Fis-FM32/32E™ for POS systems
BRIO EngineerinG, 2010.
Riga, Latvia.

This document contains a description, operations and programming of fiscal modules Fis-FM32, Fis-FM32E™.

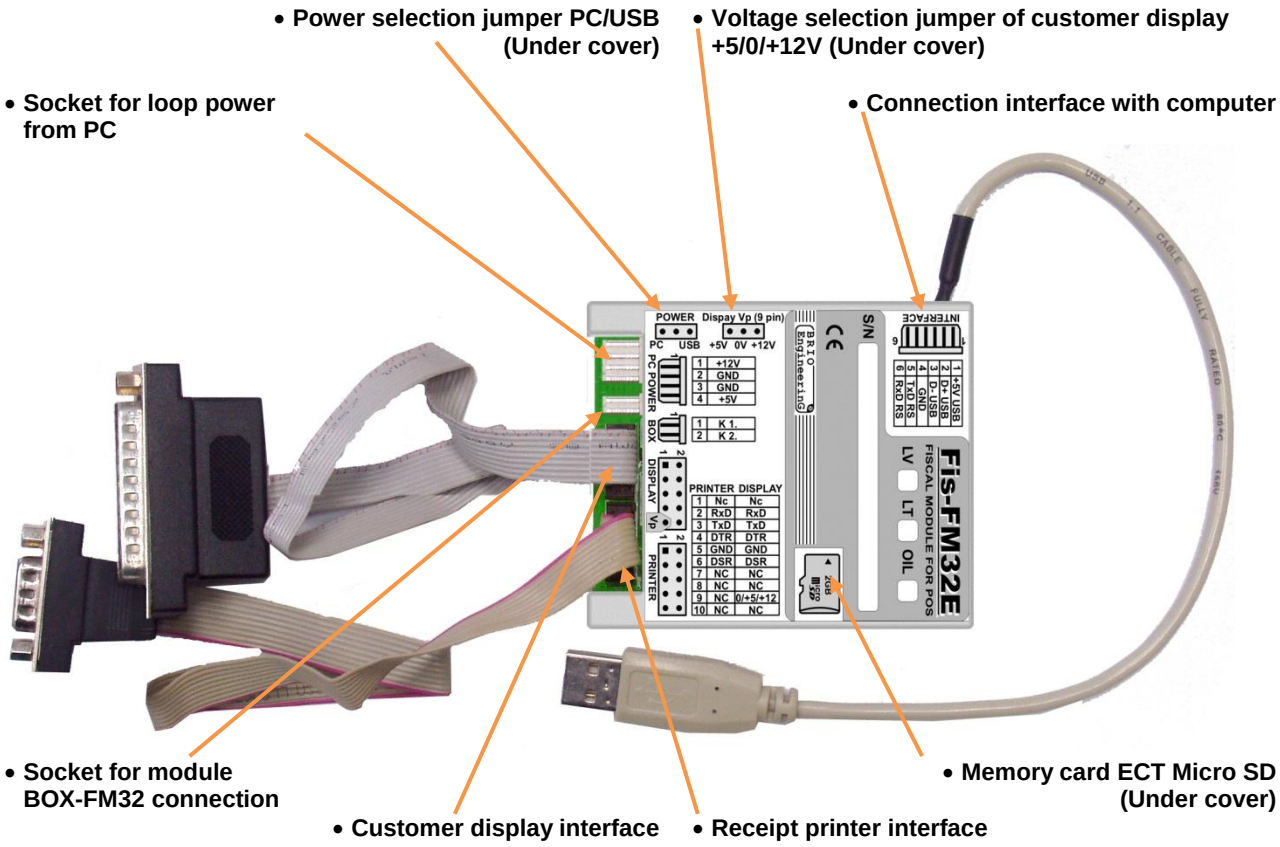
WARNING! Maintenance and adjustment of the fiscal module can be made only by authorized BRIO SRC service centres!!!

TABLE OF CONTENS

1. SPECIFICATIONS

	Fis-FM32	Fis-FM32E
Connection interface with PC, with automatic selection of interface type:	• USB V2.0, • RS-232C (115200 Bits/sec. 8 Bits, NP, 1-Stop, Non flow control)	
Built in processor type:	• ARM	
Maximum current consumption:	• 100 mA (Max)	
The volume of non-volatile memory for the current data:	• 4 MByte	
The volume of fiscal memory ROM:	• 8 MByte	
Type of memory for electronic control tape:	• No	• Micro SD (2 GByte)
Exchange state PC ↔ Fis-FM32/32E:	• Packet exchange used special protocol.	
Fiscaliziation and initialization:	• Single	
The maximum number of Z-Report:	• 8192	• 1800
Hardware diagnostics:	• Built-in diagnostic of errors	
Protection from deleting:	• Blocking system for delete command.	
Printer interface:	• RS-232C	
Types of supported printers:	PRINTERS WITH CONTROL TAPE: • Epson TM-U210A • Epson TM-U220A • Epson TM-U260A • Bixolon SRP-270D • + Any printer compatible with the system commands and interface.	PRINTERS WITH CONTROL TAPE: • Epson TM-U210A • Epson TM-U220A • Epson TM-U260A • Bixolon SRP-270D ТЕРМОПРИНТЕРА: • Bixolon SRP-350Plus • Partner Tech RP-300 • Bixolon SRP-275C • Bixolon SRP-370 • Bixolon SRP-500 • Partner Tech PT-6200 • + Any printer compatible with the system commands and interface.
Customer display interface:	• RS-232C • At the 9th output connector can apply voltage +5 V, or +12 V to supply the display.	
Types of supported customer displays:	• CD7220 • Epson • Bixolon • + Any display compatible with the system commands and interface.	
Cash drawer control:	• Command via receipt printer.	• Command via receipt printer. • Directly through the power module **BOX-FM32E**

2. CONNECTORS AND INTERFACES



2.1. CONNECTION INTERFACE WITH PC: RS-232C/USB

Num.	Signal	Direction	Assignment
1	DC +5V USB		Voltages DC +5V
2	D + USB		Bus USB
3	D - USB		Bus USB
4	GND.L		Common wire
5	TxD	Output	Data transfer
6	RxD	Input	Data receive

2.2. RECEIPT PRINTER INTERFACE: RS-232C

			PINS IN LOOP CONNECTORS	
Num.	Printer	Signal assignment	DB 25 (M)	DB 9 (M)
1	Nc	Not used		
2	RxD	Data receive	3	2
3	TxD	Data transfer	2	3
4	DTR	Request of availability	20	4
5	GND	Common	7	5
6	DSR	Printer availability	6	6
7	Nc.	Not used		
8	Nc.	Not used		
9	Nc.	Not used		

2.1. MODULE BOX-FM32 INTERFACE: BOX

Num.	Symbol	Signal assignment
1	K1.	Optorelay contact
2	K2.	Optorelay contact

2.1. CUSTOMER DISPLAY INTERFACE: RS-232C

			PINS IN LOOP CONNECTORS	
Num.	**DISPLAYS**	**SIGNAL ASSIGNMENT**	**DB 25 (M)**	**DB 9 (M)**
1	Nc	Not used		
2	RxD	Data receive	3	2
3	TxD	Data transfer	2	3
4	DTR	Request of availability	20	4
5	GND	Common	7	5
6	DSR	Display availability	6	6
7	Nc	Not used		
8	Nc	Not used		
9	0/+5V/+12V	Customer display power	22	9
10	Nc	Not used	Nc	Nc

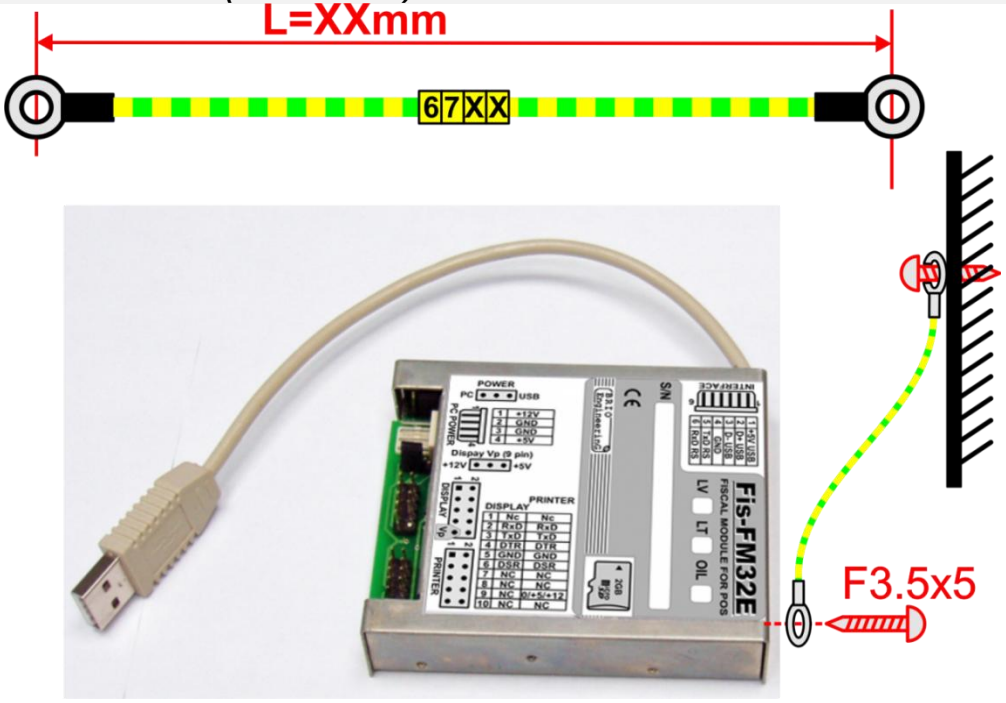
2.1. CONNECTOR FOR POWER CABEL CONNECTION FROM PC

Num.	**Symbol**	**SIGNAL ASSIGNMENT**
1	+ 12V	DC +12V
2	GND	Common wire
3	GND	Common wire
4	+5V	DC +5V

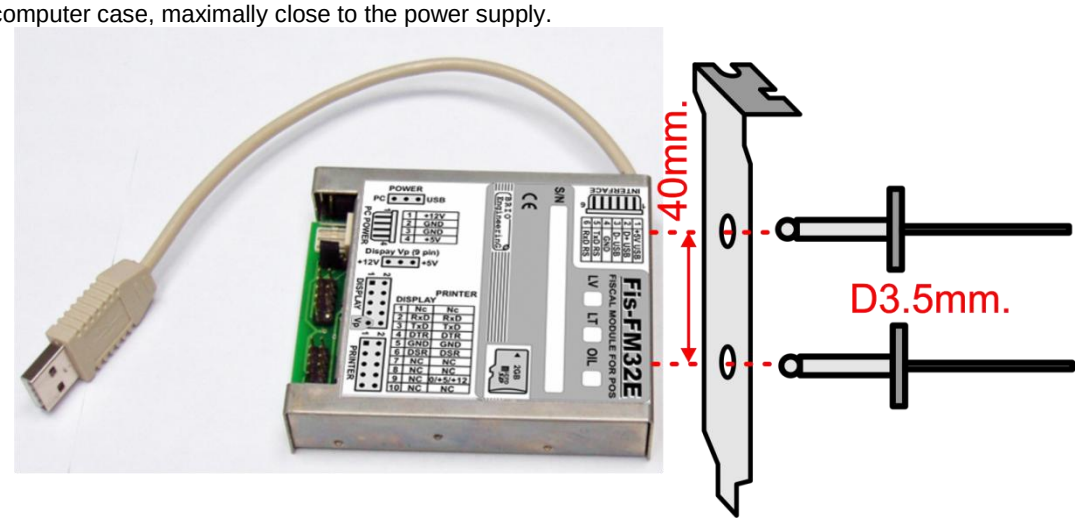
3. FISCAL MODULE CONNECTION

3.1. INSTALLATION AND GROUNDING CASE

WARNING! The metal case of the fiscal module must be electrically safety connected to the device case, to the interface of witch one (USB or RS-232) fiscal module is connected!

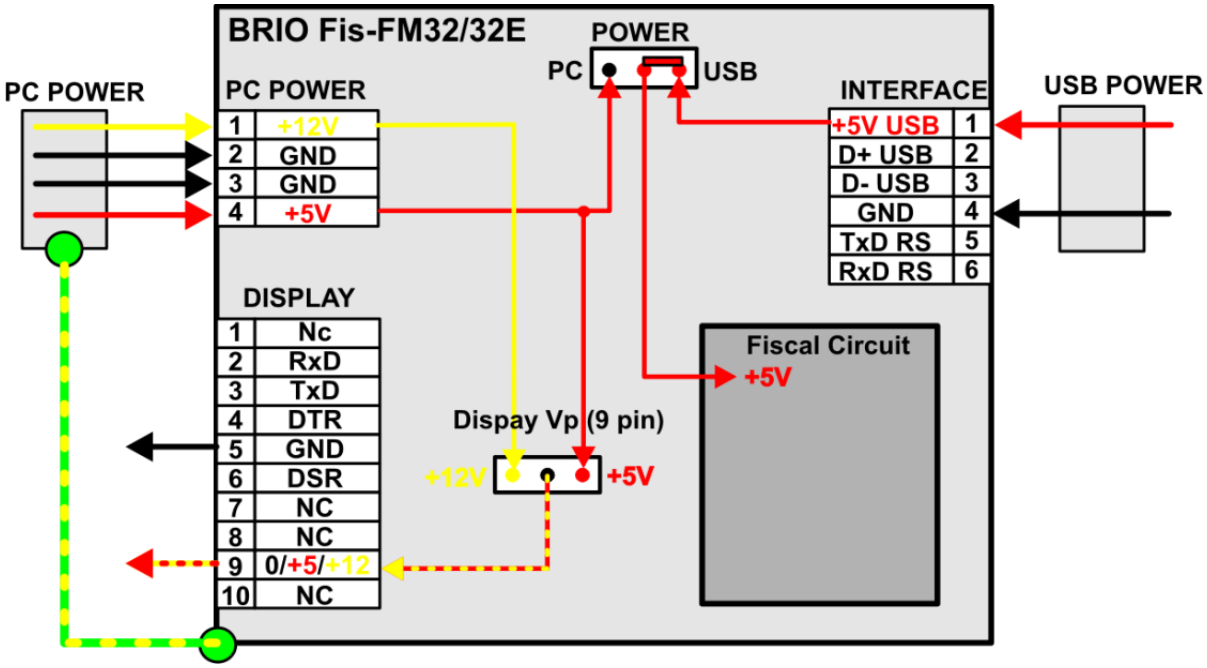


- To ensure safety grounding of fiscal module case, should use a separate cable **CB-67XX.**
- Cable must be fixed with screw F3.5x5 **(No longer than 5mm!)** in one of the holes on the front part of module case.
- The second end of the cable must be fixed on metal part, safety related to the common-grounding of the computer case, maximally close to the power supply.



- Fiscal module can be installed in a computer using a standard plate. Plate should be preattached to the case of the fiscal module with two exhaust rivets with diameter 3,5 mm.
- If the fiscal module is installed in the computer case using a standard plate, then the additional grounding cable is not required.

3.2. FISCAL MODULE AND DISPLAY POWER



- Power supply for the fiscal module can be supplied from the connector INTERFACE, and from connector PC POWER too, that can select with jumper POWER position, located inside the case.
- If for fiscal module is used USB interface, the power should be produced from the interface connector INTERFACE (Jumper position POWER – USB).
- If is used RS-232 interface, to the connector PC POWER should connect standard cable from computer power supply and set jumper POWER position to PC position.

WARNING!!! Strictly forbidden, using the USB interface, to use the power from the connector PC POWER (Jumper position - PC)!!!

- For the customer display supply at (9) connector contact DISLPAY can apply voltage. For this to connector PC POWER must connect standard cable from the computer power supply, and a jumper (Display Vp) set to +5 V, or +12 V, depending on the required voltage.

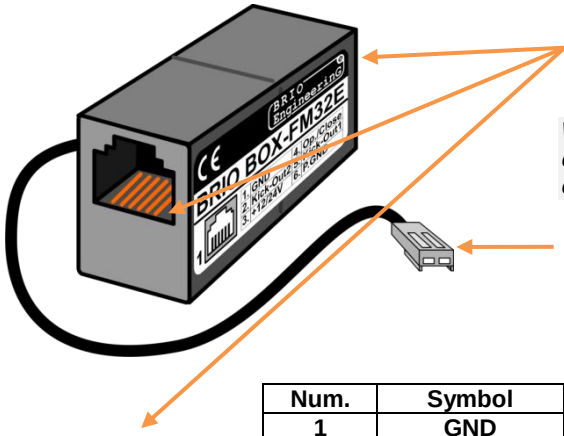
4. CASH DRAWER CONTROL

Fiscal module integrated program allows to open a cash drawer only in cases defined in the current legislation. And does it simultaneously in two ways:

- As single command for receipt printer to open connected cash drawer.
- As the closure of K1 and K2 contacts on the BOX connector, to that must be connected the power-module BOX-FM32E. (Only for versions of the fiscal module with this connector.)

WARNING!!! Strictly forbidden to use fiscal module's BOX output for direct cash drawer electromagnet switching without the use of power-module BRIO BOX-FM32E!

4.1. POWER-MODULE BRIO BOX-FM32E



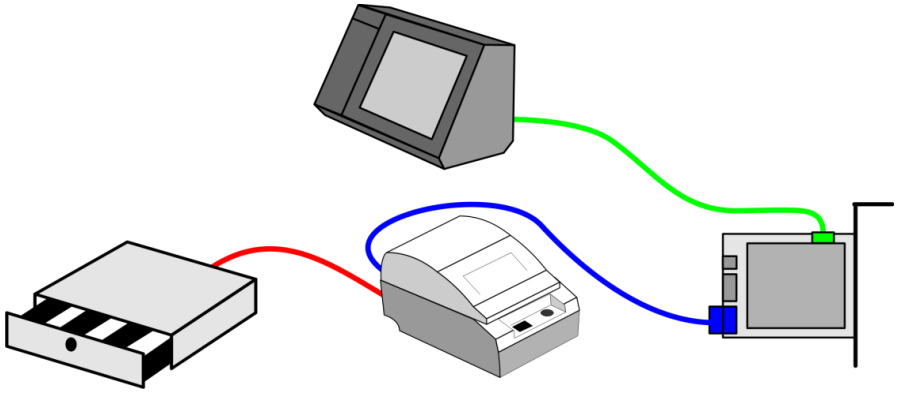
- Connectors for cable connecting from the cash drawer, or cables for power supply.

WARNING!!! All contacts of one of these connectors are connected to similar pins of another connector. Therefore, connectors are used interchangeably.

- Cable for power-module connecting to fiscal module **BOX** connector.

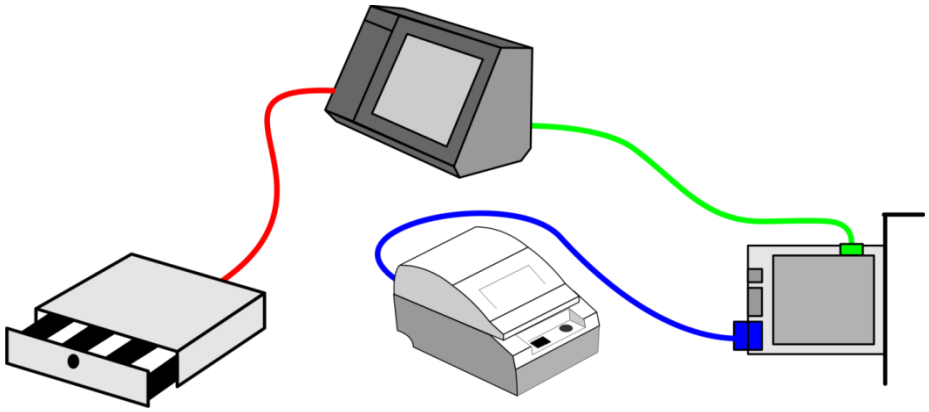
Num.	Symbol	SIGNAL ASSIGNMENT
1	GND	Common wire
2	Kick Out-2	Key-2 of box's electromagnet (Not used)
3	DC +12/24V	Voltage supply
4	Open/Close	Sensor of box opening
5	Kick Out-1	Key-1 of box's electromagnet
6	P.GND	Grounding

4.2. OPTION FOR BOX OPENING BY COMMANDS TO PRINTER



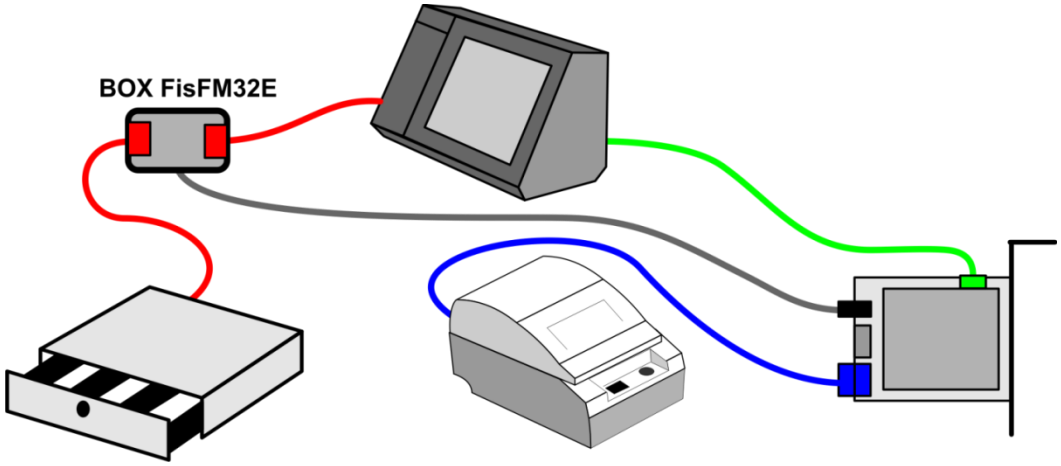
- This management option is the preferred and used if the receipt printer has a special interface for cash drawer connection.
- Cash drawer is connected to proper receipt printer's connector.
- Fiscal module at the right time send a command to open the box, and the printer opens the cash drawer.

4.3. OPTION FOR BOX OPENING DIRECTLY BY POS



- This management option is used if receipt printer hasn't interface for cash drawer connecting, but it is in the POS. In addition, this option can be used when it is required to open the box not only in cases specified by law.
- Cash drawer is connected to proper POS connector. (Is not in all models).
- Cash drawer opening directly by the POS software commands. (This function is not available in all software versions).

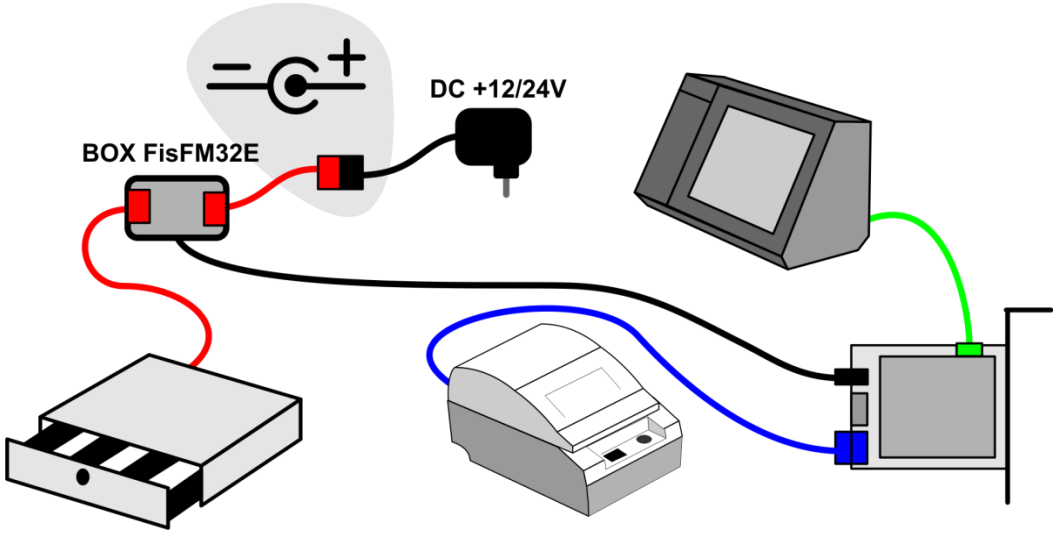
4.4. OPTION FOR BOX OPENING BY FISCAL MODULE



- This management option is used if receipt printer hasn't interface for cash drawer connecting, in the POS this interface is available, but the POS software not supported the box opening.
- Cash drawer is connected to the power-module BRIO BOX-FM32E, and this power-module is connected to proper POS connector by **CB-70XX** cable with minimum length.
- Cash drawer opening directly through the BOX interface by fiscal module.

WARNING!!! In this connection cash drawer can be opened in two ways - by a fiscal module and by POS interface, if later install on it software that allows you to manage the box opening.

4.5. OPTION FOR BOX OPENING BY FISCAL MODULE WITHOUT POS



- This management option is applied if the receipt printer and POS hasn't interface for connecting cash drawer.
- Cash drawer is connected to the power-module BRIO BOX-FM32E, which via cable **CB-71XX** with minimum length and in the correct polarity is connected to external power supply.
- The power supply should have output voltage required for this cash drawer model (usually DC +12, or +24 V), and the maximum current of 500 - 800 mA
- The cash drawer opening directly through the BOX interface by fiscal module.

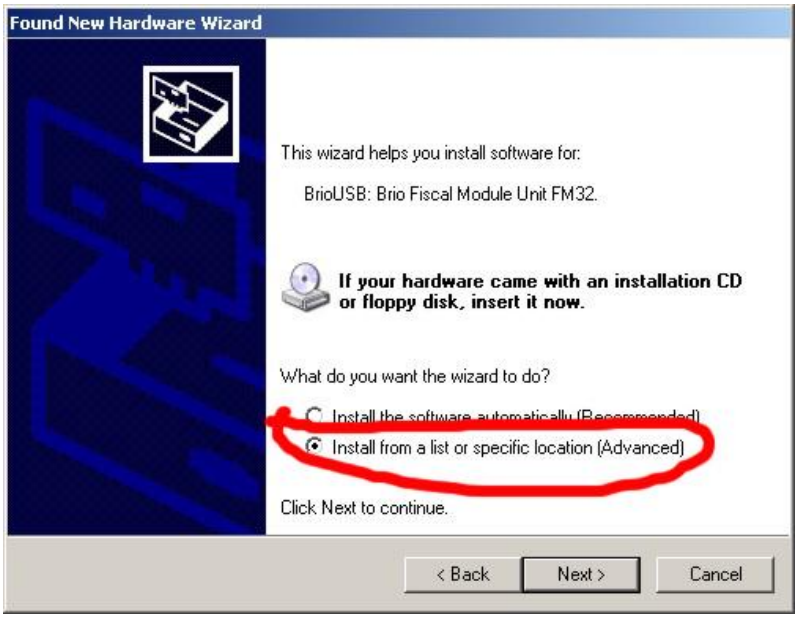
5. INSTALLATION AND SETTINGS

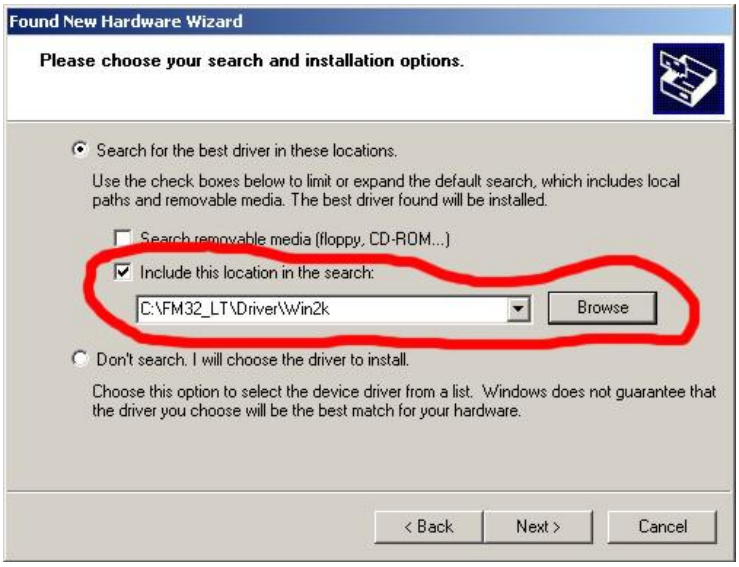
5.1. INSTALLATION

- Connect the fiscal module to the proper computer.

WARNING!!! If it is used USB interface, Windows automatically detects the fiscal module and offers to install drivers for it.

- Specify the path to the disc's partition that contains the driver for the fiscal module.
- When is used RS-232 interface (serial port on computer), the driver installation is not required.





- Specify the path to the disc's partition that contains the driver for the fiscal module, properly the version of operating system.

- Ignore systems warning and press CONTINUE.

WARNING!!! If it is used a serial interface RS-232C (COM port on computer) for communication with the computer, the driver installation is not required.

5.2. SETUP KONFIGURATION FILE

- Open the configuration file FisUnit.ini in any text editor and make the necessary corrections.

```
[Log]
  Log=1

[Port]
   ;port=com1
   ;baud=19200
    port=usb

[Journal]

UseJournal=1
WorkingDir=..\Journal
PathCopy=..\Journal\Arhiv1
```

LOG FILE SETUP
1 – Use log file. 0 – Don't use.

INTERFACE SELECTION FOR COMMUNICATION WITH COMPUTER

- COM port number. (If it is used USB, comment out ";")
- Rate. (If it is used USB, comment out ";")
- Connecting thought USB. (If it is used COM, comment out ";")

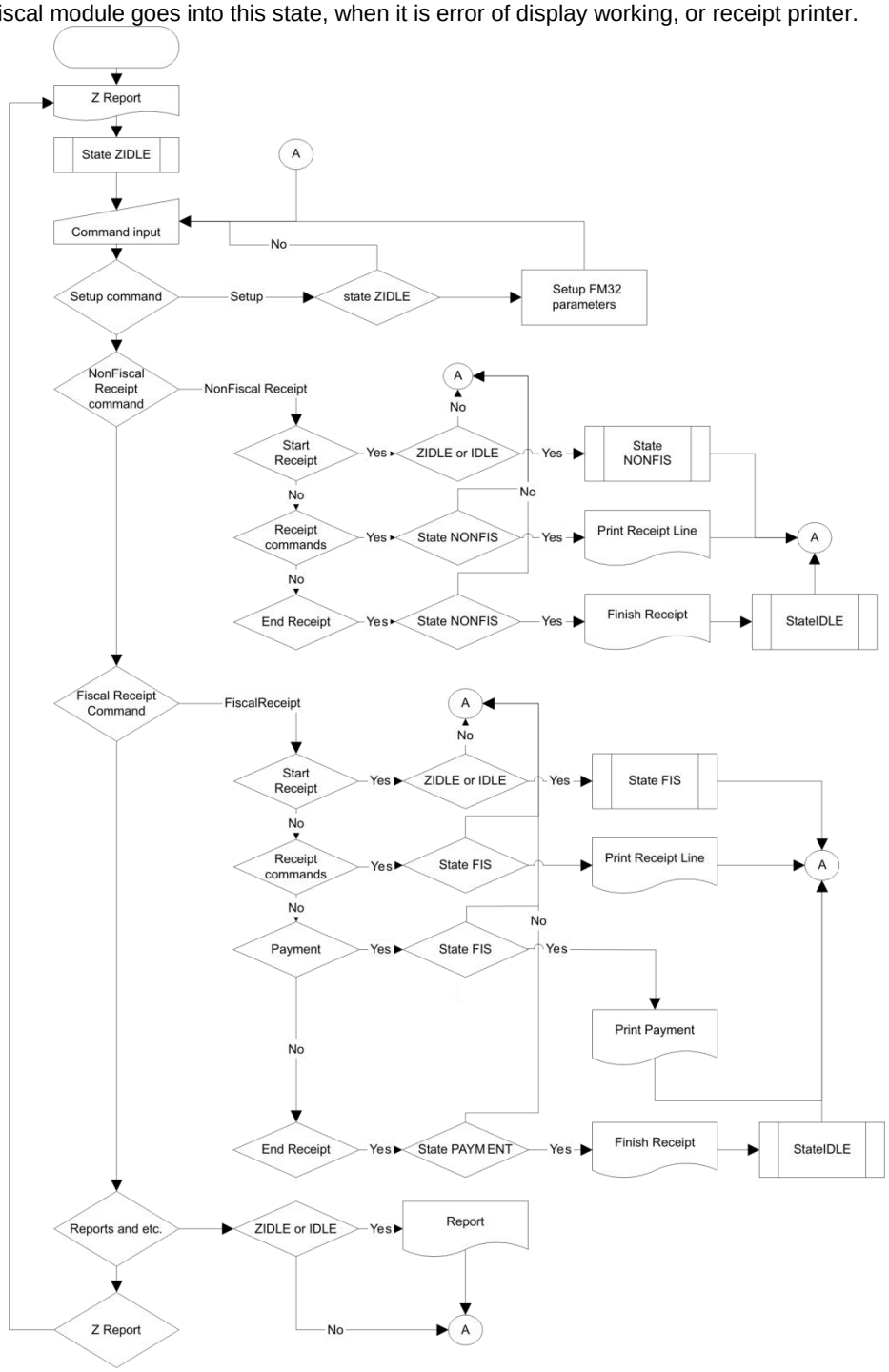
LOG SETTINGS. (Just for Fis-FM32E!!! If it is used FisFM32, all setting must be commented out ";")
1 – Use LOG. 0 – Don't use.

- The path to the section for storing log files.
- The path to the section for storing backups of control tapes.

6. FISCAL MODULE STATES

Fiscal module can be in five different states. Commands can be performed only in those states for which they are allowed. After some commands performing, the fiscal module can change its state.

- **IDLE** - Fiscal module goes into this state, after printout current receipt.
- **ZIDLE** - Fiscal module goes into this state, after Z-report processing.
- **FIS** – Fiscal module state, when started processing new fiscal receipt, but not yet processed its payment.
- **NONFIS** – State, when started but not yet completed non-fiscal receipt.
- **HARD** - Fiscal module goes into this state, when it is error of display working, or receipt printer.



ALGORITHM FOR COMMANDS PROCESSING

7. FUNCTION LIBRARY FOR WORKING WITH BRIO FisFM32/32E

RESET FISCAL MODULE STATE
int ResetFiscal(void)

RECEIPT PRINTER TYPE SETUP
int SetPrinterType(int printerType)
Used printer types:
- printerEpson210 = 0
- printerAllThermalPrinters = 10,
- printerSRP275 = 11

RECEIPT PRINTER TYPE SETUP WITH INDICATING CHARACTER ENCODING TYPE
int SetPrinterTypeEx(int printerType, int coding)
Types of character encoding:
- codingDOS = 0
- codingWin = 1

RECEIPT PRINTER TYPE SETUP WITH INDICATING CHARACTER ENCODING TYPE AND LOGO PRINT SETTINGS
int SetPrinterTypeEx2(int printerType, int coding, int image, int imagenumber, int imagelarge);
- printerType – from function SetPrinterType
- coding - from function SetPrinterTypeEx
- image – 1- use, 0 – don't use uploaded logo in printer
- imagenumber – Serial number of uploaded logo in printer 1..8
- imagelarge – 1- double, 0 – don't double logo size when printing

INTERVAL SETUP BETWEEN LINES ON RECEIPT PRINTER
int SetCompressionMode(int compression)

IMPULSE DURATION SETUP FOR CASH DRAWER OPENING
int SetTillImpuls(int time1, int time2)

FISCAL MODULE DATE SETUP
int SetDate(char * dateString)
Date in format "YYYY.MM.DD"

FISCAL MODULE TIME SETUP
int SetTime(char * timeString)
Time in format "HH:MM"

RECEIPT HEADER SETUP
int SetHeader(char * line1, char * line2, char * line3, char * line4)

RECEIPT HEADER SETUP WITH LINE FORMATTING PARAMTERS
int SetHeaderEx(int attr1, char * line1, int attr2, char * line2, int attr3, char * line3, int attr4, char * line4)

RECEIPT FOOTER LINE SETUP WITH LINE FORMATTING PARAMETRS
int SetFooter(int attr1, char * line1, int attr2, char * line2, int attr3, char * line3, int attr4, char * line4)

ENABLE/DISABLE PRINTING OF RECEIPT FOOTER LINES
int EnableFooter(int enable)

ADDITIONAL RECEIPT FOOTER LINES SETUP WITH FORMATTING
int SetFooter2(int attr1, char * line1, int attr2, char * line2)

ENABLE/DISABLE PRINTING OF ADDITIONAL RECEIPT FOOTER LINES
int EnableFooter2(int enable)

TAX RATE SETUP
int SetVAT(int number, double rate)
int SetVat(int number, double rate)

CURRENCY PARAMETER SETUP
int SetCurrency(int number, char * name, double rate)

NAME SETUP FOR CASHLESS TYPE OF PAYMENT
int SetCredit(int number, char * name)

ENABLE / DISABLE RETURN OPERATION
int AllowGoodsReturn(char * unitNumber)

INFORMATION RECEIPT FROM FISCAL MODULE
int GetFiscalInfo(int infoType, char * data)

- infoReceiptSumm = 0 – total sum of started receipt
- infoShiftTurnover = 1 – total turnover of the shift
- infoReceiptNumber = 2 – current receipt number
- infoUnitNumber = 3 – fiscal block number
- intUnitVersion = 4 – fiscal block version
- infoDate = 5 – date in a fiscal block
- infoTime = 6 – time in a fiscal block
- infoReportNumber = 7 – current Z-report number
- infoDayReceiptNumber = 8 – fiscal receipt number of the shift
- infoUnitState = 9 – fiscal module state
- infoHeader = 10 – receipt header lines
- infoFooter = 11 – receipt footer lines
- infoFooter2 = 12 – additional receipt footer lines
- infoFooterEnabled = 13 – enable receipt footer lines printing
- infoFooter2Enabled = 14 – enable additional receipt footer lines printing
- infoCurDescription = 30 – table elements of currency
- infoTax = 40 – table elements of tax
- infoCurCash = 50 – table elements of money remaining in cash drawer
- infoCreditDescription = 60 – table elements of cashless type payment
- infoTaxTurnover = 70 – turnover table elements by tax type
- infoTaxSumm = 80 – tax sum table elements be tax type

EXCHANGE MONEY ENTER
int MoneyIn(double money)

EXCHANGE MONEY ENTER IN CURRENCY
int MoneyInCurr(int number, double money);
// Extract money from cash drawer

ENCASHMENT
int MoneyOut(double money)

ENCASHMENT IN CURRENCY
int MoneyOutCurr(int number, double money)
int MoneyOutCurr2(int number, double money)

OPEN CASH DRAWER
int OpenCashDrawer(void)
int OpenCachDLINEer(void)

PRINT TAX TABLE ON RECEIPT PRINTER
int PrintVATTable(void)
int PrintVatTable(void)

PRINT CURRENCY TABLE ON RECEIPT PRINTER
int PrintCurrencyTable(void)

PRINT CASHLESS TYPE'S NAMES OF PAYMENT ON RECEIPT PRINTER
int PrintCreditTable(void)

PRINT Z-REPORT
int PrintZReport(void)

PRINT X-REPORT
int PrintXReport(void)

PRINT MINI X-REPORT
int PrintMiniXReport(void)

PRINT SUMMARY PERIODIC REPORT BY DATE RANGE
int PrintSumPeriodicReport(char * date1, char * date2)
Dates in format "YYYY.MM.DD".

PRINT SUMMARY PERIODIC REPORT BY NUMBER RANGE
int PrintSumPeriodicReportByNumber(int number1, int number2)

PRINT PERIODIC REPORT BY DATE RANGE
int PrintPeriodicReport(char * date1, char * date2)
Dates in format "YYYY.MM.DD".

PRINT PERIODIC REPORT BY NUMBER RANGE
int PrintPeriodicReportByNumber(int number1, int number2)

PRINT INFORMATION ON CUSTOMER DISPLAY
int CustomerDisplay(int displayType, char * line1, char * line2)
int CustomerDisplay2(char * line1, char * line2)
int CustomerDisplayPro(char * command)
displayType – Display type : 1

BEGIN NONFISCAL RECEIPT
int BeginNonFiscalReceipt(void)

PRINT LINE WITH TARE
int PrintTareItem(char * name, double quantity, double price)

NO TARE POSITION
int PrintTareItemVoid(char * name, double quantity, double price)

PRINT LINE WITH DEPOSIT
int PrintDepositReceive(char * name, double quantity, double price)

NO DEPOSIT POSITION
int PrintDepositRefund(char * name, double quantity, double price)

PRINT INFORMATION LINE IN A NONFISCAL RECEIPT
int PrintNonFiscalLine(char * line, int attribute)

END NONFISCAL RECEIPT
int EndNonFiscalReceipt(void)

BEGIN FISCAL RECEIPT
int BeginFiscalReceipt(void)

PRINT FISCAL RECEIPT LINE
int PrintRecItem(char * name, double quantity, double price, int taxNumber, char * unit)

NO FISCAL RECEIPT LINE
int ItemReturn(char * name, double quantity, double price,
int taxNumber, char * unit, int depart,
double discountPercent, double discountSumm)

int ItemReturnEx(char * name, double quantity, double price,
int taxNumber, char * unit, int depart,
int discountType, double discount)

PRINT COMMENT LINE IN FISCAL RECEIPT
int PrintCommentLine(char * line, int attribute)

DISCOUNT FOR RECEIPT POSITION
int DiscountAdditionForItem(int type, double val)
Discount type:
- dtPcnt = 1 – in percent
- dtSumm = 2 – absolute value

DISCOUNT FOR RECEIPT
int DiscountAdditionForReceipt(int type, double val)
Discount type:
- dtPcnt = 1 – in percent
- dtSumm = 2 – absolute value

END FISCAL RECEIPT PROCESSING
int EndFiscalReceipt(double summCash, double summCredit1, double summCredit2,
double summCredi3, double summCredi4)
int EndFiscalReceiptCurr(double summCash, double summCredit1,
double summCredit2, double summCredit3,
double summCredit4, double summCur1, double summCur2,
double summCur3)
int EndFiscalReceiptCurrEx(double summCash, double summCredit1,
double summCredit2, double summCredit3,
double summCredit4, double summCredit5,
double summCredit6, double summCredit7,
double summCredit8,
double summCur1, double summCur2,
double summCur3)

END REFUND RECEIPT PROCESSING
int GoodsReturn(double summCash, double summCredit1,
double summCredit2, double summCredit3,
double summCredit4)
int GoodsReturnCurr(double summCash, double summCredit1,
double summCredit2, double summCredit3,
double summCredit4,
double summCur1, double summCur2,
double SummCur3)
int GoodsReturnCurrEx(double summCash, double summCredit1,
double summCredit2, double summCredit3,
double summCredit4, double summCredit5,
double summCredit6, double summCredit7,
double summCredit8,
double summCur1, double summCur2,
double SummCur3)

PRINT COPY OF RECEIPT
int PrintCopyOfLastReceipt(void)

PRINT NOTIFICATION ON RECEIPT PRINTER
int PrintErrorMessage(char * message)

MODULE FISCALIZATION
int Fiscalization(char * date)

VAT CODE SETUP (JUST FOR LITHUANIAN VERSION)
int SetCompanyVATCode(char regNum)

DEPARTMENT NAME SETUP
int SetDepartName(int number, char * name)

GET A DATE FROM FISCAL MODULE
int GetFiscalData(int infoType, char data[10])

RECORD A SYSTEM IDENTIFICATION NUMBER
int SetId(char * id)

READ A SYSTEM IDENTIFICATION NUMBER
int GetIdNumber(char * date)

READ CHASSIS NUMBER
int GetUnitNumber(char * date)

PRODUCT CORRECTION IN RECEIPT

```
int ItemReturnDepart( char * name, double quantity, double price,
                      int taxNumber, char * unit, int depart,
                      double discountPercent, double discountSumm )
```

PRODUCT SALE IN RECEIPT

```
int PrintRecItemDepart( char * name, double quantity, double price,
                        int taxNumber, char * unit, int depart )
```

RECORD CHASSIS NUMBER

```
int SetShassi( char * number )
```

8. ERROR CODES

Fis-FM32/32E ERRORS	MEANING		NOTES
	DEC	HEX	
FM32_OK	0	0x00	No errors.
ERR_LENGTH	4	0x04	Invalid data packet length.
ERR_DATA	5	0x05	Invalid data in packet.
ERR_XOR	6	0x06	Not correct checksum of data packet.
ERR_ETX	7	0x07	No symbol end of packet.
ERR_ILLEGAL	16	0x10	Incorrect or unsupported command.
ERR_IDLE_STATE	17	0x11	The command can not be performed because the FM32 is in IDLE state.
ERR_NONFIS_STATE	18	0x12	The command can not be performed because the FM32 is in NONFIS state.
ERR_FIS_STATE	19	0x13	The command can not be performed because the FM32 is in FIS state.
ERR_HARD_STATE	20	0x14	Error of working with external devices. For example with printer or customer.
ERR_PARAMETERS	21	0x15	Wrong parameters in the package. Invalid parameters for commands or parameters contain invalid values.
ERR_ITEM_DESC_LENGTH	22	0x16	Parameter length in a command is out of acceptable.
ERR_ITEM_OUANTITY	23	0x17	Invalid quantity.
ERR_ITEM_PRICE	24	0x18	Invalid price.
ERR_VAT	25	0x19	Invalid tax number. Numbers are in a range from 0 to 4.
ERR_ITEM_DIM	26	0x1A	Invalid item name. Name can not contain more than 4 characters.
ERR_DEFICIENT_PAYMENT	27	0x1B	Received from the customer money less than the sum of purchase.
ERR_OVERPAYMENT_CREDIT	28	0x1C	The sum of non-cash payment exceeds the total sum on the receipt.
ERR_ITEM_DISCOUNT	29	0x1D	Invalid discount / addition on the goods. Possible causes: the percentage of discount exceeds 100%, etc.
ERR_DISCOUNT_TYPE	30	0x1E	Invalid type of discount / addition.
ERR_COMMENT_LENGTH	31	0x1F	Invalid length of comment line.
ERR_PRINTER	32	0x20	Printer error.
ERR_DISPLAY	33	0x21	Customer display error.
ERR_PRICE_AMOUNT_OVERFL OW	34	0x22	Price or amount of goods is more than 99999.99
ERR_FLASH_WRITE	35	0x23	Writing error in to fiscal memory FM32.
ERR_NOT_FISCAL	36	0x24	FM32 not fiscal. Some reports not allow.
ERR_BAD_DATE	37	0x25	Invalid date format.
ERR_REPORT_NOT_FOUND	38	0x26	In the specified date range reports are not available.
ERR_FLASH_ERASE	39	0x27	Fiscal memory FM32 is not available for deleting or writing.
ERR_DISCOUNT_RECEIPT	40	0x28	Invalid discount / addition on the goods.
ERR_NO_ITEMS	41	0x29	Discount or addition is not corresponding to a specific product.
ERR_CANT_RETURN	42	0x2A	An attempt to correct the sale of not yet sold goods.
ERR_OVER_ADDITION_PERCE NT	43	0x2B	Percent of addition is too high.
ERR_OVER_DISCOUNT_PERCE NT	44	0x2C	Percent of discount is too high.
ERR_OVER_ADDITION_FIXED	45	0x2D	Absolute addition is too high.
ERR_OVER_DISCOUNT_FIXED	46	0x2E	Absolute discount is too high.
ERR_NO_MONEY_FOR_DA	47	0x2F	It is not enough cash in the cash drawer.
ERR_ZERO_TOTAL	48	0x30	Receipt total sum = 0
ERR_PAYMENT_NOT_EQUAL	49	0x31	The amount of refund exceeds the total amount of receipts.
ERR_DEFICIENT_CASH_DRAWER	50	0x32	It is not enough cash in the cash drawer.
ERR_UNIQUE_FM32_NUMBER	51	0x33	Invalid FM32 serial number.
ERR_NOTALLOW_GOODS_RETURN	52	0x34	Goods return not allows.
ERR_ALREADYFISCAL	54	0x36	FM32 is jet fiscal.
ERR_NOT_SL_PRN	55	0x37	
ERR_RECEIPT_AMOUNT_OVER FLOW	56	0x38	Receipt total sum is more than 9900000.
ERR_FLASH_FULL	57	0x39	FM32 fiscal memory is full.
ERR_YEAR_VALUE	58	0x3A	Invalid year value in a date.
ERR_MONTH_VALUE	59	0x3B	Invalid year month in a date.
ERR_DAY_VALUE	60	0x3C	Invalid year day in a date.

ERR_JCLOUSE_STATE	63	0x3F	After Z-report is not considered a control strip.
ERR_PAY_STATE	64	0x40	
ERR_CURRENCY_NUMBER	68	0x44	Invalid currency number.
ERR_CURRENCY_RATE	69	0x45	Invalid currency exchange rate.
ERR_CURRENCY_NOT_SET	70	0x46	Currency is not setup.
ERR_CREDIT_OVERFLOW	72	0x48	The sum of non-cash payment is more than amount of receipts.
ERR_TARE_QUANTITY	73	0x49	Invalid number of tare.
ERR_CREDIT_ID	76	0x4C	Invalid ID of non-cash payment
ERR_CREDIT_DESC	77	0x4D	Invalid name of non-cash payment.
ERR_REFUND_IN_CURRENCY	78	0x4E	Money issue for returned goods is possible only in the base currency.
ERR_CREDIT_NOT_SET	79	0x4F	Non-cash payment is not defined.
ERR_VAT_AMOUNT	80	0x50	No sales with this tax.
ERR_OVER_BUF	100	0x64	Goods buffer overflow in current receipt.
ERR_DEPART_ID	128	0x80	Invalid department number.
ERR_DEPART_DESC	129	0x81	Invalid department name.
ERR_DEPART_SET	130	0x82	Error of department parameters preset.
Just for fiscal module Fis-FM32E:			
Err_NOFILE	131	0x83	File is not found.
Err_DELFILE	132	0x84	Can not to delete a file.
Err_FILE3YEAR	133	0x85	Can not to delete a file, because it is not passed three years since its formation.

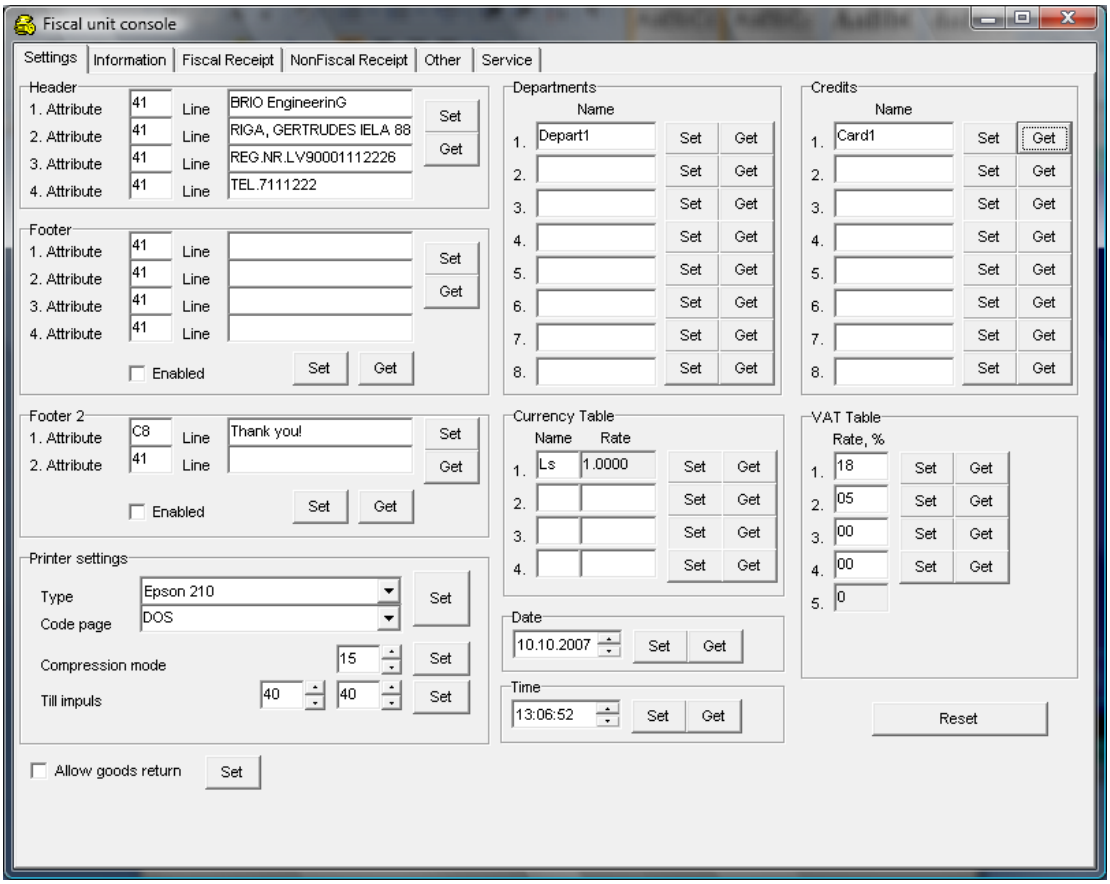
FiscalUnit.DLL ERRORS	VALUE	NOTES
ErrorDLLIllegalPacketStructure	-1	Illegal structure of the packages fields.
ErrorDLLExecuteCmd	-2	Internal error of command executes.
ErrorUnknownCommand	16	Unknown command.
ErrorDidNotSendPacket	160	Unable to sent packet.
ErrorDidNotReceivePacket	161	Unable to receive packet.
ErrorBadJournal	162	Invalid checksum of the electronic tape.

9. FISCAL MODULE PROGRAMMING

9.1. INITIAL SETUP FOR FM32

To get started with fiscal, need with a program Fiscal Console to install at least the following data:

- Receipt header (Header)
- Receipt footer (it recommend use Footer 2)
- Department (Department)
- One credit (Credits)
- One currency. First currency is base, so its Rate should be 1.00
- Taxes (VAT Table)



9.2. AVAILABLE FUNCTIONS

For the correct operation of the fiscal and the prevention of erroneous actions in various states of fiscal are available various functions FiscalUnit.dll.

Possible states of fiscal are described in paragraph " FISCAL MODULE STATES ". When a function called in a not appropriate state may appear errors with number 17,18,19 (see "Error Codes").

For return to the IDLE state or if an error come, it need to call function ResetFiscal ().

9.3. SEQUENCE OF FUNCTION LIBRARY USE

All functions, other than that correspond to receipt (fiscal and non fiscal), can use an IDLE state. For functions related to the receipt, the typical sequence of function calls can be described as follows:

- Function call “receipt header”. In this time fiscal state changed.
- Function call “receipt body”.
- Function call “receipt footer”. In this time fiscal state return in a starting state.

9.4. EXAMPLE OF FUNCTION LIBRARY USE (Delphi)

LIBRARY LOADING:

```
procedure TTraceForm.FormCreate(Sender: TObject);
begin
{You can use FiscalUnit.dll for local connect to FM32 or NetFiscalUnit.dll for using FM32 remote
}
//  if lUse_Fm32Net then
//    FisHandle := LoadLibrary('NetFiscalUnit.dll')
//  else
   FisHandle := LoadLibrary('FiscalUnit.dll');

  if FisHandle>0 then
    nFatalError:=0
  else
    nFatalError:=999;
end;
```

PRINT FISCAL RECEIPT:

```
   //Return – Flag that means a sale or return
   Function SendReceipt(lReturn:boolean):integer;
   var
      sName, sUnit:string;
      nQuant, nPrice, nCash,nCred1,nCred2,nCred3,nCred4:double;
      nVat, nTotDisc, nDisc:integer;
      PrintRecItem_proc: function(sN:string;nQuant,nPrice:double;
           nVat:integer;sUnit:string):integer;stdcall;
      BeginFiscalReceipt_proc: function():integer;stdcall;
      PrintCommentLine_proc: function(_Line : string; _attrib : integer):integer;stdcall;
      DiscountAdditionForItem_proc: function(_type : integer; _amount : double):integer;stdcall;
      DiscountAdditionForReceipt_proc: function(ntype:integer;amount:double):integer;stdcall;
      EndReceipt_proc: function(_Pay,_K1,_K2,_K3,_K4:double):integer;stdcall;
   begin
     //variables define for receipt
     sName:='Item1';
     sUnit:='Kg.';
     nQuant:=3;
     nPrice:=1.18;
     nDisc:=10;
     nTotDisc:=5;
     nVat:=0;//0..3
     nCash:=3.03;
     nCred1:=0;
     nCred2:=0;
     nCred3:=0;
     nCred4:=0;
     //Get procedure address of receipt header
     @BeginFiscalReceipt_proc := GetProcAddress(FisHandle, 'BeginFiscalReceipt');
     result:=BeginFiscalReceipt_proc();// and call
     if Result<>0 then
       exit;
     //comment
     @PrintCommentLine_proc := GetProcAddress(FisHandle, 'PrintCommentLine');
     result:=PrintCommentLine_proc('This is fiscal receipt!',65);//65-text formatting mask
     if Result<>0 then
       exit;
```

```
  //Goods in receipt.
  @PrintRecItem_proc := GetProcAddress(FisHandle, 'PrintRecItem');
  result:=PrintRecItem_proc(sName,abs(nQuant),abs(nPrice),nVat,sUnit);
  if Result<>0 then
    exit;
  {PrintRecItem with DiscountAdditionForItem can repeat one by one for each good in receipt.
  DiscountAdditionForItem use optional.
  }
  //Discount for position
  @DiscountAdditionForItem_proc := GetProcAddress(FisHandle, 'DiscountAdditionForItem');
  //First parametr – discount type. 1 – in percent
  //Negative number - discount. Positive - addition
  result:=DiscountAdditionForItem_proc(1,-1*nDisc);
  if Result<>0 then
    exit;
  //Discount for receipt. Use optional
  @DiscountAdditionForReceipt_proc :=
    GetProcAddress(FisHandle, 'DiscountAdditionForReceipt');
  // First parametr – discount type. 1 – in percent
  result:=DiscountAdditionForReceipt_proc(1,-1*nTotDisc);
  if Result<>0 then
    exit;
  //Depending on the parameter (flag) we end receipt by sale or return command
  if lReturn then
  //In return money sum must be equal to the receipt sum and ALL positions
  //goods must have a negative number
    @EndReceipt_proc := GetProcAddress(FisHandle, 'GoodsReturn')
  else
    @EndReceipt_proc := GetProcAddress(FisHandle, 'EndFiscalReceipt');
  result:=EndReceipt_proc(nCash,nCred1,nCred2,nCred3,nCred4);
end;
```

PRINT NON-FISCAL RECEIPT:

```
Function SendNonFisReceipt():integer;
var
  NonFiscalReceipt_proc: function():integer;stdcall;
  PrintNonFiscalLine_proc: function(_Line : string; _attrib : integer):integer;stdcall;
begin
  // Get procedure address of receipt header for non-fiscal receipt
  @NonFiscalReceipt_proc := GetProcAddress(FisHandle, 'BeginNonFiscalReceipt');
  result:=NonFiscalReceipt_proc();
  if Result<>0 then
    exit;
  //Print 2 lines
  @PrintNonFiscalLine_proc := GetProcAddress(FisHandle, 'PrintNonFiscalLine');
  result:=PrintNonFiscalLine_proc('Line1',65);//65-text formatting mask
  if Result<>0 then
    exit;
  result:=PrintNonFiscalLine_proc('Line2',65);//65- text formatting mask
  if Result<>0 then
    exit;
  //End non-fiscal receipt
  @NonFiscalReceipt_proc := GetProcAddress(FisHandle, 'EndNonFiscalReceipt');
  Result:=NonFiscalReceipt_proc();
  if Result<>0 then
    exit;
end;
```

LIBRARY UNLOADING:

```
FreeLibrary(FisHandle);
```

WARNING!!! Library unloading is required, because it is necessary for correct fiscal operating.

10. EU CONFORMITY DECLARATION

BRIO SRC
SCIENTIFIC RESEARCH COMPANY

EC
Declaration of Conformity

We, BRIO SRC, SIA, declare that product:

FISKĀLAIS MODULIS

Model: **BRIO-FisFM32/32E/M/U**
is in conformity with 2006/95/EC (LVD Directive)

For the evaluation of the compliance with this Directive, the following standards or standardized documents were applied:

EN 60950-1:2006+ A11:2009 + A1:2010 + A12:2011
- Information technology equipments – Safety

Person responsible for making this declaration

Name, Surname: Oleg Khalatov

Position/ Title: Director

(Signature)

CE

LATVIA, RIGA Oct/22/2012

BRIO SRC, SIA. Gertrudes 88, Riga, LV-1009, Latvia. Ph.: (+371) 67365828 Fax: +(371) 67365845 www.brio.com.lv