

Confidential

EPSON

LOW LEVEL DRIVER

SPECIFICATION MANUAL

FISCAL PRINTERS

STANDARD	
Rev. Num.	1.4
Notes	

Copy date	9/26/2006
Copy by	

EPSON AMERICA INC.
SEIKO EPSON CORPORATION




Confidential

REVISION SHEET

Rev.	SHEET	CHANGES
1.0	All	Initial revision. Library version 1.00
1.1	All	Function parameters bugs fixed. Library version 1.02.
1.2	All	Change in GetAPIVersion() function definition. Library version 1.03.
1.3	All	Library “.LIB” for Win32 added. Library version 1.03.
1.06	All	Change in documentation nomenclature for revision numbers. Time-out problems fixed for RS232-USB adapters Library version 1.06.
1.4	7 All	Change in C++ example for OpenSerialPort() function. Bug fixed when receive a full data buffer. Library version 1.4.0.0
TITLE		
LOW LEVEL DRIVER		
SPECIFICATION MANUAL		
FISCAL PRINTERS		

Confidential

Nomenclature used in this document

Symbol	Description
	Indicates the paragraph references other topics included in this document.
	Indicates how is possible to use the library.
	Indicates the text will describe the control point for a correct use of the library.

Nomenclature for Win 32 library

Name	Description
<Library_name>.dll	Win 32 library as Dynamic Link Library.

Nomenclature for .LIB Win 32 library

Name	Description
< Library_name >.lib	Static library for Win 32.

Nomenclature for Linux library

Name	Description
< Library_name >.a	Library name with "a" extension indicates a static library
< Library_name >.so	Library name with "so" extension indicates a dynamic library

Nomenclature for DOS driver

Nombre	Significado
< Library_name ><S>.lib	Library name with "S" indicates that the library must be compiled using Small compilation.
< Library_name ><L>.lib	Library name with "L" indicates that the library must be compiled using Large compilation.

Confidential

TABLE OF CONTENTS

Chapter 1 - Introduction	1
Chapter 2 - Library Functions	2
2.1 Serial Port management functions	5
2.1.1 OpenSerialPort.....	6
2.1.2 CloseSerialPort	8
2.1.3 SendSerialMessage	10
2.1.4 ResendSerialMessage	12
2.2 Output buffer management functions	14
2.2.1 AddMessageField	15
2.2.2 AddMessageFieldEx	19
2.2.3 PurgeMessage	24
2.3 Input buffer management functions	25
2.3.1 Win 32 and Linux environment	25
2.3.2 DOS environment.....	26
2.3.3 GetMessageField	27
2.3.4 GetMessageFieldEx.....	31
2.3.5 GetFieldCounter	35
2.4 Status functions	37
2.4.1 GetCommunicationState	38
2.4.2 GetLastCommError	40
2.5 General Information functions	42
2.5.1 GetApiVersion	43
2.5.2 GetSentFrame.....	45
2.5.3 GetReceivedFrame	47

Chapter 1 - Introduction

The low level driver developed by EPSON manages the communication protocol between EPSON fiscal printers and host applications. This library will guaranty the communication and will speed up development times.

This library has passed several test under the environments supported.

One of the objectives of this library is to be portable to be compatible with different operating systems. This version supports the following operating systems:

- ✓ Win 32 – Dynamic Link Library (DLL)
- ✓ Win 32 – Static Library (Lib)
- ✓ Dos – Small and Large Library (Lib)
- ✓ Linux – Static and Dynamic Library

For Win32 and Linux environment, the library has an internal “Multithread” operation; this means the functions will not block the application flow that uses the library allowing other tasks to be performed. Fiscal Printer and library can make background operations.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 1	SHEET 1

Chapter 2 - Library Functions

Library functions are defined using the following prototypes:

DWORD **Function_Name** (...)
VOID **Function_Name** (...)
BYTE * **Function_Name** (...)

The following table describes the data types used for parameters and returns and their size.

Type	Description	C/C++ declaration example
byte	Variable, 8 bits	unsigned char
word	Variable, 16 bits	unsigned short
dword	Variable, 32 bits	unsigned int
bool	0 (FALSE) or ≠ 0 (TRUE)	int/bool

Possible error codes returned by functions are listed in the following table. These errors are separated into categories to provide an easy identification.

Constant	Value (Hex)	Description
EFPROT_SUCCESS	0 (0000)	Success.
Errors from 1 to 5: Error in Serial Port use		
EFPROT_PORT_ALREADY_OPEN	1 (0001)	Serial port already open.
EFPROT_PORT_IN_USE	2 (0002)	Serial port used by another application.
EFPROT_PORT_NOT_OPEN	3 (0003)	Serial port is closed.
EFPROT_INVALID_COM	4 (0004)	Serial port invalid or doesn't exist.
EFPROT_INVALID_BAUD_RATE	5 (0005)	Invalid or non-supported baud rate.
Errors from 6 to 8: Error in data transmission		
EFPROT_TIMEOUT_ERROR	6 (0006)	Communication time-out.
EFPROT_SEND_ERROR	7 (0007)	Error while send data.
EFPROT_OFF_OR_DISCONNECTED	8 (0008)	Printer off-line or disconnected.
Error 9: Error in frame preparation manager		
EFPROT_BUFFER_OVERFLOW	9 (0009)	Buffer overflow.

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 2	SHEET 2

Confidential

Errors from 10 to 13: Errors in buffer add/retrieve operations.

EFPROT_FIELD_INVALID	10 (000A)	Invalid format in send data field.
EFPROT_ASWR_INVALID	11 (000B)	Answer field invalid.
EFPROT_ASWR_FIELD_INEXISTENT	12 (000C)	Answer field doesn't exist.
EFPROT_FIELD_INVALID_TYPE	13 (000D)	Answer field type invalid.

Error 14: Internal error

EFPROT_INVALID_PARAM	14 (000E)	Invalid parameter.
----------------------	-----------	--------------------

Error 15: Internal error

EFPROT_INTERNAL_ERROR	15 (000F)	Library internal error.
-----------------------	-----------	-------------------------

The variable types used by functions that add or retrieve data from the buffer are defined as follows:

Constant	Value (Hex)	Description
Constant used in “add” type functions		
EFPROT_TYPE_BINARY	1 (0001)	All values are in binary format.
EFPROT_TYPE_ASCII	2 (0002)	All values are in ASCII format.
Constants used to retrieve data with specific data types		
EFPROT_TYPE_UCHAR	0 (0000)	Unsigned numeric field, 8 bits (0 to 255)
EFPROT_TYPE_CHAR	1 (0001)	Numeric field, 8 bits (-128 to +127)
EFPROT_TYPE_USHORT	2 (0002)	Unsigned numeric field, 16 bits (0 to 65.535)
EFPROT_TYPE_SHORT	3 (0003)	Numeric field, 16 bits (-32.768 to 32.767)
EFPROT_TYPE_UINT	4 (0004)	Unsigned numeric field, 32 bits (0 to 4.294.967.295)
EFPROT_TYPE_INT	5 (0005)	Numeric field, 32 bits (-2.147.483.648 to 2.148.483.647)
EFPROT_TYPE_ULONG	6 (0006)	Unsigned numeric field, 32 bits (0 a 4.294.967.295)
EFPROT_TYPE_LONG	7 (0007)	Numeric field, 32 bits (-2.147.483.648 a 2.148.483.647)
EFPROT_TYPE_STRING	8 (0008)	Array of characters.
Constants used to get data from answer buffer		
EFPROT_PRT_ANSWER	1 (0001)	Printer status after command execution.
EFPROT_FSC_ANSWER	2 (0002)	Fiscal status after command exucution.

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 3	SHEET 3

Confidential

EFPROT_RSV_ANSWER_1	3 (0003)	Reserved.
EFPROT_CMD_ANSWER	4 (0004)	Command answer after command execution.
EFPROT_RSV_ANSWER_2	5 (0005)	Reserved.
EFPROT_ASW_FIELD_XX	XX (XXXX)	Answer fields.

Constants used in library management functions

EFPROT_CLOSED	0 (0000)	Communication closed.
EFPROT_IDLE	1 (0001)	Library is idle.
EFPROT_BUSY	2 (0002)	Library is working in background.



Constants EFPROT_ASW_FIELD_XX indicate the answers fields that can be received after a command execution. This XX values can go from 6 to n

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 4	SHEET 4

2.1 Serial Port management functions

Functions described in this section are used to configure and manage serial port operations. It also includes functions to send frames with commands to fiscal printers after frames are prepared

Function	Description
OpenSerialPort	Opens serial port by allocating hardware and software resources.
CloseSerialPort	Closes the serial port and free resources allocated by Function OpenSerialPort .
SendSerialMessage	Send a frame to fiscal printer.
ResendSerialMessage	Re-send the latest frame to fiscal printer.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 5	SHEET 5

Confidential

2.1.1 OpenSerialPort

This is the first functions that must be called to get communication between fiscal printer and application. It initializes the serial port if all resources are available.

Syntax:

```
DWORD OpenSerialPort( WORD wCommNumber, DWORD dwBaudRate );
```

Input:

wCommNumber	Serial port COM number. Only 1 to 4 are allowed
dwBaudRate	Serial port baud rate. (9600, 19200 or 38400).

Output:

None.

Possible returns:

EFPROT_SUCCESS
EFPROT_PORT_ALREADY_OPEN
EFPROT_PORT_IN_USE
EFPROT_INVALID_COM
EFPROT_INVALID_BAUD_RATE
EFPROT_INTERNAL_ERROR

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 6	SHEET 6

Confidential

C declaration:



```
typedef unsigned int  dword;
typedef unsigned short word;
```

```
dword OpenSerialPort( word wCommNumber, dword dwBaudRate ); //Prototype.
```

```
# define COM1                1
# define BAUD_RATE           9600
```

```
dword dwReturn; //Variable declaration.
dwReturn = OpenSerialPort( COM1, BAUD_RATE ); //Opens COM1 at 9600
```

Visual Basic declaration:



```
Private Declare Function OpenSerialPort Lib "EpsonFiscalProtocol.dll" (ByVal iSerialPort As Integer, ByVal iBaudRate As Long) As Long
```

Visual C++ / C++ Builder declaration:



```
typedef int (*LPOPENPORT)( WORD, DWORD );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");
LPOPENPORT pOpenPort = (LPOPENPORT) GetProcAddress( HandleLib, "OpenSerialPort");
```

Warning:



Variables **HandleLib** and **pOpenPort** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 7	SHEET 7

Confidential

2.1.2 CloseSerialPort

This function closes and releases serial port resources.

Syntax:

```
dword CloseSerialPort( void );
```

Input:

None.

Output:

None.

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_NOT_OPEN
```

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned int dword;
```

```
dword CloseSerialPort( void ); //Prototype.
```

```
dword dwReturn; //Variable declaration.
```

```
dwReturn = CloseSerialPort (); //Close serial port
```

Visual Basic declaration:



```
Private Declare Function CloseSerialPort Lib "EpsonFiscalProtocol.dll" () As Long
```

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 8	SHEET 8

Confidential

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pClosePort = GetProcAddress( HandleLib, "CloseSerialPort");
```

Warning:



Variables **HandleLib** and **pClosePort** must be different from NULL to be used properly.

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 9	SHEET 9

Confidential

2.1.3 SendSerialMessage

This function sends a frame to serial printer. The frame must be ready before use this function. For Win32 and Linux environment this function operates in asynchronous mode and it will return the control to the application while it executes the operation; for DOS it operates in synchronous mode and will return control only after finish the execution.

Syntax:

```
dword SendSerialMessage( void );
```

Input:

None.

Output:

None.

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_INTERNAL_ERROR
```

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned int dword;
```

```
dword SendSerialMessage( void ); //Prototype.
```

```
dword dwReturn; //Variable declaration.
```

```
dwReturn = SendSerialMessage(); //Send a frame.
```

Visual Basic declaration:



```
Private Declare Function SendSerialMessage Lib "EpsonFiscalProtocol.dll" () As Long
```

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 10	SHEET 10

Confidential

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pSendMessage = GetProcAddress( HandleLib, "SendSerialMessage");
```

Warning:



Variables **HandleLib** and **pSendMessage** must be different from NULL to be used properly.

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 11	SHEET 11

Confidential

2.1.4 ResendSerialMessage

This function provides a “retry” operation if sending fails at first attempt. It operates in the same way as SendSerialMessage() function

Syntax:

```
dword ResendSerialMessage( void );
```

Input:

None.

Output:

None.

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_INTERNAL_ERROR
```

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned int dword;
```

```
dword ResendSerialMessage( void ); //Prototype.
```

```
dword dwReturn; //Variable declaration.
```

```
dwReturn = ResendSerialMessage (); //Re-send last frame
```

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 12	SHEET 12

Confidential

Visual Basic declaration:



Private Declare Function ResendSerialMessage Lib "EpsonFiscalProtocol.dll" () As Long

Visual C++ / C++ Builder declaration:



HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");
FARPROC pResendMessage = GetProcAddress(HandleLib, "ResendSerialMessage");

Warning:



Variables **HandleLib** and **pResendMessage** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 13	SHEET 13

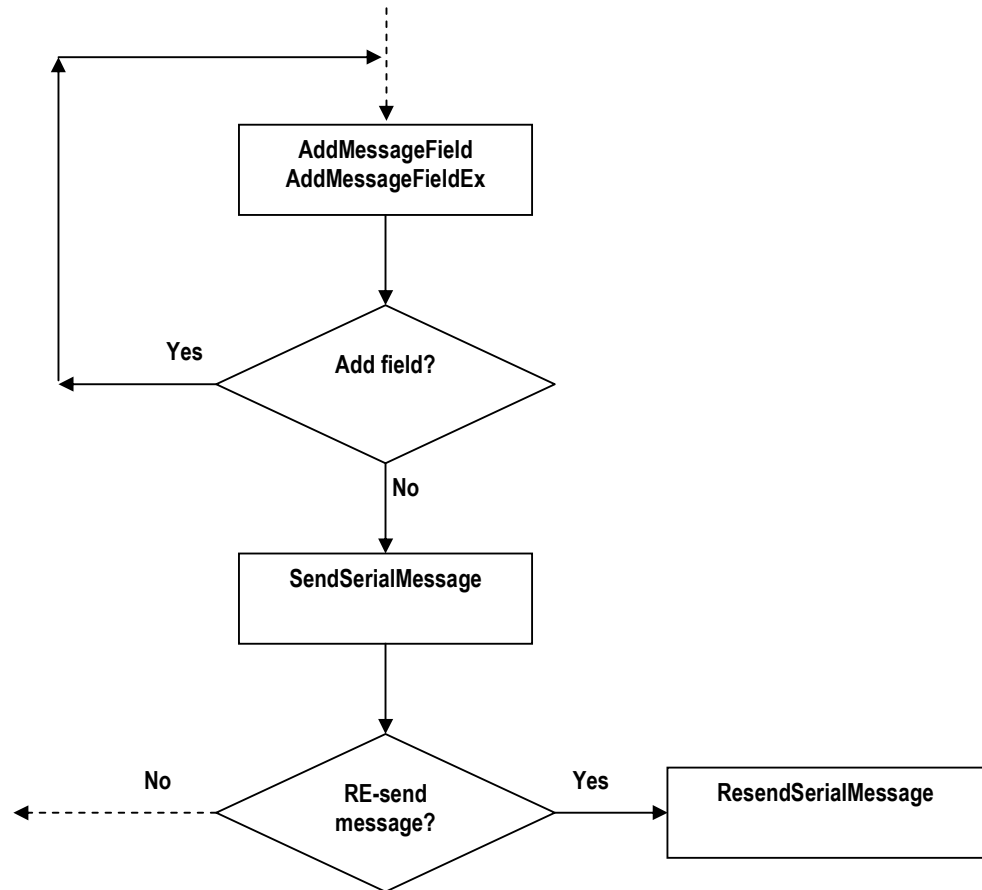
2.2 Output buffer management functions

Functions described below are used to manage the output buffer before send the frame to the printer using the functions in previous section. Functions are the following:

Function	Description
AddMessageField	Add binary data format to the output buffer
AddMessageFieldEx	Add data to output buffer. Data format can be specified.
PurgeMessage	Purge output buffer.



Following data flow shows how to use these frame management functions together with **SendSerialMessage** and **ResendSerialMessage** functions in order to follow the communication protocol of Epson Fiscal Printers.



EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 14	SHEET 14

Confidential

2.2.1 AddMessageField

This function adds data to the output buffer in binary format. Each field is separated automatically by the function following the Fiscal Printer communication protocol.

Syntax:

```
dword AddMessageField( byte * szField, dword dwFieldLength);
```

Input:

szField	Field that will be added to the frame.
dwFieldLength	Size of szField .

Output:

None.

Possible returns:

EFPROT_SUCCESS	
EFPROT_BUFFER_OVERFLOW	
EFPROT_FIELD_INVALID	Invalid format in szField .
EFPROT_INVALID_PARAM	szField parameter can not be null.

Compatibility:



☒ Win 32 ☒ Dos ☒ Linux

C declaration:



```
dword AddMessageField( byte * szField, dword dwFieldLength); //Prototype.
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 15	SHEET 15

Confidential

Visual Basic declaration:



Private Declare Function AddMessageField Lib "EpsonFiscalProtocol.dll" (ByVal szField As String, ByVal IFieldLength As Long) As Long

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPAMSG)( BYTE *, DWORD );  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
LPAMSG pAddMsgField = (LPAMSG) GetProcAddress( HandleLib, "AddMessageField");
```

Warning:



Variables **HandleLib** and **pAddMsgField** must be different from NULL to operate functions properly.

This function will interpretate the data in **szField** parameter as binary data, this means it will use the equivalent ASCII code for each character. Example: if **szField** value is "0802", this function will add 30h, 38h, 30h and 32h into the output buffer (not 08h, 02h).

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 16	SHEET 16

Confidential

Visual C++ Example:

```
typedef int ( * LPAMSG )( BYTE *,DWORD );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");

void AddMessageField( void )
{
    LPAMSG      fAddMessageField;
    unsigned char szDados[2];

    if( HandleLib == NULL )
    {
        /* Error at DLL open. Should show error and return */
        return;
    }

    fAddMessageField = (LPAMSG) GetProcAddress( HandleLib,"AddMessageField");

    if( fAddMessageField == NULL )
    {
        /* Error reading function from DLL. Should show error and return */
        return;
    }

    /* Add data 0802h to the output buffer */
    szDados[0] = 0x08;
    szDados[1] = 0x02;
    if( fAddMessageField( szDados, 0x02 ) != 0 )
    {
        /* Error while add data. Should show error and return */
        return;
    }
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 17	SHEET 17

Confidential

C Example:

```
typedef unsigned int  dword;
typedef unsigned char bool;
typedef unsigned char byte;

//Prototype
dword AddMessageField( byte *uchField, dword iFieldLength );

int main( void )
{

//Variable declaration
dword dwReturn;
byte bField[] = { 0x08, 0x02 }; //Init bField


//Add data to output buffer
dwReturn = AddMessageField( bField, 2 );

//Test dwReturn
if( dwReturn != 0 )
{
    //Error while add data. Should show error and return.
    return;
}

}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 18	SHEET 18

Confidential

2.2.2 AddMessageFieldEx

This functions adds data to the output buffer using the specified format.

Syntax:

```
dword AddMessageFieldEx( byte * szField, dword dwLength, dword dwType, bool bSeparator );
```

Input:

szField	Field that will be added to the frame.
dwLength	Size of szField .
dwType	Field data type. It can be EFPROT_TYPE_BINARY or EFPROT_TYPE_ASCII .
bSeparator	Indicates if a field separator will be added to the end of the field.

Output:

None.

Possible returns:

EFPROT_SUCCESS	
EFPROT_BUFFER_OVERFLOW	
EFPROT_FIELD_INVALID	Invalid format in szField .
EFPROT_FIELD_INVALID_TYPE	dwType parameter is invalid.
EFPROT_INVALID_PARAM	szField parameter can not be NULL.

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 19	SHEET 19

Confidential

C declaration:



```
dword AddMessageFieldEx( byte * szField, dword dwLength, dword dwType, bool bSeparator
);
```

Visual Basic declaration:



```
Private Declare Function AddMessageFieldEx Lib "EpsonFiscalProtocol.dll" (ByVal szField As
String, ByVal IFieldLength As Long, ByVal IFieldType As Long, ByVal bSeparator As Boolean)
As Long
```

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPAMSGEX )( BYTE *, DWORD );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");
LPAMSGEX pAddMsgFieldEx = (LPAMSG) GetProcAddress( HandleLib,
"AddMessageFieldEx");
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 20	SHEET 20

Confidential

Warning:

Variables **HandleLib** and **pAddMsgFieldEx** must be different from NULL to be used properly.

Function behaviour is different from the previous one in the the following way:

If is necessary to add the 0802h value, is posible to call **AddMessageFieldEx** function with paramter **szField** equal to "0802" and **dwType** equal to **EFPROT_TYPE_BINARY**. In this way, the DLL will save 08h 02h to output buffer. If **dwType** is equal to **EFPROT_TYPE_ASCII**, this function behaviour will be the same as **AddMessageField**.

To add data in binary format, it should be considered that data size must be multiply of two; if not, an error will be returned.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 21	SHEET 21

Confidential

Visual C++ example:

```
typedef int ( * LPAMSGEX )( BYTE *,DWORD, DWORD, BOOL );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");

void AddMessageField( void )
{
    LPAMSGEX    fAddMessageFieldEx;
    unsigned char szDados[5];

    if( HandleLib == NULL )
    {
        /* Error at DLL open. Should show error and return */
        return;
    }

    fAddMessageFieldEx = (LPAMSGEX) GetProcAddress( HandleLib,"AddMessageFieldEx");

    if( fAddMessageFieldEx == NULL )
    {
        /* Error reading function from DLL. Should show error and return */
        return;
    }

    /* Add data 0802h to the output buffer */
    strcpy( (char *) szDados, "0802");
    if(fAddMessageFieldEx(szDados,strlen(szDados),EFPROT_TYPE_BINARY,true ) != 0 )
    {
        /* Error while add data. Should show error and return */
        return;
    }
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 22	SHEET 22

Confidential

C example:

```
typedef unsigned int  dword;
typedef unsigned char bool;
typedef unsigned char byte;

dword AddMessageFieldEx( byte * szField, dword dwLength, dword dwType, bool bSeparator );
//Prototype.

#define FALSE          0
#define TRUE           1
#define EFPROT_TYPE_BINARY 1

int main( void )
{
    //Variable declaration
    dword dwReturn;
    byte bField[] = "0802"; //Inits szField

    //Adds a message in binary format (08h 02h) with field separator.
    dwReturn = AddMessageFieldEx( bField, 4, EFPROT_TYPE_BINARY, TRUE );

    //Tests the return
    if( dwReturn != 0 )
    {
        // Error while add data. Should show error and return.
        return;
    }
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 23	SHEET 23

Confidential

2.2.3 PurgeMessage

Purge all data that are in output data buffer.

Syntax:

```
void PurgeMessage( void );
```

Input:

None.

Output:

None.

Possible returns:

None.

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
void PurgeMessage( void ); //Prototye
```

```
PurgeMessage();
```

Visual Basic declaration:



```
Private Declare Sub PurgeMessage Lib "EpsonFiscalProtocol.dll" ()
```

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pAddMsgField = (FARPROC) GetProcAddress( HandleLib, "PurgeMessage");
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 24	SHEET 24

2.3 Input buffer management functions

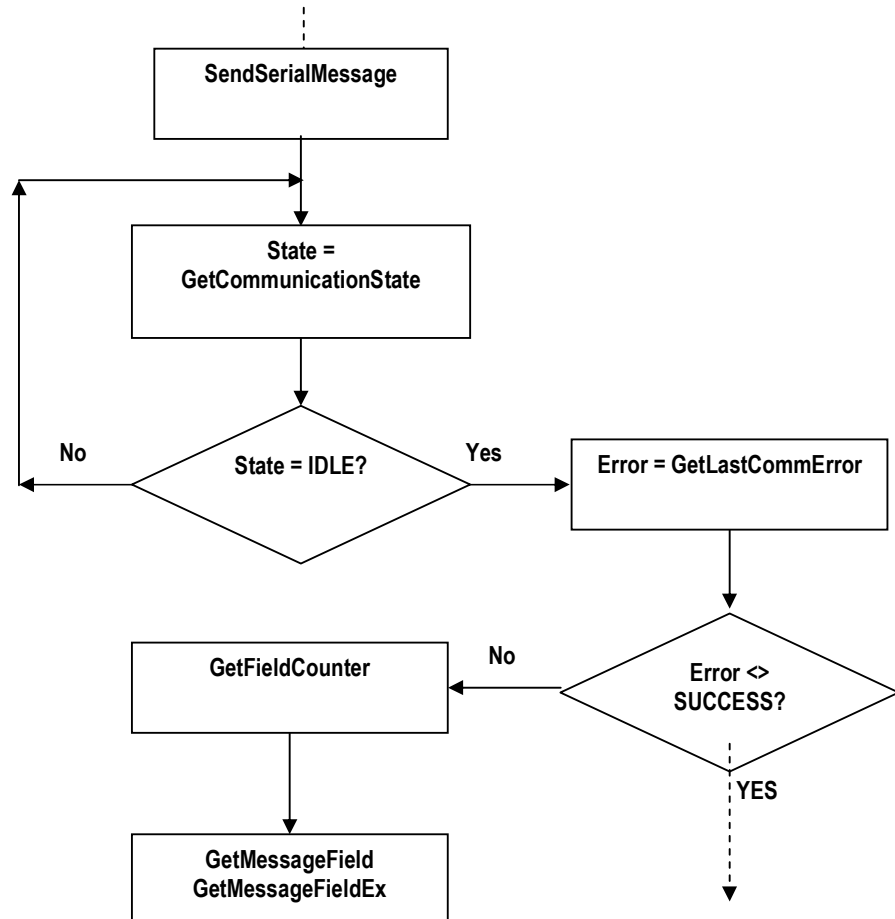
Functions in this section will allow the application to get answer data from input buffer. This buffer is filled automatically after a frame is sent by DLL using the functions mentioned in section 2.1. Functions are the following:

Function	Description
GetMessageField	Returns the value of an answer field in binary format.
GetMessageFieldEx	Returns the value of an answer field in the specified format.
GetFieldCounter	Return the number of fields in received in the input buffer.

2.3.1 Win 32 and Linux environment




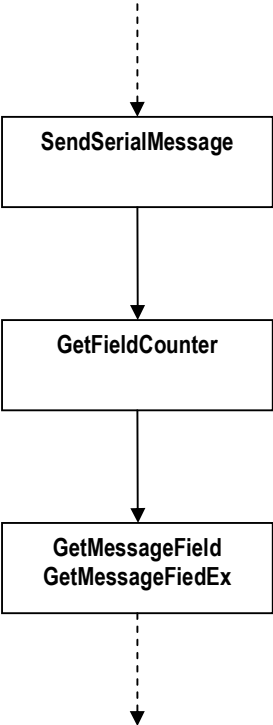
The following flow diagram shows how to use these functions together with functions described in section 2.1.



EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 25	SHEET 25

2.3.2 DOS environment

 This is the basic flow diagram to use these functions.



EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 26	SHEET 26

Confidential

2.3.3 GetMessageField

This function gets a specific field from the input buffer. All values returned from this function are in binary format.

Syntax:

```
DWORD GetMessageField( byte * szField, DWORD * dwFieldLength, DWORD dwAnswerField);
```

Input:

dwAnswerField	Field number to be retrieved.
----------------------	-------------------------------

Output:

szField	Field content. Empty in case of error.
dwFieldLength	Size of szField .

Possible returns:

EFPROT_SUCCESS	
EFPROT_ASWR_INVALID	Invalid answer field.
EFPROT_ASWR_FIELD_INEXISTENT	Answer field does not exist.
EFPROT_FIELD_INVALID_TYPE	
EFPROT_INTERNAL_ERROR	
EFPROT_INVALID_PARAM	szField , dwFieldLength can't be NULL

Compatibility:

☒ Win 32☒ Dos☒ Linux

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 27	SHEET 27

Confidential

C declaration:



```
typedef unsigned int  dword;  
typedef unsigned char byte;
```

//Prototype.

```
dword GetMessageField( byte * szField, dword * dwFieldLength, dword dwAnswerField);
```

```
# define EFPROT_PRT_ANSWER          1
```

```
dword dwReturn, dwLen; //Variable declaration.  
byte bAnswer[5];
```

// Gets answer field

```
dwReturn = GetMessageField ( bAnswer, &dwLen, EFPROT_PRT_ANSWER );
```

Visual Basic declaration:



```
Private Declare Function GetMessageField Lib "EpsonFiscalProtocol.dll" (ByVal szField As  
String, ByRef lFieldLength As Long, ByVal lAnswerField As Long) As Long
```

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPGMSG)( BYTE *, DWORD *, DWORD );  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
LPGMSG pGetMsgField = (LPGMSG) GetProcAddress( HandleLib, "GetMessageField");
```

Warning:



Variables **HandleLib** and **pGetMsgField** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 28	SHEET 28

Confidential

Visual C++ example:

```
typedef int ( * LPGMSG )( BYTE *,DWORD *, DWORD );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");

void GetMessageField( void )
{
    LPGMSG      fGetMessageField;
    unsigned char szDados[2];
    int         iFieldLength;

    if( HandleLib == NULL )
    {
        /* Error opening DLL.*/
        return;
    }

    fGetMessageField = (LPGMSG) GetProcAddress( HandleLib,"GetMessageField");

    if( fGetMessageField == NULL )
    {
        /* Error getting DLL function from library*/
        return;
    }

    /* Reads "Printer Answer" from input buffer*/
    if( fGetMessageField( szDados, &iFieldLength, EFPROT_PRT_ANSWER ) != 0 )
    {
        /* Error getting field. */
        return;
    }

    /* SUCCESS */
    ...
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 29	SHEET 29

Confidential

C example:

```
typedef unsigned int  dword;
typedef unsigned char bool;
typedef unsigned char byte;

//Prototype
dword GetMessageField( byte * szField, dword * dwFieldLength, dword dwAnswerField);

#define EFPROT_PRT_ANSWER      1

int main( void )
{
    //Variable declaration
    dword dwReturn, dwLen;
    byte bField[5];

    //Get "Printer Answer" from input buffer
    dwReturn = GetMessageField( bAnswer, 4, &dwLen, EFPROT_PRT_ANSWER );

    //Tests return
    if( dwReturn != 0 )
    {
        //Error getting value.
        return;
    }
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 30	SHEET 30

Confidential

2.3.4 GetMessageFieldEx

This function gets a field in a specific format from the input buffer. Field data type can be choosed using **dwFieldType**.

Syntax:

```
DWORD GetMessageFieldEx( void * vField, DWORD * dwLength, DWORD dwField, DWORD dwType);
```

Input:

dwField	Field number to be retrieved.
dwType	Data type of vField .

Output:

vField	Field content. Empty is case of error.
dwLength	Size of szField .

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_ASWR_INVALID  
EFPROT_ASWR_FIELD_INEXISTENT  
EFPROT_FIELD_INVALID_TYPE  
EFPROT_INTERNAL_ERROR  
EFPROT_INVALID_PARAM
```

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 31	SHEET 31

Confidential

C declaration:



```
typedef unsigned int  dword;  
typedef unsigned char byte;
```

//Prototype.

```
dword GetMessageFieldEx( void * vField, dword * dwLength, dword dwField, dword dwType);
```

```
# define EFPROT_PRT_ANSWER          1  
# define EFPROT_TYPE_STRING        8
```

```
dword dwReturn, dwLen; //Variable declaration.  
byte bAnswer[ 5 ]; .
```

// Gets the field 1 from input buffer.

```
dwReturn = GetMessageFieldEx( bAnswer, &dwLen, EFPROT_PRT_ANSWER,  
EFPROT_TYPE_STRING );
```

Visual Basic declaration:



```
Private Declare Function GetMessageFieldEx Lib "EpsonFiscalProtocol.dll" (ByRef iField As  
Long, ByRef iFieldLength As Long, ByVal lAnswerField As Long, ByVal iType As Long ) As  
Long
```

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPGMSGEX)( VOID *, DWORD *, DWORD, DWORD );  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
LPGMSGEX pGetMsgFieldEx = (LPGMSGEX) GetProcAddress( HandleLib,  
"GetMessageFieldEx");
```

Warning:



Variables **HandleLib** and **pGetMsgFieldEx** must be different from NULL to be used properly.

Due different data type this function can return in Visual Basic, it might be necessary to declare the function several times, one for each data type.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 32	SHEET 32

Confidential

Visual C++ example:

```
typedef int ( * LPGMSGEX )( VOID *,DWORD *, DWORD, DWORD );
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");

void GetMessageFieldEx( void )
{
    LPGMSGEX    fGetMessageFieldEx;
    unsigned short sDados;
    int         iFieldLength;

    if( HandleLib == NULL )
    {
        /* Error opening DLL.*/
        return;
    }

    fGetMessageFieldEx = (LPGMSGEX) GetProcAddress( HandleLib,"GetMessageFieldEx");

    if( fGetMessageFieldEx == NULL )
    {
        /* Error getting function from library. */
        return;
    }

    /* Gets "Printer Answer" field from input buffer. */
    if( fGetMessageFieldEx( sDados, &iFieldLength, EFPROT_PRT_ANSWER,
                          EFPROT_TYPE_USHORT ) != 0 )
    {
        /* Error getting field */
        return;
    }

    /* SUCCESS */
    ...
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 33	SHEET 33

Confidential

C example:

```
typedef unsigned int  dword;
typedef unsigned char byte;

//prototipo
dword GetMessageFieldEx( void * vField, dword * dwLength, dword dField, dword dwType);

#define EFPROT_PRT_ANSWER      1
#define EFPROT_TYPE_STRING    8

int main( void )
{
    //Variable declaration
    dword dwReturn, dwLen;
    byte bAnswer[5];

    //Gets "Printer Answer" field from input buffer
    dwReturn = GetMessageFieldEx( bAnswer, &dwLen, EFPROT_PRT_ANSWER,
    EFPROT_TYPE_STRING );

    //Tests return
    if( dwReturn != 0 )
    {
        //Error.
        return;
    }
}
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 34	SHEET 34

Confidential

2.3.5 GetFieldCounter

This function returns the number of retrieve fields that are in the input buffer.

Syntax:

```
dword GetMessageField( void );
```

Input:

None.

Output:

None.

Possible returns:

Number of answer fields.

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned int dword;
```

```
dword GetMessageField( void ); //Prototype
```

```
dword dwReturn;
```

```
dwReturn = GetFieldCounter();
```

Visual Basic declaration:



```
Private Declare Function GetFieldCounter Lib "EpsonFiscalProtocol.dll" () As Long
```

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 35	SHEET 35

Confidential

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pFieldCounter = GetProcAddress( HandleLib, "GetFieldCounter");
```

Warning:



Variables **HandleLib** y **pGetFieldCounter** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 36	SHEET 36

2.4 Status functions

These functions are used to get the current status of the library as well as the latest error. These functions should be called before send or retrieve data in order to guaranty the library will get communication with the Fiscal Printer.

Function	Description
GetCommunicationState	Returns current status of communication process.
GetLastCommError	Returns the latest error after it happens in a communication.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 37	SHEET 37

Confidential

2.4.1 GetCommunicationState

This function returns the current status of communication.

Syntax:

```
dword GetCommunicationState( void );
```

Input:

None.

Output:

None.

Possible returns:

```
EFPROT_CLOSED  
EFPROT_IDLE  
EFPROT_BUSY
```

Compatibility:



☒ Win 32

☒ Linux

C declaration:



```
typedef unsigned int dword;
```

```
dword GetCommunicationState( void ); //Prototype
```

```
dword dwReturn;
```

```
dwReturn = GetCommunicationState ();
```

Visual Basic declaration:



Private Declare Function GetCommunicationState Lib "EpsonFiscalProtocol.dll" () As Long

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 38	SHEET 38

Confidential

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pCommState = GetProcAddress( HandleLib, "GetCommunicationState");
```

Warning:



Variables **HandleLib** and **pGetCommStatus** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 39	SHEET 39

Confidential

2.4.2 GetLastCommError

This function returns the latest communication status.

Syntax:

```
dword GetLastCommError( void );
```

Input:

None.

Output:

None.

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_TIMEOUT_ERROR  
EFPROT_SEND_ERROR  
EFPROT_OFF_OR_DISCONNECTED
```

Compatibility:

☒ Win 32

☒ Linux

C declaration:

```
typedef unsigned int dword;
```

```
dword GetLastCommError( void ); //Prototype
```

```
dword dwReturn;
```

```
dwReturn = GetLastCommError();
```

Visual Basic declaration:

Private Declare Function GetLastCommError Lib "EpsonFiscalProtocol.dll" () As Long

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 40	SHEET 40

Confidential

Visual C++ / C++ Builder declaration:



```
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pCommError = GetProcAddress( HandleLib, "GetLastCommError");
```

Warning:



Variables **HandleLib** and **pCommError** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 41	SHEET 41

2.5 General Information functions

These functions are used to get library information.

Function	Description
GetApiVersion	Returns current library version.
GetSentFrame	Gets the latest frame that was sent to Fiscal Printer.
GetReceivedFrame	Return the latest frame received from Fiscal Printer.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 42	SHEET 42

Confidential

2.5.1 GetApiVersion

This function returns current library version.

Syntax:

```
void GetApiVersion(Byte * szVersion);
```

Input:

None.

Output:

None.

Possible returns:

Library version in format XX.YY.

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned char byte;
```

```
void GetApiVersion( byte * szVersion ); //Prototype
```

Visual Basic declaration:



```
Private Declare Sub GetApiVersion Lib "EpsonFiscalProtocol.dll" (szVersion As String)
```

Visual C++ / C++ Builder declaration:



```
typedef void (*FARPROC)( BYTE *);  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
FARPROC pApiVersion = (FARPROC) GetProcAddress( HandleLib, "GetApiVersion");
```

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 43	SHEET 43

Warning:



Variables **HandleLib** and **pApiVersion** must be different from NULL to be used properly.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 44	SHEET 44

Confidential

2.5.2 GetSentFrame

This functions allows to get the latest frame that was sent to Fiscal Printer.

Syntax:

```
dword GetSentFrame( byte * szBuffer, dword * dwBufferLength );
```

Input:

None.

Output:

szBuffer	Latest frame.
dwBufferLength	Size of szBuffer .

Possible returns:

```
EFPROT_SUCCESS  
EFPROT_INTERNAL_ERROR
```

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

C declaration:



```
typedef unsigned int  dword;  
typedef unsigned char byte;
```

```
dword GetSentFrame( byte * szBuffer, dword * dwBufferLength ); //Prototype.
```

```
dword dwReturn, dwLen; //Variables declaration.  
byte bBuffer[2048];
```

```
dwReturn = GetSentFrame( bBuffer, &dwLen); //Gets latest frame.
```

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 45	SHEET 45

Confidential

Visual Basic declaration:



Private Declare Function GetSentFrame Lib "EpsonFiscalProtocol.dll" (ByVal szBuffer As String, ByRef iBufferLength As Long) As Long

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPGSENTF)( BYTE *, DWORD * );  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
LPGSENTF pGetSentFrame = (LPGSENTF) GetProcAddress( HandleLib, "GetSentBuffer");
```

Warning:



Variables **HandleLib** y **pGetSentFrame** must be different from NULL to be used properly.

Size of **szBuffer** must be enough to support the entire frame that will return this function.
Maximum size could be 4102 bytes.

	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 46	SHEET 46

Confidential

2.5.3 GetReceivedFrame

This function returns the latest frame received by the library from the Fiscal Printer.

Syntax:

```
DWORD GetReceivedFrame( byte * szBuffer, DWORD * dwBufferLength );
```

Input:

None.

Output:

szBuffer	Latest frame received.
dwBufferLength	Size of szBuffer .

Possible returns:

EFPROT_SUCCESS
EFPROT_INTERNAL_ERROR

Compatibility:



☒ Win 32

☒ Dos

☒ Linux

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 47	SHEET 47

Confidential

C declaration:



```
typedef unsigned int  dword;  
typedef unsigned char byte;
```

//Prototype.

```
dword GetReceivedFrame( byte * szBuffer, dword * dwBufferLength );
```

```
dword dwReturn, dwLen; //Variables declaration.  
byte bBuffer[2048];
```

```
dwReturn = GetReceivedFrame ( bBuffer, &dwLen); //Gets latest frame received
```

Visual Basic declaration:



```
Private Declare Function GetReceivedFrame Lib "EpsonFiscalProtocol.dll" (ByVal szBuffer As  
String, ByRef iBufferLength As Long) As Long
```

Visual C++ / C++ Builder declaration:



```
typedef int ( *LPGRECVF)( BYTE *, DWORD * );  
HINSTANCE HandleLib = LoadLibrary("EpsonFiscalProtocol.dll");  
LPGRECVF pGetRecvFrame = (LPGRECVF) GetProcAddress( HandleLib,  
GetReceivedBuffer");
```

Warning:



Variibles **HandleLib** and **pGetRecvFrame** must be different from NULL to be used properly.

Size of **szBuffer** must be enough to support the entire frame that will return this function.
Maximum size could be 4102 bytes.

EPSON	TITLE LOW LEVEL DRIVER SPECIFICATION MANUAL FISCAL PRINTERS	REVISION SHEET 1.4	NUM.	
			TOTAL 48	SHEET 48