

# MANUAL DA INTERFACE EPSON NÃO FISCAL

Ver. 1.1.4

## INTERFACE DE ALTO NÍVEL PARA IMPRESSORAS NÃO FISCAIS EPSON



A EPSON disponibiliza exemplos de programação em diversas linguagens e sistemas operacionais, para ter acesso a estes arquivos cadastre-se no **PEPS** (Programa Epson de Parcerias de Software). Basta acessar o site do EpsonStars e realizar sua inscrição, não demora mais do que 1 minuto.



[www.epsonstars.com.br](http://www.epsonstars.com.br)

0800 7768 6668

## Índice

<b>1</b>	<b>CONVENÇÕES</b>	<b>3</b>
	CONVENÇÃO DE SÍMBOLOS	3
	TIPOS DE DADOS SUPORTADOS	3
<b>2</b>	<b>INTRODUÇÃO</b>	<b>3</b>
<b>3</b>	<b>RETORNOS DAS FUNÇÕES</b>	<b>4</b>
<b>4</b>	<b>FUNÇÕES DA INTERFACE</b>	<b>4</b>
4.1	FUNÇÕES	5
4.1.1	<i>ConfiguraTaxaSerial</i>	5
4.1.2	<i>IniciaPorta</i>	6
4.1.3	<i>FechaPorta</i>	7
4.1.4	<i>ImprimeTexto</i>	8
4.1.5	<i>ImprimeTextoTag</i>	9
4.1.6	<i>FormataTX</i>	11
4.1.7	<i>AcionaGuilhotina</i>	13
4.1.8	<i>ComandoTX</i>	14
4.1.9	<i>Le_Status</i>	15
4.1.10	<i>Le_Status_Gaveta</i>	16
4.1.11	<i>ConfiguraCodigoBarras</i>	17
4.1.12	<i>ImprimeCodigoBarrasCODABAR</i>	19
4.1.13	<i>ImprimeCodigoBarrasCODE128</i>	20
4.1.14	<i>ImprimeCodigoBarrasCODE39</i>	21
4.1.15	<i>ImprimeCodigoBarrasCODE93</i>	22
4.1.16	<i>ImprimeCodigoBarrasEAN13</i>	23
4.1.17	<i>ImprimeCodigoBarrasEAN8</i>	24
4.1.18	<i>ImprimeCodigoBarrasITF</i>	25
4.1.19	<i>ImprimeCodigoBarrasUPCA</i>	26
4.1.20	<i>ImprimeCodigoBarrasUPCE</i>	27
4.1.21	<i>ImprimeCodigoBarrasPDF417</i>	28
4.1.22	<i>ImprimeCodigoQRCODE</i>	29
4.1.23	<i>GerarQRCodeArquivo</i>	31
4.1.24	<i>ImprimeBmpEspecial</i>	32
4.1.25	<i>Habilita_Log</i>	33
4.1.26	<i>ImprimeCheque</i>	34
4.1.27	<i>LeMICR</i>	36
<b>5</b>	<b>LAYOUT DO CHEQUE</b>	<b>37</b>

## 1 Convenções

### Convenção de Símbolos

Símbolo	Significado...
	Este símbolo indica que o texto que vem logo em seguida é uma referência a outros tópicos deste documento.
	Este símbolo indica que em seguida encontra-se uma dica de como utilizar a interface.

Tabela 1 – Convenção de Símbolos

### Tipos de Dados Suportados

Tipo de Dados	Abrev.	Valores permitidos
Alfanumérico	(A)	'a'-'z', 'A'-'Z', '0'-'9'
Alfabético	(L)	'a'-'z', 'A'-'Z'
Numérico	(N)	'0'-'9'
Binário	(B)	0x00-0xFF
Imprimível	(P)	0x20-0xFF
Hexadecimal	(H)	'0'-'9', 'a'-'f', 'A'-'F'
Data	(D)	ddmmaaaa (ex: "30012002")
Hora	(T)	hhmmss (ex: "113034")
Booleano	(E)	'S', 'N'
Texto com atributos de impressão	(RT)	0x20-0xFF, aceitando atributos e códigos de barras.
Opcional	(O)	Campo opcional

Tabela 2 – Tipos de Dados

## 2 Introdução

Este documento descreve em detalhes a interface de alto nível para Impressoras Não Fiscais Epson. Esta interface pode ser usada em qualquer linguagem de desenvolvimento para o sistema operacional Windows 32/64-bits.

A Interface Epson de alto nível é uma API avançada com funções de máxima performance para a impressora não fiscal e foi concebida de maneira a permitir fácil integração entre a impressora e o aplicativo.

Nas seções seguintes encontram-se informações de como utilizar esta interface e uma descrição detalhada das funções, com seus protótipos e exemplos em diversas linguagens de desenvolvimento.

### 3 Retornos das Funções

A tabela abaixo lista os valores dos retornos das funções e seus respectivos significados.

Símbolo	Valor Hexa	Descrição
FUNC_SUCESSO	0x01	Operação realizada com sucesso.
FUNC_ERRO	Diferente de 0x01	Erro durante a execução.

Tabela 3 – Retornos das Funções

### 4 Funções da Interface

As funções da interface Epson foram definidas utilizando o seguinte protótipo:

```
function Nome_Função(...)
```

A tabela abaixo define os tipos de dados utilizados como parâmetros nas funções e seus respectivos tamanhos em bits.

Tipo	Descrição	Declaração C/C++	Declaração VB6/VB.Net	Delphi
SHORTINT	16-bit	unsigned short	Short	ShortInt
INTEGER	32-bit	unsigned int	Integer	Integer
BOOLEAN	0 (FALSE) or ≠ 0 (TRUE)	int/bool	Boolean	LongBool

Tabela 4 – Tipos de Dados

A tabela abaixo define os tipos de ponteiros utilizados como parâmetros de retorno de dados nas funções.

Ponteiro	Tipo	Declaração C/C++	Declaração VB6/VB.Net	Delphi
@BOOLEAN	BOOL*	int*/bool*	Boolean	LongBool
PCHAR	char*	char*	String	Pchar / PAnsiChar

Tabela 5 – Tipos de Ponteiros

Por convenção, todas as tabelas que detalham as posições em buffers retornados pela InterfaceEpson utilizam a posição "0" como sendo a posição inicial do mesmo (notação utilizada por linguagens como C/C++, Delphi e Java). Caso a linguagem de programação utilizada utilize por convenção a posição "1" como sendo a posição inicial de um buffer, todas as posições das tabelas devem ser acrescidas de uma unidade.

## 4.1 Funções

### 4.1.1 ConfiguraTaxaSerial

- Esta função indica para a biblioteca qual deve ser a velocidade de comunicação utilizada para a porta serial.

#### Sintaxe:

```
function ConfiguraTaxaSerial (dwTaxa:Integer):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

#### Entradas:

Variável	Tipo	Tam.	Descrição
dwTaxa	INTEGER	-	Velocidade da porta serial 115200 – 57600 – 38400 – 19200 – 9600

#### Saídas:

Nenhum.

#### Retornos:

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

#### Requisitos:

O canal de comunicação deve estar fechado.

#### Exemplo em C / C++ / C++ Builder / C#:



```
Retorno = ConfiguraTaxaSerial( 115200 );
```

#### Exemplo em Visual Basic 6.0 / Visual Basic.NET:



```
Retorno = ConfiguraTaxaSerial( 115200 )
```

#### Exemplo em Delphi / Delphi XE2:



```
Retorno := ConfiguraTaxaSerial( 115200 );
```

### 4.1.2 IniciaPorta

- Esta função abre a porta de comunicação com a impressora não fiscal. A execução bem sucedida desta função é necessária para o funcionamento de todos os demais comandos da interface.

**Sintaxe:**

Function IniciaPorta (pszPorta:Pchar):Integer; StdCall; External 'InterfaceEpsonNF.dll';

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszPorta	PCHAR	-	Nome da Porta de Comunicação que será utilizada. Podemos utilizar: COM1, COM2, COM3, etc, para porta serial; USB ou o endereço I.P para Ethernet

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

O canal de comunicação deve estar fechado.

**Exemplo em C / C++ / C++ Builder / C#:**

```
Retorno = IniciaPorta( "USB" );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = IniciaPorta( "COM1" )
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := IniciaPorta ( '192.168.192.168' );
```

### 4.1.3 FechaPorta

Esta função efetua o fechamento do canal de comunicação entre a impressora e a biblioteca. Após a execução deste comando será necessário reabrir o canal de comunicação para o envio de novos comandos.

**Sintaxe:**

```
function FechaPorta ():Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Nenhum.

**Saídas:**

Nenhum.

**Retornos:**

.....  
FUNC\_SUCESSO

Operação realizada com sucesso.

.....  
FUNC\_ERRO

Erro durante a execução.

**Requisitos:**

O canal de comunicação deve estar aberto.

**Exemplo em C / C++ / C++ Builder / C#:**

```
Retorno = FechaPorta( );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = FechaPorta( )
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := FechaPorta( );
```

#### 4.1.4 ImprimeTexto

Esta função envia uma cadeia de caracteres para ser impresso. A quebra de linhas será efetuada caso seja inserido um caracter do tipo LINE FEED, ou caso seja atingido o limite máximo de caracteres suportados pela linha (limite variável, dependendo do tipo de fonte e impressora utilizada).

##### Sintaxe:

```
function ImprimeTexto(pszTexto:Pchar):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

##### Entradas:

Variável	Tipo	Tam.	Descrição
pszTexto	PCHAR	2048	Texto que será impresso

##### Saídas:

Nenhum.

##### Retornos:

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

##### Requisitos:

O canal de comunicação deve estar aberto.

##### Exemplo em C / Visual C++ / C++ Builder / C# :



```
Retorno = ImprimeTexto("Linha 01\nLinha 02");
```

##### Exemplo em Visual Basic 6.0 / Visual Basic.NET:



```
Retorno = ImprimeTexto("Linha 01" + Chr(10) + "Linha 02")
```

##### Exemplo em Delphi / Delphi XE2:



```
Retorno := ImprimeTexto('Linha 01' + #10 + 'Linha 02');
```

### 4.1.5 ImprimeTextoTag

Esta função envia uma cadeia de caracteres para ser impresso. A quebra de linhas será efetuada caso seja inserido um caracter do tipo LINE FEED, ou caso seja atingido o limite máximo de caracteres suportados pela linha (limite variável, dependendo do tipo de fonte e impressora utilizada). Nesta função podemos incluir TAGS que podem indicar:

- Formatação de texto
- Impressão de código de barras
- Impressão de códigos bi-dimensionais (QR-Code, PDF 417)
- Impressão de logotipo

#### Sintaxe:

```
function ImprimeTextoTag(pszTexto:Pchar):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

#### Entradas:

Variável	Tipo	Tam.	Descrição
pszTexto	PCHAR	4096	Texto que será impresso

#### Tags suportadas:

Tag	Descrição
<b></b>	Negrito
<ad></ad>	Alinhamento de texto à direita
<s></s>	Sublinhado
<e></e>	Expandido
<c></c>	Condensado
<n></n>	Fonte Padrão
<l></l>	Salta Linha
<ce></ce>	Centraliza o Texto
<da></da>	Altura Dupla
<x></x>	Altura e Largura Dupla
<g></g>	Aciona a Gaveta
<gui></gui>	Aciona a Guilhotina
<bmp>XX,XX</bmp>	Imprime logotipo pré carregado na impressora. Caso seja passado como parâmetro entre as tags o índice do logotipo, o mesmo será utilizado na impressão. Caso contrário será utilizada a primeira posição da tabela de logotipos (32,32).
<ibmp>XXX</ibmp>	Imprime uma imagem gravada em um arquivo BMP. Deve ser passado o path completo do arquivo, incluindo a extensão .bmp
<cespl>XX</cespl>	Configura o espaço entre as linhas utilizado pela impressora.
<upc-a> </upc-a>	Imprime Código de Barras padrão UPC-A
<ean13> </ean13>	Imprime Código de Barras padrão EAN-13
<ean8> </ean8>	Imprime Código de Barras padrão EAN-8
<code39> </code39>	Imprime Código de Barras padrão Code 39

<code93> </code93>	Imprime Código de Barras padrão Code 93
<codabar> </codabar>	Imprime Código de Barras padrão CODABAR
<i2of5> </i2of5>	Imprime Código de Barras padrão ITF
<code128> </code128>	Imprime Código de Barras padrão Code 128
<pdf> </pdf>	Imprime Código Bidimensional padrão PDF 417
<qrcode> </qrcode>	Imprime Código Bidimensional padrão QR-Code
<lmodulo> </lmodulo>	Define a largura do módulo utilizada no QR-Code
<correcao> </correcao>	Define o fator de correção utilizado no QR-Code

**Saídas:**

Nenhum.

**Retornos:**

SERIAL\_SUCESSO

Operação realizada com sucesso.

FUNC\_ERRO

Erro durante a execução.

**Requisitos:**

Nenhum requisito é necessário.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeTextoTag("<ce><qrcode>www.epson.com.br</qrcode></ce>");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeTextoTag("<ce><qrcode>www.epson.com.br</qrcode></ce>")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeTextoTag('<ce><qrcode>www.epson.com.br</qrcode></ce>');
```

#### 4.1.6 FormataTX

Esta função envia uma cadeia de caracteres a ser impressa, com a possibilidade da escolha da formatação que será aplicada no texto.

**Sintaxe:**

```
function FormataTX(pszTexto:PChar , dwTipoLetra:Integer, dwItalico:Integer, dwSublinhado:Integer,  
dwExpandido:Integer, dwEnfatizado:Integer):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszTexto	PCHAR	-	Texto que será impresso
dwTipoLetra	INTEGER	-	Tipo de letra que será utilizada: 1 – Comprimido 2 - Normal
dwItalico	INTEGER	-	Flag que habilita itálico 1 – Habilita itálico 0 – Desabilita itálico
dwSublinhado	INTEGER	-	Flag que habilita sublinhado 1 – Habilita sublinhado 0 – Desabilita sublinhado
dwExpandido	INTEGER	-	Flag que habilita impressão expandida 1 – Habilita expandido 0 – Desabilita expandido
dwEnfatizado	INTEGER	-	Flag que habilita Negrito 1 – Habilita negrito 0 – Desabilita negrito

**Saídas:**

Nenhum.

**Retornos:**

SERIAL\_SUCESSO

Operação realizada com sucesso.

FUNC\_ERRO

Erro durante a execução.

**Requisitos:**

Nenhum.

---

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = FormataTX( "Teste de impressão", 1, 0, 1, 1, 1 );
```

---

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = FormataTX( "Teste de impressão", 1, 0, 1, 1, 1 )
```

---

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := FormataTX( 'Teste de impressão', 1, 0, 1, 1, 1 );
```

### 4.1.7 AcionaGuilhotina

Esta função executa o acionamento da guilhotina da impressora.

**Sintaxe:**

```
function AcionaGuilhotina (dwTipoCorte:Integer):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
dwTipoCorte	INT	-	Tipo do corte utilizado: 0 – Parcial 1 - Total

**Saídas:**

Nenhum.

**Retornos:**

SERIAL_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = AcionaGuilhotina( 1 );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = AcionaGuilhotina( 1 )
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := AcionaGuilhotina( 1 );
```

### 4.1.8 ComandoTX

Esta função deve ser utilizada para o envio de cadeias de bytes, contendo comandos do tipo ESC/POS.

**Sintaxe:**

```
function ComandoTX( pszComando:PChar; dwTamanho:Integer):Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszComando	PCHAR	-	Cadeia de bytes do comando.
dwTamanho	INT	-	Quantidade de bytes utilizados no comando.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
 char szComando[] = { 27, 112, 0, 25, 250}; //Aciona gaveta  
Retorno = ComandoTX( szComando, 5 );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
 szComando = Chr(27) + Chr(112) + Chr(0) + Chr(25) + Chr(250)  
Retorno = ComandoTX( szComando, 5 )
```

**Exemplo em Delphi / Delphi XE2:**

```
 szComando := chr(#27) + chr(#112) + chr(#0) + chr(#25) + chr(#250);  
Retorno := ComandoTX( szComando, 5 );
```

### 4.1.9 Le\_Status

Esta função retorna o status da impressora.

**Sintaxe:**

```
function Le_Status():Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Nenhum.

**Saídas:**

Nenhum

**Retornos:**

FUNC_ERRO	Erro durante a execução.
5	Impressora com pouco papel.
9	Tampa aberta.
24	Impressora "ONLINE".
32	Impressora sem papel.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C#:**

```
Retorno = Le_Status();
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = Le_Status()
```

**Exemplo em Delphi/ Delphi XE2:**

```
Retorno := Le_Status();
```

**4.1.10 Le\_Status\_Gaveta**

Esta função efetua a leitura do status da gaveta.

**Sintaxe:**

```
function Le_Status_Gaveta ():Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Nenhum

**Saídas:**

Nenhum.

**Retornos:**

FUNC_ERRO	Erro durante a execução.
1	Gaveta Aberta.
2	Gaveta Fechada.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = Le_Status_Gaveta();
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = Le_Status_Gaveta()
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := Le_Status_Gaveta();
```

**4.1.11 ConfiguraCodigoBarras**

Esta função efetua a configuração dos parâmetros utilizados na impressão do código de barras.

**Sintaxe:**

function ConfiguraCodigoBarras (dwAltura:Integer, dwLargura:Integer, dwHRI:Integer, dwFonte:Integer, dwMargem:Integer):Integer;StdCall; External 'InterfaceEpsonNF.dll';

**Entradas:**

Variável	Tipo	Tam.	Descrição
dwAltura	INTEGER	-	Altura do código de barras (entre 1 e 255).
dwLargura	INTEGER	-	Largura da barra utilizada: 0 – Barras finas. 1 – Barras médias. 2 – Barras grossas.
dwHRI	INTEGER	-	Posição dos caracteres do código de barras: 0 – Não imprime os caracteres. 1 – Imprime os caracteres acima do código 2 – Imprime os caracteres abaixo do código 3 – Imprime os caracteres acima e abaixo do código
dwFonte	INTEGER	-	Tipo da fonte utilizada na impressão dos caracteres: 0 – Normal. 1 – Condensada.
dwMargem	INTEGER	-	Valor da margem utilizada (entre 0 e 575).

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

---

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ConfiguraCodigoBarras( 150, 0, 2, 1, 0 );
```

---

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ConfiguraCodigoBarras( 150, 0, 2, 1, 0 )
```

---

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ConfiguraCodigoBarras( 150, 0, 2, 1, 0 );
```

**4.1.12 ImprimeCodigoBarrasCODABAR**

Esta função efetua a impressão de um código de barras do tipo CODABAR.

**Sintaxe:**

```
function ImprimeCodigoBarrasCODABAR(pszCodigo:PChar):Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasCODABAR( "A1234567B" );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasCODABAR( "A1234567B" )
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasCODABAR( 'A1234567B' );
```

**4.1.13 ImprimeCodigoBarrasCODE128**

Esta função efetua a impressão de um código de barras do tipo CODE128.

**Sintaxe:**

```
function ImprimeCodigoBarrasCODE128 (pszCodigo:PChar):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasCODE128( "Linha de Texto" );
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasCODE128( "Linha de Texto" )
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasCODE128( 'Linha de Texto' );
```

**4.1.14 ImprimeCodigoBarrasCODE39**

Esta função efetua a impressão de um código de barras do tipo CODE39.

**Sintaxe:**

```
function ImprimeCodigoBarrasCODE39( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasCODE39("ABC-123");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasCODE39("ABC-123")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasCODE39('ABC-123');
```

**4.1.15 ImprimeCodigoBarrasCODE93**

Esta função efetua a impressão de um código de barras do tipo CODE93.

**Sintaxe:**

```
function ImprimeCodigoBarrasCODE93( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasCODE93("123-ABC");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasCODE93("123-ABC")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasCODE93('123-ABC');
```

**4.1.16 ImprimeCodigoBarrasEAN13**

Esta função efetua a impressão de um código de barras do tipo EAN13.

**Sintaxe:**

```
function ImprimeCodigoBarrasEAN13( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasEAN13("123456789012");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasEAN13("123456789012")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasEAN13('123456789012');
```

**4.1.17 ImprimeCodigoBarrasEAN8**

Esta função efetua a impressão de um código de barras do tipo EAN8.

**Sintaxe:**

```
function ImprimeCodigoBarrasEAN8( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasEAN8("1234567");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasEAN8("1234567")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasEAN8('1234567');
```

**4.1.18 ImprimeCodigoBarrasITF**

Esta função efetua a impressão de um código de barras do tipo ITF.

**Sintaxe:**

```
function ImprimeCodigoBarrasITF( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasITF("0123456789012345");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasITF("0123456789012345")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasITF('0123456789012345');
```

**4.1.19 ImprimeCodigoBarrasUPCA**

Esta função efetua a impressão de um código de barras do tipo UPC-A.

**Sintaxe:**

```
function ImprimeCodigoBarrasUPCA( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasUPCA("12345678901");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasUPCA("12345678901")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasUPCA('12345678901');
```

**4.1.20 ImprimeCodigoBarrasUPCE**

Esta função efetua a impressão de um código de barras do tipo UPC-E.

**Sintaxe:**

```
function ImprimeCodigoBarrasUPCE( pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

A função ConfiguraCodigoBarras deve ter sido executada.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasUPCE("02354866");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasUPCE("02354866")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasUPCE('02354866');
```

**4.1.21 ImprimeCodigoBarrasPDF417**

Esta função efetua a impressão de um código bi-dimensional do tipo PDF-417.

**Sintaxe:**

```
function ImprimeCodigoBarrasPDF417( dwCorrecao:Integer, dwAltura:Integer, dwLargura:Integer,  
dwColunas:Integer, pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
dwCorrecao	INTEGER	-	Nível de correção de erros (entre 0 e 8).
dwAltura	INTEGER	-	Altura do caracter (entre 1 e 8).
dwLargura	INTEGER	-	Largura do caracter (entre 1 e 4).
dwColunas	INTEGER	-	Número de colunas utilizadas por linha (entre 0 e 30).
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoBarrasPDF417(4, 3, 2, 0, "Código PDF 417");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoBarrasPDF417(4, 3, 2, 0, "Código PDF 417")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoBarrasPDF417(4, 3, 2, 0, 'Código PDF 417');
```

**4.1.22 ImprimeCodigoQRCODE**

Esta função efetua a impressão de um código bi-dimensional do tipo QR-Code.

**Sintaxe:**

```
function ImprimeCodigoQRCODE(dwRestauracao:Integer, dwModulo:Integer, dwTipo:Integer, dwVersao:Integer,  
dwModo:Integer, pszCodigo:PChar ):Integer; StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
dwRestauracao	INTEGER	-	Nível de restauração do código: 0 – 7% de restauração 1 – 15% de restauração 2 – 25% de restauração 3 – 30% de restauração
dwModulo	INTEGER	-	Tamanho do módulo do código (entre 1 e 127).
dwTipo	INTEGER	-	Tipo do código: 0 – Normal 1 – Reduzido
dwVersao	INTEGER	-	Versão do QR-Code (entre 1 e 40).
dwModo	INTEGER	-	Tipo dos dados que serão impressos: 0 – Somente números 1 – Alfanumérico 2 – Binário 3 – Kanji
pszCodigo	PCHAR	-	Código que será impresso.

**Saídas:**

Nenhum.

**Retornos:**

FUNC\_SUCESSO                      Operação realizada com sucesso.

FUNC\_ERRO                          Erro durante a execução.

**Requisitos:**

Nenhum.

---

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCodigoQRCODE(3, 3, 1, 1, 1, "Código QR CODE");
```

---

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCodigoQRCODE(3, 3, 1, 1, 1, "Código QR CODE ")
```

---

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := ImprimeCodigoQRCODE(3, 3, 1, 1, 1, 'Código QR CODE');
```

**4.1.23 GerarQRCodeArquivo**

Esta função efetua a criação de um arquivo BMP com um código bi-dimensional do tipo QR-Code.

**Sintaxe:**

```
function GerarQRCodeArquivo(pszFileName:PChar, pszDados:PChar ):Integer; StdCall; External  
    'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
pszFileName	PCHAR	-	Nome do arquivo de destino (Path completo, incluindo a extensão .BMP)
pszDados	PCHAR	-	Dados que serão gravados no QRCode.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = GerarQRCodeArquivo("C:\\Epson\\QRCode.bmp", "Código QR CODE");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = GerarQRCodeArquivo("C:\\Epson\\QRCode.bmp", "Código QR CODE")
```

**Exemplo em Delphi / Delphi XE2:**

```
Retorno := GerarQRCodeArquivo('C:\\Epson\\QRCode.bmp', 'Código QR CODE');
```

#### 4.1.24 ImprimeBmpEspecial

Esta função efetua a impressão de um arquivo BMP monocromático na impressora.

##### Sintaxe:

```
function ImprimeBmpEspecial(pszFileName:PChar, dwX:Integer, dwY:Integer, dwAngulo:Integer,):Integer;  
StdCall; External 'InterfaceEpsonNF.dll';
```

##### Entradas:

Variável	Tipo	Tam.	Descrição
pszFileName	PCHAR	-	Nome do arquivo de destino (Path completo, incluindo a extensão .BMP)
dwX	INTEGER	-	Escala Horizontal (não implementado).
dwY	INTEGER	-	Escala Vertical (não implementado).
dwAngulo	INTEGER	-	Ângulo de impressão (não implementado).

##### Saídas:

Nenhum.

##### Retornos:

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

##### Requisitos:

Nenhum.

##### Exemplo em C / Visual C++ / C++ Builder / C# :



```
Retorno = ImprimeBmpEspecial("C:\\Epson\\QRCode.bmp", 0,0,0);
```

##### Exemplo em Visual Basic 6.0 / Visual Basic.NET:



```
Retorno = ImprimeBmpEspecial("C:\\Epson\\QRCode.bmp", 0,0,0)
```

##### Exemplo em Delphi / Delphi XE2:



```
Retorno := ImprimeBmpEspecial('C:\\Epson\\QRCode.bmp', 0,0,0);
```

**4.1.25 Habilita\_Log**

Esta função habilita o log da comunicação efetuada entre a biblioteca e a impressora.

**Sintaxe:**

```
function Habilita_Log(dwEstado:Integer, pszCaminho:PChar):Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
dwEstado	INTEGER	-	Estado de execução do Log: <b>0</b> – Desabilitado <b>1</b> – Habilitado
pszCaminho	PCHAR	-	Nome do diretório de gravação do log.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
FUNC_ERRO	Erro durante a execução.

**Requisitos:**

Nenhum.

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = Habilita_Log(1, "C:\\Epson\\");
```

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = Habilita_Log( 1, "C:\\Epson\\" )
```

**Exemplo em Delphi / Delphi XE2:**

```
iRetorno := Habilita_Log( 1, 'C:\\Epson\\');
```

**4.1.26 ImprimeCheque**

Esta função executa a impressão de um cheque. Comando disponível apenas para modelos TM-H6000.

**Sintaxe:**

```
function ImprimeCheque(szIndice:PChar, szValor:PChar, szData:PChar, szPara:PChar, szCidade:PChar,  
szAdicional:PChar):Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

**Entradas:**

Variável	Tipo	Tam.	Descrição
szIndice	PCHAR	-	Índice das coordenadas do banco que serão utilizadas na impressão do cheque.
szValor	PCHAR	-	Valor do cheque, com 2 casas decimais.
szData	PCHAR	-	Data do cheque, no formato DD/MM/AAAA
szPara	PCHAR	-	Destinatário do cheque
szCidade	PCHAR	-	Cidade em que o cheque será emitido.
szAdicional	PCHAR	-	Texto adicional que será impresso no cheque.

**Saídas:**

Nenhum.

**Retornos:**

FUNC_SUCESSO	Operação realizada com sucesso.
-1	Timeout de inserção do cheque.
-2	Índice do cheque não encontrado no arquivo InterfaceEpsonNF.xml
-3	Coordenada do cheque com valor zerado.
-4	Arquivo InterfaceEpsonNF.xml não encontrado
-5	Data do cheque com formato inválido.
-6	Mês com valor inválido.

**Requisitos:**

As coordenadas do cheque devem estar gravadas no arquivo InterfaceEpson.xml. Para maiores informações, verifique o capítulo 5 (Layout do cheque).

---

**Exemplo em C / Visual C++ / C++ Builder / C# :**

```
Retorno = ImprimeCheque("1", "12,23", "10/12/2014", "Epson do Brasil", "São Paulo", "");
```

---

**Exemplo em Visual Basic 6.0 / Visual Basic.NET:**

```
Retorno = ImprimeCheque("1", "12,23", "10/12/2014", "Epson do Brasil", "São Paulo", "")
```

---

**Exemplo em Delphi / Delphi XE2:**

```
iRetorno := ImprimeCheque('1', '12,23', '10/12/2014', 'Epson do Brasil', 'São Paulo', '');
```

#### 4.1.27 LeMICR

Esta função executa a leitura do código MICR do cheque. Comando disponível apenas para modelos TM-H6000.

##### Sintaxe:

```
function LeMICR(pszCodigo:PChar):Integer;StdCall; External 'InterfaceEpsonNF.dll';
```

##### Entradas:

Nenhum.

##### Saídas:

Variável	Tipo	Tam.	Descrição
pszCodigo	PCHAR	-	Código do MICR lido do cheque.

##### Retornos:

FUNC_SUCESSO	Operação realizada com sucesso.
-1	Timeout de inserção do cheque.
-2	Erro de leitura do MICR.

##### Requisitos:

Nenhum.

##### Exemplo em C / Visual C++ / C++ Builder:

```
 char pszCodigo[40];  
Retorno = LeMICR( pszCodigo );
```

##### Exemplo em C#:

```
 StringBuilder pszCodigo = new StringBuilder(40, 40);  
Retorno = LeMICR( pszCodigo );
```

##### Exemplo em Visual Basic 6.0 / Visual Basic.NET:

```
 Dim pszCodigo As String  
pszCodigo = Space(40)  
iRetorno := LeMICR( pszCodigo )
```

##### Exemplo em Delphi:

```
 pszCodigo: array[0..40] of Char;  
iRetorno := LeMICR( pszCodigo );
```

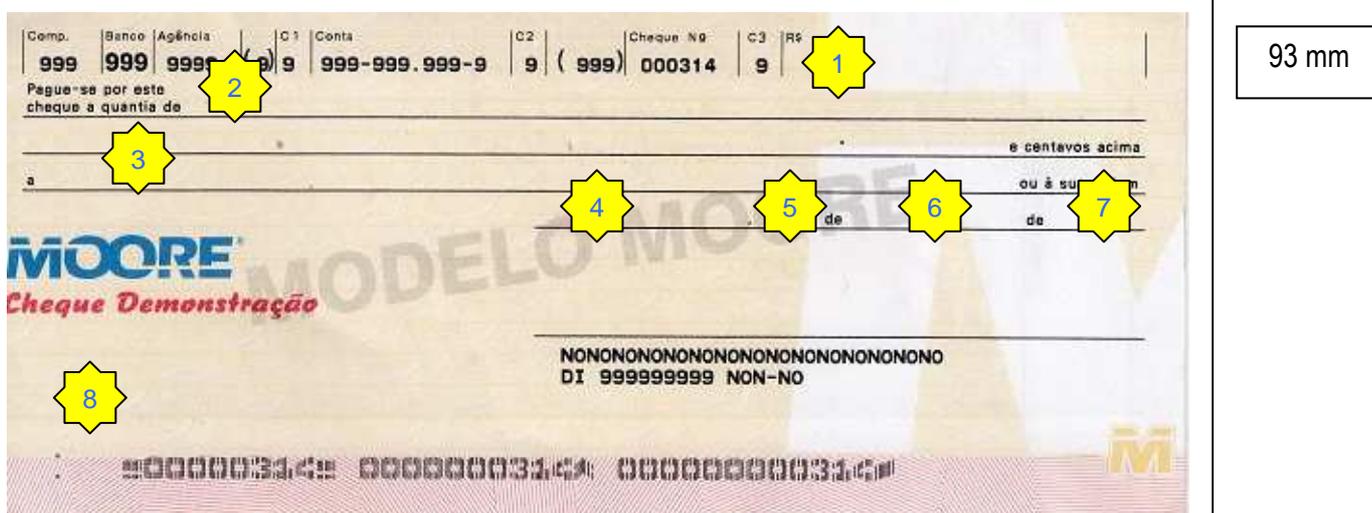
##### Exemplo em Delphi XE2:

```
 pszCodigo := AnsiString(StringOfChar(' ', 40));  
iRetorno := LeMICR( pszCodigo );
```

## 5 Layout do cheque

Devido a não padronização do layout dos cheques utilizados no país, é necessário a inclusão das principais coordenadas de impressão dos cheques que serão impressos. Estas coordenadas devem ser incluídas no arquivo de configurações da biblioteca, chamado "InterfaceEpsonNF.xml"

A imagem abaixo mostra quais devem ser as coordenadas incluídas no arquivo de configuração:



154 mm

93 mm

Campo	Tag no arquivo XML	Descrição
1	<VALOR>	Coordenadas para a impressão do valor do cheque.
2	<EXTENSO>	Coordenadas para a impressão do valor por extenso do cheque.
3	<PARA>	Coordenadas para a impressão do destinatário do cheque.
4	<CIDADE>	Coordenadas para a impressão da cidade onde o cheque foi emitido.
5	<DIA>	Coordenadas para a impressão do dia da emissão do cheque.
6	<MES>	Coordenadas para a impressão do mês da emissão do cheque.
7	<ANO>	Coordenadas para a impressão do ano da emissão do cheque.
8	<ADICIONAL>	Coordenadas para a impressão de um texto adicional no cheque.

Dentro do grupo das coordenadas, devem ser incluídas as tags <HORIZONTAL> e <VERTICAL> contendo as coordenadas de impressão em milímetros, a partir da margem inferior direita do cheque.

Além das tags de coordenadas, devem ser incluídas também uma tag referente ao índice do cheque no arquivo, e outra com a descrição do tipo do cheque.

Opcionalmente, outras configurações podem ser incluídas no arquivo XML. Na tabela abaixo são mostradas as tags opcionais que podem ser incluídas no arquivo:

Tag no arquivo XML	Descrição
<TIMEOUT>	Tempo de espera para a inserção do cheque. Caso este tempo seja expirado, a função retornará erro. O tempo deve ser incluído em segundos.
<MOEDA>	Grupo contendo as configurações de moeda.
<SINGULAR>	Descrição da moeda utilizada na impressão do cheque no singular.
<PLURAL>	Descrição da moeda utilizada na impressão do cheque no plural.
<CENTAVOS>	Flag que indica se a palavra "CENTAVOS" deve ser incluída na impressão do valor por extenso do cheque. Devem ser utilizados os valores 'S' para Sim e 'N' para Não.
<MICR>	Grupo contendo as configurações de leitura do MICR do cheque.
<TIMEOUT>	Tempo de espera para a inserção do cheque. Caso este tempo seja expirado, a função retornará erro. O tempo deve ser incluído em segundos.
<TIPO>	Tipo do MICR que será lido. Deverá ser utilizado o valor '0' para leitura do padrão CMC7.

A imagem abaixo mostra a configuração de um cheque, gravada no arquivo InterfaceEpsonNF.xml:

```
<EPSON>
  <NAO_FISCAL>
    <CHEQUES>
      <TIMEOUT>25</TIMEOUT>
      <MOEDA>
        <SINGULAR>REAL</SINGULAR>
        <PLURAL>REAIS</PLURAL>
        <CENTAVOS>S</CENTAVOS>
      </MOEDA>
      <CHEQUE>
        <CODIGO>1</CODIGO>
        <DESCRICAO>BANCO MOORE</DESCRICAO>
        <VALOR>
          <HORIZONTAL>55</HORIZONTAL>
          <VERTICAL>65</VERTICAL>
        </VALOR>
        <EXTENSO>
          <HORIZONTAL>145</HORIZONTAL>
          <VERTICAL>60</VERTICAL>
        </EXTENSO>
        <PARA>
          <HORIZONTAL>165</HORIZONTAL>
          <VERTICAL>50</VERTICAL>
        </PARA>
        <CIDADE>
          <HORIZONTAL>95</HORIZONTAL>
          <VERTICAL>43</VERTICAL>
        </CIDADE>
        <DIA>
          <HORIZONTAL>60</HORIZONTAL>
          <VERTICAL>43</VERTICAL>
        </DIA>
        <MES>
          <HORIZONTAL>49</HORIZONTAL>
          <VERTICAL>43</VERTICAL>
        </MES>
        <ANO>
          <HORIZONTAL>19</HORIZONTAL>
          <VERTICAL>43</VERTICAL>
        </ANO>
        <ADICIONAL>
          <HORIZONTAL>160</HORIZONTAL>
          <VERTICAL>10</VERTICAL>
        </ADICIONAL>
      </CHEQUE>
    </CHEQUES>
    <MICR>
      <TIMEOUT>25</TIMEOUT>
      <TIPO>0</TIPO>
    </MICR>
  </NAO_FISCAL>
</EPSON>
```

Configuração do timeout para impressão do cheque.

Configuração da moeda utilizada na impressão do cheque.

Coordenadas utilizadas na impressão do cheque.

Configurações utilizadas na leitura do MICR do cheque.