

Portuguese

bemafi32.so

2.3.1

Gerado por Doxygen 1.5.6

Mon Jul 5 11:25:13 2010

Contents

1 bemafiscal.so

A bemafiscal.so é uma biblioteca de comunicação de alto-nível, pela versatilidade e facilidade de programação em relação as outras bibliotecas. É utilizada em ambiente Linux 32 bits, realizando a comunicação com as impressoras fiscais Bematech.

Todas as funções desta biblioteca são funções de alto-nível.

A biblioteca pode ser salva em qualquer uma das pastas de bibliotecas do sistema, de acordo com a configuração de /etc/ld.so.conf*. Normalmente, basta utilizar a pasta /usr/local/lib ou até mesmo /usr/lib. Para utilizar outros diretórios, provavelmente será necessário alterar as configurações do ld.so.conf para que a biblioteca seja encontrada.

1.1 Arquivo de configuração

O arquivo bemafiscal.xml (arquivo de configuração da biblioteca) deve estar em um dos seguintes diretórios:

- ./bemaconfig.xml (diretório de execução da aplicação)
- /etc/bemaconfig.xml
- /usr/lib/bemaconfig.xml

O arquivo de configuração é um arquivo XML, com codificação ISO-8859-1, e deve estar presente para o correto funcionamento da biblioteca.

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<bematech>
<configuracoes>
<porta>/dev/ttyS0</porta>
<modelo>MP3000FI</modelo>
<log>1</log>
<path>.</path>
<controleManual>0</controleManual>
<ConfigRede>0</ConfigRede>
</configuracoes>
<ModoRemoto>
<IP>10.0.1.100</IP>
<Porta>5001</Porta>
</ModoRemoto>
<RelatorioTipo60>
<COOInicial>291</COOInicial>
<COOFinal>000085</COOFinal>
<GTFinal>000000000000272263</GTFinal>
<GTInicial>000000000000272263</GTInicial>
</RelatorioTipo60>
<Cheque>
<Favorecido>Bematech</Favorecido>
<Cidade>Curitiba</Cidade>
</Cheque>
<ChequeCopia>
<NumeroBanco/>
<Valor>0000000015000</Valor>
<Favorecido>Bematech</Favorecido>
<Cidade>Curitiba</Cidade>
```

```
<Banco>001</Banco>
<Data>05042007</Data>
<Mensagem>Mensagem promocional</Mensagem>
<ImpressaoVerso>0</ImpressaoVerso>
<Linhas>1</Linhas>
</ChequeCopia>
</bematech>
```

Os parâmetros que devem ser configurados são descritos a seguir. Os emails são internos, utilizados e alterados pela própria biblioteca a fim de permitir certas funcionalidades. Em nenhuma hipótese eles devem ser manualmente alterados.

1.1.1 modoremoto.porta

Porta serial a ser utilizada.

1.1.2 configuracoes.modelo

Modelo da impressora sendo utilizada. Impressoras suportadas: KR4IBMFI, MP25FI, MP40FI, MP50FI, MP2000FI, MP2100FI, MP3000FI e MP6000FI.

1.1.3 configuracoes.log

Habilita/desabilita o arquivo de log da biblioteca.

1.1.4 configuracoes.path

Diretório no qual serão criados os arquivos status.txt, retorno.txt e bemafiscal.log.

1.1.5 controleManual

Habilita o controle manual da abertura da porta serial. Isto é, o desenvolvedor será responsável por chamar as funções Bematech_FI_AbrePortaSerial e Bematech_FI_FechaPortaSerial.

1.1.6 Cheque.Favorecido

Nome do favorecido a ser preenchido nos cheques.

1.1.7 Cheque.Cidade

Cidade a ser preenchida nos cheques.

2 Índice dos módulos

2.1 Módulos

Lista de todos os módulos:

Cupom Fiscal	??
Operações Não Fiscais	??
Código de Barras	??
Relatórios Fiscais	??
Leitura da Memória Fiscal	??
Inicialização	??
Informações da Impressora	??
Outras Informações da Impressora	??
Informações MFD	??
Informação	??
Gaveta de Dinheiro	??
Cheque	??
Utilidades	??
Outras	??
Download MFD	??
Internas	??
Monitor	??

3 Documentação do módulo

3.1 Cupom Fiscal

Funções

- int [Bematech_FI_AbreBilhetePassagem](#) (const char *imprimeValorFim, const char *imprimeEnfatizado, const char *embarque, const char *destino, const char *linha, const char *prefixo, const char *agente, const char *agencia, const char *data, const char *hora, const char *poltrona, const char *plataforma)
- int [Bematech_FI_AbreBilhetePassagemMFD](#) (const char *embarque, const char *destino, const char *linha, const char *agencia, const char *data, const char *hora, const char *poltrona, const char *plataforma, const char *tipo, const char *rg, const char *nome, const char *endereco, const char *ufDestino)
- int [Bematech_FI_AbreCupom](#) (const char *cpf)
- int [Bematech_FI_AbreCupomMFD](#) (const char *cCPF, const char *cNome, const char *cEndereco)
- int [Bematech_FI_AcrescimoDescontoItemMFD](#) (const char *cItem, const char *cAcrescimoDesconto, const char *cTipoAcrescimoDesconto, const char *cValorAcrescimoDesconto)

- int [Bematech_FI_AcrescimoDescontoSubtotalMFD](#) (const char *cTmpFlag, const char *cTipo, const char *cTmpValor)
- int [Bematech_FI_AumentaDescricaoItem](#) (const char *descricao)
- int [Bematech_FI_Autenticacao](#) (void)
- int [Bematech_FI_AutenticacaoMFD](#) (const char *cLinhas, const char *cTexto)
- int [Bematech_FI_CancelaAcrescimoDescontoItemMFD](#) (const char *cFlag, const char *cItem)
- int [Bematech_FI_CancelaAcrescimoDescontoSubtotalMFD](#) (const char *cFlag)
- int [Bematech_FI_CancelaCupom](#) (void)
- int [Bematech_FI_CancelaCupomMFD](#) (const char *cCPF, const char *cNome, const char *cEndereco)
- int [Bematech_FI_CancelaItemAnterior](#) (void)
- int [Bematech_FI_CancelaItemGenerico](#) (const char *numeroItem)
- int [Bematech_FI_CupomAdicionalMFD](#) (void)
- int [Bematech_FI_EfetuaFormaPagamento](#) (const char *formaPagamento, const char *valorPago)
- int [Bematech_FI_EfetuaFormaPagamentoDescricaoForma](#) (const char *formaPagamento, const char *valorPago, const char *descricaoForma)
- int [Bematech_FI_EfetuaFormaPagamentoImpAntiga](#) (const char *cFormaPagamento, const char *cValorFormaPagamento)
- int [Bematech_FI_EfetuaFormaPagamentoIndice](#) (const char *indice, const char *valorForma)
- int [Bematech_FI_EfetuaFormaPagamentoIndiceDescricaoForma](#) (const char *cIndiceFormaPagamento, const char *cValorFormaPagamento, const char *cDescricaoForma)
- int [Bematech_FI_EfetuaFormaPagamentoIndiceMFD](#) (const char *cIndiceForma, const char *cValorFormaPagamento, const char *cParcelas, const char *cDescricaoForma)
- int [Bematech_FI_EfetuaFormaPagamentoMFD](#) (const char *cFormaPagamento, const char *cValorFormaPagamento, const char *cParcelas, const char *cDescricaoForma)
- int [Bematech_FI_EstornoFormasPagamento](#) (const char *formaOrigem, const char *formaDestino, const char *valor)
- int [Bematech_FI_FechaCupom](#) (const char *formaPagamento, const char *acrescimoOuDesconto, const char *tipoAcrescimoDesconto, const char *valorAcrescimoDesconto, const char *valorPago, const char *mensagem)
- int [Bematech_FI_FechaCupomResumido](#) (const char *formaPagamento, const char *mensagem)
- int [Bematech_FI_IniciaFechamentoCupom](#) (const char *acrescimoOuDesconto, const char *tipoAcrescimoDesconto, const char *valorAcrescimoDesconto)
- int [Bematech_FI_IniciaFechamentoCupomMFD](#) (const char *cAcrescimoDesconto, const char *cTipoAcrescimoDesconto, const char *cValorAcrescimo, const char *cValorDesconto)
- int [Bematech_FI_SubTotalizaCupomMFD](#) (void)
- int [Bematech_FI_TerminaFechamentoCupom](#) (const char *mensagem)
- int [Bematech_FI_TerminaFechamentoCupomCodigoBarrasMFD](#) (const char *cTmpMensagem, const char *cTipoCodigo, const char *cCodigo, int iAltura, int iTmpLargura, int iPosicaoCaracteres, int iFonte, int iMargem, int iCorrecaoErros, int iColunas)
- int [Bematech_FI_TotalizaCupomMFD](#) (void)
- int [Bematech_FI_UsaUnidadeMedida](#) (const char *unidadeMedida)
- int [Bematech_FI_VendeItem](#) (const char *codigo, const char *descricao, const char *aliquota, const char *tipoQuantidade, const char *quantidade, short int casasDecimais, const char *unitario, const char *tipoDesconto, const char *desconto)
- int [Bematech_FI_VendeItemArredondamentoMFD](#) (const char *cCodigo, const char *cDescricao, const char *cAliquota, const char *cUnidadeMedida, const char *cQtdFracionaria, const char *cUnitario, const char *cAcrescimo, const char *cDesconto, int iArredonda)
- int [Bematech_FI_VendeItemDepartamento](#) (const char *cCodigo, const char *cDescricao, const char *cAliquota, const char *cValorUnitario, const char *cQuantidade, const char *cValorAcrescimo, const char *cValorDesconto, const char *cIndiceDepartamento, const char *cUnidadeMedida)

3.1.1 Documentação das funções

3.1.1.1 `int Bematech_FI_AbreBilhetePassagem (const char * imprimeValorFim, const char * imprimeEnfatizado, const char * embarque, const char * destino, const char * linha, const char * prefixo, const char * agente, const char * agencia, const char * data, const char * hora, const char * poltrona, const char * plataforma)`

Abre Bilhete Passagem.

Abrir o cupom bilhete de passagem

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AbreBilhetePassagem("1", "1", "Curitiba", "São Paulo",
    "Leito", "123", "Carlos", "Itapemirim", "11/01/02", "23:30:00",
    "15", "D10");
```

Parâmetros:

imprimeValorFim Indicação se será impresso o valor pago no fim do cupom. "1" - imprime "0" - não imprime

imprimeEnfatizado Indica se as informações de embarque, poltrona e plataforma serão impressos enfatizado. "1" - imprime enfatizado "0" - não imprime

embarque Local de embarque, com até 40 caracteres.

destino Local de destino, com até 40 caracteres.

linha Informações sobre a linha (Ex: Curitiba - São Paulo - Leito), com até 40 caracteres.

prefixo Prefixo, com até 40 caracteres.

agente Nome do vendedor, com até 40 caracteres.

agencia Nome da agência, com até 40 caracteres.

data Data de embarque, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

hora Hora de embarque, hhmmss ou hh:mm:ss.

poltrona Número da poltrona, com até 2 caracteres.

plataforma Número da plataforma, com até 3 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Para o registro de itens, utilize a função [Bematech_FI_VendeItem\(\)](#).

Os campos de código e descrição devem ser informados, porém não serão impressos.

É obrigatório pelo convênio SINIEF 06/89 no mínimo o registro da tarifa (num totalizador tributado) e do seguro (no totalizador não incidência).

O canhoto do motorista somente será impresso se a emissão do cupom adicional for ativada durante a lacração da impressora.

3.1.1.2 int Bematech_FI_AbreBilhetePassagemMFD (const char * *embarque*, const char * *destino*, const char * *linha*, const char * *agencia*, const char * *data*, const char * *hora*, const char * *poltrona*, const char * *plataforma*, const char * *tipo*, const char * *rg*, const char * *nome*, const char * *endereco*, const char * *ufDestino*)

Abre Bilhete Passagem MFD.

Abrir o cupom bilhete de passagem

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AbreBilhetePassagemMFD("Curitiba", "São Paulo",
    "Curitiba/São Paulo - Leito", "Itapemirim", "22/04/02",
    "11:30:00", "15", "D10", "3", "5.021.659-66", "Fulano de Tal",
    "Rua sem Fim, 1000", "SP");
```

Parâmetros:

embarque Local de embarque, com até 40 caracteres.

destino Local de destino, com até 40 caracteres.

linha informações sobre a linha (Ex: Curitiba - São Paulo - Leito).

agencia Nome da agência, com até 40 caracteres.

data Data de embarque.

hora Hora de embarque.

poltrona Número da poltrona, com até 2 caracteres.

plataforma Número da plataforma, com até 3 caracteres.

tipo Tipo da passagem.

rg RG, com até 29 caracteres.

nome Nome do cliente, com até 30 caracteres.

endereco Endereço do cliente, com até 80 caracteres.

ufDestino UF destino, com 2 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Para o registro de itens, utilize a função [Bematech_FI_VendeItem\(\)](#).

O primeiro item a ser vendido deve possuir a descrição "TARIFA".

O tipo da passagem deve ser de acordo com a seguinte tabela:

```
0 (zero)    passagem Rodoviário Intermunicipal
1 (um)     passagem Ferroviário Intermunicipal
2 (dois)   passagem Aquaviário Intermunicipal
3 (três)   passagem Rodoviário Interestadual
4 (quatro) passagem Ferroviário Interestadual
5 (cinco)  passagem Aquaviário Interestadual
6 (seis)   passagem Rodoviário Internacional
7 (sete)   passagem Ferroviário Internacional
8 (oito)   passagem Aquaviário Internacional
```

3.1.1.3 int Bematech_FI_AbreCupom (const char * *cpf*)

Abre Cupom.

Abre um cupom fiscal na impressora.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AbreCupom("11.111.111-11");
// ou
iRetorno = Bematech_FI_AbreCupom("");
```

Parâmetros:

cpf CPF/CGC do cliente a ser impresso no cabeçalho do cupom com até 29 caracteres (opcional)

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom fiscal deve estar fechado.

3.1.1.4 int Bematech_FI_AbreCupomMFD (const char * *cCPF*, const char * *cNome*, const char * *cEndereco*)

Abre Cupom MFD.

Abre o cupom fiscal na impressora MFD e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AbreCupomMFD("10.123.154-98", "Fulano de Tal",
"Rua Sem Fim, 1000");
```

Parâmetros:

- cCPF* CPF ou CGC do cliente com até 29 caracteres.
- cNome* Nome do cliente com até 30 caracteres.
- cEndereco* Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

Na impressora fiscal Bematech MP-2100 TH FI o tamanho da descrição do endereço, impresso no cabeçalho do cupom fiscal, foi reduzido de 80 para 79 caracteres. Para manter a compatibilidade, será aceito o tamanho de 80 caracteres, mas internamente a impressora registrará apenas os 79 caracteres.

3.1.1.5 int Bematech_FI_A acrescimoDescontoItemMFD (const char * cItem, const char * cAcrescimoDesconto, const char * cTipoAcrescimoDesconto, const char * cValorAcrescimoDesconto)

Acrécimo Desconto Item MFD.

Efetua o acréscimo ou desconto em qualquer item enquanto o cupom fiscal não estiver totalizado.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_A acrescimoDescontoItemMFD("005", "D", "$", "1000");
```

Parâmetros:

- cItem* Número do item pertencente ao cupom fiscal com até 3 dígitos.
- cAcrescimoDesconto* Indica se é acréscimo "A" ou desconto "D".
- cTipoAcrescimoDesconto* Indica se o acréscimo ou desconto é por valor "\$" ou por percentual "%".
- cValorAcrescimoDesconto* Valor do acréscimo ou desconto a ser efetuado, com até 14 dígitos para acréscimo ou desconto por valor e 4 dígitos para percentual.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação

- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.1.1.6 int Bematech_FI_A acrescimoDescontoSubtotalMFD (const char * *cTmpFlag*, const char * *cTipo*, const char * *cTmpValor*)

Acrécimo Desconto SubTotal MFD.

Efetua acréscimo ou desconto no subtotal do cupom.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_A acrescimoDescontoItemMFD("A", "%", "10,00");
```

Parâmetros:

cTmpFlag Indicador de acréscimo ("A" ou "a") ou desconto ("D" ou "d").

cTipo Indicador de acréscimo ou desconto por valor "\$" ou por percentual "%".

cTmpValor Valor do acréscimo ou desconto com até 14 dígitos por valor ou 4 dígitos por percentual.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom deve estar subtotalizado.

O cupom não deve estar totalizado.

É permitido o registro de apenas uma operação válida para acréscimo e uma para desconto no subtotal.

3.1.1.7 int Bematech_FI_AumentaDescricaoItem (const char * *descricao*)

Aumenta Descrição Item.

Essa função permite aumentar a descricao do item em até 200 caracteres.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AumentaDescricaoItem("Produto 123/776 - 001 abc");
```

Parâmetros:

descricao Descrição do item (até 200 caracteres)

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 30 Função incompatível com a impressora Yanco.

Nota:

Essa função deve ser utilizada antes da Bematech_FI_VendeItem, pois assim o item será impresso com a nova descrição.

Este comando tem validade somente para a impressão de um item, voltando ao default que é a impressão com 29 caracteres na descricao do item.

3.1.1.8 int Bematech_FI_Autenticacao (void)

Autenticação.

Fazer a autenticação de documentos.

```
// exemplo em C/C++  
int iRetorno;  
iRetorno = Bematech_FI_Autenticacao();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2).

Nota:

Deverá ser executada imediatamente após um recebimento não fiscal ou o término de um cupom fiscal. Poderá ser repetido até 5 vezes para cada recebimento, após isso o comando será ignorado.

Na impressora fiscal Bematech, serão aguardados 5 segundos para o posicionamento do documento e autenticará no documento a data, hora, número da loja, número do caixa e o número do último cupom fiscal. Se decorrido os 5 segundos sem o posicionamento do documento, a impressora retornará ao seu estado normal, retornando o status de "Comando Não Executado".

3.1.1.9 int Bematech_FI_AutenticacaoMFD (const char * *cLinhas*, const char * *cTexto*)

Autenticação MFD.

Efetua a autenticação de documentos na impressora MFD e mantém a compatibilidade com a impressora MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AutenticacaoMFD("5", "Autenticacao Mecanica !!!");
```

Parâmetros:

cLinhas Número de linhas que serão saltadas para imprimir o texto, sendo este um valor entre 1 e 99.

cTexto Texto a ser impresso com até 47 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Deverá ser executada imediatamente após um recebimento não fiscal ou o término de um cupom fiscal. Poderá ser repetido até 5 vezes para cada recebimento. Após isto, o comando será ignorado.

A impressora irá aguardar 5 segundos para o posicionamento do documento e autenticará no documento a data, hora, número da loja, número do caixa e o número do último cupom fiscal. Se decorrido os 5 segundos sem o posicionamento do documento, a impressora retornará ao seu estado normal, retornando o status de "comando não executado".

3.1.1.10 int Bematech_FI_CancelaAcredicoDescontoItemMFD (const char * *cFlag*, const char * *cItem*)

Cancela Acréscimo Desconto Item MFD.

Cancela o acréscimo ou desconto dado ao item.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaAcredicoDescontoItemMFD("A", "005");
```

Parâmetros:

cFlag Indicador de cancelamento de acréscimo "A" ou desconto "D".

cItem Número do item a ser cancelado com até 3 dígitos e restrito aos 300 últimos registros efetuados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.1.1.11 int Bematech_FI_CancelaAcredicoDescontoSubtotalMFD (const char * *cFlag*)

Cancela Acréscimo Desconto SubTotal MFD.

Cancela o acréscimo ou desconto efetuados no subtotal do cupom.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaAcredicoDescontoSubtotalMFD("D");
```

Parâmetros:

cFlag Indicador de cancelamento de acréscimo ("A" ou "a") ou cancelamento de desconto ("D" ou "d").

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

- O cupom deve estar subtotalizado.
- O cupom não deve estar totalizado.
- O cupom deve ter operações anteriores de acréscimo ou desconto no subtotal.

3.1.1.12 int Bematech_FI_CancelaCupom (void)

Cancela Cupom.

Cancela o último cupom fiscal.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaCupom();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicacao
- 2 Erro de parametros
- 4 Arquivo de inicialização não encontrado ou inválido
- 8 Erro ao criar ou gravar no arquivo texto

Nota:

No caso de cancelamento com o cupom ainda aberto, pelo menos um item deve ter sido vendido. Não poderá ter ocorrido Redução Z e nem Leitura X após a emissão do cupom.

3.1.1.13 int Bematech_FI_CancelaCupomMFD (const char * *cCPF*, const char * *cNome*, const char * *cEndereco*)

Cancela Cupom MFD.

Cancela o cupom fiscal na impressora MFD e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_CancelaCupomMFD("11.111.111-11", "Fulano de Tal",
"Rua Sem Fim, 1000");
```

Parâmetros:

- cCPF* CPF ou CGC do cliente com até 29 caracteres.
- cNome* Nome do cliente com até 30 caracteres.
- cEndereco* Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.1.1.14 int Bematech_FI_CancelaItemAnterior (void)

Cancela Item Anterior.

Cancela o último item vendido.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaItemAnterior();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Ok
- 2* Erro de parâmetros
- 4* Arquivo de inicialização não encontrado ou inválido
- 8* Erro ao criar ou gravar no arquivo texto

Nota:

- O cupom fiscal deve estar aberto
- Ao menos um ítem deve ter sido vendido
- Não deve ter sido cancelado nenhum item imediatamente anterior

3.1.1.15 int Bematech_FI_CancelaItemGenerico (const char * numeroItem)

Cancela Item Genérico.

Cancelar qualquer item vendido no cupom fiscal

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaItemGenerico("005");
```

Parâmetros:

numeroItem Número do item a ser cancelado

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Comando não executado
- 2* Erro de parâmetros
- 4* Arquivo de inicialização não encontrado ou inválido
- 8* Erro ao criar ou gravar no arquivo texto

Nota:

- O cupom fiscal deve estar aberto.
- Ao menos um ítem deve ter sido vendido.
- O ítem não pode ter sido cancelado anteriormente e nem fora da faixa dos últimos itens vendidos.

3.1.1.16 int Bematech_FI_CupomAdicionalMFD (void)

Cupom Adicional MFD.

Emite um cupom fiscal adicional com as informações do COO e valor do cupom fiscal.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CupomAdicionalMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.1.1.17 int Bematech_FI_EfetuaFormaPagamento (const char * *formaPagamento*, const char * *valorPago*)

Efetua Forma Pagamento. Efetuar a forma de pagamento. Verifica o índice da forma de pagamento, se ainda não estiver cadastrada, ela será programada.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_EfetuaFormaPagamento("Dinheiro", "10,00");
```

Parâmetros:

- formaPagamento* Forma de pagamento
- valorPago* valor da forma de pagamento

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido na função
- 4* Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O fechamento do cupom com formas de pagamento deve ter sido iniciado.

3.1.1.18 int Bematech_FI_EfetuaFormaPagamentoDescricaoForma (const char *formaPagamento, const char *valorPago, const char *descricaoForma)

Efetua Forma Pagamento Descrição Forma.

Imprime a(s) forma(s) de pagamento e o(s) valor(es) pago(s). Permite a impressão de comentários na(s) forma(s) de pagamento.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EfetuaFormaPagamentoDescricaoForma("Cheque PRE", "75,00",
"Vencimento em 15/02/02");
```

Parâmetros:

- formaPagamento** Forma de pagamento, com até 16 caracteres.
- valorPago** Valor da forma de pagamento, com até 14 dígitos.
- descricaoForma** Descrição da forma de pagamento, com até 80 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora YANCO

Nota:

Descrição será impressa uma linha após a forma de pagamento.
O fechamento do cupom com formas de pagamento deve ter sido iniciado.
A função verifica o índice da forma de pagamento se ainda não estiver cadastrada será programada.

3.1.1.19 int Bematech_FI_EfetuaFormaPagamentoImpAntiga (const char * *cFormaPagamento*, const char * *cValorFormaPagamento*)

Efetua Forma Pagamento Impressora Antiga.

Efetuar a forma de pagamento na impressora antiga (v2.12)

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EfetuaFormaPagamentoImpAntiga("Dinheiro", "10,00");
```

Parâmetros:

cFormaPagamento Descrição da forma de pagamento

cValorFormaPagamento Valor da forma de pagamento

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação

3.1.1.20 int Bematech_FI_EfetuaFormaPagamentoIndice (const char * *indice*, const char * *valorForma*)

Efetua Forma Pagamento Índice.

Efetua a forma de pagamento através de seu índice.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_EfetuaFormaPagamentoIndice("01", "75,00");
```

Parâmetros:

indice Índice da forma de pagamento, com até 2 caracteres.

valorForma Valor da forma de pagamento, com até 14 dígitos.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.1.1.21 int Bematech_FI_EfetuaFormaPagamentoIndiceDescricaoForma (const char * *cIndiceFormaPagamento*, const char * *cValorFormaPagamento*, const char * *cDescricaoForma*)

Efetua Forma Pagamento Índice Descrição Forma.

Imprime a(s) forma(s) de pagamento e o(s) valor(es) pago(s) através de seu índice. Permite a impressão de comentários na(s) forma(s) de pagamento.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EfetuaFormaPagamentoIndiceDescricaoForma("02", "75,00",
"Vencimento em 15/02/02");
```

Parâmetros:

cIndiceFormaPagamento Índice da forma de pagamento com 2 caracteres

cValorFormaPagamento Valor da forma de pagamento com até 14 dígitos

cDescricaoForma Descrição opcional da forma de pagamento com no máximo 80 caracteres

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-24 Forma de pagamento não programada

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

-30 Função não compatível com a impressora Yanco

Nota:

A descrição será impressa uma linha após a forma de pagamento.

O fechamento do cupom com formas de pagamento deve ter sido iniciado.

3.1.1.22 int Bematech_FI_EfetuaFormaPagamentoIndiceMFD (const char * *cIndiceForma*, const char * *cValorFormaPagamento*, const char * *cParcelas*, const char * *cDescricaoForma*)

Efetua Forma Pagamento Índice MFD.

Efetua a forma de pagamento na impressora MFD usando o parâmetro opcional "descrição" e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EfetuaFormaPagamentoIndiceMFD("01", "50,00", "2",
"Compra parcelada");
```

Parâmetros:

cIndiceForma Índice da forma de pagamento, com até 2 caracteres.

cValorFormaPagamento Valor da forma de pagamento com até 14 caracteres.

cParcelas Número de parcelas em que o pagamento será realizado, sendo este um valor entre 1 e 24.

cDescricaoForma Descrição da forma de pagamento, com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-24 Forma de pagamento não programada

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A função verifica se o índice da forma de pagamento encontra-se cadastrado. Caso não esteja, o mesmo será programado.

O número de parcelas é utilizado para emissão do comprovante não fiscal vinculado. Poderá ser emitido um comprovante para cada parcela.

A descrição será impressa uma linha após a forma de pagamento.

O fechamento do cupom com formas de pagamento deve ter sido indicado.

3.1.1.23 int Bematech_FI_EfetuaFormaPagamentoMFD (const char * cFormaPagamento, const char * cValorFormaPagamento, const char * cParcelas, const char * cDescricaoForma)

Efetua Forma Pagamento MFD.

Efetua a forma de pagamento na impressora MFD usando o parâmetro opcional "descrição" e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EfetuaFormaPagamentoMFD("Cartao VISA", "50,00", "2",
"Compra parcelada");
```

Parâmetros:

cFormaPagamento Forma de pagamento com até 16 caracteres.

cValorFormaPagamento Valor da forma de pagamento com até 14 caracteres.

cParcelas Indicador do número de parcelas em que o pagamento será realizado, devendo este número ser entre 1 e 24.

cDescricaoForma Descrição da forma de pagamento com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O número de parcelas é utilizado para emissão do comprovante não fiscal vinculado. Poderá ser emitido um comprovante para cada parcela.

A descrição será impressa uma linha após a forma de pagamento.

O fechamento do cupom com formas de pagamento deve ter sido iniciado.

3.1.1.24 int Bematech_FI_EstornoFormasPagamento (const char * *formaOrigem*, const char * *formaDestino*, const char * *valor*)

Estorno Formas Pagamento.

Estornar os valores de uma forma de pagamento e inserir em outra. O valor estornado não pode exceder o total da forma de origem e nem ter Dinheiro como origem.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EstornoFormasPagamento("Ticket", "Dinheiro", "50,00");
```

Parâmetros:

formaOrigem Descrição da forma de pagamento de origem, com até 16 caracteres.

formaDestino Descrição da forma de pagamento de destino, com até 16 caracteres.

valor Valor a ser estornado, com até 14 dígitos.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.1.1.25 `int Bematech_FI_FechaCupom (const char * formaPagamento, const char * acrescimoOuDesconto, const char * tipoAcrescimoDesconto, const char * valorAcrescimoDesconto, const char * valorPago, const char * mensagem)`

Fecha Cupom.

Fechar o cupom fiscal, permitindo acréscimo ou desconto no total do cupom, troco e mensagem promocional.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_FechaCupom("Dinheiro", "A", "$", "0000", "35,00",
    "Obrigado, volte sempre !!!");
```

Parâmetros:

formaPagamento Forma de pagamento, com até 16 caracteres.

acrescimoOuDesconto "A" para Acréscimo, "D" para Desconto

tipoAcrescimoDesconto "\$" para Acréscimo ou desconto por valor e "%" para percentual

valorAcrescimoDesconto Valor do acréscimo ou desconto.

valorPago Valor pago, com até 14 dígitos.

mensagem Mensagem promocional.

Retorna:

int com a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom deve estar aberto.

O valor do acréscimo/desconto deve ter no máximo 14 dígitos se for por valor ou 4 dígitos se for por percentual.

A mensagem promocional deve ter até 384 caracteres (8 linhas X 48 colunas) para a impressora fiscal MP-20 FI II, e 320 caracteres (8 linhas X 40 colunas) para a impressora fiscal MP-40 FI II.

3.1.1.26 `int Bematech_FI_FechaCupomResumido (const char * formaPagamento, const char * mensagem)`

Fecha Cupom Resumido.

Permite fechar o cupom de forma resumida, ou seja, sem acréscimo ou desconto no cupom e com apenas uma forma de pagamento. Essa função lê o subtotal do cupom para fechá-lo.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_FechaCupomResumido("Ticket",
    "Obrigado, volte sempre !!!");
```

Parâmetros:

formaPagamento Forma de pagamento, com até 16 caracteres.

mensagem Mensagem promocional.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2).

Nota:

A mensagem promocional deve ter até 384 caracteres (8 linhas X 48 colunas) para a impressora fiscal MP-20 FI II, e 320 caracteres (8 linhas X 40 colunas) para a impressora fiscal MP-40 FI II.

O cupom deve estar aberto.

Pelo menos 1 (um) item deve ter sido vendido e não pode ter sido cancelado.

A utilização dessa função elimina a obrigatoriedade de uso das funções Bematech_FI_IniciaFechamentoCupom, Bematech_FI_EfetuaFormaPagamento e Bematech_FI_TerminaFechamentoCupom que estão implementadas internamente na função.

3.1.1.27 int Bematech_FI_IniciaFechamentoCupom (const char * *acrescimoOuDesconto*, const char * *tipoAcrescimoDesconto*, const char * *valorAcrescimoDesconto*)

Inicia Fechamento Cupom.

Inicia o fechamento do cupom com o uso das formas de pagamento.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_IniciaFechamentoCupom("A", "%", "1000");
```

Parâmetros:

acrescimoOuDesconto "A" para Acréscimo e "D" para desconto

tipoAcrescimoDesconto "\$" para Acréscimo ou desconto por valor e "%" para percentual

valorAcrescimoDesconto Valor do acréscimo ou desconto.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom deve estar aberto.
Pelo menos um item deve ter sido vendido.
O valor do acréscimo/desconto deve ser menor que o subtotal do cupom.
O valor do acréscimo/desconto deve ter no máximo 14 dígitos se for por valor ou 4 dígitos se for por percentual.

3.1.1.28 int Bematech_FI_IniciaFechamentoCupomMFD (const char * cAcrescimoDesconto, const char * cTipoAcrescimoDesconto, const char * cValorAcrescimo, const char * cValorDesconto)

Inicia Fechamento Cupom MFD.

Inicia o fechamento do cupom fiscal. Permite acréscimo e/ou desconto no fechamento do cupom.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_IniciaFechamentoCupomMFD("X", "%", "1200", "1000");
```

Parâmetros:

- cAcrescimoDesconto** Indicador de acréscimo "A", desconto "D" ou ambos "X" no fechamento do cupom fiscal.
- cTipoAcrescimoDesconto** Indica se o acréscimo ou desconto será por valor "\$" ou percentual "%".
- cValorAcrescimo** Valor do acréscimo a ser efetuado com até 14 dígitos por valor e 4 dígitos por percentual.
- cValorDesconto** Valor do desconto a ser efetuado com até 14 dígitos por valor e 4 dígitos por percentual.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para efetuar a forma de pagamento, use a função Bematech_FI_EfetuaFormaPagamentoMFD.

3.1.1.29 int Bematech_FI_SubTotalizaCupomMFD (void)

Subtotaliza Cupom MFD.

Subtotaliza o cupom fiscal, ou seja, inicia o fechamento, imprimindo o valor total do cupom.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_SubTotalizaCupomMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função habilita as seguintes operações:

Acréscimo ou desconto em subtotal.
Cancelamento de acréscimo ou desconto em subtotal.
Totalização do cupom fiscal.

O cupom fiscal deve estar aberto e com item vendido.

Caso o valor total do cupom seja zero, o mesmo será cancelado.

Para usar esta função, não se pode ter iniciado o fechamento do cupom (Bematech_FI_IniciaFechamentoCupomMFD).

3.1.1.30 int Bematech_FI_TerminaFechamentoCupom (const char * mensagem)

Termina Fechamento Cupom.

Termina o fechamento do cupom com mensagem promocional.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_TerminaFechamentoCupom("Obrigado, volte sempre !!!");
```

Parâmetros:

mensagem Mensagem promocional

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 36 Forma de pagamento não finalizada

Nota:

A mensagem promocional deve ter até 384 caracteres (8 linhas X 48 colunas) para a impressora fiscal MP-20 FI II, e 320 caracteres (8 linhas X 40 colunas) para a impressora fiscal MP-40 FI II.

O cupom deve estar aberto.

A forma de pagamento deve ter sido efetuada.

Na impressora fiscal Bematech, serão impressas 8 linhas de texto.

3.1.1.31 `int Bematech_FI_TerminaFechamentoCupomCodigoBarrasMFD (const char * cTmpMensagem, const char * cTipoCodigo, const char * cCodigo, int iAltura, int iTmpLargura, int iPosicaoCaracteres, int iFonte, int iMargem, int iCorrecaoErros, int iColunas)`

Termina Fechamento Cupom Código de Barras MFD.

Termina o fechamento do cupom com mensagem promocional e impressão do código de barras.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_TerminaFechamentoCupomCodigoBarrasMFD
("Obrigado, volte sempre !!!", "EAN13", "123456789012", 100, 1, 3,
0, 5, 4, 5);
```

Parâmetros:

cTmpMensagem Mensagem promocional com até 384 caracteres (8 linhas x 48 colunas).

cTipoCodigo Tipo do código que será impresso. Este poderá ser um dos 12 formatos suportados pela impressora:

```
"EAN13"
"PDF417"
"ITF"
"EAN8"
"CODABAR"
"CODE128"
"CODE39"
"ISBN"
"MSI"
"PLESSEY"
"UPCA"
"UPCE"
```

cCodigo Código a ser impresso, respeitando a definição do tipo de cada código.

iAltura Altura do código de barras. Deve ser um número inteiro entre 1 e 255. (162 - default).

iTmpLargura Largura do código de barras. Deve ser um número inteiro com valor entre 0 e 2.

Largura	Formato das barras
0	barras finas
1	barras médias (default)
2	barras grossas

iPosicaoCaracteres Posição do código de barras. Deve ser um número inteiro com valor entre 0 e 3.

Posição	Formato de impressão
0	não imprime os caracteres do código
1	imprime os caracteres acima do código
2	imprime os caracteres abaixo do código (default)
3	imprime os caracteres acima e abaixo do código

iFonte Formato da fonte do código de barras. Deve ser um número inteiro com valor entre 0 e 1.

Fonte	Formato da fonte
0	normal
1	condensado

iMargem Formato da margem do código de barras. Deve ser um número inteiro com valor entre 0 e 575 (dots pitch). O valor default "0", representa a não impressão de margem.

iCorrecaoErros Correção de erros de impressão do código. Deve ser um número inteiro com valor entre 0 e 8.

iColunas Tamanho da coluna gráfica. Deve ser um inteiro com valor entre 0 e 30.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 2** Parâmetro inválido
- 4** Arquivo de inicialização não encontrado ou inválido
- 5** Erro ao abrir a porta de comunicação
- 27** Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A forma de pagamento deve ter sido efetuada.
Esta função só está disponível para as impressoras fiscais térmicas.

3.1.1.32 int Bematech_FI_TotalizaCupomMFD (void)

Totaliza Cupom MFD.

Totaliza o cupom fiscal habilitando o uso das formas de pagamento.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_TotalizaCupomMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 4** Arquivo de inicialização não encontrado ou inválido
- 5** Erro ao abrir a porta de comunicação
- 27** Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.1.1.33 int Bematech_FI_UsaUnidadeMedida (const char * *unidadeMedida*)

Usa Unidade Medida.

Imprime a unidade de medida após a quantidade do produto na venda de item.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_UsaUnidadeMedida("KG");
```

Parâmetros:

unidadeMedida Descrição da unidade de medida com no máximo 2 caracteres

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicacao
- 30 Função incompatível com a imp. yanco

Nota:

Esta função deve ser usada, antes do envio da função Bematech_FI_VendeItem.

Esta função tem validade somente para a impressão de um item, voltando ao default que é a impressão de 2 (dois) espaços em branco após a quantidade do item vendido.

Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI está programação não existe. Caso você deseje utilizar a unidade de medida, utilize a função Bematech_FI_VendeItemDepartamento.

3.1.1.34 int Bematech_FI_VendeItem (const char * *codigo*, const char * *descricao*, const char * *aliquota*, const char * *tipoQuantidade*, const char * *quantidade*, short int *casasDecimais*, const char * *unitario*, const char * *tipoDesconto*, const char * *desconto*)

Vende Item.

Vende item após a abertura do cupom fiscal. Essa função permite também a venda de itens com 3 casas decimais no valor unitário.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_VendeItem("123", "Caneta", "1200", "I", "10", 2, "0,25",
"%", "0000");
```

Parâmetros:

codigo Código do produto, com até 14 caracteres para MP7000 TH FI e 13 caracteres para as demais.

descricao Descrição do produto, com até 29 caracteres.

aliquota Alíquota tributária a ser utilizada.

tipoQuantidade Tipo de quantidade I = Inteira, F = Fracionaria

quantidade Quantidade.

casasDecimais Número de casas decimais para quantidades inteiras, restrito a 2 ou 3.

unitario Valor unitário, com até 8 dígitos.

tipoDesconto Forma do desconto: '\$' para desconto por valor e '%' para desconto por percentual.

desconto Valor do desconto.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-3 Alíquota não programada

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

A alíquota pode ser em forma de valor, o qual deve ser informado com o tamanho de 4 caracteres ou 5 com a vírgula, ou em forma de índice, então com 2 caracteres. Ex. (18,00 para o valor ou 05 para o índice).

A quantidade deve ter até 4 dígitos para quantidades inteiras e até 7 dígitos para quantidades fracionárias, sendo sempre 3 casas decimais.

O valor do desconto deve ser até 8 dígitos para descontos por valor (2 casas decimais) ou 4 dígitos para descontos percentuais.

Imagine que há duas alíquotas com o valor 12% cadastradas na impressora. A primeira cadastrada na posição 01 como ICMS e a outra na posição 05 como ISS. Se você informar o valor 1200 ou 12,00 no parâmetro "alíquota" a função irá imprimir o item usando a alíquota 01 de ICMS. A função lê as alíquotas da impressora e usa o índice da primeira ocorrência. Para usar a alíquota de ISS você deverá passar o índice 05 e não o valor 1200 ou 12,00. Se você não tiver duas alíquotas com o mesmo valor cadastradas na impressora use sempre o valor no parâmetro alíquota.

valida codigo

valida descricao

valida tipoQuantidade

valida quantidade

Valida valor unitario

3.1.1.35 int Bematech_FI_VendeItemArredondamentoMFD (const char * cCodigo, const char * cDescricao, const char * cAliquota, const char * cUnidadeMedida, const char * cQtdFracionaria, const char * cUnitario, const char * cAcrescimo, const char * cDesconto, int iArredonda)

Vende Item Arredondamento MFD.

Vende item com arredondamento ou truncamento.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_VendeItemArredondamentoMFD("123", "Caneta", "1200", "UN",
    "12", "0,25", "0", "0", false);
```

Parâmetros:

- cCodigo* Código do produto com até 14 caracteres.
- cDescricao* Descrição do produto com até 200 caracteres.
- cAliquota* Índice, com 2 caracteres, ou valor, com até 5 caracteres (duas casas decimais), da alíquota tributária.
- cUnidadeMedida* Unidade de medida com até 2 caracteres.
- cQtyFracionaria* Quantidade fracionaria com até 7 caracteres (três casas decimais).
- cUnitario* Valor unitário com até 8 caracteres (três casas decimais).
- cAcrescimo* Acréscimo, com até 10 dígitos por valor (duas casas decimais) ou 4 dígitos para percentual.
- cDesconto* Desconto, com até 10 dígitos por valor (duas casas decimais) ou 4 dígitos para percentual.
- iArredonda* Variável, inteira, indicando se o item será arredondado "1" ou truncado "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 3* Alíquota não programada
- 4* Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta
- 5* Erro ao abrir a porta de comunicação

3.1.1.36 int Bematech_FI_VendeItemDepartamento (const char * *cCodigo*, const char * *cDescricao*, const char * *cAliquota*, const char * *cValorUnitario*, const char * *cQuantidade*, const char * *cValorAcrescimo*, const char * *cValorDesconto*, const char * *cIndiceDepartamento*, const char * *cUnidadeMedida*)

Vende Item Departamento.

Vender item com departamento e unidade de medida

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_VendeItemDepartamento("123", "Caneta", "1200", "0,25",
    "10", "0", "0", "03", "UN");
```

Parâmetros:

- cCodigo* Código do produto, com até 49 caracteres.
- cDescricao* Descrição do produto, com até 201 carecteres.

cAliquota Valor ou o índice da alíquota tributária.

cValorUnitario Valor unitário, com até 9 dígitos para o valor (três casas decimais).

cQuantidade Quantidade, com até 7 dígitos para a quantidade.

cValorAcrescimo Valor acréscimo, com até 10 dígitos, sendo 2 casas decimais.

cValorDesconto Valor desconto, com até 10 dígitos, sendo 2 casas decimais.

cIndiceDepartamento Índice do departamento, com até 2 dígitos.

cUnidadeMedida Unidade de medida, com até 2 caracteres.

Retorna:

int com a informação sobre a execução do comando

Nota:

Se a alíquota for o valor, deve ser informado com o tamanho de 4 caracteres ou 5 com a vírgula. Se for o índice da alíquota, deve ser 2 caracteres. Ex. (18,00 para o valor ou 05 para o índice)

Na venda com departamento a quantidade é fracionária e são 3 casas decimais.

Caso não seja passado nenhum caracter, a unidade de medida não será impressa.

Obedece as mesmas situações descrita na função Bematech_FI_VendeItem.

3.2 Operações Não Fiscais

Módulos

- [Código de Barras](#)

Funções

- int [Bematech_FI_AbreComprovanteNaoFiscalVinculado](#) (const char *formaPagamento, const char *valorPago, const char *numeroCupom)
- int [Bematech_FI_AbreComprovanteNaoFiscalVinculadoMFD](#) (const char *cFormaPagamento, const char *cValor, const char *cNumeroCupom, const char *cCPF, const char *cNome, const char *cEndereco)
- int [Bematech_FI_AbreRecebimentoNaoFiscalMFD](#) (const char *cCPF, const char *cNome, const char *cEndereco)
- int [Bematech_FI_AbreRelatorioGerencialMFD](#) (const char *cTotalizador)
- int [Bematech_FI_AcionaGuilhotinaMFD](#) (short int iModo)
- int [Bematech_FI_AcrescimoDescontoSubtotalRecebimentoMFD](#) (const char *cTmpFlag, const char *cTipo, const char *cTmpValor)
- int [Bematech_FI_AcrescimoItemNaoFiscalMFD](#) (const char *NumeroItem, const char *cAcrescimoDesconto, const char *cTipoAcrescimoDesconto, const char *cValorAcrescimoDesconto)
- int [Bematech_FI_AtivaDesativaAlinhamentoEsquerdaMFD](#) (short flag)
- int [Bematech_FI_AtivaDesativaGuilhotinaMFD](#) (short int flag)
- int [Bematech_FI_AtivaDesativaTratamentoONOFFLineMFD](#) (short flag)
- int [Bematech_FI_AvancaPapelAccionaGuilhotinaMFD](#) (short int iLinhas, short int iModo)
- int [Bematech_FI_CancelaAcrescimoDescontoSubtotalRecebimentoMFD](#) (const char *cFlag)
- int [Bematech_FI_CancelaAcrescimoNaoFiscalMFD](#) (const char *NumeroItem, const char *cAcrescimoDesconto)
- int [Bematech_FI_CancelaItemNaoFiscalMFD](#) (const char *NumeroItem)

- int `Bematech_FI_CancelaRecebimentoNaoFiscalMFD` (const char *cCPF, const char *cNome, const char *cEndereco)
- int `Bematech_FI_EfetuaRecebimentoNaoFiscalMFD` (const char *cIndiceTotalizador, const char *cValorRecebimento)
- int `Bematech_FI_EstornoNaoFiscalVinculadoMFD` (const char *cCPF, const char *cNome, const char *cEndereco)
- int `Bematech_FI_EstornoNaoFiscalVinculadoPosteriorMFD` (const char *cFormaPagamento, const char *cValor, const char *cCOOCupom, const char *cCOOCD, const char *cCPF, const char *cNome, const char *cEndereco)
- int `Bematech_FI_FechaComprovanteNaoFiscalVinculado` (void)
- int `Bematech_FI_FechaRecebimentoNaoFiscalMFD` (const char *cMensagem)
- int `Bematech_FI_FechaRelatorioGerencial` (void)
- int `Bematech_FI_IniciaFechamentoRecebimentoNaoFiscalMFD` (const char *cAcrescimoDesconto, const char *cTipoAcrescimoDesconto, const char *cValorAcrescimo, const char *cValorDesconto)
- int `Bematech_FI_LinhasEntreCupons` (int linhas)
- int `Bematech_FI_RecebimentoNaoFiscal` (const char *indiceTotalizador, const char *valorPago, const char *formaPagamento)
- int `Bematech_FI_ReimpressaoNaoFiscalVinculadoMFD` (void)
- int `Bematech_FI_RelatorioGerencial` (const char *texto)
- int `Bematech_FI_RelatorioGerencialImpAntiga` (const char *cTexto)
- int `Bematech_FI_ResetaImpressora` (void)
- int `Bematech_FI_Sangria` (const char *valorPago)
- int `Bematech_FI_SegundaViaNaoFiscalVinculadoMFD` (void)
- int `Bematech_FI_SubTotalizaRecebimentoMFD` (void)
- int `Bematech_FI_Suprimento` (const char *valorPago, const char *formaPagamento)
- int `Bematech_FI_TotalizaRecebimentoMFD` (void)
- int `Bematech_FI_UsaComprovanteNaoFiscalVinculado` (const char *texto)
- int `Bematech_FI_UsaComprovanteNaoFiscalVinculadoTEF` (const char *texto)
- int `Bematech_FI_UsaRelatorioGerencialMFD` (const char *cTexto)
- int `Bematech_FI_UsaRelatorioGerencialMFDTEF` (const char *cTexto)

3.2.1 Documentação das funções

3.2.1.1 int Bematech_FI_AbreComprovanteNaoFiscalVinculado (const char * *formaPagamento*, const char * *valorPago*, const char * *numeroCupom*)

Abre Comprovante Não Fiscal Vinculado.

Abrir um comprovante não fiscal vinculado.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AbreComprovanteNaoFiscalVinculado("Cartao VISA", "", "");
```

Parâmetros:

formaPagamento Forma de pagamento, com até 16 caracteres.

valorPago Valor pago, com até 14 dígitos, sendo 2 casas decimais.

numeroCupom Número do cupom a que se refere o comprovante, com até 6 dígitos.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido na função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

A forma de pagamento utilizada não pode ser dinheiro.
A forma de pagamento deve ter sido utilizada no cupom.
Só pode ser emitido um comprovante não fiscal por forma de pagamento.
Os parâmetros "Valor" e "Numero do Cupom" tornam-se obrigatórios se o comprovante emitido não for referente ao último cupom fiscal emitido.

3.2.1.2 int Bematech_FI_AbreComprovanteNaoFiscalVinculadoMFD (const char * *cFormaPagamento*, const char * *cValor*, const char * *cNumeroCupom*, const char * *cCPF*, const char * *cNome*, const char * *cEndereco*)

Abre Comprovante Não Fiscal Vinculado MFD.

Abre um comprovante não fiscal vinculado na impressora MFD e mantém compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AbreComprovanteNaoFiscalVinculadoMFD("Cartao", "50,00",
"000165", "1.111.111-1", "Fulano de Tal", "Rua Sem Fim, 1000");
```

Parâmetros:

- cFormaPagamento* Descrição da forma de pagamento com até 16 caracteres.
- cValor* Valor pago na forma de pagamento do cupom a que se refere o comprovante, com até 14 dígitos (2 casas decimais).
- cNumeroCupom* Número do cupom a que se refere o comprovante com até 6 dígitos.
- cCPF* CPF do cliente com até 29 caracteres.
- cNome* Nome do cliente com até 30 caracteres.
- cEndereco* Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 4* Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A forma de pagamento utilizada não pode ser dinheiro.
A forma de pagamento deve ter sido utilizada no cupom ao qual se refere o comprovante.
Só pode ser emitido um comprovante não fiscal por forma de pagamento.
Os parâmetros "Valor" e "Número do Cupom" tornam-se obrigatórios se o comprovante emitido não for referente ao último cupom fiscal emitido.

3.2.1.3 int Bematech_FI_AbreRecebimentoNaoFiscalMFD (const char * *cCPF*, const char * *cNome*, const char * *cEndereco*)

Abre Recebimento Não Fiscal MFD.

Abre o comprovante não fiscal não vinculado para que sejam lançados os recebimentos não fiscais.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AbreRecebimentoNaoFiscalMFD("1.111.111-11",
    "Fulano de Tal", "Rua Sem Fim, 1000");
```

Parâmetros:

- cCPF* CPF ou CGC do cliente com até 29 caracteres.
- cNome* Nome do cliente com até 30 caracteres.
- cEndereco* Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 2** Parâmetro inválido
- 4** Arquivo de inicialização não encontrado ou inválido
- 5** Erro ao abrir a porta de comunicação
- 27** Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.2.1.4 int Bematech_FI_AbreRelatorioGerencialMFD (const char * *cTotalizador*)

Abre Relatório Gerencial MFD.

Abre o relatório gerencial na impressora fiscal MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AbreRelatorioGerencialMFD("05");
```

Parâmetros:

cTotalizador Índice do relatório sendo este um valor entre 1 e 30.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O relatório deve estar programado.
O relatório permanecerá aberto por 2 minutos. Se não for enviado comando de fechamento durante este período, o relatório será fechado automaticamente.

3.2.1.5 int Bematech_FI_AcionaGuilhotinaMFD (short int iModo)

Aciona Guilhotinha MFD.

Aciona a guilhotina, para o corte de papel, das impressoras fiscais MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AcionaGuilhotinaMFD(1);
```

Parâmetros:

iModo Variável, inteira, destinada a definir o tipo do corte do papel: "0" corte parcial e "1" corte total.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A função só terá efeito dentro do Comprovante Não Fiscal Vinculado e Relatório Gerencial.
Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01 e MP-2100 TH FI.

3.2.1.6 int Bematech_FI_AcrescimoDescontoSubtotalRecebimentoMFD (const char * *cTmpFlag*, const char * *cTipo*, const char * *cTmpValor*)

Acréscimo Desconto SubTotal Recebimento MFD. .

Efetua acréscimo ou desconto no subtotal do cupom.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AcrescimoDescontoSubtotalRecebimentoMFD("D", "$",
"15,00");
```

Parâmetros:

cTmpFlag Indicador de acréscimo ("A" ou "a") ou desconto ("D" ou "d").

cTipo Indicador de acréscimo ou desconto por valor "\$" ou por percentual "%".

cTmpValor Valor do acréscimo ou desconto com até 14 dígitos por valor ou 4 dígitos por percentual.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O recebimento não fiscal deve estar subtotalizado.

O recebimento não fiscal não pode estar totalizado.

É permitido o registro de apenas uma operação válida para acréscimo e uma para desconto no subtotal.

3.2.1.7 int Bematech_FI_AcrescimoItemNaoFiscalMFD (const char * *NumeroItem*, const char * *cAcrescimoDesconto*, const char * *cTipoAcrescimoDesconto*, const char * *cValorAcrescimoDesconto*)

Acréscimo Item Não Fiscal MFD.

Efetua um acréscimo ou desconto em um item não fiscal na impressora fiscal MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_AcrescimoItemNaoFiscalMFD("005", "A", "$", "10,00");
```

Parâmetros:

NumeroItem Número do item não fiscal, com até 3 caracteres.

cAcrescimoDesconto Indica se é acréscimo "A" ou desconto "D".

cTipoAcreditoDesconto Indica se o acréscimo ou desconto é por valor "\$" ou por percentual "%".

cValorAcreditoDesconto Valor do acréscimo ou desconto a ser efetuado.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.
O cupom não fiscal não pode estar subtotalizado.

3.2.1.8 int Bematech_FI_AtivaDesativaAlinhamentoEsquerdaMFD (short flag)

Ativa Desativa Alinhamento Esquerda MFD.

Ativa ou desativa o alinhamento da descrição à esquerda da impressão da descrição do item.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaAlinhamentoEsquerdaMFD(1);
```

Parâmetros:

flag Indicador de ativação "1" ou desativação "0", do alinhamento da descrição à esquerda.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só será executada, caso a impressora não tenha nenhum movimento no dia ou após a redução Z.

Não será necessária sua execução no início de cada dia.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.2.1.9 int Bematech_FI_AtivaDesativaGuilhotinaMFD (short int flag)

Ativa Desativa Guilhotina MFD.

Ativa ou desativa a guilhotina.

```
// exemplo em C/C++
int iRetorno;
int iFlag = 1;
iRetorno = Bematech_FI_AtivaDesativaGuilhotinaMFD(iFlag);
```

Parâmetros:

flag Variável, inteira, para definir o tipo de corte do papel, sendo "0" para desativar e "1" para ativar a guilhotina.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido

3.2.1.10 int Bematech_FI_AtivaDesativaTratamentoONOFFLineMFD (short flag)

Ativa Desativa Tratamento ON/OFF Line MFD.

Ativa ou desativa o tratamento da tecla "ON LINE" da impressora.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaTratamentoONOFFLineMFD(1);
```

Parâmetros:

flag Indicador de tratamento da tecla "ON LINE" da impressora: "0" desativado, "1" ativado.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Se esta funcionalidade estiver ativada, o "ON LINE" da impressora é bloqueado, evitando que ela entre em "OFF LINE", caso a tecla "ON LINE" seja pressionada.

Se esta funcionalidade estiver desativada, para se entrar no menu de "DUMP/RUNNING", deve-se pressionar os dois botões da impressora, e não somente a tecla "PAPER".

Esta função só será executada, caso a impressora não tenha nenhum movimento no dia ou após a Redução Z.

Não é necessária sua execução no início de cada dia.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.2.1.11 int Bematech_FI_AvancaPapelAcionaGuilhotinaMFD (short int *iLinhas*, short int *iModo*)

Avança Papel Aciona Guilhotina MFD.

Aciona a guilhotina das impressoras MFD avançando "N" linhas antes de cortar o papel.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_Bematech_FI_AvancaPapelAcionaGuilhotinaMFD(5, 1);
```

Parâmetros:

iLinhas Número de linhas para avançar antes de cortar o papel. Valor entre 0 e 255.

iModo Modo do corte de papel, sendo "0" corte total, "1" corte parcial ou "2" sem corte.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta

-5 Erro ao abrir a porta de comunicação

3.2.1.12 int Bematech_FI_CancelaAcréscimoDescontoSubtotalRecebimentoMFD (const char * *cFlag*)

Cancela Acréscimo Desconto SubTotal Recebimento MFD.

Cancela acréscimo ou desconto efetuados no subtotal do recebimento não fiscal.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_CancelaAcréscimoDescontoSubtotalRecebimentoMFD("D");
```

Parâmetros:

cFlag Indicador de cancelamento de acréscimo ("A" ou "a") ou desconto ("D" ou "d"), dado no subtotal do recebimento.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

- O recebimento não fiscal deve estar subtotalizado.
- O recebimento não fiscal não pode estar totalizado.
- O recebimento não fiscal deve ter operações anteriores de acréscimo ou desconto no subtotal.

3.2.1.13 int Bematech_FI_CancelaAcredicoNaoFiscalMFD (const char * NumeroItem, const char * cAcredicoDesconto)

Cancela Acréscimo Não Fiscal MFD.

Cancela o acréscimo ou desconto em um item não fiscal na impressora fiscal MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaAcredicoNaoFiscalMFD("005", "A");
```

Parâmetros:

NumeroItem Número do item não fiscal, com até 3 caracteres.

cAcredicoDesconto Indica se é o cancelamento é por acréscimo "A" ou por desconto "D".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

- Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.
- O cupom não fiscal não pode estar subtotalizado.

3.2.1.14 int Bematech_FI_CancelaItemNaoFiscalMFD (const char * *NumeroItem*)

Cancela Item Não Fiscal MFD.

Cancela um item não fiscal na impressora fiscal MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaItemNaoFiscalMFD("005");
```

Parâmetros:

NumeroItem Número do item não fiscal a ser cancelado, com até 3 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.

3.2.1.15 int Bematech_FI_CancelaRecebimentoNaoFiscalMFD (const char * *cCPF*, const char * *cNome*, const char * *cEndereco*)

Cancela Recebimento Não Fiscal MFD.

Cancela o último recebimento não fiscal emitido (Comprovante não fiscal não vinculado).

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_CancelaRecebimentoNaoFiscalMFD("11.111.111-11",
"fulano de tal", "Rua Sem Fim, 1000");
```

Parâmetros:

- cCPF* CPF ou CGC do cliente com até 29 caracteres.
- cNome* Nome do cliente com até 30 caracteres.
- cEndereco* Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.2.1.16 `int Bematech_FI_EfetuaRecebimentoNaoFiscalMFD (const char * cIndiceTotalizador, const char * cValorRecebimento)`

Efetua Recebimento Não Fiscal MFD.

Efetua o recebimento não fiscal.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_EfetuaRecebimentoNaoFiscalMFD("03", "35,00");
```

Parâmetros:

cIndiceTotalizador Índice do totalizador com até 2 dígitos para o recebimento.

cValorRecebimento Valor do recebimento com até 14 dígitos (duas casas decimais).

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O comprovante deve ter sido aberto anteriormente.

O totalizador não fiscal deve estar programado.

3.2.1.17 `int Bematech_FI_EstornoNaoFiscalVinculadoMFD (const char * cCPF, const char * cNome, const char * cEndereco)`

Estorno Não Fiscal Vinculado MFD.

Estorna os lançamentos de um comprovante de crédito ou débito vinculado.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EstornoNaoFiscalVinculadoMFD("11.111.111-11",
    "Fulano de Tal", "Rua Sem Fim, 1000");
```

Parâmetros:

- cCPF* CPF do cliente com até 29 caracteres.
cNome Nome do cliente com até 30 caracteres.
cEndereco Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
0 Erro de comunicação
-2 Parâmetro inválido
-4 Arquivo de inicialização não encontrado ou inválido
-5 Erro ao abrir a porta de comunicação
-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Deve ser executado imediatamente após a impressão do comprovante vinculado.
Para imprimir um texto qualquer, dentro do cupom de estorno, a função Bematech_FI_UsaComprovanteNaoFiscalVinculado deve ser usada.
Para fechar o cupom de estorno, deve ser usada a função Bematech_FI_FechaComprovanteNaoFiscalVinculado.

3.2.1.18 int Bematech_FI_EstornoNaoFiscalVinculadoPosteriorMFD (const char * cFormaPagamento, const char * cValor, const char * cCOOCupom, const char * cCOOCDC, const char * cCPF, const char * cNome, const char * cEndereco)

Estorno Não Fiscal Vinculado Posterior MFD.

Estorna um CDC mesmo não imediatamente após o cupom fiscal e permite o cancelamento do cupom fiscal após o estorno de todos os CDCs vinculados a ele.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_EstornoNaoFiscalVinculadoPosteriorMFD("Dinheiro", "12,50",
"1712", "133", "123.456.789-10", "Fulano de Tal",
"Rua Sem Fim, 1000");
```

Parâmetros:

- cFormaPagamento* Descrição da forma de pagamento, com até 16 caracteres.
cValor Valor pago, com até 14 caracteres.
cCOOCupom COO do cupom fiscal com até 6 caracteres.
cCOOCDC COO do CDC a ser estornado com até 6 caracteres.
cCPF CPF do cliente com até 29 caracteres.
cNome Nome do cliente com até 30 caracteres.
cEndereco Endereço do cliente com até 80 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação

3.2.1.19 int Bematech_FI_FechaComprovanteNaoFiscalVinculado (void)

Fecha Comprovante Não Fiscal Vinculado.

Encerrar o comprovante não fiscal vinculado.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_FechaComprovanteNaoFiscalVinculado();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O comprovante não fiscal vinculado deve ter sido aberto.

3.2.1.20 int Bematech_FI_FechaRecebimentoNaoFiscalMFD (const char * *cMensagem*)

Fecha Recebimento Não Fiscal MFD.

Termina o fechamento do recebimento não fiscal.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_FechaRecebimentoNaoFiscalMFD
("Obrigado, volte sempre !!!");
```

Parâmetros:

cMensagem Mensagem promocional com até 490 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O recebimento não fiscal é finalizado pelo comando de fechamento do cupom fiscal.

3.2.1.21 int Bematech_FI_FechaRelatorioGerencial (void)

Fecha Relatório Gerencial.

Encerra um relatório gerencial

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_FechaRelatorioGerencial();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O relatório gerencial deve estar aberto.

3.2.1.22 int Bematech_FI_IniciaFechamentoRecebimentoNaoFiscalMFD (const char * cAcrecimoDesconto, const char * cTipoAcrecimoDesconto, const char * cValorAcrecimo, const char * cValorDesconto)

Inicia Fechamento Recebimento Não Fiscal MFD.

Inicia o fechamento do recebimento não fiscal. Permite acréscimo e desconto no fechamento do mesmo.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_IniciaFechamentoRecebimentoNaoFiscalMFD("X", "%", "1200",
"1000");
```

Parâmetros:

cAcrecimoDesconto Indicador de acréscimo "A", desconto "D" ou ambos "X" no fechamento do cupom.

cTipoAcrecimoDesconto Indica se o acréscimo ou desconto será por valor "\$" ou percentual "%".

cValorAcrecimo Valor do acréscimo a ser efetuado, com até 14 dígitos por valor e 4 dígitos por percentual.

cValorDesconto Valor do desconto a ser efetuado, com até 14 dígitos por valor e 4 dígitos por percentual.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para efetuar a forma de pagamento, use a função Bematech_FI_EfetuaFormaPagamentoMFD.
O totalizador não fiscal deve estar programado.

3.2.1.23 int Bematech_FI_LinhasEntreCupons (int linhas)

Linhas Entre Cupons.

Programar número de linhas entre cupons. O valor default da impressora é 8 linhas

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_LinhasEntreCupons(5);
```

Parâmetros:

linhas Número de linhas, entre 0 e 255.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora Yanco

3.2.1.24 int Bematech_FI_RecebimentoNaoFiscal (const char * *indiceTotalizador*, const char * *valorPago*, const char * *formaPagamento*)

Recebimento Não Fiscal.

Emitir um comprovante não fiscal não vinculado. Usado para recebimentos. Ex: conta de água, luz, etc.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_RecebimentoNaoFiscal("05", "30,00", "Dinheiro");
```

Parâmetros:

indiceTotalizador Índice do totalizador para recebimento parcial, com até 2 dígitos.

valorPago Valor do recebimento, com até 14 dígitos por valor, separadas por ',' e até 5 valores.

formaPagamento Forma de pagamento, com até 16 caracteres por forma, separadas por ',' e até 5 formas.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

- O cupom fiscal deve estar fechado.
- O totalizador deve estar cadastrado.

3.2.1.25 int Bematech_FI_ReimpressaoNaoFiscalVinculadoMFD (void)

Reimpressão Não Fiscal Vinculado MFD.

Reimprime o último comprovante não fiscal vinculado.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ReimpressaoNaoFiscalVinculadoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação

- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O comando somente será executado se este for enviado imediatamente após a impressão do comprovante.

3.2.1.26 int Bematech_FI_RelatorioGerencial (const char * texto)

Relatório Gerencial.

Imprime o relatório gerencial

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_RelatorioGerencial
("Digite o texto a ser impresso aqui !!!");
```

Parâmetros:

texto Texto a ser impresso no relatório gerencial, com até 620 bytes.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2).

Nota:

O cupom fiscal deve estar fechado.

Se qualquer função diferente da Bematech_FI_RelatorioGerencial for chamada com o relatório aberto, efetuará seu fechamento automaticamente.

3.2.1.27 int Bematech_FI_RelatorioGerencialImpAntiga (const char * cTexto)

Relatório Gerencial Impressora Antiga.

Imprimir o relatório gerencial na impressora antiga MP20 FI

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RelatorioGerencialImpAntiga("Mensagem!!");
```

Parâmetros:

cTexto Texto a ser impresso, com até 620 bytes

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicacao
- 2* Parametro inválido
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicacao

3.2.1.28 int Bematech_FI_ResetaImpressora (void)

Reseta Impressora.

Reseta a impressora em caso de erro.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ResetaImpressora();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30* Função não compatível com a impressora Yanco

Nota:

Será executada somente se a impressora estiver em erro.

3.2.1.29 int Bematech_FI_Sangria (const char * valorPago)

Sangria.

Faz uma sangria na impressora (retirada de dinheiro).

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_Sangria("25,00");
```

Parâmetros:

valorPago Valor da sangria, com até 14 dígitos (2 casas decimais).

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido na função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom fiscal deve estar fechado.

3.2.1.30 int Bematech_FI_SegundaViaNaoFiscalVinculadoMFD (void)

Segunda Via Não Fiscal Vinculado MFD.

Imprime a segunda via do comprovante não fiscal vinculado.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_SegundaViaNaoFiscalVinculadoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O comando deverá ser executado imediatamente após a emissão da primeira via.

3.2.1.31 int Bematech_FI_SubTotalizaRecebimentoMFD (void)

SubTotaliza Recebimento MFD.

Inicia o fechamento do recebimento não fiscal (comprovante não fiscal vinculado), imprimindo o valor total do recebimento.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_SubTotalizaRecebimentoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função habilita as seguintes operações:

Acréscimo ou desconto em subtotal de recebimento não fiscal.
Cancelamento de acréscimo ou desconto em subtotal de recebimento não fiscal.
Totalização do recebimento não fiscal.

O recebimento deve estar aberto e com item vendido.

Caso o valor total do recebimento seja zero, o mesmo será cancelado.

Para usar esta função, não se pode ter iniciado o fechamento do recebimento. (Bematech_FI_IniciaFechamentoRecebimentoNaoFiscalMFD).

3.2.1.32 int Bematech_FI_Suprimento (const char * *valorPago*, const char * *formaPagamento*)

Suprimento.

Faz um suprimento na impressora (entrada de dinheiro)

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_Suprimento("50,00", "Dinheiro");
```

Parâmetros:

- valorPago*** Valor para suprimento, com até 14 dígitos (2 casas decimais)
- formaPagamento*** Forma de pagamento, com até 16 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Caso a forma de pagamento não seja informada, o suprimento será feito em Dinheiro.

3.2.1.33 int Bematech_FI_TotalizaRecebimentoMFD (void)

Totaliza Recebimento MFD.

Totaliza o recebimento não fiscal habilitando o uso das formas de pagamento.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_TotalizaRecebimentoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para efetuar a forma de pagamento, use a função Bematech_FI_EfetuaFormaPagamentoMFD.

3.2.1.34 int Bematech_FI_UsaComprovanteNaoFiscalVinculado (const char * texto)

Usa Comprovante Não Fiscal Vinculado.

Imprimir o comprovante não fiscal vinculado

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_UsaComprovanteNaoFiscalVinculado
("Digite o texto a ser impresso aqui !!!");
```

Parâmetros:

texto Texto a ser impresso na forma de pagamento, com até 618 bytes.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Na impressora fiscal BEMATECH, só pode ser usado durante 2 (dois) minutos após a abertura do comprovante não fiscal vinculado. Se esse tempo for ultrapassado, o comprovante será fechado automaticamente.

3.2.1.35 int Bematech_FI_UsaComprovanteNaoFiscalVinculadoTEF (const char * *texto*)

Usa Comprovante Não Fiscal Vinculado TEF.

Imprime o comprovante não fiscal vinculado, sem travar automaticamente o teclado e o mouse. Esta função pode ser utilizada para a impressão do texto da transação TEF. É necessário usar alguma outra função ou API do sistema operacional para travar o teclado e o mouse.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_UsaComprovanteNaoFiscalVinculadoTEF
("Digite o texto a ser impresso aqui !!!");
```

Parâmetros:

texto Texto a ser impresso no comprovante não fiscal vinculado, com até 618 bytes.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Na impressora fiscal BEMATECH, só pode ser usado durante 2 (dois) minutos após a abertura do comprovante não fiscal vinculado. Se esse tempo for ultrapassado o comprovante é fechado automaticamente.

Antes de executar a função Bematech_FI_UsaComprovanteNaoFiscalVinculadoTEF, você deverá abrir o comprovante não-fiscal vinculado através da função Bematech_FI_-AbreComprovanteNaoFiscalVinculado.

Após a impressão do texto no comprovante não-fiscal vinculado, use a função Bematech_FI_-FechaComprovanteNaoFiscalVinculado para fechá-lo.

3.2.1.36 int Bematech_FI_UsaRelatorioGerencialMFD (const char * *cTexto*)

Usa Relatório Gerencial MFD.

Imprime as informações do relatório gerencial na impressora MFD.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_UsaRelatorioGerencialMFD("Entre com o texto aqui !!!");
```

Parâmetros:

cTexto Texto a ser impresso no relatório com até 618 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O relatório permanecerá aberto por 2 minutos. Se não for enviado o comando de fechamento durante este período, o mesmo será fechado automaticamente.

Para abrir o relatório gerencial use a função Bematech_FI_AbreRelatórioGerencialMFD.

Para fechar o relatório gerencial use a função Bematech_FI_FechaRelatórioGerencial.

3.2.1.37 int Bematech_FI_UsaRelatorioGerencialMFDTEF (const char * *cTexto*)

Usa Relatório Gerencial MFD TEF.

Imprime as informações do relatório gerencial na impressora MFD com bloqueio de teclado e mouse.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_UsaRelatorioGerencialMFDTEF("Entre com o texto aqui !!!");
```

Parâmetros:

cTexto Texto a ser impresso no relatório com até 618 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função pode ser utilizada para a impressão do texto da transação TEF.

Com o uso desta função, torna-se desnecessário o uso de outra função da API do sistema operacional para travar o teclado e mouse.

O relatório permanecerá aberto por 2 minutos. Se não for enviado o comando de fechamento durante este período, o mesmo será fechado automaticamente.

Para abrir o relatório gerencial use a função Bematech_FI_AbreRelatórioGerencialMFD.

Para fechar o relatório gerencial use a função Bematech_FI_FechaRelatórioGerencial.

3.3 Código de Barras

Funções

- `int Bematech_FI_CodigoBarrasCODABARMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasCODE128MFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasCODE39MFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasCODE93MFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasEAN13MFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasEAN8MFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasISBNMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasITFMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasMSIMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasPDF417MFD` (int iCorrecaoErros, int iAltura, int iLargura, int iColunas, const char *cCodigo)
- `int Bematech_FI_CodigoBarrasPLESSEYMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasUPCAMFD` (const char *cCodigo)
- `int Bematech_FI_CodigoBarrasUPCEMFD` (const char *cCodigo)
- `int Bematech_FI_ConfiguraCodigoBarrasMFD` (int Altura, int Largura, int PosicaoCaracteres, int Fonte, int Margem)

3.3.1 Documentação das funções

3.3.1.1 `int Bematech_FI_CodigoBarrasCODABARMFD` (const char * *cCodigo*)

Código de Barras CODABAR MFD.

Realiza a impressão do código de barras CODABAR.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasCODABARMFD("123-ABC/001");
```

Parâmetros:

cCodigo Código a ser gerado no padrão CODABAR. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	20
1	12
2	08

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 4* Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.
Aceita dígitos entre 0 e 9.
Aceita as letras A, B, C e D (maiúsculas ou minúsculas).
Aceita os caracteres: "\$", "+", "-", ".", "/", e ":".

3.3.1.2 int Bematech_FI_CodigoBarrasCODE128MFD (const char * *cCodigo*)

Código de Barras CODE128 MFD.

Realiza a impressão do código de barras no padrão CODE128.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasCODE128MFD("Bematech");
```

Parâmetros:

cCodigo Código a ser gerado no padrão CODE128. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	42
1	28
2	16

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Aceita os caracteres da tabela ASCII, na faixa de valores entre 001 e 127.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.3 int Bematech_FI_CodigoBarrasCODE39MFD (const char * cCodigo)

Código de Barras CODE39 MFD.

Realiza a impressão do código de barras no padrão CODE39.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasCODE39MFD("abc-123");
```

Parâmetros:

cCodigo Código a ser gerado no padrão CODE39. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	15
1	09
2	06

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.

Será acrescentado, automaticamente, o dígito verificador.

A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

Aceita dígitos entre 0 e 9.

Aceita as letras entre A e Z (maiúsculas ou minúsculas).

Aceita os caracteres: "espaço em branco", "\$", "%", "+", "-", ".", "e /".

As letras não podem ser maiúsculas e minúsculas simultaneamente.

3.3.1.4 int Bematech_FI_CodigoBarrasCODE93MFD (const char * cCodigo)

Código de Barras CODE93 MFD.

Realiza a impressão do código de barras no padrão CODE93.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasCODE93MFD("123-ABC");
```

Parâmetros:

cCodigo Código a ser gerado no padrão CODE93. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	15
1	09
2	06

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas. Será acrescentado, automaticamente, o dígito verificador. A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado. Aceita os caracteres da tabela ASCII, na faixa de valores entre 001 e 127.

3.3.1.5 int Bematech_FI_CodigoBarrasEAN13MFD (const char * *cCodigo*)

Código de Barras EAN13 MFD.

Realiza a impressão do código de barras no padrão EAN13.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasEAN13MFD("123456789012");
```

Parâmetros:

cCodigo Código a ser gerado no padrão EAN13, com tamanho de 12 dígitos compreendidos entre 0 e 9.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação

- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.6 int Bematech_FI_CodigoBarrasEAN8MFD (const char * cCodigo)

Código de Barras EAN8 MFD.

Realiza a impressão do código de barras no padrão EAN8.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasEAN8MFD("1234567");
```

Parâmetros:

cCodigo Código a ser gerado no padrão EAN8, com tamanho de 7 dígitos compreendidos entre 0 e 9..

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.7 int Bematech_FI_CodigoBarrasISBNMFD (const char * cCodigo)

Código de Barras ISBN MFD.

Realiza a impressão do código de barras no padrão ISBN.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasISBNMFD("1-56592-292-X 90000");
```

Parâmetros:

cCodigo Código a ser gerado no padrão ISBN, com 9 dígitos, composto por dígitos entre 0 e 9, "-"(hifen) e "X".

Os caracteres "X" e "-"(hifen) não são somados.

Após o 9º dígito, podem aparecer ainda hifens seguidos por "X" ou algum número com o tamanho de 5 caracteres, não somando o espaço após o "-X" ou após o número, como por exemplo:

```
1-56592-292-X 9000
1-56592-291-1 900000
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.3.1.8 int Bematech_FI_CodigoBarrasITFMFD (const char * cCodigo)

Código de Barras ITF MFD.

Realiza a impressão do código de barras ITF (Intercalado 2/5).

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasITFMFD("0123456789012345");
```

Parâmetros:

cCodigo Código a ser gerado no padrão ITF. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	30
1	20
2	14

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Aceita dígitos entre 0 e 9.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.9 int Bematech_FI_CodigoBarrasMSIMFD (const char * cCodigo)

Código de Barras MSI MFD.

Realiza a impressão do código de barras no padrão MSI.

```
// Exemplo em C/C++  
int iRetorno;  
iRetorno = Bematech_FI_CodigoBarrasMSIMFD("123");
```

Parâmetros:

cCodigo Código a ser gerado no padrão MSI. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	16
1	10
2	07

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
Aceita dígitos entre 0 e 9.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.10 int Bematech_FI_CodigoBarrasPDF417MFD (int iCorrecaoErros, int iAltura, int iLargura, int iColunas, const char * cCodigo)

Código de Barras PDF417 MFD.

Realiza a impressão do código de barras PDF417.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_CodigoBarrasPDF417MFD(4, 3, 2, 0,
    "Bematech. Sempre presente nas melhores solucoes !!!");
```

Parâmetros:

iCorrecaoErros Quanto mais alto o nível, melhor a leitura do código, maior a impressão e menor o número de informações que poderão ser impressas. Deverá ser um número inteiro entre 0 e 8.

iAltura Altura do caractere do código (pitch - 1 pitch = altura de 0,125mm). Deverá ser um número inteiro entre 1 e 8.

iLargura Largura do caractere do código (pitch - 1 pitch = altura de 0,125mm). Deverá ser um número inteiro entre 1 e 4.

iColunas "0" (zero) utiliza o máximo de colunas que o mecanismo permite para a largura informada (pitch). Caso não caiba na linha, a impressora ajusta automaticamente para o máximo de colunas permitido. Deverá ser um número inteiro entre 0 e 30.

cCodigo Código a ser gerado no padrão PFD417 com até 1024 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.

3.3.1.11 int Bematech_FI_CodigoBarrasPLESSEYMFD (const char * cCodigo)

Código de Barras PLESSEY MFD.

Realiza a impressão do código de barras PLESSEY.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasPLESSEYMFD("123-ABC");
```

Parâmetros:

cCodigo Código a ser gerado no padrão PLESSEY. A quantidade de caracteres é dada através da relação com a largura das barras:

Largura	Caracteres
0	13
1	07
2	04

Por default a largura das barras é "1".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
Aceita dígitos entre 0 e 9.
Aceita as letras A, B, C, D, E e F (maiúsculas ou minúsculas).
As letras não podem ser maiúsculas e minúsculas simultaneamente.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.12 int Bematech_FI_CodigoBarrasUPCAMFD (const char * cCodigo)

Código de Barras UPCA MFD.

Realiza a impressão do código de barras no padrão UPCA.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasUPCAMFD("12345678901");
```

Parâmetros:

cCodigo Código a ser gerado no padrão UPCA, com tamanho de 11 dígitos compreendidos entre 0 e 9.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.13 int Bematech_FI_CodigoBarrasUPCEMFD (const char * *cCodigo*)

Código de Barras UPCE MFD.

Realiza a impressão do código de barras no padrão UPCE.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CodigoBarrasUPCEMFD("123456");
```

Parâmetros:

cCodigo Código a ser gerado no padrão UPCE, com tamanho de 6 dígitos compreendidos entre 0 e 9.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.
Será acrescentado, automaticamente, o dígito verificador.
A impressão do código de barras só será executado dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.3.1.14 int Bematech_FI_ConfiguraCodigoBarrasMFD (int *Altura*, int *Largura*, int *PosicaoCaracteres*, int *Fonte*, int *Margem*)

Configura Código de Barras MFD.

Configura a impressão do código de barras.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ConfiguraCodigoBarrasMFD(100, 1, 3, 0, 5);
```

Parâmetros:

Altura Altura do código de barras. Deve ser um número inteiro entre 1 e 255. (162 - default).

Largura Largura do código de barras. Deve ser um número inteiro com valor entre 0 e 2.

Largura	Formato das barras
0	barras finas
1	barras médias (default)
2	barras grossas

PosicaoCaracteres Posição do código de barras. Deve ser um número inteiro com valor entre 0 e 3.

Posição	Formato de impressão
0	não imprime os caracteres do código
1	imprime os caracteres acima do código
2	imprime os caracteres abaixo do código (default)
3	imprime os caracteres acima e abaixo do código

Fonte Formato da fonte do código de barras. Deve ser um número inteiro com valor entre 0 e 1.

Fonte	Formato da fonte
0	normal
1	condensado

Margem Formato da margem do código de barras. Deve ser número um inteiro com valor entre 0 e 575 (dots pitch). O valor default "0", representa a não impressão de margem.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.

A configuração do código de barras deve ser realizada dentro do Relatório Gerencial ou dentro do Comprovante Não Fiscal Vinculado.

3.4 Relatórios Fiscais**Módulos**

- [Leitura da Memória Fiscal](#)

Funções

- int [Bematech_FI_FechaRelatorioXouZ](#) (void)
- int [Bematech_FI_LeituraX](#) (void)
- int [Bematech_FI_LeituraXSerial](#) (void)
- int [Bematech_FI_ReducacaoZ](#) (const char *data, const char *hora)
- int [Bematech_FI_ReducacaoZImpAntiga](#) (void)

3.4.1 Documentação das funções

3.4.1.1 int Bematech_FI_FechaRelatorioXouZ (void)

Fecha Relatório X ou Z.

Esta função realiza o fechamento do Relatório X ou do Relatório Z, quando o mesmo estiver aberto e não finalizado.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_FechaRelatorioXouZ();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Essa função funciona apenas na impressora Yanco. Foi implementada apenas para compatibilização.

3.4.1.2 int Bematech_FI_LeituraX (void)

Leitura X.

Emite a Leitura X na impressora.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_LeituraX();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicacao
- 2 Erro de parametros
- 4 Arquivo de inicialização não encontrado ou inválido
- 8 Erro ao criar ou gravar no arquivo texto

3.4.1.3 int Bematech_FI_LeituraXSerial (void)

Leitura X Serial.

Recebe os dados da Leitura X pela serial e grava em um arquivo texto.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_LeituraXSerial();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.4.1.4 int Bematech_FI_ReducacaoZ (const char * data, const char * hora)

Redução Z.

Emite a Redução Z na impressora. Permite ajustar o relógio interno da impressora em até 5 minutos.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ReducacaoZ("08/01/2000", "18:00:00");
// ou
iRetorno = Bematech_FI_ReducacaoZ("", "");
```

Parâmetros:

data Data atual no formato ddmmaa ou dd/mm/aa ou dd/mm/aa.

hora Hora a ser alterada no formato hhmss ou hh:mm:ss.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Comando não executado
- 2* Erro de parâmetros
- 4* Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação.
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Somente será aceito um ajuste de +/- 5 minutos. Se os valores estiverem fora dessa faixa serão limitados a 5 minutos.
O cupom fiscal deve estar fechado.

3.4.1.5 int Bematech_FI_ReducacaoZImpAntiga (void)

Redução Z Impressora Antiga.

Imprimir a redução Z antiga

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ReducacaoZImpAntiga();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação

3.5 Leitura da Memória Fiscal**Funções**

- int [Bematech_FI_LeituraMemoriaFiscalData](#) (const char *dataInicial, const char *dataFinal)
- int [Bematech_FI_LeituraMemoriaFiscalDataMFD](#) (const char *cDataInicial, const char *cDataFinal, const char *cFlagLeitura)
- int [Bematech_FI_LeituraMemoriaFiscalReducao](#) (const char *reducaoInicial, const char *reducaoFinal)
- int [Bematech_FI_LeituraMemoriaFiscalReducaoMFD](#) (const char *cReducaoInicial, const char *cReducaoFinal, const char *cFlagLeitura)
- int [Bematech_FI_LeituraMemoriaFiscalSerialData](#) (const char *dataInicial, const char *dataFinal)
- int [Bematech_FI_LeituraMemoriaFiscalSerialDataMFD](#) (const char *cDataInicial, const char *cDataFinal, const char *cFlagLeitura)
- int [Bematech_FI_LeituraMemoriaFiscalSerialReducao](#) (const char *reducaoInicial, const char *reducaoFinal)
- int [Bematech_FI_LeituraMemoriaFiscalSerialReducaoMFD](#) (const char *cReducaoInicial, const char *cReducaoFinal, const char *cFlagLeitura)

3.5.1 Documentação das funções

3.5.1.1 `int Bematech_FI_LeituraMemoriaFiscalData (const char * dataInicial, const char * dataFinal)`

Leitura Memória Fiscal Data.

Imprimir os dados de leitura da memória fiscal da impressora

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalData("01/01/2002", "05/01/2002");
```

Parâmetros:

dataInicial Data inicial, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

dataFinal Data final, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.2 `int Bematech_FI_LeituraMemoriaFiscalDataMFD (const char * cDataInicial, const char * cDataFinal, const char * cFlagLeitura)`

Leitura Memória Fiscal Data MFD.

Imprime os dados da leitura da memória fiscal na impressora MFD, por período de datas.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalDataMFD("01/04/02", "15/04/02", "c");
```

Parâmetros:

cDataInicial Data inicial dos dados da memória fiscal a serem lidos, no formato dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

cDataFinal Data final dos dados da memória fiscal a serem lidos, no formato dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

cFlagLeitura Indicador de leitura completa "c" ou simplificada "s".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.3 int Bematech_FI_LeituraMemoriaFiscalReducao (const char * *reducaoInicial*, const char * *reducaoFinal*)

Leitura Memória Fiscal Redução Z.

Emite a leitura da memória fiscal da impressora por intervalo de reduções.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_LeituraMemoriaFiscalReducao("0100", "0110");
```

Parâmetros:

- reducaoInicial* Redução inicial, com até 4 dígitos
- reducaoFinal* Redução final, com até 4 dígitos

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.4 int Bematech_FI_LeituraMemoriaFiscalReducaoMFD (const char * *cReducaoInicial*, const char * *cReducaoFinal*, const char * *cFlagLeitura*)

Leitura Memória Fiscal Redução MFD.

Imprime os dados da leitura da memória fiscal na impressora MFD, por período de reduções Z.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalReducaoMFD("0010", "0015", "c");
```

Parâmetros:

cReducaoInicial Redução inicial dos dados da memória fiscal a serem lidos com até 4 dígitos.

cReducaoFinal Data final dos dados da memória fiscal a serem lidos com até 4 dígitos.

cFlagLeitura Indicador de leitura completa "c" ou simplificada "s".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.5 int Bematech_FI_LeituraMemoriaFiscalSerialData (const char * *dataInicial*, const char * *dataFinal*)

Leitura Memória Fiscal Serial Data.

Recebe os dados da memória fiscal por intervalo de datas pela serial e grava em arquivo texto.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalSerialData("01/04/2002",
"15/04/2002");
```

Parâmetros:

dataInicial Data inicial da leitura da memória fiscal, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

dataFinal Data final da leitura da memória fiscal, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.6 int Bematech_FI_LeituraMemoriaFiscalSerialDataMFD (const char * *cDataInicial*, const char * *cDataFinal*, const char * *cFlagLeitura*)

Leitura Memória Fiscal Serial Data MFD.

Faz a leitura dos dados da memória fiscal pela serial na impressora MFD por período de datas.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalSerialDataMFD("01/04/02", "15,04,02",
"c");
```

Parâmetros:

- cDataInicial* Data inicial da leitura no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.
- cDataFinal* Data final da leitura no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.
- cFlagLeitura* Indicador de leitura da memória fiscal completa "c" ou simplificada "s".

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.7 int Bematech_FI_LeituraMemoriaFiscalSerialReducao (const char * *reducaoInicial*, const char * *reducaoFinal*)

Leitura Memória Fiscal Serial Redução Z.

Recebe os dados da memória fiscal por intervalo de reduções pela serial e grava em arquivo texto.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalSerialReducao("0100", "0110");
```

Parâmetros:

reducaoInicial Número da redução inicial da leitura da memória fiscal, com até 4 dígitos.

reducaoFinal Número da redução final da leitura da memória fiscal, com até 4 dígitos.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.5.1.8 int Bematech_FI_LeituraMemoriaFiscalSerialReducaoMFD (const char * *cReducaoInicial*, const char * *cReducaoFinal*, const char * *cFlagLeitura*)

Leitura Memória Fiscal Serial Redução MFD.

Faz a leitura dos dados da memória fiscal pela serial na impressora MFD por período de reduções Z.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_LeituraMemoriaFiscalSerialReducaoMFD("01/04/02",
"15,04,02", "c");
```

Parâmetros:

cReducaoInicial Número da redução inicial, com até 4 caracteres.

cReducaoFinal Número da redução final, com até 4 caracteres.

cFlagLeitura Indicador de leitura da memória fiscal completa "c" ou simplificada "s".

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O cupom fiscal deve estar fechado.

3.6 Inicialização**Funções**

- int [Bematech_FI_AlteraSimboloMoeda](#) (const char *simboloMoeda)
- int [Bematech_FI_AtivaDesativaCancelamentoCupom2HorasMFD](#) (int flag)
- int [Bematech_FI_AtivaDesativaCorteTotalMFD](#) (int flag)
- int [Bematech_FI_AtivaDesativaSensorPoucoPapelMFD](#) (int flag)
- int [Bematech_FI_AtivaDesativaVendaUmaLinhaMFD](#) (short flag)
- int [Bematech_FI_ConfiguraCorteGuilhotinaMFD](#) (int tempo)
- int [Bematech_FI_EspacoEntreLinhas](#) (int dots)
- int [Bematech_FI_ForcaImpactoAguilhas](#) (short int impacto)
- int [Bematech_FI_NomeiaDepartamento](#) (int indice, const char *departamento)
- int [Bematech_FI_NomeiaRelatorioGerencialMFD](#) (const char *cIndice, const char *cDescricao)
- int [Bematech_FI_NomeiaTotalizadorNaoSujeitoIcms](#) (int indice, const char *totalizador)
- int [Bematech_FI_ProgramaAliquota](#) (const char *aliquota, int vinculo)
- int [Bematech_FI_ProgramaArredondamento](#) (void)
- int [Bematech_FI_ProgramaCaracterAutenticacao](#) (const char *parametros)
- int [Bematech_FI_ProgramaFormaPagamentoMFD](#) (const char *cFormaPagamento, const char *cOperacaoTef)
- int [Bematech_FI_ProgramaHorarioVerao](#) (void)
- int [Bematech_FI_ProgramaIdAplicativoMFD](#) (const char *cIdAplicativo)
- int [Bematech_FI_ProgramaTruncamento](#) (void)
- int [Bematech_FI_RetornoImpressoraMFD](#) (short int *iACK, short int *iST1, short int *iST2, short int *iST3)
- int [Bematech_FI_VerificaCancelamentoCupom2HorasMFD](#) (char *cFlag)

3.6.1 Documentação das funções**3.6.1.1 int Bematech_FI_AlteraSimboloMoeda (const char * simboloMoeda)**

Alterar Símbolo Moeda.

Altera o símbolo da moeda programada na Impressora Fiscal.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AlteraSimboloMoeda(" R");
```

Parâmetros:

simboloMoeda Símbolo da moeda. O \$ (cifrão) é inserido automaticamente.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora Yanco

Nota:

O cupom fiscal deve estar fechado.

Será executada somente após uma Redução Z.

Na impressora fiscal Bematech modelo MP-2100 TH FI, a programação do símbolo da moeda é feita somente por Intervenção Técnica.

3.6.1.2 int Bematech_FI_AtivaDesativaCancelamentoCupom2HorasMFD (int flag)

Ativa Desativa Cancelamento Cupom 2 Horas MFD.

Ativa ou desativa o cancelamento automático do cupom às duas horas da manhã.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaCancelamentoCupom2HorasMFD(1);
```

Parâmetros:

flag Variável, inteira, destinada a indicar se o cancelamento automático encontra-se habilitado "1" ou desabilitado "0". desabilitado.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só será executada, caso a impressora não tenha nenhum movimento no dia ou após a Redução Z.

Não será necessária sua execução no início de cada dia.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.03.03 ou MP-2100 TH FI.

3.6.1.3 int Bematech_FI_AtivaDesativaCorteTotalMFD (int flag)

Ativa Desativa Corte Total MFD.

Ativa ou desativa o corte total do papel.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaCorteTotalMFD(1);
```

Parâmetros:

flag 1 para ativar e 0 para desativar o corte total do papel.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta

-5 Erro ao abrir a porta de comunicação

3.6.1.4 int Bematech_FI_AtivaDesativaSensorPoucoPapelMFD (int flag)

Ativa Desativa Sensor Pouco Papel MFD.

Ativa ou desativa o corte total do papel.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaSensorPoucoPapelMFD(1);
```

Parâmetros:

flag 1 para ativar e 0 para desativar o corte total do papel.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta

-5 Erro ao abrir a porta de comunicação

3.6.1.5 `int Bematech_FI_AtivaDesativaVendaUmaLinhaMFD (short flag)`

Ativa Desativa Venda Uma Linha MFD.

Ativa ou desativa a venda de item em apenas uma linha.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaVendaUmaLinhaMFD(1);
```

Parâmetros:

flag Indicador de ativação "1" ou desativação "0", da venda de item em apenas uma linha.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só será executada, caso a impressora não tenha nenhum movimento no dia ou após a redução Z.

Não será necessária sua execução no início de cada dia.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

Para que o item possa ser impresso em uma única linha, o seu código e sua descrição (juntos) não podem ultrapassar 16 caracteres; a quantidade deve possuir até 2 dígitos (entre 1 e 99) e o valor unitário deve possuir até 3 dígitos inteiros (entre 0,01 e 999,99).

3.6.1.6 `int Bematech_FI_ConfiguraCorteGuilhotinaMFD (int tempo)`

Configura Corte Guilhotina MFD.

Configura o tempo de acionamento da guilhotina. Valor entre 0 e 254 ms

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ConfiguraCorteGuilhotinaMFD(100);
```

Parâmetros:

tempo Tempo de acionamento

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação

3.6.1.7 `int Bematech_FI_EspacoEntreLinhas (int dots)`

Espaço Entre Linhas.

Programar espaços entre linhas. O valor default da impressora é 0 dots.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_EspacoEntreLinhas(2);
```

Parâmetros:

dots Espaço entre linhas, entre 0 e 255 dots

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora Yanco

3.6.1.8 `int Bematech_FI_ForcaImpactoAguilhas (short int impacto)`

Força Impacto Agulhas.

Permite tornar a impressão mais forte nos equipamentos baseados na MP-20 FI II.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ForcaImpactoAguilhas(2);
```

Parâmetros:

impacto Força de impacto das agulhas de impressão, de 1 (fraco) a 3 (forte).

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

O padrão das impressoras é utilizar impactos fracos.
A seleção de uma força de impacto diferente da default implica na diminuição da vida útil do cabeçote de impressão.

3.6.1.9 int Bematech_FI_NomeiaDepartamento (int *indice*, const char * *departamento*)

Nomeia Departamento.

Programar departamentos na impressora

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_NomeiaDepartamento(5, "Gasolina");
```

Parâmetros:

- indice* Índice do departamento a ser nomeado.
- departamento* Nome do departamento, com até 10 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora Yanco

Nota:

O cupom fiscal deve estar fechado.
Será executada somente após uma Redução Z.

3.6.1.10 int Bematech_FI_NomeiaRelatorioGerencialMFD (const char * *cIndice*, const char * *cDescricao*)

Nomeia Relatório Gerencial MFD.

Programa relatório gerencial.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_NomeiaRelatorioGerencialMFD("2", "Resumo de vendas");
```

Parâmetros:

cIndice Índice do relatório, sendo este um valor entre 2 e 30.

cDescricao Nome do relatório, com até 17 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A impressora possui um relatório default pré-programado: "Relatório Gerencial", no índice "01". Só será possível nomear um relatório gerencial, caso a impressora ainda não tenha iniciado o seu movimento.

Não é possível alterar ou apagar um relatório gerencial já gravado.

Na impressora fiscal Bematech MP-2100 TH FI, o tamanho da descrição das formas de pagamento foi reduzido de 17 para 15 caracteres. Para manter a compatibilidade, será aceito o tamanho de 17 caracteres, mas internamente a impressora registrará apenas os 15 caracteres.

3.6.1.11 int Bematech_FI_NomeiaTotalizadorNaoSujeitoIcms (int *indice*, const char * *totalizador*)

Nomeia Totalizador Não Sujeito ICMS.

Programar totalizadores não sujeitos ao ICMS.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_NomeiaTotalizadorNaoSujeitoIcms(5, "Conta de Luz");
```

Parâmetros:

indice Índice do totalizador a ser nomeado

totalizador Descrição do totalizador, com até 19 caracteres

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação

3.6.1.12 `int Bematech_FI_ProgramaAliquota (const char * aliquota, int vinculo)`

Programa Alíquota.

Programar alíquotas na impressora. A alíquota deve ser informada no formato XX.YY%, sem a pontuação. Por exemplo, para uma alíquota de 12,75%, deve-se informar "1275".

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaAliquota("0500", 1);
```

Parâmetros:

aliquota Valor da alíquota a ser programada.

vinculo 0 indica que a alíquota está vinculada ao ICMS e 1 ao ISS

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicacao
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2).
- 30 Função incompatível com a imp. yanco

Nota:

O cupom fiscal deve estar fechado.

Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI a programação da alíquota somente será permitida após uma Redução Z.

valida código

3.6.1.13 `int Bematech_FI_ProgramaArredondamento (void)`

Programa Arredondamento.

Programa o modo arredondamento na impressora. Este arredondamento se refere à venda de item com quantidade fracionária.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaArredondamento();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 30 Função incompatível com impressora Yanco.

Nota:

O cupom fiscal deve estar fechado.
 Será executada somente após uma Redução Z.
 Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI está programação não existe. Elas somente operam em modo de truncamento.

3.6.1.14 int Bematech_FI_ProgramaCaracterAutenticacao (const char * *parametros*)

Programa Caracter Autenticação.

Programar o caracter gráfico para autenticação.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaCaracterAutenticacao
  ("001,002,004,008,016,032,064,128,064, 032,016,008,004,002,129,129,129,129");
```

Parâmetros:

parametros String com os 18 valores para programação do caracter gráfico, separados por vírgula.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Exemplo de programação do caracter gráfico:

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
bit 0 (1)		X														X	X	X	X
bit 1 (2)			X																X
bit 2 (4)				X											X				
bit 3 (8)					X									X					
bit 4 (16)						X							X						
bit 5 (32)							X				X								
bit 6 (64)								X	X										
bit 7 (128)									X							X	X	X	X

Para programar o caracter acima deve-se passar a seguinte string de parâmetros: "001,002,004,008,016,032,064,128,064,032,016,008,004,002,129,129,129,129"

Uma vez programado, este caracter será mantido na memória da impressora mesmo que a mesma seja desligada.

3.6.1.15 `int Bematech_FI_ProgramaFormaPagamentoMFD (const char * cFormaPagamento, const char * cOperacaoTef)`

Programa Forma Pagamento MFD.

Programa as formas de pagamento.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaFormaPagamentoMFD("Cartao", "1");
```

Parâmetros:

cFormaPagamento Forma de pagamento a ser realizada, com até 16 caracteres.

cOperacaoTef Se setado para "1", indica que a forma de pagamento permite TEF. Se "0" a forma de pagamento não permite TEF.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Na impressora fiscal Bematech MP-2100 TH FI, o tamanho da descrição das formas de pagamento foi reduzido de 16 para 15 caracteres. Para manter a compatibilidade, será aceito o tamanho de 16 caracteres, mas internamente a impressora registrará apenas os 15 caracteres.

3.6.1.16 `int Bematech_FI_ProgramaHorarioVerao (void)`

Programa/Desprograma Horário Verão.

Programa e desprograma o horário de verão. Se a impressora já estiver no horário de verão o mesmo será desprogramado atrasando o relógio em 1 (uma) hora, caso contrário será adiantado 1 (uma) hora.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaHorarioVerao();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK,ST1,ST2)

Nota:

A programação do horário de verão será realizada somente após uma Redução Z. Para desprogramar, somente 1 (uma) hora após a Redução Z e não pode ter havido movimento na impressora nesse período.

3.6.1.17 int Bematech_FI_ProgramaIdAplicativoMFD (const char * *cIdAplicativo*)

Programa ID Aplicativo MFD.

Programa a identificação do aplicativo no cupom fiscal e comprovante não fiscal.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaIdAplicativoMFD("APPTESTE.EXE");
```

Parâmetros:

cIdAplicativo Identificação do nome do aplicativo, com até 84 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só será executada, caso o cupom fiscal esteja fechado. Não será necessária sua execução a cada início de dia. Função disponível para a impressora fiscal MP-2100 TH FI.

3.6.1.18 int Bematech_FI_ProgramaTruncamento (void)

Programa Truncamento.

Programar a impressora para o modo truncamento

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaTruncamento();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)
- 30 Função não compatível com a impressora Yanco

Nota:

O cupom fiscal deve estar fechado.
Será executada somente após uma Redução Z.
Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI está programação não existe. Elas somente operam em modo de truncamento.

3.6.1.19 int Bematech_FI_RetornoImpressoraMFD (short int * iACK, short int * iST1, short int * iST2, short int * iST3)

Retorno Impressora MFD. Ler o status da impressora após cada comando enviado.

Parâmetros:

- *iACK* ponteiro para o recebimento do ACK
- *iST1* ponteiro para o recebimento do
- *iST2* ponteiro para o recebimento do
- *iST3* ponteiro para o recebimento do

Retorna:

int com a informação sobre a execução do comando

3.6.1.20 int Bematech_FI_VerificaCancelamentoCupom2HorasMFD (char * cFlag)

Verifica Cancelamento Cupom 2 Horas MFD.

Verifica se o cancelamento automático do cupom às duas horas está habilitado ou desabilitado.

```
// Exemplo em C/C++
char cVerificaCancelamentoCupom2Horas[3];
int iRetorno;
iRetorno =
Bematech_FI_VerificaCancelamentoCupom2HorasMFD
(cVerificaCancelamentoCupom);
```

Parâmetros:

→ *cFlag* Variável, com 3 posições, destinada a receber a verificação referente ao cancelamento automático do cupom às duas horas habilitado "1" ou desabilitado "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7 Informações da Impressora

Módulos

- [Outras Informações da Impressora](#)
- [Informações MFD](#)
- [Informação](#)

Funções

- int [Bematech_FI_Acrescimos](#) (char *acrescimos)
- int [Bematech_FI_Cancelamentos](#) (char *cancelamentos)
- int [Bematech_FI_CGC_IE](#) (char *cgcie)
- int [Bematech_FI_ContadoresTotalizadoresNaoFiscais](#) (char *contadores)
- int [Bematech_FI_DadosUltimaReducao](#) (char *dados)
- int [Bematech_FI_Descontos](#) (char *descontos)
- int [Bematech_FI_FlagsFiscais](#) (short int *flag)
- int [Bematech_FI_GrandeTotal](#) (char *grandeTotal)
- int [Bematech_FI_MonitoramentoPapel](#) (int *linhas)
- int [Bematech_FI_NumeroCupom](#) (char *numeroCupom)
- int [Bematech_FI_NumeroCuponsCancelados](#) (char *numeroCancelamentos)
- int [Bematech_FI_NumeroReducoes](#) (char *reducoes)
- int [Bematech_FI_NumeroSerie](#) (char *numeroSerie)
- int [Bematech_FI_ReturnoAliquotas](#) (char *valorAliquotas)
- int [Bematech_FI_ReturnoImpressora](#) (short int *ack, short int *st1, short int *st2)
- int [Bematech_FI_ReturnoIndiceAliquota](#) (const char *aliquota)
- int [Bematech_FI_SubTotal](#) (char *subTotal)
- int [Bematech_FI_TotalIcmsCupom](#) (char *ICMS)
- int [Bematech_FI_UltimoItemVendido](#) (char *item)
- int [Bematech_FI_VerificaAliquotasIss](#) (char *flags)

- int `Bematech_FI_VerificaAliquotasIssImpAntiga` (char *cFlags)
- int `Bematech_FI_VerificaDepartamentos` (char *departamentos)
- int `Bematech_FI_VerificaFormasPagamento` (char *formasPagamento)
- int `Bematech_FI_VerificaIndiceAliquotasIss` (char *flags)
- int `Bematech_FI_VerificaIndiceAliquotasIssImpAntiga` (char *cIndices)
- int `Bematech_FI_VerificaRecebimentoNaoFiscal` (char *recebimentos)
- int `Bematech_FI_VerificaReducaoZAutomatica` (short *flag)
- int `Bematech_FI_VerificaSensorPoucoPapelMFD` (char *cFlag)
- int `Bematech_FI_VerificaTotalizadoresNaoFiscais` (char *totalizadores)
- int `Bematech_FI_VerificaTotalizadoresParciais` (char *totalizadores)
- int `Bematech_FI_VersaoFirmware` (char *versao)

3.7.1 Documentação das funções

3.7.1.1 int Bematech_FI_Acrescimos (char * *acrescimos*)

Acréscimos.

Retorna o valor acumulado dos acréscimos efetuados nos cupons.

```
// Exemplo em C/C++
char cAcrescimos[15];
int iRetorno;
iRetorno = Bematech_FI_Acrescimos(cAcrescimos);
```

Parâmetros:

- *acrescimos* Variável, com 15 caracteres, destinado a receber o valor acumulado dos acréscimos efetuados nos cupons.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.2 int Bematech_FI_Cancelamentos (char * *cancelamentos*)

Cancelamentos.

Retorna o valor acumulado dos itens e dos cupons cancelados.

```
// Exemplo em C/C++
char cCancelamentos[15];
int iRetorno;
iRetorno = Bematech_FI_Cancelamentos(cCancelamentos);
```

Parâmetros:

→ *cancelamentos* Variável, com 15 caracteres, destinada a receber o valor acumulado dos itens e dos cupons cancelados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.3 int Bematech_FI_CGC_IE (char * *cgcie*)

CGC/IE.

Retorna o CGC e a Inscrição Estadual do cliente/proprietário cadastrado na impressora.

```
// Exemplo em C/C++
char cGCC_IE[34];
int iRetorno;
iRetorno = Bematech_FI_CGC_IE(cGCC_IE);
```

Parâmetros:

→ *cgcie* Variável, com 34 caracteres, destinada a receber o CGC/IE cadastrado na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os primeiros 18 caracteres são referentes ao CGC cadastrado no equipamento, os 15 demais a inscrição estadual.

3.7.1.4 int Bematech_FI_ContadoresTotalizadoresNaoFiscais (char * *contadores*)

Contadores Totalizadores Não Fiscais.

Retorna o número de vezes em que cada totalizador não sujeito a ICMS foi utilizado.

```
// Exemplo em C/C++
char cContadoresTotalizadoresNF[45];
int iRetorno;
iRetorno =
Bematech_FI_ContadoresTotalizadoresNaoFiscais
(cContadoresTotalizadoresNF);
```

Parâmetros:

→ *contadores* Variável, com 45 caracteres, destinada a receber os contadores dos totalizadores.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A impressora fiscal permite a programação de até 50 totalizadores não fiscais, porém esta função retorna somente os contadores dos 9 primeiros totalizadores cadastrados.

O conteúdo da variável retornada será 36 dígitos separados de 4 em 4 por vírgula, representando os 9 primeiros totalizadores, por exemplo:

```
0001,0003,0001,0005,0004,0002,0003,0004,0007
```

O primeiro valor corresponde ao número de vezes que o totalizador 01 foi usado e assim sucessivamente.

3.7.1.5 int Bematech_FI_DadosUltimaReducao (char * *dados*)

Dados Última Redução Z.

Retorna os dados da impressora no momento da última redução Z.

```
// Exemplo em C/C++
char cDadosUltimaReducao[632];
int iRetorno;
iRetorno = Bematech_FI_DadosUltimaReducao(cDadosUltimaReducao);
```

Parâmetros:

→ *dados* Variável, com 632 caracteres, destinado a receber os dados da impressora no momento da última redução Z.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

São retornados os valores das seguintes informações separados por vírgulas:

```

Modo de Redução Z.....: 2 bytes
Grande Total.....: 18 bytes
Cancelamentos.....: 14 bytes
Descontos.....: 14 bytes
Tributos.....: 64 bytes
Totalizadores Parciais Tributados.....:266 bytes
Sangria.....: 14 bytes
Suprimentos.....: 14 bytes
Totalizadores não sujeitos ao ICMS.....:126 bytes
Contadores dos TP's não sujeitos ao ICMS..: 36 bytes
Contador de ordem de operação.....: 6 bytes
Contador de operações não sujeitas ao ICMS: 6 bytes
Número de Alíquotas cadastradas.....: 2 bytes
Data do movimento.....: 6 bytes
Acréscimos.....: 14 bytes
Acréscimo financeiro.....: 14 bytes

```

No campo "Modo de Redução Z" o valor "00" indica redução por comando e "01" redução automática. Nos campos "Grande Total", "Cancelamentos", "Descontos", "Sangria" e "Suprimentos", estão incluídas as 2 casas decimais.

No campo "Tributos" o retorno são de 16 alíquotas x 4 dígitos.

O valor do campo "Totalizadores Parciais Tributados" inclui as alíquotas Isenção (II), Não Incidência (NN) e Substituição Tributária (FF).

A estrutura do campo comentado acima é de 16 alíquotas (224 bytes) mais II, NN e FF (42 bytes). No total 19 alíquotas são retornadas. Para cada alíquota, são 14 dígitos para o valor, incluindo 2 casas decimais.

No campo "Totalizadores não sujeitos ao ICMS", o retorno são 9 totalizadores não fiscais, com 14 dígitos, incluindo as 2 casas decimais.

O campo "Contadores dos TP's não sujeitos ao ICMS", indica quanto cada totalizador não fiscal foi utilizado, sendo 9 totalizadores de 4 dígitos cada.

O campo "Acréscimos" informa o que foi concedido na venda do item ou no fechamento do cupom fiscal, com 14 dígitos, incluindo as 2 casas decimais.

O campo "Acréscimo financeiro" é um totalizador extinto nas versões 3.10 ou posterior, das impressoras fiscais.

3.7.1.6 int Bematech_FI_Descontos (char * *descontos*)

Descontos.

Retorna o valor acumulado dos descontos.

```
// exemplo em C/C++
char cDescontos[15];
int iRetorno;
iRetorno = Bematech_FI_Descontos(cDescontos);
```

Parâmetros:

→ *descontos* Variável, com 15 caracteres, destinada a receber o valor acumulado dos descontos com 2 casas decimais.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.7 int Bematech_FI_FlagsFiscais (short int * *flag*)

Flags Fiscais.

Retorna um número referente ao flag fiscal da impressora.

```
// Exemplo em C/C++
int iRetorno, iFlagFiscais;
iRetorno = Bematech_FI_FlagsFiscais(iFlagFiscais);
```

Parâmetros:

→ *flag* Variável, inteira, destinada a receber um número referente ao flag fiscal da impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Discriminação dos valores para o flag fiscal:

Descrição	Valor
Cupom fiscal aberto	1
Fechamento de formas de pagamento iniciado	2
Horário de verão selecionado	4
Já houve redução Z no dia	8
Não utilizado	16
Permite cancelar cupom fiscal	32
Não utilizado	64
Memória fiscal sem espaço	128

Os valores podem vir somados, o que indica mais de um estado.

3.7.1.8 int Bematech_FI_GrandeTotal (char * *grandeTotal*)

Grande Total.

Retorna o grande total da impressora.

```
// Exemplo em C/C++
char cGrandeTotal[19];
int iRetorno;
iRetorno = Bematech_FI_GrandeTotal(cGrandeTotal);
```

Parâmetros:

→ *grandeTotal* Variável, com 19 caracteres, destinada a receber o grande total da impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.9 int Bematech_FI_MonitoramentoPapel (int * *linhas*)

Monitoramento Papel.

Retorna o número de linhas impressas após o status de pouco papel.

```
// Exemplo em C/C++
int iRetorno, int iMonitoramentoPapel;
iRetorno = Bematech_FI_MonitoramentoPapel(iMonitoramentoPapel);
```

Parâmetros:

→ *linhas* Variável, inteira, destinado a receber o número de linhas impressas após o status de pouco papel.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.10 int Bematech_FI_NumeroCupom (char * *numeroCupom*)

Número Cupom.

Retorna o número do último cupom emitido.

```
// Exemplo em C/C++
char cNumeroCupom[7];
int iRetorno;
iRetorno = Bematech_FI_NumeroCupom(cNumeroCupom);
```

Parâmetros:

→ *numeroCupom* Variável, com 7 caracteres, destinada a receber o número do último cupom emitido.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.11 int Bematech_FI_NumeroCuponsCancelados (char * *numeroCancelamentos*)

Número Cupons Cancelados.

Retorna o número de cupons cancelados.

```
// exemplo em C/C++
char cNumeroCuponsCancelados[5];
int iRetorno;
iRetorno =
Bematech_FI_NumeroCuponsCancelados(cNumeroCuponsCancelados);
```

Parâmetros:

→ *numeroCancelamentos* Variável, com 5 caracteres, destinada a receber o número de cupons cancelados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.12 int Bematech_FI_NumeroReducoes (char * *reducoes*)

Número Reduções Z.

Retorna o número de reduções Z realizadas na impressora.

```
// exemplo em C/C++
char cNumeroReducoes[5];
int iRetorno;
iRetorno = Bematech_FI_NumeroReducoes(cNumeroReducoes);
```

Parâmetros:

→ *reducoes* Variável, com 5 caracteres, destinada a receber o número de reduções Z realizadas na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.13 int Bematech_FI_NumeroSerie (char * *numeroSerie*)

Número Série.

Retorna o número de série da impressora.

```
// Exemplo em C/C++
char cNumeroSerie[16];
int iRetorno;
iRetorno = Bematech_FI_NumeroSerie(cNumeroSerie);
```

Parâmetros:

→ *numeroSerie* Variável, com 16 caracteres, destinada a receber o número de série da impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Nas impressoras fiscasi MP-2000 TH FI, MP-25 TH FI e MP-50 TH FI, serão retornados os 15 primeiros caracteres de seu número serial. O número serial nestas impressoras é alfanumérico com o tamanho de 20 caracteres.

3.7.1.14 int Bematech_FI_RetornoAliquotas (char * *valorAliquotas*)

Retorno Aliquotas.

Retorna as alíquotas cadastradas na impressora.

```
// Exemplo em C/C++
char cRetornoAliquotas[80];
int iRetorno;
iRetorno = Bematech_FI_RetornoAliquotas(cRetornoAliquotas);
```

Parâmetros:

→ *valorAliquotas* Variável, com 80 caracteres, destinada a receber o retorno das alíquotas cadastradas na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação

- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As alíquotas retornadas estarão separadas por vírgula. Ex: 1200,1700,0500,1800.

3.7.1.15 int Bematech_FI_RetornoImpressora (short int * ack, short int * st1, short int * st2)

Retorno Impressora.

Lê o retorno da impressora referente ao último comando enviado.

```
// exemplo em C/C++
int iRetorno, iACK = 0, iST1 = 0, iST2 = 0;
iRetorno = Bematech_FI_RetornoImpressora(iACK, iST1, iST2);
```

Parâmetros:

- *ack* Variável, inteira, destinada a receber o valor do registrador ACK.
- *st1* Variável, inteira, destinada a receber o valor do registrador ST1.
- *st2* Variável, inteira, destinad a receber o valor do registrador ST2.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT

Nota:

Essa função deve ser usada após a execução de qualquer outra função, para ler o status da impressora, referente a função executada.

Se o retorno o ACK, ST1 e ST2 for "0,0,0", corresponde que a impressora não possui nenhum status de execução, significando que nenhuma função fora executada anteriormente.

3.7.1.16 int Bematech_FI_RetornoIndiceAliquota (const char * aliquota)

Retorno Índice Alíquota.

Retorna o índice em que a aliquota está cadastrada na impressora.

```
// Exemplo em C/C++
int iRetornoIndiceAliquota;
iRetornoIndiceAliquota = Bematech_FI_RetornoIndiceAliquota("1200");
```

Parâmetros:

- aliquota* Alíquota a ser pesquisada.

Retorna:

Índice em que a alíquota está cadastrada na impressora.

Nota:

O retorno igual a "0" indica alíquota não cadastrada.

3.7.1.17 int Bematech_FI_SubTotal (char * subTotal)

SubTotal.

Retorna o valor do subtotal do cupom.

```
// Exemplo em C/C++
char cSubTotal[15];
iRetorno = Bematech_FI_SubTotal(cSubTotal);
```

Parâmetros:

→ *subTotal* Variável, com 15 caracteres, destinada a receber o retorno do valor do subtotal do cupom.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução da função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.18 int Bematech_FI_TotalIcmsCupom (char * ICMS)

Total ICMS Cupom.

Lê o valor total do ICMS pago no cupom.

```
// Exemplo em C/C++
char cTotalIcmsCupom[14];
int iRetorno;
iRetorno = Bematech_FI_TotalIcmsCupom(cTotalIcmsCupom);
```

Parâmetros:

→ *ICMS* Variável, com 14 caracteres, destinada a receber o valor total do ICMS pago no cupom.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar no arquivo RetornoStatus STATUS.TXT ou RETORNO.TXT

3.7.1.19 int Bematech_FI_UltimoItemVendido (char * *item*)

Último Item Vendido.

Retorna o número do último item vendido.

```
// Exemplo em C/C++
char cNumeroUltimoItemVendido[5];
int iRetorno;
iRetorno = Bematech_FI_UltimoItemVendido(cNumeroUltimoItemVendido);
```

Parâmetros:

→ *item* Variável, com 5 caracteres, destinada a receber o número do último item vendido.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.20 int Bematech_FI_VerificaAliquotasIss (char * *flags*)

Verifica Alíquota ISS.

Retorna as alíquotas de vinculação ao ISS.

```
// exemplo em C/C++
char cAliquotasISS[80];
int iRetorno;
iRetorno = Bematech_FI_VerificaAliquotasIss(cAliquotasISS);
```

Parâmetros:

→ *flags* Variável, com 80 caracteres, destinado a receber as alíquotas de vinculação ao ISS.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.21 int Bematech_FI_VerificaAliquotasIssImpAntiga (char * *cFlags*)

Verifica Alíquotas ISS Impressora Antiga.

Retorna as alíquotas de ISS.

```
// Exemplo em C/C++
char cAliquotasISSImpAntiga[82];
int iRetorno;
iRetorno =
Bematech_FI_VerificaAliquotasIssImpAntiga(cAliquotasISSImpAntiga);
```

Parâmetros:

- *cFlags* Variável, com 82 posições, destinada a receber os flags de vinculação ISS (Indicam quais alíquotas são ISS).

Retorna:

- int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.22 int Bematech_FI_VerificaDepartamentos (char * *departamentos*)

Verifica Departamentos.

Retorna os departamentos e seus valores acumulados.

```
// Exemplo em C/C++
char cDepartamentos[1020];
int iRetorno;
iRetorno = Bematech_FI_VerificaDepartamentos(cDepartamentos);
```


Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função retorna as informações das formas de pagamento na seguinte ordem:

```
Descricao da forma de pagamento.....
Valor acumulado (2 casas decimais).....
Valor recebido no último cupom (2 casas decimais).....
Valor indicando se a forma foi usada para a emissão do cupom não fiscal vinculado (1 usado, 0 não us
```

São retornadas 50 formas de pagamento mais a forma "Valor Recebido" e "Troco" obedecendo a ordem descrita anteriormente. As formas de pagamento são separadas por vírgula, como no exemplo abaixo:

```
Dinheiro      0000000000000000415580000000000000000000000015580,
Cartao Credito 0000000000000000003947000000000000000000000000000000,
Cheque        0000000000000000002894000000000000000000000000000000,
Ticket        0000000000000000000900000000000000000000000000000000,
              00000000000000000000000000000000000000000000000000000,
              00000000000000000000000000000000000000000000000000000,
              .
              .
              .
Valor Recebido 0000000000000000001189680000000000000000000000000000,
Troco          00000000000000000000000000000000000000000000000000000
```

A impressora permite programar até 50 formas de pagamento.

As formas que não estiverem programadas estarão com os valores zerados e a descrição em branco, como mostrado no exemplo acima.

Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI serão retornadas somente 20 formas de pagamento, que é a quantidade permitida, as demais posições serão retornadas com espaços em branco.

3.7.1.24 int Bematech_FI_VerificaIndiceAliquotasIss (char * flags)

Verifica Índice Alíquota ISS.

Retorna os índices das alíquotas de vinculação ao ISS.

```
// Exemplo em C/C++
char VerificaIndiceAliquotasIss[49];
int iRetorno;
iRetorno =
Bematech_FI_VerificaIndiceAliquotasIss(VerificaIndiceAliquotasIss);
```

Parâmetros:

→ *flags* Variável, com 49 caracteres, destinado a receber os índices das alíquotas de vinculação ao ISS.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os índices retornados estarão separados por vírgula, por exemplo: 03,05.

3.7.1.25 int Bematech_FI_VerificaIndiceAliquotasIssImpAntiga (char * cIndices)

Verifica Índice Alíquotas ISS Impressora Antiga.

Retorna a posição das alíquotas de ISS.

```
// Exemplo em C/C++
char cIndiceAliquotasISSImpAntiga[49];
int iRetorno;
iRetorno =
Bematech_FI_VerificaIndiceAliquotasIssImpAntiga
(cIndiceAliquotasISSImpAntiga);
```

Parâmetros:

→ *cIndices* Variável, com 49 posições, destinada a receber a posição das alíquotas de ISS.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os índices retornados estarão separados por vírgula. Por exemplo: 01,03,05.

3.7.1.26 int Bematech_FI_VerificaRecebimentoNaoFiscal (char * *recebimentos*)

Verifica Recebimento Não Fiscal.

Retorna os recebimentos não fiscais não vinculados programados na impressora.

```
// Exemplo em C/C++
char cRecebimentoNF[2201];
int iRetorno;
iRetorno = Bematech_FI_VerificaRecebimentoNaoFiscal(cRecebimentoNF);
```

Parâmetros:

→ *recebimentos* Variável, com 2201 caracteres, destinado a receber os recebimentos não fiscais não vinculados programados na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função retorna as informações para os 50 totalizadores não fiscais não vinculados na seguinte ordem:

```
Valor indicando quantas vezes cada recebimento foi utilizado: 4 bytes
Valor acumulado para cada recebimento (2 casas decimais)....: 20 bytes
Descrição do recebimento.....: 19 bytes
```

Segue abaixo o exemplo de um retorno desta função:

```
000200000000000000000005460Conta de água      ,
0015000000000000000000145850Conta de Luz      ,
0000000000000000000000000000000000000000    ,
0000000000000000000000000000000000000000    ,
0000000000000000000000000000000000000000    ,
0000000000000000000000000000000000000000    ,
0005000000000000000000078437Conta de Telefone ,
.
.
.
0000000000000000000000000000000000000000    ,
Prestacao de contas
```

A impressora permite programar até 50 totalizadores não fiscais não vinculados.

Os totalizadores que não estiverem programados estarão com os valores zerados e a descrição em branco, conforme exemplo acima.

Nas impressoras fiscais MP-2000 TH FI, MP-6000 TH FI, MP-25 FI e MP-50 FI serão retornados somente 30 recebimentos não fiscais, que é a quantidade permitida, as demais posições serão retornadas com espaços em branco.

3.7.1.27 int Bematech_FI_VerificaReducaoZAutomatica (short *flag)

Verifica Redução Z Automática.

Verifica se a última redução Z foi executada automaticamente.

```
// Exemplo em C/C++
int iVerificaReducaoZAutomatica = 0;
int iRetorno;
iRetorno =
Bematech_FI_VerificaReducaoZAutomatica(iVerificaReducaoZAutomatica);
```

Parâmetros:

→ **flag** Variável, inteira, destinada a receber a verificação da execução de redução Z automática "1" ou por comando "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 4** Arquivo de inicialização não encontrado ou inválido
- 5** Erro ao abrir a porta de comunicação
- 8** Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27** Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.28 int Bematech_FI_VerificaSensorPoucoPapelMFD (char *cFlag)

Verifica Sensor Pouco Papel MFD.

Verifica se o sensor de pouco papel está habilitado ou desabilitado.

```
// Exemplo em C/C++
char cVerificaSensorPoucoPapel[3];
int iRetorno;
iRetorno =
Bematech_FI_VerificaSensorPoucoPapelMFD(cVerificaSensorPoucoPapel);
```

Parâmetros:

→ **cFlag** Variável, com 3 posições, destinada a receber a verificação referente ao sensor de pouco papel habilitado "1" ou desabilitado "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 1** Erro de execução

- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.7.1.29 `int Bematech_FI_VerificaTotalizadoresNaoFiscais (char * totalizadores)`

Verifica Totalizadores Não Fiscais.

Retorna a descrição dos totalizadores não fiscais programados na impressora.

```
// Exemplo em C/C++
char cTotalizadoresNF[180];
int iRetorno;
iRetorno =
Bematech_FI_VerificaTotalizadoresNaoFiscais(cTotalizadoresNF);
```

Parâmetros:

- *totalizadores* Variável, com 180 caracteres, destinada a receber a descrição dos totalizadores não fiscais programados.

Retorna:

- int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

- Os totalizadores virão separados por vírgula.
- A impressora permite a programação de até 50 totalizadores não fiscais, porém esta função retorna somente os 9 primeiros cadastrados.

3.7.1.30 `int Bematech_FI_VerificaTotalizadoresParciais (char * totalizadores)`

Verifica Totalizadores Parciais.

Retorna os totalizadores parciais cadastrados na impressora.

```
// Exemplo em C/C++
char cTotalizadoresParciais[446];
int iRetorno;
iRetorno =
Bematech_FI_VerificaTotalizadoresParciais(cTotalizadoresParciais);
```

Parâmetros:

→ *totalizadores* Variável, com 446 caracteres, destinada a receber os totalizadores parciais cadastrados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função retorna as seguintes informações, separadas por vírgula:

```
Totalizadores parciais tributados.....: 224 bytes
Isenção.....: 14 bytes
Não incidência.....: 14 bytes
Substituição.....: 14 bytes
Totalizadores parciais não sujeitos ao ICMS: 126 bytes
Sangria.....: 14 bytes
Suprimento.....: 14 bytes
Grande Total.....: 18 bytes
```

3.7.1.31 int Bematech_FI_VersaoFirmware (char * versao)

Versão Firmware.

Retorna a versão do firmware da impressora.

```
// Exemplo em C/C++
char cVersaoFirmware[5];
int iRetorno;
iRetorno = Bematech_FI_VersaoFirmware(cVersaoFirmware);
```

Parâmetros:

→ *versao* Variável, com 5 caracteres, destinada a receber a versão do firmware da impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Nas impressoras fiscasi MP-2000 TH FI, MP-6000 TH FI, MP-25 TH FI e MP-50 TH FI, serão retornados os 4 primeiros caracteres de sua versão de firmware. A versão de firmware, nestas impressoras, tem o tamanho de 6 caracteres.

3.8 Outras Informações da Impressora

Funções

- int `Bematech_FI_ClicheProprietario` (char *cliche)
- int `Bematech_FI_ContadorBilhetePassagem` (char *contadorBilhete)
- int `Bematech_FI_ContadorRelatoriosGerenciaisMFD` (char *cContadorRelatorios)
- int `Bematech_FI_DadosUltimaReducaoMFD` (char *dados)
- int `Bematech_FI_DataHoraImpressora` (char *data, char *hora)
- int `Bematech_FI_DataHoraReducao` (char *data, char *hora)
- int `Bematech_FI_DataMovimento` (char *data)
- int `Bematech_FI_MarcaModeloTipoImpressoraMFD` (char *marca, char *modelo, char *tipo)
- int `Bematech_FI_MinutosImprimindo` (char *minutos)
- int `Bematech_FI_MinutosLigada` (char *minutos)
- int `Bematech_FI_NumeroCaixa` (char *caixa)
- int `Bematech_FI_NumeroIntervencoes` (char *intervencoes)
- int `Bematech_FI_NumeroLoja` (char *loja)
- int `Bematech_FI_NumeroOperacoesNaoFiscais` (char *operacoes)
- int `Bematech_FI_NumeroSerieMFD` (char *numeroSerie)
- int `Bematech_FI_NumeroSubstituicoesProprietario` (char *substituicoes)
- int `Bematech_FI_SimboloMoeda` (char *moeda)
- int `Bematech_FI_ValorFormaPagamento` (const char *formaPagamento, char *valorForma)
- int `Bematech_FI_ValorPagoUltimoCupom` (char *valor)
- int `Bematech_FI_ValorTotalizadorNaoFiscal` (const char *totalizador, char *valorTotalizador)
- int `Bematech_FI_VerificaEpromConectada` (char *flag)
- int `Bematech_FI_VerificaEstadoImpressora` (short int *ack, short int *st1, short int *st2)
- int `Bematech_FI_VerificaModoOperacao` (char *modo)
- int `Bematech_FI_VerificaTipoImpressora` (short int *tipoImpressora)
- int `Bematech_FI_VerificaTruncamento` (char *flag)

3.8.1 Documentação das funções

3.8.1.1 int Bematech_FI_ClicheProprietario (char * *cliche*)

Clichê Proprietário.

Retorna o clichê do proprietário cadastrado na impressora.

```
// Exemplo em C/C++
char cClicheProprietario[187];
int iRetorno;
iRetorno = Bematech_FI_ClicheProprietario(cClicheProprietario);
```

Parâmetros:

→ *cliche* Variável, com 187 caracteres, destinada a receber o clichê do proprietário cadastrado na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.2 int Bematech_FI_ContadorBilhetePassagem (char * contadorBilhete)

Contador Bilhete Passagem.

Retorna o número de bilhetes de passagem emitidas.

```
// Exemplo em C/C++
char cContadorBilhetePassagem[7];
int iRetorno;
iRetorno =
Bematech_FI_ContadorBilhetePassagem(cContadorBilhetePassagem);
```

Parâmetros:

→ *contadorBilhete* Variável, com 7 caracteres, destinado a receber o número de bilhetes de passagem emitidas.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.3 int Bematech_FI_ContadorRelatoriosGerenciaisMFD (char * *cContadorRelatorios*)

Contador Relatórios Gerenciais MFD.

Lê o contador de relatórios gerenciais na MFD.

```
// Exemplo em C/C++
char cContadorRelatoriosGerenciais[7];
int iRetorno;
iRetorno =
Bematech_FI_ContadorRelatoriosGerenciaisMFD
(cContadorRelatoriosGerenciais);
```

Parâmetros:

→ *cContadorRelatorios* Variável, com 7 posições, destinada a receber o número de relatórios gerenciais emitidos.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.4 int Bematech_FI_DadosUltimaReducaoMFD (char * *dados*)

Dados Última Redução MFD.

Ler os dados da última Redução Z.

```
// exemplo em C/C++
int iRetorno;
char sDados[1279];
iRetorno = Bematech_FI_DadosUltimaReducaoMFD (sDados);
```

Parâmetros:

→ *dados* Buffer, com 1279 caracteres, destinado a receber os dados da última redução.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo ini não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (Ack, St1, St2 e St3)

Nota:

São retornados os valores das seguintes informações separados por vírgulas:

Modo de redução Z:	2 bytes (se 00 redução por comando se 01
Contador de reinício de operação:	4 bytes
Contador de redução z:	4 bytes
Contador de ordem de operação:	6 bytes
Contador Geral de operações não fiscais:	6 bytes
Contador de cupom fiscal:	6 bytes
Contador Geral de relatório gerencial:	6 bytes
Contador de fita detalhe emitida:	6 bytes
Contador de operação não fiscal cancelada:	4 bytes
Contador de cupom fiscal cancelado:	4 bytes
Contadores específicos de operações não fiscais:	120 bytes
Contadores específicos de relatórios gerenciais:	120 bytes
Contador de comprovantes de débito ou crédito:	4 bytes
Contador de comprovantes de débito ou crédito não emitidos:	4 bytes
Contador de comprovantes de débito ou crédito cancelados:	4 bytes
Totalizador geral:	18 bytes
Totalizadores Parciais Tributados:	224 bytes
Totalizador de isenção de ICMS:	14 bytes
Totalizador de não incidência de ICMS:	14 bytes
Totalizador de substituição tributária de ICMS:	14 bytes
Totalizador de isenção de ISSQN:	14 bytes
Totalizador de não incidência de ISSQN:	14 bytes
Totalizador de substituição tributária de ISSQN:	14 bytes
Totalizador de descontos em ICMS:	14 bytes
Totalizador de descontos em ISSQN:	14 bytes
Totalizador de acréscimos em ICMS:	14 bytes
Totalizador de acréscimos em ISSQN:	14 bytes
Totalizador de cancelamentos em ICMS:	14 bytes
Totalizador de cancelamentos em ISSQN:	14 bytes
Totalizadores parciais não sujeitos ao ICMS:	392 bytes
Totalizador de sangria:	14 bytes
Totalizador de suprimento:	14 bytes
Totalizador de cancelamentos de não fiscais:	14 bytes
Totalizador de descontos de não fiscais:	14 bytes
Totalizador de acréscimos de não fiscais:	14 bytes
Alíquotas tributárias:	64 bytes
Data do movimento:	6 bytes

3.8.1.5 int Bematech_FI_DataHoraImpressora (char * data, char * hora)

Data Hora Impressora.

Retorna a data e a hora atual da impressora.

```
// Exemplo em C/C++
char cData[7];
char cHora[7];
int iRetorno;
iRetorno = Bematech_FI_DataHoraImpressora(cData, cHora);
```

Parâmetros:

- **data** Variável, com 7 caracteres, destinada a receber a data atual da impressora no formato ddmmaa.
- **hora** Variável, com 7 caracteres, destinada a receber a hora atual da impressora no formato hh-mmss.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.6 int Bematech_FI_DataHoraReducao (char * data, char * hora)

Data Hora Redução Z.

Retorna a data e a hora da última Redução Z.

```
// Exemplo em C/C++
char cData[7];
char cHora[7];
int iRetorno;
iRetorno = Bematech_FI_DataHoraReducao(cData, cHora);
```

Parâmetros:

- *data* Variável, com 7 caracteres, destinada a receber a data da última redução no formato ddmmaa.
- *hora* Variável, com 7 caracteres, destinada a receber a hora da última redução no formato hh-mmss.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.7 int Bematech_FI_DataMovimento (char * data)

Data Movimento.

Retorna a data do último movimento.

```
// Exemplo em C/C++
char cData[7];
int iRetorno;
iRetorno = Bematech_FI_DataMovimento(cData);
```

Parâmetros:

→ *data* Variável, com 7 caracteres, destinada a receber a data do último movimento.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.8 int Bematech_FI_MarcaModeloTipoImpressoraMFD (char * *marca*, char * *modelo*, char * *tipo*)

Marca Modelo Tipo Impressora MFD.

Retorna a marca, o modelo e o tipo da impressora.

```
// exemplo em C/C++
int iRetorno;
char sMarca[16];
char sModelo[21];
char sTipo[8];
iRetorno =
Bematech_FI_MarcaModeloTipoImpressoraMFD(sMarca, sModelo, sTipo);
```

Parâmetros:

- *marca* Buffer com 16 bytes para receber a marca da impressora.
- *modelo* Buffer com 21 bytes para receber o modelo da impressora.
- *tipo* Buffer com 8 bytes para receber o tipo da impressora.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo ini não encontrado ou parâmetro inválido para o nome da porta
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (Ack, St1, St2 e St3)

3.8.1.9 int Bematech_FI_MinutosImprimindo (char * minutos)

Minutos Imprimindo.

Retorna o tempo em que a impressora está ou esteve imprimindo.

```
// Exemplo em C/C++
char cMinutosImprimindo[5];
int iRetorno;
iRetorno = Bematech_FI_MinutosImprimindo(cMinutosImprimindo);
```

Parâmetros:

→ *minutos* Variável, com 5 caracteres, destinada a receber o tempo em que a impressora está ou esteve imprimindo.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.10 int Bematech_FI_MinutosLigada (char * minutos)

Minutos Ligada.

Retorna o tempo em que a impressora está ligada.

```
// Exemplo em C/C++
char cMinutosLigada[5];
int iRetorno;
iRetorno = Bematech_FI_MinutosLigada(cMinutosLigada);
```

Parâmetros:

→ *minutos* Variável, com 5 caracteres, destinada a receber o tempo em que a impressora está ligada.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.11 int Bematech_FI_NumeroCaixa (char * caixa)

Número Caixa.

Retorna o número do caixa cadastrado na impressora.

```
// Exemplo em C/C++
char cNumeroCaixa[5];
int iRetorno;
iRetorno = Bematech_FI_NumeroCaixa(cNumeroCaixa);
```

Parâmetros:

→ *caixa* Variável, com 5 caracteres, destinada a receber o número do caixa cadastrado na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.12 int Bematech_FI_NumeroIntervencoes (char * intervencoes)

Número Intervenções.

Retorna o número de intervenções técnicas feitas na impressora.

```
// Exemplo em C/C++
char cNumeroIntervencoes[5];
int iRetorno;
iRetorno = Bematech_FI_NumeroIntervencoes(cNumeroIntervencoes);
```

Parâmetros:

→ *intervencoes* Variável, com 5 caracteres, destinada a receber o número de intervenções técnicas feitas na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.13 int Bematech_FI_NumeroLoja (char * loja)

Número Loja.

Retorna o número da loja cadastrada na impressora.

```
// Exemplo em C/C++
char cNumeroLoja[5];
int iRetorno;
iRetorno = Bematech_FI_NumeroLoja(cNumeroLoja);
```

Parâmetros:

→ *loja* Variável, com 5 caracteres, destinada a receber o número da loja cadastrada na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.14 int Bematech_FI_NumeroOperacoesNaoFiscais (char * operacoes)

Número Operações Não Fiscais.

Retorna o número de operações não fiscais executadas na impressora.

```
// Exemplo em C/C++
char cNumeroOperacoesNF[7];
int iRetorno;
iRetorno = Bematech_FI_NumeroOperacoesNaoFiscais(cNumeroOperacoesNF);
```

Parâmetros:

→ *operacoes* Variável, com 7 caracteres, destinada a receber o número de operações não fiscais executadas na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.15 int Bematech_FI_NumeroSerieMFD (char * *numeroSerie*)

Número Série MFD.

Ler o número de série das impressoras convenio novo - 20 bytes alfanuméricos

```
// exemplo em C/C++
int iRetorno;
char sNumeroSerie[21];
iRetorno = Bematech_FI_NumeroSerieMFD(sNumeroSerie);
```

Parâmetros:

→ *numeroSerie* Buffer com 21 bytes para receber o número de série da impressora.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo ini não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (Ack, St1, St2 e St3).

3.8.1.16 int Bematech_FI_NumeroSubstituicoesProprietario (char * *substituicoes*)

Número Substituições Proprietário.

Retorna o número de substituições de proprietário.

```
// Exemplo em C/C++
char cNumeroSubstituicoesProprietario[5];
int iRetorno;
iRetorno =
Bematech_FI_NumeroSubstituicoesProprietario
(cNumeroSubstituicoesProprietario);
```

Parâmetros:

→ *substituicoes* Variável, com 5 caracteres, destinada a receber o número de substituições de proprietário.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.17 int Bematech_FI_SimboloMoeda (char * moeda)

Símbolo Moeda.

Retorna o símbolo da moeda cadastrado na impressora.

```
// Exemplo em C/C++
char cSimboloMoeda[3];
int iRetorno;
iRetorno = Bematech_FI_SimboloMoeda(cSimboloMoeda);
```

Parâmetros:

→ *moeda* Variável, com 3 caracteres, destinada a receber o símbolo da moeda cadastrado na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.18 int Bematech_FI_ValorFormaPagamento (const char * formaPagamento, char * valorForma)

Valor Forma Pagamento.

Ler o valor acumulado em uma determinada forma de pagamento.

```
// exemplo em C/C++
int iRetorno;
char Valor[16];
iRetorno = Bematech_FI_ValorFormaPagamento("Cheque", Valor);
```

Parâmetros:

formaPagamento Forma de pagamento a ser lida, com até 16 caracteres.

→ *valorForma* Buffer com 16 bytes para receber o valor da forma de pagamento.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 2 Parâmetro inválido na função
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.8.1.19 int Bematech_FI_ValorPagoUltimoCupom (char * valor)

Valor Pago Último Cupom.

Retorna o valor pago no último cupom.

```
// Exemplo em C/C++
char cValorPagoUltimoCupom[15];
int iRetorno;
iRetorno = Bematech_FI_ValorPagoUltimoCupom(cValorPagoUltimoCupom);
```

Parâmetros:

→ **valor** Variável, com 15 caracteres, destinada a receber o valor pago no último cupom.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 1** Erro de execução da função
- 4** Arquivo de inicialização não encontrado ou inválido
- 5** Erro ao abrir a porta de comunicação
- 8** Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27** Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Valor pago no último cupom é diferente do valor total do cupom, por exemplo: o Valor do cupom é de R\$15,00 e o cliente pagou R\$20,00. O valor retornado por esta função será o valor pago pelo cliente, ou seja, R\$20,00.

3.8.1.20 int Bematech_FI_ValorTotalizadorNaoFiscal (const char * totalizador, char * valorTotalizador)

Valor Totalizador Não Fiscal.

Ler o valor acumulado de um determinado totalizador não fiscal.

```
// exemplo em C/C++
int iRetorno;
char Valor[15];
iRetorno =
Bematech_FI_ValorTotalizadorNaoFiscal("Rec. Prestacao", Valor);
```

Parâmetros:

totalizador Totalizador, com até 19 caracteres.

→ **valorTotalizador** Buffer com 15 bytes para receber o valor do totalizador.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 2 Parâmetro inválido na função
- 25 Totalizador não fiscal não programado
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.8.1.21 int Bematech_FI_VerificaEpromConectada (char *flag)

Verifica EPROM Conectada.

Verifica se a Eprom fiscal está conectada.

```
// Exemplo em C/C++
char cEpromConectada[2];
int iRetorno;
iRetorno = Bematech_FI_VerificaEpromConectada(cEpromConectada);
```

Parâmetros:

- **flag** Variável, com 2 caracteres, destinada a receber o flag de Eprom conectada "1" ou desconectada "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os retornos provenientes da impressora são: Eprom conectada "55h" e desconcetada "AAh".

3.8.1.22 int Bematech_FI_VerificaEstadoImpressora (short int *ack, short int *st1, short int *st2)

Verifica Estado Impressora.

Retorna o estado da impressora.

```
// Exemplo em C/C++
int iRetorno, iACK, iST1, iST2;
iRetorno = Bematech_FI_VerificaEstadoImpressora(iACK, iST1, iST2);
```

Parâmetros:

- *ack* Variável, inteira, destinada a receber o primeiro byte de retorno.
- *st1* Variável, inteira, destinada a receber o segundo byte de retorno.
- *st2* Variável, inteira, destinada a receber o terceiro byte de retorno.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Todas as funções, executadas na impressora, possuem bytes de retorno, que significam o estado atual após o envio.

Cada byte é tratado da seguinte maneira:

ACK (06h ou 6d)	: Byte indicativo de recebimento correto.
ST1 e ST2	: Bytes de estado da impressora.
NAK (15h ou 21d)	: Byte indicativo de recebimento incorreto.

Nas impressoras fiscais matriciais, os retornos de execução são obtidos, apenas, pelos bytes ACK, ST1 e ST2.

Nas impressoras fiscal MFD (somente), existe um outro retorno chamado ST3 (retorno estendido), onde informações mais detalhadas podem ser obtidas, sendo este habilitado pela função [Bematech_FI_HabilitaDesabilitaRetornoEstendidoMFD\(\)](#).

Abaixo seguem as estruturas do ST1 e ST2:

ST1:

Composição do Byte de Retorno	Mensagem da Impressora	Valor de Retorno
bit 7	Fim de papel	128
bit 6	Pouco papel	64
bit 5	Erro no relógio	32
bit 4	Impressora em erro	16
bit 3	Primeiro dado de CMD não foi ESC (18h)	8
bit 2	Comando inexistente	4
bit 1	Cupom fiscal aberto	2
bit 0	Número de parâmetro de CMD inválido	1

ST2:

Composição do Byte de Retorno	Mensagem da Impressora	Valor de Retorno
bit 7	Tipo de parâmetro de CMD inválido	128
bit 6	Memória fiscal lotada	64
bit 5	Erro na memória RAM CMOS não volátil	32
bit 4	Alíquota não programada	16
bit 3	Capacidade de alíquotas esgotada	8
bit 2	Cancelamento não permitido	4
bit 1	CNPJ/IE do proprietário não programados	2
bit 0	Comando não executado	1

Lógica de Tratamento:

Cada byte é composto de 8 bits.

Cada bit, dentro do byte, tem um valor, conforme a tabela acima.

O valor recebido da impressora para ST1 e/ou ST2 deve ser comparado com cada bit.

3.8.1.23 int Bematech_FI_VerificaModoOperacao (char * modo)

Verifica Modo Operação.

Verifica se a impressora está em modo normal ou em intervenção técnica.

```
// exemplo em C/C++
char cModoOperacao[2];
int iRetorno;
iRetorno = Bematech_FI_VerificaModoOperacao(cModoOperacao);
```

Parâmetros:

→ *modo* Variável, com 2 caracteres, destinada a receber o modo de operação da impressora, sendo modo normal "1" e intervenção técnica "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução da função

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os retornos provenientes da impressora são: Modo normal "55h" e intervenção técnica "AAh".

3.8.1.24 int Bematech_FI_VerificaTipoImpressora (short int * tipoImpressora)

Verifica Tipo Impressora.

Retorna o tipo de impressora.

```
// Exemplo em C/C++
int iRetorno, iTipoImpressora;
iRetorno = Bematech_FI_VerificaTipoImpressora(iTipoImpressora);
```

Parâmetros:

→ *tipoImpressora* Variável, inteira, destinada a receber o tipo da impressora, esta podendo ser:

- 1 - Impressora fiscal, gaveta, autenticação.
- 2 - Impressora fiscal, gaveta, cutter.
- 3 - Impressora fiscal, presenter, autenticação.
- 4 - Impressora fiscal, presenter, cutter.
- 5 - Impressora bilhete de passagem, gaveta, autenticação.
- 6 - Impressora bilhete de passagem, gaveta, cutter.
- 7 - Impressora bilhete de passagem, presenter, autenticação.
- 8 - Impressora bilhete de passagem, presenter, cutter.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.8.1.25 int Bematech_FI_VerificaTruncamento (char *flag)

Verifica Truncamento.

Verifica se a impressora encontra-se em modo de truncamento ou arredondamento.

```
// exemplo em C/C++
char cModoTruncamento[2];
int iRetorno;
iRetorno = Bematech_FI_VerificaTruncamento(cModoTruncamento);
```

Parâmetros:

- **flag** Variável, com 2 caracteres, destinado a receber o flag indicativo de impressora em modo de truncamento "1" ou arredondamento "0".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9 Informações MFD

Funções

- int `Bematech_FI_CNPJMFD` (char *cCNPJ)
- int `Bematech_FI_ComprovantesNaoFiscaisNaoEmitidosMFD` (char *cComprovantes)
- int `Bematech_FI_ContadorComprovantesCreditoMFD` (char *cComprovante)
- int `Bematech_FI_ContadorCupomFiscalMFD` (char *cContadorCupom)
- int `Bematech_FI_ContadoresTotalizadoresNaoFiscaisMFD` (char *cContadores)
- int `Bematech_FI_ContadorFitaDetalheMFD` (char *cContadorFita)
- int `Bematech_FI_ContadorOperacoesNaoFiscaisCanceladasMFD` (char *cContadorOperacoes)
- int `Bematech_FI_DataHoraUltimoDocumentoMFD` (char *cData)
- int `Bematech_FI_DataMovimentoUltimaReducaoMFD` (char *cData)
- int `Bematech_FI_FlagsFiscais3MFD` (short int *flag)
- int `Bematech_FI_GrandeTotalUltimaReducaoMFD` (char *cGrandeTotal)
- int `Bematech_FI_InicioFimCOOsMFD` (char *COOini, char *COOfim)
- int `Bematech_FI_InicioFimGTsMFD` (char *GTini, char *GTfim)
- int `Bematech_FI_InscricaoEstadualMFD` (char *cIE)
- int `Bematech_FI_InscricaoMunicipalMFD` (char *cIM)
- int `Bematech_FI_MinutosEmitindoDocumentosFiscaisMFD` (char *cMinutos)
- int `Bematech_FI_NumeroSerieMemoriaMFD` (char *cNumeroSerie)
- int `Bematech_FI_PercentualLivreMFD` (char *cValor)
- int `Bematech_FI_ReducoesRestantesMFD` (char *cReducoes)
- int `Bematech_FI_StatusEstendidoMFD` (short int *status)
- int `Bematech_FI_SubTotalComprovanteNaoFiscalMFD` (char *cSubTotal)
- int `Bematech_FI_TamanhoTotalMFD` (char *cTamanho)
- int `Bematech_FI_TempoOperacionalMFD` (char *cTempo)
- int `Bematech_FI_TempoRestanteComprovanteMFD` (char *cTempo)
- int `Bematech_FI_TotalIssCupomMFD` (char *ISS)
- int `Bematech_FI_TotalLivreMFD` (char *cTamanho)
- int `Bematech_FI_UFProprietarioMFD` (char *cUF)
- int `Bematech_FI_ValorFormaPagamentoMFD` (const char *cFormaPagamento, char *cValorForma)
- int `Bematech_FI_ValorTotalizadorNaoFiscalMFD` (const char *cTotalizador, char *cValorTotalizador)
- int `Bematech_FI_VerificaEstadoImpressoraMFD` (short int *ack, short int *st1, short int *st2, short int *st3)
- int `Bematech_FI_VerificaFlagCorteMFD` (short *flag)
- int `Bematech_FI_VerificaFormasPagamentoMFD` (char *cFormas)
- int `Bematech_FI_VerificaRecebimentoNaoFiscalMFD` (char *cValorRecebimento)
- int `Bematech_FI_VerificaRelatorioGerencialMFD` (char *cRelatorios)
- int `Bematech_FI_VerificaTotalizadoresNaoFiscaisMFD` (char *cTotalizadores)
- int `Bematech_FI_VerificaTotalizadoresParciaisMFD` (char *cTotalizadores)
- int `Bematech_FI_VersaoFirmwareMFD` (char *versao)

3.9.1 Documentação das funções

3.9.1.1 `int Bematech_FI_CNPJMFD (char * cCNPJ)`

CNPJ MFD.

Lê o CNPJ cadastrado na impressora MFD e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
char cCNPJMFD[21];
int iRetorno;
iRetorno = Bematech_FI_CNPJMFD(cCNPJMFD);
```

Parâmetros:

→ *cCNPJ* Variável, com 21 posições, destinada a receber o CNPJ cadastrado na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.2 `int Bematech_FI_ComprovantesNaoFiscaisNaoEmitidosMFD (char * cComprovantes)`

Comprovantes Não Fiscais Não Emitidos MFD.

Lê o contador de comprovantes não fiscais não emitidos na MFD.

```
// Exemplo em C/C++
char cComprovantesNFNaoEmitidos[5];
int iRetorno;
iRetorno =
Bematech_FI_ComprovantesNaoFiscaisNaoEmitidosMFD
(cComprovantesNFNaoEmitidos);
```

Parâmetros:

→ *cComprovantes* Variável, com 5 posições, destinada a receber o número de comprovantes não fiscais não emitidos.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.3 int Bematech_FI_ContadorComprovantesCreditoMFD (char * cComprovante)

Contador Comprovante Crédito MFD.

Lê o contador de comprovantes de crédito na MFD.

```
// Exemplo em C/C++
char cComprovantesCredito[5];
int iRetorno;
iRetorno =
Bematech_FI_ContadorComprovantesCreditoMFD(cComprovantesCredito);
```

Parâmetros:

→ *cComprovante* Variável, com 5 posições, destinada a receber o número de comprovantes de crédito emitidos.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.4 int Bematech_FI_ContadorCupomFiscalMFD (char * cContadorCupom)

Contador Cupom Fiscal MFD.

Lê o contador de cupom fiscal na MFD.

```
// Exemplo em C/C++
char cContadorCupomFiscal[7];
int iRetorno;
iRetorno = Bematech_FI_ContadorCupomFiscalMFD(cContadorCupomFiscal);
```

Parâmetros:

→ *cContadorCupom* Variável, com 7 posições, destinada a receber o número de cupons fiscais emitidos.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.5 int Bematech_FI_ContadoresTotalizadoresNaoFiscaisMFD (char * *cContadores*)

Contadores Totalizadores Não Fiscais MFD.

Lê o número de vezes em que cada totalizador não sujeito a ICMS foi usado e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
char cTotalizadoresNaoFiscais[150];
int iRetorno;
iRetorno =
Bematech_FI_ContadoresTotalizadoresNaoFiscaisMFD
(cTotalizadoresNaoFiscais);
```

Parâmetros:

→ *cContadores* Variável, com 150 posições, destinada a receber o número de vezes em que os totalizadores não sujeitos a ICMS foram usados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O conteúdo da variável retornada será 120 dígitos separados de 4 em 4 por vírgula que representam os 30 totalizadores, como por exemplo:

```
"0001,0003,0001,0005,0004,0002,0003,0004,0007,..."
```

O primeiro valor corresponde ao número de vezes que o totalizador "01" fora usado, o segundo corresponde ao totalizador "02" e assim sucessivamente.

3.9.1.6 int Bematech_FI_ContadorFitaDetalheMFD (char * *cContadorFita*)

Contador Fita Detalhe MFD.

Lê o contador de fita detalhe na MFD.

```
// Exemplo em C/C++
char cContadorFitaDetalhe[7];
int iRetorno;
iRetorno = Bematech_FI_ContadorFitaDetalheMFD(cContadorFitaDetalhe);
```

Parâmetros:

→ *cContadorFita* Variável, com 7 posições, destinada a receber o número de vezes em que foi impressa a fita detalhe.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.7 int Bematech_FI_ContadorOperacoesNaoFiscaisCanceladasMFD (char * cContadorOperacoes)

Contador Operações Não Fiscais Canceladas MFD.

Lê o contador de operações não fiscais canceladas na MFD.

```
// Exemplo em C/C++
char cContadorOperacoesNFCanceladas[5];
int iRetorno;
iRetorno =
Bematech_FI_ContadorOperacoesNaoFiscaisCanceladasMFD
(cContadorOperacoesNFCanceladas);
```

Parâmetros:

- *cContadorOperacoes* Variável, com 5 posições, destinada a receber o número de de operações não fiscais canceladas na MFD.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.8 int Bematech_FI_DataHoraUltimoDocumentoMFD (char * cData)

Data Hora Último Documento MFD.

Lê a data e hora do último documento armazenado na MFD no formato dd/mm/aa hh/mm/ss (sem barra e espaços).

```
// Exemplo em C/C++
char cDataHora[13];
int iRetorno;
iRetorno = Bematech_FI_DataHoraUltimoDocumentoMFD(cDataHora);
```

Parâmetros:

- *cData* Variável, com 13 posições, destinada a receber a data e hora do último documento armazenado na MFD.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.9 int Bematech_FI_DataMovimentoUltimaReducaoMFD (char * *cData*)

Data Movimento Última Redução Z MFD.

Lê a data do movimento da última redução Z no formato DDMMAA.

```
// Exemplo em C/C++
char cDataMovimento[7];
int iRetorno;
iRetorno = Bematech_FI_DataMovimentoUltimaReducaoMFD(cDataMovimento);
```

Parâmetros:

→ *cData* Variável, com 7 caracteres, destinada a receber a data do movimento da última redução Z.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.10 int Bematech_FI_FlagsFiscais3MFD (short int * *flag*)

Flags Fiscais 3 MFD.

Lê os flags fiscais III da impressora fiscal térmica.

```
// Exemplo em C/C++
int iFlagsFiscais3;
int iRetorno;
iRetorno = Bematech_FI_FlagsFiscais3MFD(iFlagsFiscais3);
```

Parâmetros:

→ *flag* Variável, inteira, destinada a receber os flags fiscais III.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Discriminação dos valores para o flag fiscal:

Descrição	Valor
Impressora com pouco papel	1
Sensor de pouco papel habilitado	2
Cancelamento automático de cupom às duas horas habilitado	4
Desconto em ISSQN desabilitado "0" ou habilitado "1"	8
Redução Z automática desabilitada "0" ou habilitada "1"	16
Impressora ON-LINE "0" ou OFF-LINE "1"	32
NÃO UTILIZADO	64
Abertura detectada	128

Os valores citados acima podem vir somados, o que indica mais de um estado.

Esta função está disponível apenas para os modelos MP-4000 TH FI e MP-7000 TH FI.

3.9.1.11 int Bematech_FI_GrandeTotalUltimaReducaoMFD (char * cGrandeTotal)

Grande Total Última Redução MFD.

Retorna o grande total (GT) da última redução Z.

```
// Exemplo em C/C++
char cGrandeTotalMFD[19];
int iRetorno;
iRetorno = Bematech_FI_GrandeTotalUltimaReducaoMFD(cGrandeTotalMFD);
```

Parâmetros:

→ *cGrandeTotal* Variável, com 19 posições, destinada a receber o grande total da última redução Z.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação

- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.12 int Bematech_FI_InicioFimCOOsMFD (char * *COOini*, char * *COOfim*)

Início Fim COOs MFD.

Retorna o contador de ordem de operação (COO) inicial e final.

```
// Exemplo em C/C++
char cCOOInicial[7];
char cCOOFinal[7];
int iRetorno;
iRetorno = Bematech_FI_InicioFimCOOsMFD(cCOOInicial, cCOOFinal);
```

Parâmetros:

- *COOini* Variável, com 7 posições, destinada a receber a informação do COO inicial.
- *COOfim* Variável, com 7 posições, destinada a receber a informação do COO final.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função deve ser usada após a redução Z, para que o COO inicial e final tenha referência ao dia do movimento atual.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.13 int Bematech_FI_InicioFimGTsMFD (char * *GTini*, char * *GTfim*)

Início Fim GTs MFD.

Retorna o contador do grande total (GT) inicial e final.

```
// Exemplo em C/C++
char cGTInicial[19];
char cGTFinal[19];
int iRetorno;
iRetorno = Bematech_FI_InicioFimGTsMFD(cGTInicial, cGTFinal);
```

Parâmetros:

- *GTini* Variável, com 19 posições, destinada a receber a informação do GT inicial.
- *GTfim* Variável, com 19 posições, destinada a receber a informação do GT final.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função deve ser usada após a redução Z, para que o GT inicial e final tenha referência ao dia do movimento atual.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.14 int Bematech_FI_InscricaoEstadualMFD (char * *cIE*)

Inscrição Estadual MFD.

Lê a inscrição estadual cadastrada na impressora MFD e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
char cIEMFD[21];
int iRetorno;
iRetorno = Bematech_FI_InscricaoEstadualMFD(cIEMFD);
```

Parâmetros:

- *cIE* Variável, com 21 posições, destinada a receber a inscrição estadual cadastrada na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.15 int Bematech_FI_InscricaoMunicipalMFD (char * *cIM*)

Inscrição Municipal MFD.

Retorna a inscrição municipal do cliente cadastrada na impressora.

```
// Exemplo em C/C++
char cIMMFD[21];
int iRetorno;
iRetorno = Bematech_FI_InscricaoMunicipalMFD(cIMMFD);
```

Parâmetros:

→ *cIM* Variável, com 21 posições, destinada a receber a inscrição municipal cadastrada na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.16 int Bematech_FI_MinutosEmitindoDocumentosFiscaisMFD (char * *cMinutos*)

Minutos Emitindo Documentos Fiscais MFD.

Retorna o tempo em que a impressora emitiu documentos fiscais.

```
// Exemplo em C/C++
char cMinutosDocumentosFiscais[5];
int iRetorno;
iRetorno =
Bematech_FI_MinutosEmitindoDocumentosFiscaisMFD
(cMinutosDocumentosFiscais);
```

Parâmetros:

→ *cMinutos* Variável, com 5 posições, destinada a receber o tempo em que a impressora emitiu documentos fiscais.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.17 int Bematech_FI_NumeroSerieMemoriaMFD (char * cNumeroSerie)

Número Série Memória MFD.

Retorna o número de série da memória de fita detalhe (MFD).

```
// Exemplo em C/C++
char cNumeroSerieMFD[21];
int iRetorno;
iRetorno = Bematech_FI_NumeroSerieMemoriaMFD(cNumeroSerieMFD);
```

Parâmetros:

→ *cNumeroSerie* Variável, com 21 posições, destinada a receber o número de série da MFD.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.18 int Bematech_FI_PercentualLivreMFD (char * cValor)

Percentual Livre MFD.

Retorna o percentual livre da Memória Fita Detalhe (MFD) no formato XX,XX% (com a vírgula e o "%").

```
// Exemplo em C/C++
char cValorPercentual[7];
int iRetorno;
iRetorno = Bematech_FI_PercentualLivreMFD(cValorPercentual);
```

Parâmetros:

→ *cValor* Variável, com 7 posições, destinada a receber o percentual livre da MFD.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.

3.9.1.19 int Bematech_FI_ReducoesRestantesMFD (char * *cReducoes*)

Reduções Restantes MFD.

Lê o número de reduções restantes na MFD.

```
// Exemplo em C/C++
char cReducoesRestantes[5];
int iRetorno;
iRetorno = Bematech_FI_ReducoesRestantesMFD(cReducoesRestantes);
```

Parâmetros:

→ *cReducoes* Variável, com 5 posições, destinada a receber o número de reduções restantes na impressora.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.20 int Bematech_FI_StatusEstendidoMFD (short int * *status*)

Status Estendido MFD.

Retorna o status de relatório gerencial ou comprovante não-fiscal aberto.

```
// Exemplo em C/C++
int iStatus;
int iRetorno;
iRetorno = Bematech_FI_StatusEstendidoMFD(iStatus);
```

Parâmetros:

→ *status* Variável inteira destinada a receber o status de relatório gerencial ou comprovante não-fiscal aberto.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.21 int Bematech_FI_SubTotalComprovanteNaoFiscalMFD (char * *cSubTotal*)

SubTotal Comprovante Não Fiscal MFD.

Retorna o subtotal do comprovante não fiscal não vinculado (recebimento).

```
// Exemplo em C/C++
char cSubTotalComprovanteNF[15];
int iRetorno;
iRetorno =
Bematech_FI_SubTotalComprovanteNaoFiscalMFD(cSubTotalComprovanteNF);
```

Parâmetros:

→ *cSubTotal* Variável, com 15 posições, destinada a receber o subtotal do comprovante não fiscal.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.22 int Bematech_FI_TamanhoTotalMFD (char * *cTamanho*)

Tamanho Total MFD.

Lê o tamanho total da MFD em bytes.

```
// Exemplo em C/C++
char cTamanhoTotalMFD[11];
int iRetorno;
iRetorno = Bematech_FI_TamanhoTotalMFD(cTamanhoTotalMFD);
```

Parâmetros:

→ *cTamanho* Variável, com 11 posições, destinada a receber o tamanho total da MFD.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.23 int Bematech_FI_TempoOperacionalMFD (char * *cTempo*)

Tempo Operacional MFD.

Retorna o tempo em que a impressora está operacional.

```
// Exemplo em C/C++
char cTempoOperacional[5];
int iRetorno;
iRetorno = Bematech_FI_TempoOperacionalMFD(cTempoOperacional);
```

Parâmetros:

→ *cTempo* Variável, com 5 posições, destinada a receber o tempo em que a impressora está operacional.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.24 int Bematech_FI_TempoRestanteComprovanteMFD (char * *cTempo*)

Tempo Restante Comprovante MFD.

Retorna o tempo restante do relatório gerencial ou do comprovante não-fiscal aberto.

```
// Exemplo em C/C++
char cTempoRestante[5];
int iRetorno;
iRetorno = Bematech_FI_TempoRestanteComprovanteMFD(cTempoRestante);
```

Parâmetros:

→ *cTempo* Variável, com 5 posições, destinada a receber o tempo restante para o comprovante aberto.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O tempo é retornado à variável em segundos.

O limite destes comprovantes é de 2 minutos.

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.25 int Bematech_FI_TotalIssCupomMFD (char * ISS)

Total ISS Cupom MFD.

Lê o valor total do ISS pago no cupom.

```
// Exemplo em C/C++
char cTotalIssCupom[14];
int iRetorno;
iRetorno = Bematech_FI_TotalIssCupomMFD (cTotalIssCupom);
```

Parâmetros:

→ *ISS* Variável, com 14 posições, destinada a receber o valor total do ISS pago no cupom.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta

-5 Erro ao abrir a porta de comunicação

-8 Erro ao criar ou gravar no arquivo texto

3.9.1.26 int Bematech_FI_TotalLivreMFD (char * cTamanho)

Total Livre MFD.

Lê o espaço livre da MFD em bytes.

```
// Exemplo em C/C++
char cTamanhoEspacoLivreMFD[11];
int iRetorno;
iRetorno =
Bematech_FI_Bematech_FI_TotalLivreMFD (cTamanhoEspacoLivreMFD);
```

Parâmetros:

→ *cTamanho* Variável, com 11 posições, destinada a receber a quantidade de bytes livres na MFD.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.27 int Bematech_FI_UFProprietarioMFD (char * *cUF*)

UF Proprietário MFD.

Retorna a UF do proprietário cadastrada na impressora.

```
// Exemplo em C/C++
char cUFProprietario[3];
int iRetorno;
iRetorno = Bematech_FI_UFProprietarioMFD(cUFProprietario);
```

Parâmetros:

→ *cUF* Variável, com 3 posições, destinada a receber a UF do proprietário cadastrada na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.28 int Bematech_FI_ValorFormaPagamentoMFD (const char * *cFormaPagamento*, char * *cValorForma*)

Valor Forma Pagamento MFD.

Lê o valor acumulado na forma de pagamento na impressora MFD e compatibiliza com a MP20 FI II.

```
// Exemplo em C/C++
char cValorPagamento[14];
int iRetorno;
iRetorno =
Bematech_FI_ValorFormaPagamentoMFD("Cheque", cValorPagamento);
```

Parâmetros:

cFormaPagamento Descrição da forma de pagamento a qual se deseja retornar seu valor, com até 16 caracteres.

→ *cValorForma* Variável, com 15 posições, destinada a receber o valor da forma de pagamento desejada.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.29 int Bematech_FI_ValorTotalizadorNaoFiscalMFD (const char * cTotalizador, char * cValorTotalizador)

Valor Totalizador Não Fiscal MFD.

Lê o valor acumulado de um determinado totalizador não fiscal.

```
// Exemplo em C/C++
char cValorTotalizadorNF[15];
int iRetorno;
iRetorno =
Bematech_FI_ValorTotalizadorNaoFiscalMFD("Rec. Prestacao",
cValorTotalizadorNF);
```

Parâmetros:

cTotalizador Descrição do totalizador, com até 19 caracteres.

→ *cValorTotalizador* Variável, com 15 posições, destinada a receber o valor acumulado do totalizador não fiscal.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 2 Parâmetro inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 25 Totalizador não fiscal não programado
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.9.1.30 int Bematech_FI_VerificaEstadoImpressoraMFD (short int * ack, short int * st1, short int * st2, short int * st3)

Verifica Estado Impressora MFD.

Ler o estado da impressora

```
// exemplo em C/C++
int iRetorno;
short int iAck, iSt1, iSt2, iSt3;
iRetorno =
Bematech_FI_VerificaEstadoImpressoraMFD(&iAck, &iSt1, &iSt2, &iSt3);
```

Parâmetros:

- **ack** Buffer para receber o ack
- **st1** Buffer para receber o st1
- **st2** Buffer para receber o st2
- **st3** Buffer para receber o st3

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1** Ok
- 0** Erro de comunicação
- 1** Erro de execução da função
- 4** O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows
- 5** Erro ao abrir a porta de comunicação
- 8** Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT
- 27** Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.9.1.31 int Bematech_FI_VerificaFlagCorteMFD (short * flag)

Verifica Flag Corte MFD.

Retorna o flag de acionamento da guilhotina.

```
// Exemplo em C/C++
int iFlagCorteMFD;
int iRetorno;
iRetorno = Bematech_FI_VerificaFlagCorteMFD(iFlagCorteMFD);
```

Parâmetros:

- **flag** Variável, inteira, destinada a receber o flag de acionamento da guilhotina.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1** Ok

- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01.

3.9.1.32 int Bematech_FI_VerificaFormasPagamentoMFD (char * cFormas)

Verifica Formas Pagamento MFD.

Lê as formas de pagamento e seus valores e mantém a compatibilidade com MP20 FI II.

```
// Exemplo em C/C++
char cFormasPagamento[920];
int iRetorno;
iRetorno = Bematech_FI_VerificaFormasPagamentoMFD(cFormasPagamento);
```

Parâmetros:

→ *cFormas* Variável, com 920 posições, destinada a receber as formas de pagamento programadas na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Essa função retorna as informações das formas de pagamento na seguinte ordem:

```
Descricao da forma de pagamento.....: 16 bytes
Valor acumulado da forma (duas casas decimais).....: 14 bytes
Valor recebido no último cupom (duas casas decimais).....: 14 bytes
Valor indicando se a forma permite operação TEF (1 permite, 0 não permite ): 1 byte
```

São retornadas 20 formas de pagamento mais a forma "Valor Recebido" e "Troco" obedecendo a ordem descrita anteriormente. As formas de pagamento vêm separadas por vírgula.

Veja exemplo a seguir:

```

Dinheiro          00000000000000041558000000000000000015580,
Cartao Credito    0000000000000003947000000000000000000000,
Cheque           0000000000000002894000000000000000000000,
Ticket           00000000000000090000000000000000000000,
.
.
.
Valor Recebido   00000000000000118968000000000000000015580,
Troco           00000000000000000000000000000000000000

```

A impressora MFD permite programar até 20 formas de pagamento e essas formas não são mais apagadas na redução Z (conforme ocorria na impressora MP20 FI II).

As formas que não estiverem programadas estarão com os valores zerados e a descrição em branco (veja exemplo acima).

3.9.1.33 int Bematech_FI_VerificaRecebimentoNaoFiscalMFD (char * *cValorRecebimento*)

Verifica Recebimento Não Fiscal MFD.

Retorna os recebimentos não fiscais não vinculados programados na impressora.

```

// Exemplo em C/C++
char cRecebimentos[1078];
int iRetorno;
iRetorno = Bematech_FI_VerificaRecebimentoNaoFiscalMFD(cRecebimentos);

```

Parâmetros:

→ *cValorRecebimento* Variável, com 1078 posições, destinada aos recebimentos não fiscais não vinculados programados na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função retorna as informações para os 30 totalizadores não fiscais não vinculados na seguinte ordem:

```

Descrição do recebimento.....: 19 bytes
Valor acumulado para cada recebimento (duas casas decimais): 14 bytes

```

Um exemplo do retorno desta função é mostrado abaixo:

```

Conta de água          00000000005460,
Conta de Luz          00000000145850,
Conta de Telefone     00000000078437,
.
.
.
Sangria              00000000000000,
Suprimento           00000000010000,

```

A impressora permite programar até 28 totalizadores não fiscais não vinculados. Os totalizadores "Sangria" e "Suprimento" são pré-programados.

3.9.1.34 int Bematech_FI_VerificaRelatorioGerencialMFD (char * *cRelatorios*)

Verifica Relatório Gerencial MFD.

Lê a descrição e os valores dos relatórios gerenciais cadastrados na MFD.

```
// Exemplo em C/C++
char cRelatoriosGerenciais[660];
int iRetorno;
iRetorno =
Bematech_FI_VerificaRelatorioGerencialMFD(cRelatoriosGerenciais);
```

Parâmetros:

→ *cRelatorios* Variável, com 660 posições, destinada a receber os relatórios gerenciais programados e seus valores acumulados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função retorna as informações para os 30 relatórios gerenciais na seguinte ordem:

Número de vezes que o relatório foi utilizado.: 4 bytes
Descrição do relatório.....: 17 bytes

Exemplo:

```
0005Relatorio Geral ,
0002Relatorio 3 ,
0001Relatorio 4 ,
.
.
.
0004Relatorio 30 ,
```

3.9.1.35 int Bematech_FI_VerificaTotalizadoresNaoFiscaisMFD (char * *cTotalizadores*)

Verifica Totalizadores Não Fiscais MFD.

Lê os totalizadores não sujeitos a ICMS cadastrados na impressora e mantém a compatibilidade com a MP20 FI II.

```
// Exemplo em C/C++
char cTotalizadoresNF[600];
int iRetorno;
iRetorno =
Bematech_FI_VerificaTotalizadoresNaoFiscaisMFD(cTotalizadoresNF);
```

Parâmetros:

→ *cTotalizadores* Variável, com 600 posições, destinada a receber a descrição dos totalizadores não fiscais programados.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

A impressora MFD permite programar até 28 totalizadores não fiscais e possui dois (2) pré-programados: Sangria e Suprimento.
Os totalizadores virão separados por vírgula.

3.9.1.36 int Bematech_FI_VerificaTotalizadoresParciaisMFD (char * cTotalizadores)

Verifica Totalizadores Parciais MFD.

Lê os totalizadores parciais e mantém a compatibilidade com a MP20 FI II

```
// Exemplo em C/C++
char cTotalizadoresParciais[890];
int iRetorno;
iRetorno =
Bematech_FI_VerificaTotalizadoresParciaisMFD(cTotalizadoresParciais);
```

Parâmetros:

→ *cTotalizadores* Variável, com 890 posições, destinada a receber os totalizadores parciais programados na impressora.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação

- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

São retornadas as seguintes informações separadas por vírgula:

Totalizadores parciais tributados:	224 bytes
Isenção de ICMS:	14 bytes
Não incidência de ICMS:	14 bytes
Substituição de ICMS:	14 bytes
Isenção de ISSQN:	14 bytes
Não incidência de ISSQN:	14 bytes
Substituição de ISSQN:	14 bytes
Descontos sobre ICMS:	14 bytes
Acréscimos sobre ICMS:	14 bytes
Cancelamentos sobre ICMS:	14 bytes
Descontos sobre ISSQN:	14 bytes
Acréscimos sobre ISSQN:	14 bytes
Cancelamentos sobre ISSQN:	14 bytes
Totalizadores não fiscais + sangria e suprimento:	420 bytes
Descontos sobre não fiscais:	14 bytes
Acréscimos sobre não fiscais:	14 bytes
Cancelamentos sobre não fiscais:	14 bytes
Grande Total:	18 bytes

3.9.1.37 int Bematech_FI_VersaoFirmwareMFD (char * versao)

Versão Firmware MFD.

Ler a versão do firmware MFD.

```
// exemplo em C/C++
int iRetorno;
char sVersaoFirmware[7];
iRetorno = Bematech_FI_VersaoFirmwareMFD(sVersaoFirmware);
```

Parâmetros:

→ *versao* Buffer com 7 caracteres para receber a versão do firmware.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo ini não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (Ack, St1, St2 e St3)

3.10 Informação

Funções

- int `Bematech_FI_FlagsVinculacaoIss` (int *iFlag1, int *iFlag2)
- int `Bematech_FI_ModeloImpressora` (char *modelo)
- int `Bematech_FI_ModeloVersaoImpressora` (char *modelo, char *versao)
- int `Bematech_FI_RegistroTipo60M` (void)
- int `Bematech_FI_VendaBruta` (char *vendaBruta)
- int `Bematech_FI_VersaoSO` (char *versao)

3.10.1 Documentação das funções

3.10.1.1 int `Bematech_FI_FlagsVinculacaoIss` (int * *iFlag1*, int * *iFlag2*)

Flags Vinculação ISS.

Retorna os flags das alíquotas de vinculação ao ISS.

```
// Exemplo em C/C++
int iRetorno, iFlagVinculacaoISS1, iFlagVinculacaoISS2;
iRetorno =
Bematech_FI_FlagsVinculacaoIss (iFlagVinculacaoISS1,
    iFlagVinculacaoISS2);
```

Parâmetros:

- *iFlag1* Variável, inteira, destinada a receber o flag1, referente as alíquotas da posição 1 a 8.
- *iFlag2* Variável, inteira, destinada a receber o flag2, referente as alíquotas da posição 9 a 16.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Os valores podem vir somados, o que indica mais de uma alíquota vinculada.

3.10.1.2 int `Bematech_FI_ModeloImpressora` (char * *modelo*)

Modelo Impressora.

Retorna o modelo da impressora fiscal em uso.

```
// Exemplo em C/C++
char cModeloImpressora[11];
int iRetorno;
iRetorno = Bematech_FI_ModeloImpressora(cModeloImpressora);
```

Parâmetros:

→ **modelo** Modelo da impressora com 11 caracteres. Os tipos de retorno poderão ser:

```
MP20FI
MP40FI
MP25FI
MP50FI
MP2000FI
MP2100FI
MP3000FI
MP4000FI
MP6000FI
MP7000FI
4610-KR4
4679-3B4
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução (impressora desconhecida)
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.10.13 int Bematech_FI_ModeloVersaoImpressora (char * modelo, char * versao)

Modelo Versão Impressora.

Lê o modelo da impressora e a versão do firmware.

```
// Exemplo em C/C++
char cModeloImpressora[11];
char cVersaoFirmware[7];
int iRetorno;
iRetorno =
Bematech_FI_ModeloVersaoImpressora(cModeloImpressora,
cVersaoFirmware);
```

Parâmetros:

→ **modelo** Modelo da impressora com 11 caracteres. Os tipos de retorno poderão ser:

```
MP20FI
MP40FI
MP25FI
MP50FI
MP2000FI
MP2100FI
```

```

MP3000FI
MP4000FI
  MP6000FI
    4610-KR4
    4679-3B4

```

→ *versao* Versão do firmware com 7 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função independe da configuração de Impressora MFD no arquivo ".INI".

3.10.1.4 int Bematech_FI_RegistroTipo60M (void)

Registros Tipo 60M.

Gerar os registros tipo 60M Contidos no relatório Sintegra.

```

// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RegistroTipo60M();

```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo ini não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicacao
- 8 Erro ao criar ou gravar no arquivo status.txt ou retorno.txt.

Nota:

Informações geradas pela função:

Tipo do relatório	2 bytes
Subtipo	1 byte
Data de emissão	8 bytes no formato AAAAMMDD
Número de série	20 bytes
Situacao Trib.	4 bytes
Valor acumulado situacao trib.	12 bytes
Espacos em branco	79 bytes

Será gerado um registro para cada situação tributária utilizada no dia, um para FI, N, um para CANC, DESC e ISS.

Algumas informações geradas no relatório tipo 60 mestre são referentes aos dados da última redução Z. Portanto, essa função deve ser executada após a redução Z.

3.10.1.5 `int Bematech_FI_VendaBruta (char * vendaBruta)`

Venda Bruta.

Ler o valor da venda bruta na impressora fiscal

```
// exemplo em C/C++
int iRetorno;
char Valor[19];
iRetorno = Bematech_FI_VendaBruta(Valor);
```

Parâmetros:

→ *vendaBruta* Buffer com 19 bytes para receber o valor da venda bruta.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 2 Parâmetro inválido na função
- 24 Forma de pagamento não programada
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.10.1.6 `int Bematech_FI_VersaoSO (char * versao)`

Versão SO. Retornar a versão da biblioteca.

```
// exemplo em C/C++
int iRetorno;
char cVersao[5];
iRetorno = Bematech_FI_VersaoSO(cVersao);
```

Parâmetros:

→ *versao* Buffer com 5 bytes para receber a versão da biblioteca.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO
- 27 Status da impressora diferente de 6,0,0,0 (Ack, St1, St2 e St3)

3.11 Gaveta de Dinheiro

Funções

- int `Bematech_FI_AcionaGaveta` (void)
- int `Bematech_FI_VerificaEstadoGaveta` (short int *flag)

3.11.1 Documentação das funções

3.11.1.1 int `Bematech_FI_AcionaGaveta` (void)

Aciona Gaveta.

Abre a gaveta de dinheiro.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AcionaGaveta();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.11.1.2 int `Bematech_FI_VerificaEstadoGaveta` (short int **flag*)

Verifica Estado Gaveta.

Retorna se a gaveta está aberta ou fechada.

```
// exemplo em C/C++
int iRetorno, iEstadoGaveta;
iRetorno = Bematech_FI_VerificaEstadoGaveta(iEstadoGaveta);
```

Parâmetros:

→ *flag* Variável, inteira, destinada a receber o estado da gaveta, onde "0" indica gaveta aberta e "1" fechada.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 1* Erro de execução da função

- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Em alguns modelos de gaveta, este retorno poderá ser invertido, ou seja, o retorno que corresponde a "gaveta aberta", poderá ser o retorno para "gaveta fechada". Caso isso ocorra, mude a chave "ModoGaveta" (encontrado nas configurações do arquivo Bemafi32.ini), para "1" e efetue o teste de retorno, abrindo e fechando a gaveta.

3.12 Cheque

Funções

- int [Bematech_FI_CancelaImpressaoCheque](#) (void)
- int [Bematech_FI_ImprimeCheque](#) (const char *cNumeroBanco, const char *cValor, const char *cFavorecido, const char *cCidade, const char *cData, const char *cMsg)
- int [Bematech_FI_ImprimeChequeMFD](#) (const char *cNumeroBanco, const char *cValor, const char *cFavorecido, const char *cCidade, const char *cData, const char *cMsg, const char *cImpressaoVerso, const char *cLinhas)
- int [Bematech_FI_ImprimeChequeMFDEx](#) (const char *cNumeroBanco, const char *cValor, const char *cFavorecido, const char *cCidade, const char *cData, const char *cMsg, const char *cFonte)
- int [Bematech_FI_ImprimeCopiaCheque](#) (void)
- int [Bematech_FI_ImprimeInformacaoChequeMFD](#) (int iPosicao, int iLinhas, const char *cMensagem)
- int [Bematech_FI_IncluiCidadeFavorecido](#) (const char *cCidade, const char *cFavorecido)
- int [Bematech_FI_LeituraChequeMFD](#) (char *cCMC7)
- int [Bematech_FI_ProgramaMoedaPlural](#) (const char *moedaPlural)
- int [Bematech_FI_ProgramaMoedaSingular](#) (const char *moedaSingular)
- int [Bematech_FI_VerificaStatusCheque](#) (short int *status)
- int [Bematech_FI_ViraChequeMFD](#) (void)

3.12.1 Documentação das funções

3.12.1.1 int Bematech_FI_CancelaImpressaoCheque (void)

Cancela Impressão Cheque. cheque que está em impressão não pode ser cancelado.

Cancela a impressão do cheque que está sendo aguardado pela impressora. O

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_CancelaImpressaoCheque();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Função usada somente na impressora MP-40 FI II.

3.12.1.2 int Bematech_FI_ImprimeCheque (const char * *cNumeroBanco*, const char * *cValor*, const char * *cFavorecido*, const char * *cCidade*, const char * *cData*, const char * *cMsg*)

Imprime Cheque.

Montar e imprimir o cheque de acordo com as coordenadas do arquivo de inicialização.

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_ImprimeCheque("353", "50,00", "Bematech S/A", "Curitiba",
    "10/01/02", "");
```

Parâmetros:

- cNumeroBanco* Número do banco, com 3 dígitos.
- cValor* Valor do cheque, com até 14 dígitos.
- cFavorecido* Favorecido, com até 45 caracteres.
- cCidade* Cidade, com até 27 caracteres.
- cData* Data, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.
- cMsg* Mensagem, com até 120 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 7 Banco não encontrado no arquivo BemaFI32.ini
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

A mensagem será impressa 1 (uma) linha após a cidade.
Função usada somente na impressora BEMATECH MP-40 FI II.

3.12.1.3 `int Bematech_FI_ImprimeChequeMFD (const char * cNumeroBanco, const char * cValor, const char * cFavorecido, const char * cCidade, const char * cData, const char * cMsg, const char * cImpressaoVerso, const char * cLinhas)`

Imprime Cheque MFD.

Monta e imprime o cheque de acordo com as coordenadas do arquivo .ini.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_ImprimeChequeMFD("001", "50,00", "Fulano de Tal",
    "Curitiba", "18/04/02", "", "0", "0");
```

Parâmetros:

cNumeroBanco Número do banco, com 3 caracteres.

cValor Valor do cheque com até 14 caracteres.

cFavorecido Favorecido com até 45 caracteres.

cCidade Cidade com até 27 caracteres.

cData Data no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

cMsg Comentário com até 120 caracteres. Esta mensagem será impressa uma linha após a cidade, caso não tenha sido indicada para impressão no verso.

cImpressaoVerso Indicador de impressão na frente do cheque "0" ou no verso "1".

cLinhas Número de linhas a serem saltadas antes da impressão da mensagem, sendo este um valor entre 0 e 35. Este campo só é utilizado na impressão da mensagem no verso do cheque.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-7 Banco não localizado no arquivo de configuração - BemaFI32.ini

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.12.1.4 `int Bematech_FI_ImprimeChequeMFDEX (const char * cNumeroBanco, const char * cValor, const char * cFavorecido, const char * cCidade, const char * cData, const char * cMsg, const char * cFonte)`

Imprime Cheque MFD Ex.

Monta e imprime o cheque de acordo com as coordenadas do arquivo de configuração.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_Bematech_FI_ImprimeChequeMFDEX("001", "50,00", "Bematech",
    "Curitiba", "25/10/2008", "Obrigado, Volte Sempre!!", "0");
```

Parâmetros:

cNumeroBanco Número do banco, com 3 caracteres.

- cValor* Valor do cheque com até 20 caracteres.
cFavorecido Favorecido com até 80 caracteres.
cCidade Cidade com até 27 caracteres.
cData Data no formato dd/mm/aa ou dd/mm/aaaa.
cMsg Mensagem com até 240 caracteres.
cFonte Indicador do tamanho da fonte de impressão do cheque, sendo "0" normal ou "1" grande.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 7* Banco não localizado no arquivo de configuração - BemaFI32.ini
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.12.1.5 int Bematech_FI_ImprimeCopiaCheque (void)

Imprime Cópia Cheque.

Imprimir uma cópia do último cheque impresso.

```
// exemplo em C/C++  
int iRetorno;  
iRetorno = Bematech_FI_ImprimeCopiaCheque();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 7* Banco não encontrado no arquivo BemaFI32.ini
- 27* Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Função usada somente na impressora BEMATECH MP-40 FI II.

3.12.1.6 int Bematech_FI_ImprimeInformacaoChequeMFD (int *iPosicao*, int *iLinhas*, const char * *cMensagem*)

Imprime Informação Cheque MFD.

Imprime informações adicionais em um cheque já impresso.

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_ImprimeInformacaoChequeMFD(1, 2,
    "Teste de impressao no verso do cheque");
```

Parâmetros:

iPosicao Variável, inteira, destinada a definir a posição da impressão: "0" impressão frontal e "1" impressão no verso.

iLinhas Variável, inteira, destinada a definir o número de avanços de linha antes da impressão.

cMensagem Texto a ser impresso, limitado a 240 caracteres ou 3 linhas.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.

3.12.1.7 int Bematech_FI_IncluiCidadeFavorecido (const char * *cCidade*, const char * *cFavorecido*)

Inclui Cidade Favorecido.

Inclui a cidade e o favorecido no arquivo de configuração

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_IncluiCidadeFavorecido("Londrina", "José");
```

Parâmetros:

cCidade Nome da cidade com até 27 caracteres

cFavorecido Nome do favorecido com até 45 caracteres

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 1* Erro de execução
- 2* Parâmetro inválido na função

Nota:

Após gravar o nome da cidade e do favorecido no arquivo de configuração não é mais necessário passar essas informações como parâmetros para a função `Bematech_FI_ImprimeCheque()`, a não ser que seja necessário imprimir uma cidade ou um favorecido diferente aos que estão programados.

Função usada somente na impressora BEMATECH MP-40 FI II.

3.12.1.8 int Bematech_FI_LeituraChequeMFD (char * *cCMC7*)

Leitura Cheque MFD.

Lê o código CMC7 do cheque.

```
// Exemplo em C/C++
char cCodigoCMC7[37];
int iRetorno;
iRetorno = Bematech_FI_LeituraChequeMFD(cCodigoCMC7);
```

Parâmetros:

→ *cCMC7* Variável, com 37 posições, destinada a receber o código CMC7 do cheque.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 9* Time-out na leitura do cheque
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.12.1.9 int Bematech_FI_ProgramaMoedaPlural (const char * *moedaPlural*)

Programa Moeda Plural.

Programa o nome da moeda no plural para a impressão de cheques. Ex. (Reais)

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaMoedaPlural("Reais");
```

Parâmetros:

moedaPlural Moeda no plural, com até 22 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação

Nota:

Função usada somente na impressora BEMATECH MP-40 FI II.

3.12.1.10 int Bematech_FI_ProgramaMoedaSingular (const char * *moedaSingular*)

Programa Moeda Singular.

Programa o nome da moeda no singular para a impressão de cheques. Ex. (Real)

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ProgramaMoedaSingula("Real");
```

Parâmetros:

moedaSingular Moeda no singular, com até 19 caracteres

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Função usada somente na impressora BEMATECH MP-40 FI II.

3.12.1.11 int Bematech_FI_VerificaStatusCheque (short int * *status*)

Verifica Status Cheque.

Verifica o status do cheque.

```
// Exemplo em C/C++
int iRetorno, iStatusCheque;
iRetorno = Bematech_FI_VerificaStatusCheque(iStatusCheque);
```

Parâmetros:

→ *status* Variável, inteira, destinada a receber o status do cheque. Os valores de status são:

- 1 - Impressora OK.
- 2 - Cheque em impressão.
- 3 - Cheque posicionado.
- 4 - Aguardando o posicionamento do cheque.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Função usada somente na impressora MP-40 FI II.

3.12.1.12 int Bematech_FI_ViraChequeMFD (void)

Vira cheque MFD.

Vira o cheque para impressão de texto no verso.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ViraChequeMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de configuração não encontrado ou parâmetro inválido para o nome da porta
- 5 Erro ao abrir a porta de comunicação

3.13 Utilidades**Funções**

- [int Bematech_FI_BaudRateBalanca](#) (int baudRate)
- [int Bematech_FI_ConfiguraCharset](#) (const char *encoding)

- int `Bematech_FI_ImpressaoCarne` (const char *titulo, const char *parcela, const char *datas, int quantidade, const char *texto, const char *cliente, const char *rgcpf, const char *cupom, int vias, int assina)
- int `Bematech_FI_InfoBalanca` (const char *port, int model, char *weight, char *precoKilo, char *total)

3.13.1 Documentação das funções

3.13.1.1 int Bematech_FI_BaudRateBalanca (int *baudRate*)

Baud Rate Balança.

Altera o baudrate da porta serial utilizada pela balança.

```
// exemplo em C/C++
char cPorta[] = "/dev/ttyS0";
char cPeso[7], cPrecoKilo[7], cTotal[7];
int iRetorno;
iRetorno =
Bematech_FI_InfoBalanca(cPorta, 1, cPeso, cPrecoKilo, cTotal);
```

Parâmetros:

baudRate Baudrate da porta serial utilizada pela impressora (ex: CBR_9600)

3.13.1.2 int Bematech_FI_ConfiguraCharset (const char * *encoding*)

Info Balança.

Altera o charset utilizado pela aplicação na comunicação com a biblioteca fiscal. O comportamento padrão da biblioteca é detectar automaticamente qual charset o sistema está utilizando. Normalmente essa função deverá ser chamada como um ajuste quando utilizando a interface JNI (Java), uma vez que essa interface sempre usará UTF-8 entre a JNI e a biblioteca, independentemente do sistema estar usando UTF-8 ou não.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ConfiguraCharset("UTF-8");
```

Parâmetros:

encoding Charset a ser utilizado como interface biblioteca/aplicação. Valores permitidos: UTF-8 ou ISO-8859-1.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 2 Erro de parâmetros

3.13.1.3 int Bematech_FI_ImpressaoCarne (const char * titulo, const char * parcela, const char * datas, int quantidade, const char * texto, const char * cliente, const char * rgcpf, const char * cupom, int vias, int assina)

Impressão Carnê.

Imprime um carnê de pagamentos.

```
// exemplo em C/C++
char cTitulo[] = "Carne de Pgto";
char cValorParcela[] = "10,00;10,00;10,00";
char cData[] = "30/06/03;30/07/03;30/08/03";
int iQtdeParcela = 3;
char cTexto[] = "Texto a ser impresso no carnê";
char cCliente[] = "Fulano de Tal";
char cRGC[] = "1234567890-12345";
char cCOOCupom[] = "000257";
int iVia = 1;
int iAssinatura = 0;
int iRetorno;
iRetorno = Bematech_FI_ImpressaoCarne(cTitulo cValorParcela, cData,
    iQtdeParcela, cTexto, cCliente, cRGC, cCOOCupom,
    iVia, iAssinatura);
```

Parâmetros:

titulo Titulo do carnê que será impresso centralizado e expandido em cada via. Limitado em 20 caracteres.

parcela Valores de cada parcela, separados por ';', com duas casas decimais obrigatoriamente. Exemplos: "23,23;1.200,00;100", "2323;120000;1,00".

datas Datas de vencimento de cada parcela, separadas por ';'. Exemplo: "10/10/2003;10/11/2003;10/12/2003;10/01/2004".

quantidade Quantidade de parcelas. Deve ser diferente de zero.

texto Texto livre de até 200 caracteres.

cliente Nome do cliente para assinatura com até 30 caracteres.

rgcpf RG ou CPF do cliente. Pode ser nulo ou vazio.

cupom COO do Cupom Fiscal com 6 caracteres.

vias Quantidade de vias, 1 ou 2 apenas.

assina Habilita ou não a assinatura do cliente, no qual 1 habilita e 0 desabilita.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução

-2 Erro de parâmetros

Nota:

Caso a assinatura do cliente seja habilitada, será impressa uma linha tracejada como local de assinatura. O número do cupom fiscal e a quantidade de vias só serão impressos no cupom se forem diferentes de zero e um, respectivamente.

Para o parâmetro Parcelas, quando passado um número menor de parcelas na string do que passado no parâmetro quantidade, o valor das parcelas não passadas será igualado ao último valor válido na string. Por exemplo, caso seja passado "20,00;12000;340" em parcelas e 6 em quantidade, o valor de cada parcela será impresso da seguinte forma, respectivamente: R\$ 20,00 ... R\$ 120,00 ... R\$ 3,40 ... R\$ 3,40 ... R\$ 3,40 ... R\$ 3,40. Caso o valor passado em quantidade seja menor do que o número de parcelas encontradas em parcelas, a função retorna erro de parâmetros.

Para o parâmetro Datas, quando passado um número menor de datas do que passado no parâmetro quantidade, as datas ausentes serão calculadas e impressas com acréscimo de um mês, a partir da última data válida. Por exemplo, caso seja passado "20/09/1999;31/12/1999" no parâmetro datas e 5 no parâmetro quantidade, serão impressas no carnê as seguintes datas, respectivamente: 20/09/1999 ... 31/12/1999 ... 31/01/2000 ... 29/02/2000 (ano bissexto) ... 31/03/2000.

3.13.1.4 `int Bematech_FI_InfoBalanca (const char *port, int model, char *weight, char *precoKilo, char *total)`

Info Balança.

Requisita informações da balança pela serial.

```
// exemplo em C/C++
char cPorta[] = "/dev/ttyS0";
char cPeso[7], cPrecoKilo[7], cTotal[7];
int iRetorno;
iRetorno =
Bematech_FI_InfoBalanca(cPorta, 1, cPeso, cPrecoKilo, cTotal);
```

Parâmetros:

port Nome da porta em que a balança está conectada (Ex: "/dev/ttyS0").

model Identificador do modelo da impressora, sendo 1 para modelo BP6, 2 para modelo CS15, 3 para modelo SA-110-0 e 4 para modelo SA-110-4.

→ **weight** Buffer com 7 caracteres para o retorno do peso medido pela balança no formato "KKggg" (Ex: "01200" igual à 1Kg e 200g).

→ **precoKilo** Buffer com 7 caracteres para o retorno do preço por quilo, configurado na balança, no formato "RRRCC" (Ex: "01234" igual à R\$12,34/Kg).

→ **total** Buffer com 7 caracteres para o retorno do preço total do produto no formato "RRRCCC".

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução

-2 Erro de parâmetros

-33 Condição de alívio de prato (para modelos CS) ou peso negativo (para modelos BP)

-34 Peso instável

-35 Peso excedido

Nota:

Para as balanças da família BP, o parâmetro weight é no formato "SKKggg" (Ex: "013200" igual à 13Kg e 200g e "-00020" igual à -20g).

Os parâmetros precoKilo e total não são retornados pelas balanças da família BP.

3.14 Outras

Módulos

- [Download MFD](#)

Funções

- int [Bematech_FI_AberturaDoDia](#) (const char *valorSuprimento, const char *formaPagamento)
- int [Bematech_FI_AtivaDesativaCorteProximoMFD](#) (void)
- int [Bematech_FI_DadosSintegra](#) (const char *dataInicial, const char *dataFinal)
- int [Bematech_FI_DadosSintegraMFD](#) (const char *cDataInicial, const char *cDataFinal)
- int [Bematech_FI_FechamentoDoDia](#) (void)
- int [Bematech_FI_GeraRelatorioSintegraMFD](#) (int iRelatorios, const char *cOrigem, const char *cDestino, const char *cMes, const char *cAno, const char *cRazaoSocial, const char *cEndereco, const char *cNumeroTmp, const char *cComplemento, const char *cBairro, const char *cCidade, const char *cCEPTmp, const char *cTelefoneTmp, const char *cFax, const char *cContato)
- int [Bematech_FI_HabilitaDesabilitaRetornoEstendidoMFD](#) (const char *cFlagRetorno)
- int [Bematech_FI_ImpressaoFitaDetalhe](#) (const char *cTipo, const char *cDadoInicial, const char *cDadoFinal, const char *cUsuario)
- int [Bematech_FI_ImprimeClicheMFD](#) (void)
- int [Bematech_FI_ImprimeConfiguracoesImpressora](#) (void)
- int [Bematech_FI_ImprimeDepartamentos](#) (void)
- int [Bematech_FI_LeArquivoRetorno](#) (char *cDados)
- int [Bematech_FI_MapaResumo](#) (void)
- int [Bematech_FI_MapaResumoMFD](#) (void)
- int [Bematech_FI_NumeroSerieCriptografado](#) (BYTE *cNumSerie)
- int [Bematech_FI_NumeroSerieDescriptografado](#) (const BYTE *cNumSerieCriptografado, char *cNumSerieDescriptografado)
- int [Bematech_FI_RegistrosTipo60](#) (void)
- int [Bematech_FI_RelatorioSintegraMFD](#) (int iRelatorios, const char *cArquivo, const char *cMes, const char *cAno, const char *cRazaoSocial, const char *cEndereco, const char *cNumeroTmp, const char *cComplemento, const char *cBairro, const char *cCidade, const char *cCEPTmp, const char *cTelefoneTmp, const char *cFaxTmp, const char *cContato)
- int [Bematech_FI_RelatorioTipo60Analitico](#) (void)
- int [Bematech_FI_RelatorioTipo60AnaliticoMFD](#) (void)
- int [Bematech_FI_RelatorioTipo60Mestre](#) (void)
- void [Bematech_FI_ReloadINIFile](#) (void)
- int [Bematech_FI_VerificaImpressoraLigada](#) (void)

3.14.1 Documentação das funções

3.14.1.1 int Bematech_FI_AberturaDoDia (const char * *valorSuprimento*, const char * *formaPagamento*)

Abertura do Dia.

Fazer a abertura do dia. Faz um suprimento, emite a leitura X e atualiza as informações COOInicial e GTInicial no arquivo de inicialização usadas pelas funções de relatório tipo 60. Portanto, se você for emitir o relatório "tipo 60 mestre" é obrigatório o uso dessa função.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AberturaDoDia("50,00", "Dinheiro");
```

Parâmetros:

- valorSuprimento* Valor para suprimento, com até 14 dígitos (2 casas decimais).
formaPagamento Descrição da forma de pagamento, com até 16 caracteres.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Se a forma de pagamento não for especificada, será utilizado Dinheiro.
Caso deseje não efetuar um suprimento, informe o valor 0 (zero).

3.14.1.2 int Bematech_FI_AtivaDesativaCorteProximoMFD (void)

Ativa Desativa Corte Próximo MFD.

Ativa ou desativa o corte do papel (acionamento da guilhotina) para o documento impresso.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_AtivaDesativaCorteProximoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função ativará ou desativará o corte do papel, do documento que será impresso, retornando a configuração default, ou seja, corte sempre ativado.
Função disponível para a impressora fiscal térmica MP-2000 TH FI versão 01.00.02 ou 01.01.01 e MP-2100 TH FI.

3.14.1.3 int Bematech_FI_DadosSintegra (const char * dataInicial, const char * dataFinal)

Dados Sintegra.

Ler informações da leitura da memória fiscal usadas para a geração do relatório sintegra. O retorno das informações geradas por esta função serão gravadas no arquivo RETORNO.TXT, na seguinte ordem:

Data	8 bytes com o formato AAAAMMDD (esta data refere-se a data em que a Redução Z foi emitida)
Número de Serie	20 bytes
Número Sequencial do ECF	3 bytes
Contador de Redução Z	6 bytes
Cont. de Reinício de Operação	6 bytes
GT Final	16 bytes
GT Inicial	16 bytes
Venda Bruta	16 bytes
Venda Líquida	16 bytes
Cancelamentos	12 bytes
Descontos	12 bytes
F (Substituição Tributária)	12 bytes
I (Isenção)	12 bytes
N (Não Incidência)	12 bytes
ISS	12 bytes
Situação Tributária de ICMS	4 bytes
Valor acumulado na Situação Tributária	12 bytes

Exemplo de uma linha de um retorno.txt:

```
2004012947080005718500147004300000000144323430000000014420587000000000001175600000000000053930000000035640000000050000000000279917000000000018101200000000001788
```

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_DadosSintegra("010104", "310104");
```

Parâmetros:

dataInicial Data inicial, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

dataFinal Data final, no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido na função

-4 O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows

-5 Erro ao abrir a porta de comunicação

-8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT

-27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Os campos "Situação Tributária de ICMS" e "Valor acumulado na Situação Tributária" poderão ser repetidos de acordo com o número de alíquotas utilizadas nas vendas.

Os campos já estão formatados no tamanho que o relatório do Sintegra exige. Esta função deve ser utilizada nas impressoras fiscais matriciais (MP-20 FI II e MP-40 FI II). Para as impressoras fiscais térmicas com MFD (Memória de Fita Detalhe), utilize a função [Bematech_FI_RelatorioSintegraMFD\(\)](#).

Cada linha do arquivo RETORNO.TXT corresponde a um dia de movimento na impressora.

3.14.1.4 int Bematech_FI_DadosSintegraMFD (const char * *cDataInicial*, const char * *cDataFinal*)

Dados Sintegra MFD.

Retorna as informações da memória fiscal utilizada na geração do relatório Sintegra, para impressoras no convênio ICMS 85/01.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_DadosSintegraMFD("010104", "310104");
```

Parâmetros:

cDataInicial Data inicial no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

cDataFinal Data final no formato ddmmaa, dd/mm/aa, ddmmaaaa ou dd/mm/aaaa.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

O retorno das informações geradas por esta função, serão gravadas no arquivo "RETORNO.TXT", na seguinte ordem:

Data:	8 bytes
Número de Serie:	20 bytes
Número Sequencial do ECF:	3 bytes
Contador de Redução Z:	6 bytes
Contador de Reinício de Operação:	6 bytes
GT Final:	16 bytes
GT Inicial:	16 bytes
Venda Bruta:	16 bytes
Venda Líquida:	16 bytes
Cancelamentos:	12 bytes
Cancelamentos ISS:	12 bytes
Descontos:	12 bytes
Descontos:	12 bytes
Acréscimos:	12 bytes
Acréscimos ISS:	12 bytes

Operação Não Fiscal:	12 bytes
F (Substituição Tributária):	12 bytes
FI (Substituição ISS):	12 bytes
I (Isenção):	12 bytes
II (Isenção ISS):	12 bytes
N (Não Incidência):	12 bytes
NI (Não Incidência ISS):	12 bytes
ISS:	12 bytes
Situação Tributária (valor percentual da alíquota) de ICMS:	4 bytes
Valor Acumulado na Situação Tributária:	12 bytes

O campo "Data", no formato AAAAMMDD, refere-se a data em que a Redução Z fora emitida. Os campos "Situação Tributária de ICMS" e "Valor Acumulado na Situação Tributária" poderão ser repetidos de acordo com o número de alíquotas utilizadas nas vendas. Os campos já estão formatados no tamanho que o relatório Sintegra exige. Cada linha do arquivo "RETORNO.TXT" corresponde a um dia de movimento na impressora. Segue abaixo um exemplo do conteúdo do arquivo "RETORNO.TXT":

```
2004012947080005718500147004300000000144323430000000014420587000000000001175600000000000539300000000
0000000050000000000279917000000000018101200000000001788
```

3.14.1.5 int Bematech_FI_FechamentoDoDia (void)

Fechamento do Dia.

Fazer o fechamento do dia. Emitir uma redução Z e atualizar as informações COOFinal e GTFinal no arquivo de inicialização usadas pelas funções de relatório tipo 60

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_FechamentoDoDia();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

3.14.1.6 int Bematech_FI_GeraRelatorioSintegraMFD (int iRelatorios, const char * cOrigem, const char * cDestino, const char * cMes, const char * cAno, const char * cRazaoSocial, const char * cEndereco, const char * cNumeroTmp, const char * cComplemento, const char * cBairro, const char * cCidade, const char * cCEPTmp, const char * cTelefoneTmp, const char * cFax, const char * cContato)

Gera Relatório Sintegra MFD.

Gera os relatórios para o Sintegra, somente da impressora fiscal térmica (MFD), a partir de um arquivo ".MFD".

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_GeraRelatorioSintegraMFD(63, "DOWNLOAD.MFD",
    "SINTEGRA.TXT", "11", "2003", "BEMATECH S/A",
    "Estrada de Santa Candida", "263", "Industria", "Santa Candida",
    "Curitiba", "82630490", "41 351-2700", "41 351-2863",
    "Fulano de Tal")
```

Parâmetros:

iRelatorios Variável inteira, com o tamanho de um byte, onde são definidos os relatórios a serem gerados. Segue a seguinte estrutura:

- 1 - gera o relatório tipo 60M (Mestre).
- 2 - gera o relatório tipo 60A (Analítico).
- 4 - gera o relatório tipo 60D (Diário).
- 8 - gera o relatório tipo 60I (Item).
- 16 - gera o relatório tipo 60R (Resumo mensal).
- 32 - gera o relatório tipo 75.

cOrigem Path e nome do arquivo MFD de origem. Por exemplo: "C:/DOWNLOAD.MFD".

cDestino Path e nome do arquivo onde o relatório será gerado. Por exemplo: "C:/SINTEGRA.TXT".

cMes Mês da geração do relatório, no formato MM.

cAno Ano da geração do relatório, no formato AAAA.

cRazaoSocial Razão social, com até 35 caracteres.

cEndereco Endereço, com até 34 caracteres.

cNumeroTmp Número do endereço, com até 5 caracteres.

cComplemento Complemento do endereço, com até 22 caracteres.

cBairro Bairro, com até 15 caracteres.

cCidade Cidade, com até 30 caracteres.

cCEPTmp CEP, com 8 caracteres.

cTelefoneTmp Telefone, com até 12 caracteres.

cFax Fax, com até 10 caracteres.

cContato Nome do contato, com até 18 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Erro de execução da função
- 2 Parâmetro inválido
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para gerar mais de um relatório, deve-se enviar a soma dos valores da variável "iRelatorios", por exemplo:

Relatório tipo 60M + tipo 60A + tipo 75: iRetorno = 34.

Os registros tipo 10, tipo 11 e o tipo 90, são gerados automaticamente.

Para o uso correto desta função, é necessário ter os arquivos libbemamfd.so ou libbemamfd2.so instalados.

3.14.1.7 `int Bematech_FI_HabilitaDesabilitaRetornoEstendidoMFD (const char * cFlagRetorno)`

Habilita Desabilita Retorno Estendido MFD.

Habilita e desabilita o retorno estendido na impressora MFD (ACK, ST1, ST2, ST3).

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_HabilitaDesabilitaRetornoEstendidoMFD("1");
```

Parâmetros:

cFlagRetorno Variável destinada a habilitar (1) ou desabilitar (0) o retorno estendido.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

Tabela de retornos para ST3:

- 0 - COMANDO OK
- 1 - COMANDO INVÁLIDO
- 2 - ERRO DESCONHECIDO
- 3 - NÚMERO DE PARÂMETRO INVÁLIDO
- 4 - TIPO DE PARÂMETRO INVÁLIDO
- 5 - TODAS ALÍQUOTAS JÁ PROGRAMADAS
- 6 - TOTALIZADOR NÃO FISCAL JÁ PROGRAMADO
- 7 - CUPOM FISCAL ABERTO
- 8 - CUPOM FISCAL FECHADO
- 9 - ECF OCUPADO
- 10 - IMPRESSORA EM ERRO
- 11 - IMPRESSORA SEM PAPEL
- 12 - IMPRESSORA COM CABEÇA LEVANTADA
- 13 - IMPRESSORA OFF LINE
- 14 - ALÍQUOTA NÃO PROGRAMADA
- 15 - TERMINADOR DE STRING FALTANDO
- 16 - ACRÉSCIMO OU DESCONTO MAIOR QUE O TOTAL DO CUPOM FISCAL
- 17 - CUPOM FISCAL SEM ITEM VENDIDO
- 18 - COMANDO NÃO EFETIVADO
- 19 - SEM ESPAÇO PARA NOVAS FORMAS DE PAGAMENTO
- 20 - FORMA DE PAGAMENTO NÃO PROGRAMADA
- 21 - ÍNDICE MAIOR QUE NÚMERO DE FORMA DE PAGAMENTO
- 22 - FORMAS DE PAGAMENTO ENCERRADAS
- 23 - CUPOM NÃO TOTALIZADO
- 24 - COMANDO MAIOR QUE 7Fh (127d)

- 25 - CUPOM FISCAL ABERTO E SEM ÍTEM
- 26 - CANCELAMENTO NÃO IMEDIATAMENTE APÓS
- 27 - CANCELAMENTO JÁ EFETUADO
- 28 - COMPROVANTE DE CRÉDITO OU DÉBITO NÃO PERMITIDO OU JÁ EMITIDO
- 29 - MEIO DE PAGAMENTO NÃO PERMITE TEF
- 30 - SEM COMPROVANTE NÃO FISCAL ABERTO
- 31 - COMPROVANTE DE CRÉDITO OU DÉBITO JÁ ABERTO
- 32 - REIMPRESSÃO NÃO PERMITIDA
- 33 - COMPROVANTE NÃO FISCAL JÁ ABERTO
- 34 - TOTALIZADOR NÃO FISCAL NÃO PROGRAMADO
- 35 - CUPOM NÃO FISCAL SEM ÍTEM VENDIDO
- 36 - ACRÉSCIMO E DESCONTO MAIOR QUE TOTAL CNF
- 37 - MEIO DE PAGAMENTO NÃO INDICADO
- 38 - MEIO DE PAGAMENTO DIFERENTE DO TOTAL DO RECEBIMENTO
- 39 - NÃO PERMITIDO MAIS DE UMA SANGRIA OU SUPRIMENTO
- 40 - RELATÓRIO GERENCIAL JÁ PROGRAMADO
- 41 - RELATÓRIO GERENCIAL NÃO PROGRAMADO
- 42 - RELATÓRIO GERENCIAL NÃO PERMITIDO
- 43 - MFD NÃO INICIALIZADA
- 44 - MFD AUSENTE
- 45 - MFD SEM NÚMERO DE SÉRIE
- 46 - MFD JÁ INICIALIZADA
- 47 - MFD LOTADA
- 48 - CUPOM NÃO FISCAL ABERTO
- 49 - MEMÓRIA FISCAL DESCONECTADA
- 50 - MEMÓRIA FISCAL SEM NÚMERO DE SÉRIE DA MFD
- 51 - MEMÓRIA FISCAL LOTADA
- 52 - DATA INICIAL INVÁLIDA
- 53 - DATA FINAL INVÁLIDA
- 54 - CONTADOR DE REDUÇÃO Z INICIAL INVÁLIDO
- 55 - CONTADOR DE REDUÇÃO Z FINAL INVÁLIDO
- 56 - ERRO DE ALOCAÇÃO
- 57 - DADOS DO RTC INCORRETOS
- 58 - DATA ANTERIOR AO ÚLTIMO DOCUMENTO EMITIDO
- 59 - FORA DE INTERVENÇÃO TÉCNICA
- 60 - EM INTERVENÇÃO TÉCNICA
- 61 - ERRO NA MEMÓRIA DE TRABALHO
- 62 - JÁ HOVE MOVIMENTO NO DIA
- 63 - BLOQUEIO POR RZ
- 64 - FORMA DE PAGAMENTO ABERTA
- 65 - AGUARDANDO PRIMEIRO PROPRIETÁRIO
- 66 - AGUARDANDO RZ
- 67 - ECF OU LOJA IGUAL A ZERO
- 68 - CUPOM ADICIONAL NÃO PERMITIDO
- 69 - DESCONTO MAIOR QUE TOTAL VENDIDO EM ICMS
- 70 - RECEBIMENTO NÃO FISCAL NULO NÃO PERMITIDO
- 71 - ACRÉSCIMO OU DESCONTO MAIOR QUE TOTAL NÃO FISCAL
- 72 - MEMÓRIA FISCAL LOTADA PARA NOVO CARTUCHO
- 73 - ERRO DE GRAVAÇÃO NA MF
- 74 - ERRO DE GRAVAÇÃO NA MFD
- 75 - DADOS DO RTC ANTERIORES AO ÚLTIMO DOC ARMAZENADO
- 76 - MEMÓRIA FISCAL SEM ESPAÇO PARA GRAVAR LEITURAS DA MFD
- 77 - MEMÓRIA FISCAL SEM ESPAÇO PARA GRAVAR VERSAO DO SB
- 78 - DESCRIÇÃO IGUAL A DEFAULT NÃO PERMITIDO

- 79 - EXTRAPOLADO NÚMERO DE REPETIÇÕES PERMITIDAS
- 80 - SEGUNDA VIA DO COMPROVANTE DE CRÉDITO OU DÉBITO NÃO PERMITIDO
- 81 - PARCELAMENTO FORA DA SEQUÊNCIA
- 82 - COMPROVANTE DE CRÉDITO OU DÉBITO ABERTO
- 83 - TEXTO COM SEQUÊNCIA DE ESC INVÁLIDA
- 84 - TEXTO COM SEQUÊNCIA DE ESC INCOMPLETA
- 85 - VENDA COM VALOR NULO
- 86 - ESTORNO DE VALOR NULO
- 87 - FORMA DE PAGAMENTO DIFERENTE DO TOTAL DA SANGRIA
- 88 - REDUÇÃO NÃO PERMITIDA EM INTERVENÇÃO TÉCNICA
- 89 - AGUARDANDO RZ PARA ENTRADA EM INTERVENÇÃO TÉCNICA
- 90 - FORMA DE PAGAMENTO COM VALOR NULO NÃO PERMITIDO
- 91 - ACRÉSCIMO E DESCONTO MAIOR QUE VALOR DO ÍTEM
- 92 - AUTENTICAÇÃO NÃO PERMITIDA
- 93 - TIMEOUT NA VALIDAÇÃO
- 94 - COMANDO NÃO EXECUTADO EM IMPRESSORA BILHETE DE PASSAGEM
- 95 - COMANDO NÃO EXECUTADO EM IMPRESSORA DE CUPOM FISCAL
- 96 - CUPOM NÃO FISCAL FECHADO
- 97 - PARÂMETRO NÃO ASCII EM CAMPO ASCII
- 98 - PARÂMETRO NÃO ASCII NUMÉRICO EM CAMPO ASCII NUMÉRICO
- 99 - TIPO DE TRANSPORTE INVÁLIDO
- 100 - DATA E HORA INVÁLIDA
- 101 - SEM RELATÓRIO GERENCIAL OU COMPROVANTE DE CRÉDITO OU DÉBITO ABERTO
- 102 - NÚMERO DO TOTALIZADOR NÃO FISCAL INVÁLIDO
- 103 - PARÂMETRO DE ACRÉSCIMO OU DESCONTO INVÁLIDO
- 104 - ACRÉSCIMO OU DESCONTO EM SANGRIA OU SUPRIMENTO NÃO PERMITIDO
- 105 - NÚMERO DO RELATÓRIO GERENCIAL INVÁLIDO
- 106 - FORMA DE PAGAMENTO ORIGEM NÃO PROGRAMADA
- 107 - FORMA DE PAGAMENTO DESTINO NÃO PROGRAMADA
- 108 - ESTORNO MAIOR QUE FORMA PAGAMENTO
- 109 - CARACTER NUMÉRICO NA CODIFICAÇÃO GT NÃO PERMITIDO
- 110 - ERRO NA INICIALIZAÇÃO DA MF
- 111 - NOME DO TOTALIZADOR EM BRANCO NÃO PERMITIDO
- 112 - DATA E HORA ANTERIORES AO ÚLTIMO DOC ARMAZENADO
- 113 - PARÂMETRO DE ACRÉSCIMO OU DESCONTO INVÁLIDO
- 114 - ÍTEM ANTERIOR AOS TREZENTOS ÚLTIMOS
- 115 - ÍTEM NÃO EXISTE OU JÁ CANCELADO
- 116 - CÓDIGO COM ESPAÇOS NÃO PERMITIDO
- 117 - DESCRICAO SEM CARACTER ALFABÉTICO NÃO PERMITIDO
- 118 - ACRÉSCIMO MAIOR QUE VALOR DO ÍTEM
- 119 - DESCONTO MAIOR QUE VALOR DO ÍTEM
- 120 - DESCONTO EM ISS NÃO PERMITIDO
- 121 - ACRÉSCIMO EM ÍTEM JÁ EFETUADO
- 122 - DESCONTO EM ÍTEM JÁ EFETUADO
- 123 - ERRO NA MEMÓRIA FISCAL CHAMAR CREDENCIADO
- 124 - AGUARDANDO GRAVAÇÃO NA MEMÓRIA FISCAL
- 125 - CARACTER REPETIDO NA CODIFICAÇÃO DO GT
- 126 - VERSÃO JÁ GRAVADA NA MEMÓRIA FISCAL
- 127 - ESTOURO DE CAPACIDADE NO CHEQUE
- 128 - TIMEOUT NA LEITURA DO CHEQUE
- 129 - MÊS INVÁLIDO
- 130 - COORDENADA INVÁLIDA
- 131 - SOBREPOSIÇÃO DE TEXTO
- 132 - SOBREPOSIÇÃO DE TEXTO NO VALOR

- 133 - SOBREPOSIÇÃO DE TEXTO NO EXTENSO
- 134 - SOBREPOSIÇÃO DE TEXTO NO FAVORECIDO
- 135 - SOBREPOSIÇÃO DE TEXTO NA LOCALIDADE
- 136 - SOBREPOSIÇÃO DE TEXTO NO OPCIONAL
- 137 - SOBREPOSIÇÃO DE TEXTO NO DIA
- 138 - SOBREPOSIÇÃO DE TEXTO NO MÊS
- 139 - SOBREPOSIÇÃO DE TEXTO NO ANO
- 140 - USANDO MFD DE OUTRO ECF
- 141 - PRIMEIRO DADO DIFERENTE DE ESC OU 1C
- 142 - NÃO PERMITIDO ALTERAR SEM INTERVENÇÃO TÉCNICA
- 143 - DADOS DA ÚLTIMA RZ CORROMPIDOS
- 144 - COMANDO NÃO PERMITIDO NO MODO INICIALIZAÇÃO
- 145 - AGUARDANDO ACERTO DE RELÓGIO
- 146 - MFD JÁ INICIALIZADA PARA OUTRA MF
- 147 - AGUARDANDO ACERTO DO RELÓGIO OU DESBLOQUEIO PELO TECLADO
- 148 - VALOR FORMA DE PAGAMENTO MAIOR QUE MÁXIMO PERMITIDO
- 149 - RAZÃO SOCIAL EM BRANCO
- 150 - NOME DE FANTASIA EM BRANCO
- 151 - ENDEREÇO EM BRANCO
- 152 - ESTORNO DE CDC NÃO PERMITIDO
- 153 - DADOS DO PROPRIETÁRIO IGUAIS AO ATUAL
- 154 - ESTORNO DE FORMA DE PAGAMENTO NÃO PERMITIDO
- 155 - DESCRIÇÃO FORMA DE PAGAMENTO IGUAL JÁ PROGRAMADA
- 156 - ACERTO DE HORÁRIO DE VERÃO SÓ IMEDIATAMENTE APÓS RZ
- 157 - IT NÃO PERMITIDA MF RESERVADA PARA RZ
- 158 - SENHA CNPJ INVÁLIDA
- 159 - TIMEOUT NA INICIALIZAÇÃO DA NOVA MF
- 160 - NÃO ENCONTRADO DADOS NA MFD
- 161 - SANGRIA OU SUPRIMENTO DEVEM SER ÚNICOS NO CNF
- 162 - ÍNDICE DA FORMA DE PAGAMENTO NULO NÃO PERMITIDO
- 163 - UF DESTINO INVÁLIDA
- 164 - TIPO DE TRANSPORTE INCOMPATÍVEL COM UF DESTINO
- 165 - DESCRIÇÃO DO PRIMEIRO ÍTEM DO BILHETE DE PASSAGEM DIFERENTE DE "TAR-IFA"
- 166 - AGUARDANDO IMPRESSÃO DE CHEQUE OU AUTENTICAÇÃO
- 167 - NÃO PERMITIDO PROGRAMAÇÃO CNPJ IE COM ESPAÇOS EM BRANCO
- 168 - NÃO PERMITIDO PROGRAMAÇÃO UF COM ESPAÇOS EM BRANCO
- 169 - NÚMERO DE IMPRESSÕES DA FITA DETALHE NESTA INTERVENÇÃO TÉCNICA ESGOTADO
- 170 - CF JÁ SUBTOTALIZADO
- 171 - CUPOM NÃO SUBTOTALIZADO
- 172 - ACRÉSCIMO EM SUBTOTAL JÁ EFETUADO
- 173 - DESCONTO EM SUBTOTAL JÁ EFETUADO
- 174 - ACRÉSCIMO NULO NÃO PERMITIDO
- 175 - DESCONTO NULO NÃO PERMITIDO
- 176 - CANCELAMENTO DE ACRÉSCIMO OU DESCONTO EM SUBTOTAL NÃO PERMITIDO
- 177 - DATA INVÁLIDA
- 178 - VALOR DO CHEQUE NULO NÃO PERMITIDO
- 179 - VALOR DO CHEQUE INVÁLIDO
- 180 - CHEQUE SEM LOCALIDADE NÃO PERMITIDO
- 181 - CANCELAMENTO ACRÉSCIMO EM ÍTEM NÃO PERMITIDO
- 182 - CANCELAMENTO DESCONTO EM ÍTEM NÃO PERMITIDO
- 183 - NÚMERO MÁXIMO DE ÍTENS ATINGIDO
- 184 - NÚMERO DE ÍTEM NULO NÃO PERMITIDO

- 185 - MAIS QUE DUAS ALÍQUOTAS DIFERENTES NO BILHETE DE PASSAGEM NÃO PERMITIDO
- 186 - ACRÉSCIMO OU DESCONTO EM ITEM NÃO PERMITIDO
- 187 - CANCELAMENTO DE ACRÉSCIMO OU DESCONTO EM ITEM NÃO PERMITIDO
- 188 - CLICHE JÁ IMPRESSO
- 189 - TEXTO OPCIONAL DO CHEQUE EXCEDEU O MÁXIMO PERMITIDO
- 190 - IMPRESSÃO AUTOMÁTICA NO VERSO NÃO PERMITIDO NESTE EQUIPAMENTO
- 191 - TIMEOUT NA INSERÇÃO DO CHEQUE
- 192 - OVERFLOW NA CAPACIDADE DE TEXTO DO COMPROVANTE DE CRÉDITO OU DÉBITO
- 193 - PROGRAMAÇÃO DE ESPAÇOS ENTRE CUPONS MENOR QUE O MÍNIMO PERMITIDO
- 194 - EQUIPAMENTO NÃO POSSUI LEITOR DE CHEQUE
- 195 - PROGRAMAÇÃO DE ALÍQUOTA COM VALOR NULO NÃO PERMITIDO
- 196 - PARÂMETRO BAUD RATE INVÁLIDO
- 197 - CONFIGURAÇÃO PERMITIDA SOMENTE PELA PORTA DOS FISCO
- 198 - VALOR TOTAL DO ITEM EXCEDE 11 DÍGITOS
- 199 - PROGRAMAÇÃO DA MOEDA COM ESPAÇOS EM BRACO NÃO PERMITIDO
- 200 - CASAS DECIMAIS DEVEM SER PROGRAMADAS COM 2 OU 3
- 201 - NÃO PERMITE CADASTRAR USUÁRIOS DIFERENTES NA MESMA MFD
- 202 - IDENTIFICAÇÃO DO CONSUMIDOR NÃO PERMITIDA PARA SANGRIA OU SUPRIMENTO
- 203 - CASAS DECIMAIS EM QUANTIDADE MAIOR DO QUE A PERMITIDA
- 204 - CASAS DECIMAIS DO UNITÁRIO MAIOR DO QUE O PERMITIDA
- 205 - POSIÇÃO RESERVADA PARA ICMS
- 206 - POSIÇÃO RESERVADA PARA ISS
- 207 - TODAS AS ALÍQUOTAS COM A MESMA VINCULAÇÃO NÃO PERMITIDO
- 208 - DATA DE EMBARQUE ANTERIOR A DATA DE EMISSÃO
- 209 - ALÍQUOTA DE ISS NÃO PERMITIDA SEM INSCRIÇÃO MUNICIPAL
- 210 - RETORNO PACOTE CLICHE FORA DA SEQUÊNCIA
- 211 - ESPAÇO PARA ARMAZENAMENTO DO CLICHE ESGOTADO
- 212 - CLICHE GRÁFICO NÃO DISPONÍVEL PARA CONFIRMAÇÃO
- 213 - CRC DO CLICHE GRÁFICO DIFERENTE DO INFORMADO
- 214 - INTERVALO INVÁLIDO
- 215 - USUÁRIO JÁ PROGRAMADO
- 217 - DETECTADA ABERTURA DO EQUIPAMENTO
- 218 - CANCELAMENTO DE ACRÉSCIMO/DESCONTO NÃO PERMITIDO

3.14.1.8 `int Bematech_FI_ImpressaoFitaDetalhe (const char * cTipo, const char * cDadoInicial, const char * cDadoFinal, const char * cUsuario)`

Impressão Fita Detalhe.

Imprimir a Fita Detalhe das Impressoras com MFD

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_ImpressaoFitaDetalhe("2", "000001", "000100", "1");
```

Parâmetros:

cTipo 0 - total, 1 - por data, 2 - por COO

cDadoInicial Data ou COO inicial, sendo data no formato DDMMAA ou DDMMAAA ou ainda COO com no máximo 6 dígitos

cDadoFinal Data ou COO final, sendo data no formato DDMMAA ou DDMMAAA ou ainda COO com no máximo 6 dígitos

cUsuario Número de ordem do proprietário do ECF com até 3 dígitos

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 5* Erro ao abrir a porta de comunicação
- 8* Erro ao gerar o arquivo

Nota:

Os parâmetros *cDadoInicial* e *cDadoFinal* são obrigatórios se o tipo de download for por data ou por COO.

O parâmetro *cUsuario* é obrigatório se o download for por COO.

3.14.1.9 int Bematech_FI_ImprimeClicheMFD (void)

Imprime Clichê MFD.

Imprime as linhas do clichê.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ImprimeClicheMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1* Ok
- 0* Erro de comunicação
- 2* Parâmetro inválido
- 4* Arquivo de inicialização não encontrado ou inválido
- 5* Erro ao abrir a porta de comunicação
- 27* Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função só está disponível para a versão 01.00.00, da impressora fiscal MFD.

Este comando permite que as informações fixas do clichê do proprietário sejam impressas antes da definição do tipo do próximo documento a ser impresso. A utilização deste comando associado à programação previa do espaço entre cupons com valor adequado permitirá ao usuário diminuir o consumo da bobina de papel.

3.14.1.10 int Bematech_FI_ImprimeConfiguracoesImpressora (void)

Imprime Configurações Impressora.

Imprime configurações da impressora fiscal em um relatório gerencial. Será emitida uma leitura X antes. Veja abaixo em "Nota" as informações que serão impressas.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ImprimeConfiguracoesImpressora();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar no arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0 (ACK, ST1 e ST2)

Nota:

Serão impressas as seguintes configurações:

Data da impressora	27/06/2001
Hora da impressora	11:38:37
Versão do firmware	0310
Número de série	4708991023070
Número da loja	0001
Número do caixa	0001
Símbolo da moeda	R\$
Modo de operação	Arredondamento
Horário de verão	Não
Já houve redução Z	Não
Memória fiscal	Com espaço
Versão da dll	1.9
log da dll	Desabilitado
Lin. Imp. após pouco papel	0
Tipo Impressora	fiscal + gaveta+autenticação

3.14.1.11 int Bematech_FI_ImprimeDepartamentos (void)

Imprime Departamentos.

Retorna os departamentos e seus valores acumulados em um relatório gerencial.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ImprimeDepartamentos();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido na função
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Uma leitura X será emitida antes.

Essas informações eram impressas na leitura X até a versão 3.0 e foram retiradas por solicitação do fisco.

3.14.1.12 int Bematech_FI_LeArquivoRetorno (char * cDados)

Lê Arquivo Retorno.

Lê o conteúdo do arquivo "RETORNO.TXT".

```
// exemplo em C/C++
char cCupom[6];
int iACK, iST1, iST2;
int iRetorno;
iRetorno = Bematech_FI_NumeroCupom(cCupom);
iRetorno = Bematech_FI_RetornoImpressora(iACK, iST1, iST2);
iRetorno = Bematech_FI_LeArquivoRetorno(cCupom)
```

Parâmetros:

→ *cDados* Variável destinada a receber a informação lida do arquivo "RETORNO.TXT".

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação

Nota:

O tamanho de "cDados" varia de acordo com a informação a ser lida. Por exemplo: número do cupom (6 bytes), número de série (15 bytes).

Esta função só deverá ser usada, caso esteja trabalhando como "Cliente/Servidor" (Windows Terminal Service ou Frame-Relay).

Esta função deve ser usada logo após a função que originou o retorno da informação da impressora.

3.14.1.13 int Bematech_FI_MapasResumo (void)

Mapa Resumo.

Gera o relatório "Mapa Resumo" referente ao movimento do dia.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_MapasResumo();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As informações serão geradas no arquivo "RETORNO.TXT", no diretório configurado no parâmetro "path" do arquivo .ini.

O diretório default configurado é o raiz (C:\).

As informações contidas no mapa resumo podem variar de estado para estado.

Esta função gera o relatório com as informações padrões que são usadas na maiorias dos estados. Este relatório terá o seguinte layout:

Contador de Redução Z..:	0312
COO.....:	000026
Venda Bruta.....:	43,73
Venda Líquida.....:	25,50
Cancelamentos.....:	11,14
Acréscimos.....:	2,33
Descontos.....:	7,09
ISS.....:	0,00
Isenção.....:	0,00
Não Incidência.....:	1,95
Substituição Tributária:	23,55
1200.....:	0,00
1700.....:	0,00

Após a linha "Substituição Tributária" serão gravadas as informações de vendas referente as alíquotas de ICMS. Portanto, o número de linhas irá variar de acordo com o número de alíquotas de ICMS cadastradas na impressora.

As informações contidas no mapa resumo serão referentes aos dados da última redução Z. Desta forma, para gerar o mapa resumo referente ao movimento do dia, esta função deve ser executada após a redução Z. Caso contrário, o mapa resumo será gerado com as informações referentes ao movimento do dia anterior.

3.14.1.14 int Bematech_FI_MapasResumoMFD (void)

Mapa Resumo MFD.

Gera o relatório "Mapa Resumo" referente ao movimento do dia.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_MapaResumoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As informações serão geradas no arquivo "RETORNO.TXT", no diretório configurado no parâmetro "path" do arquivo .ini.

O diretório default configurado é o raiz (C:\).

As informações contidas no mapa resumo podem variar de estado para estado.

Esta função gera o relatório com as informações padrões que são usadas na maiorias dos estados. Este relatório terá o seguinte layout:

Contador de Redução Z:	0312
COO.....	000026
Venda Bruta.....	43,73
Venda Líquida.....	25,50
Cancelamentos ICMS...	11,14
Cancelamentos ISSQN..	0,00
Acrescimos ICMS.....	2,33
Acrescimos ISSQN.....	0,00
Descontos ICMS.....	7,09
Descontos ISSQN.....	0,00
Substituicao ICMS.....	23,55
Substituicao ISSQN...	0,00
Isencao ICMS.....	0,00
Isencao ISSQN.....	0,00
Nao incidencia ICMS..	0,00
Nao incidencia ISSQN..	0,00
1200.....	0,00
1700.....	0,00
ISS.....	0,00

O campo "Venda Bruta" é calculado seguindo a fórmula: Soma das alíquotas de ICMS e ISS + II + NN + FF + SI + SN + SF + Cancelamentos + Descontos.

O campo "Venda líquida", é calculado seguindo a fórmula: Venda bruta - ISS - SI - SN - SF - Cancelamentos (ICMS e ISSQN) - Descontos (ICMS e ISSQN) da última redução.

Após a linha "Não incidencia ISQN" serão gravadas as informações de vendas referente as alíquotas de ICMS. Portanto, o número de linhas irá variar de acordo com o número de alíquotas de ICMS cadastradas na impressora.

As informações contidas no mapa resumo serão referentes aos dados da última redução Z. Desta forma, para gerar o mapa resumo referente ao movimento do dia, esta função deve ser executada após a redução Z. Caso contrário, o mapa resumo será gerado com as informações referentes ao movimento do dia anterior.

3.14.1.15 int Bematech_FI_NumeroSerieCriptografado (BYTE * *cNumSerie*)

Número Série Criptografado.

Retorna o número de série da impressora criptografado.

```
// Exemplo em C/C++
char cNumeroSerieCriptografado[21];
int iRetorno;
iRetorno =
Bematech_FI_NumeroSerieCriptografado(cNumeroSerieCriptografado);
```

Parâmetros:

→ *cNumSerie* Variável, destinada a receber o número de série criptografado da impressora. Seu tamanho deverá ser de 16 caracteres para impressoras do convênio 156/94 (MP-20 FI II e MP-40 FI II) ou 21 caracteres para demais impressoras - convênio 85/01.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para os modelos MP-25 FI, MP-50 FI, MP-2000 TH FI, MP-2100 TH FI e MP-6000 TH FI, a chave "Impressora" deve estar igual a "1" no arquivo de configuração ".INI", para que sejam retornados os 20 caracteres para a criptografia. Caso esta chave seja igual a "0", o retorno possuirá somente 15 caracteres.

3.14.1.16 int Bematech_FI_NumeroSerieDescriptografado (const BYTE * *cNumSerieCriptografado*, char * *cNumSerieDescriptografado*)

Número Série Descriptografado.

Retorna o número de serie da impressora, descriptografado, a partir do número de série criptografado informado.

```
// Exemplo em C/C++
char cNumeroSerieCriptografado[21];
char cNumeroSerieDescriptografado[21];
int iRetorno;
iRetorno =
Bematech_FI_NumeroSerieCriptografado(cNumeroSerieCriptografado);
iRetorno =
Bematech_FI_NumeroSerieDescriptografado(cNumeroSerieDescriptografado);
```

Parâmetros:

cNumSerieCriptografado Variável, com o número de série criptografado pela função Bematech_FI_NumeroSerieCriptografado.

→ *cNumSerieDescriptografado* Variável, destinada a receber o número de série descriptografado. Seu tamanho deverá ser de 16 caracteres para impressoras do convênio 156/94 (MP-20 FI II e MP-40 FI II) ou 21 caracteres para demais impressoras - convênio 85/01.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

3.14.1.17 int Bematech_FI_RegistrosTipo60 (void)

Registros Tipo 60.

Retorna os registros tipo 60 Analítico e Mestre completos da impressora.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RegistrosTipo60();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Esta função deve ser utilizada diariamente, após a emissão da Redução Z.

Os campos já estão formatados no tamanho que o relatório do Sintegra exige.

Esta função deve ser utilizada nas impressoras fiscais matriciais (MP-20 FI II e MP-40 FI II).

Para as impressoras fiscais térmicas com MFD, utilize a função [Bematech_FI_RelatorioSintegraMFD\(\)](#).

O retorno das informações geradas por esta função, serão gravadas no arquivo RETORNO.TXT, na seguinte ordem:

Registro Tipo 60 Analítico:

Campo

Conteúdo

Tipo	60
Subtipo	A
Data de Emissão	Data de Emissão dos Cupons Fiscais
Número de Série do ECF	Número de Série do ECF
Situação Tributária/Alíquota	Identificador da Situação Tributária/Alíquota do ICMS
Valor Acumulado no Totalizador Parcial Brancos	Valor acumulado no final do dia no totalizador parcial Brancos

Registro Tipo 60 Mestre:

Campo	Conteúdo
Tipo	60
Subtipo	M
Data de Emissão	Data de Emissão dos Cupons Fiscais
Número de Série do ECF	Número de Série do ECF
Número do ECF	Número do ECF
Modelo do Documento Fiscal	Código do Modelo Documento Fiscal
COO Inicial	Primeiro Cupom Fiscal Emitido
COO Final	Último Cupom Fiscal Emitido
Contador de Redução Z	Reduções Z
Contador de Reinício de Operação	Reinício de Operações
Venda Bruta	Valor da Venda Bruta
Grande Total	Valor do GT
Brancos	

3.14.1.18 `int Bematech_FI_RelatorioSintegraMFD (int iRelatorios, const char * cArquivo, const char * cMes, const char * cAno, const char * cRazaoSocial, const char * cEndereco, const char * cNumeroTmp, const char * cComplemento, const char * cBairro, const char * cCidade, const char * cCEPTmp, const char * cTelefoneTmp, const char * cFaxTmp, const char * cContato)`

Relatório Sintegra MFD.

Gera os relatórios para o Sintegra, somente na impressora fiscal térmica (MFD).

```
// Exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_RelatorioSintegraMFD(63, "SINTEGRA.TXT", "11", "2003",
    "BEMATECH S/A", "Estrada de Santa Candida", "263", "Industria",
    "Santa Candida", "Curitiba", "82630490", "41 351-2700",
    "41 351-2863", "Fulano de Tal");
```

Parâmetros:

iRelatorios Variável, inteira, com o tamanho de um byte, onde são definidos os relatórios a serem gerados. Segue a seguinte estrutura:

- 1 - gera o relatório tipo 60M (Mestre).
- 2 - gera o relatório tipo 60A (Analítico).
- 4 - gera o relatório tipo 60D (Diário).
- 8 - gera o relatório tipo 60I (Item).
- 16 - gera o relatório tipo 60R (Resumo mensal).
- 32 - gera o relatório tipo 75.

cArquivo Path e nome do arquivo onde o relatório será gerado. Por exemplo: "C:/SINTEGRA.TXT".

cMes Mês da geração do relatório, no formato MM.

cAno Ano da geração do relatório, no formato AAAA.

cRazaoSocial Razão social, com até 35 caracteres.

cEndereco Endereço, com até 34 caracteres.

cNumeroTmp Número do endereço, com até 5 caracteres.

cComplemento Complemento do endereço, com até 22 caracteres.

cBairro Bairro, com até 15 caracteres.

cCidade Cidade, com até 30 caracteres.

cCEPTmp CEP, com 8 caracteres.

cTelefoneTmp Telefone, com até 12 caracteres.

cFaxTmp Fax, com até 10 caracteres.

cContato Nome do contato, com até 18 caracteres.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução da função

-2 Parâmetro inválido

-4 Arquivo de inicialização não encontrado ou inválido

-5 Erro ao abrir a porta de comunicação

-27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

Para gerar mais de um relatório, deve-se enviar a soma dos valores da variável "iRelatorios", por exemplo:

```
Relatório tipo 60M + tipo 60A + tipo 75: iRetorno = 34.
```

Os registros tipo 10, tipo 11 e o tipo 90, são gerados automaticamente.

Para o uso correto desta função, é necessário ter os arquivos libbemamfd.so ou libbemamfd2.so instalados.

3.14.1.19 int Bematech_FI_RelatorioTipo60Analitico (void)

Relatório Tipo 60 Analítico.

Gera o relatório "Tipo 60 analítico" exigido pelo convênio de ICMS 85/2001.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RelatorioTipo60Analitico();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

1 Ok

0 Erro de comunicação

-4 Arquivo de inicialização não encontrado ou inválido

- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As informações serão geradas no arquivo "RETORNO.TXT", no diretório configurado no parâmetro "path" do arquivo .ini.

O diretório default configurado é o raiz (C:\).

O relatório gerado terá o seguinte layout:

```
Tipo do relatório.....:          60
Subtipo.....:                A
Data de emissão.....:          12/05/03
Número de série.....:          5708990400028
Cancelamentos.....:           11,14
Descontos.....:                7,09
F.....:                       18,55
I.....:                         0,00
N.....:                         1,95
1200.....:                      0,00
1700.....:                      0,00
ISS.....:                       0,00
```

Após a linha "N..." serão gravadas as informações de vendas referentes às alíquotas tributárias. Portanto, o número de linhas irá variar de acordo com o número de alíquotas cadastradas na impressora. As informações contidas no relatório Tipo 60 Analítico são referentes aos dados da última Redução Z. Desta forma, para gerar o relatório referente ao movimento do dia você deve executar essa função após a Redução Z, caso contrário as informações geradas serão referentes ao movimento do dia anterior.

3.14.1.20 int Bematech_FI_RelatorioTipo60AnaliticoMFD (void)

Relatório Tipo 60 Analítico MFD.

Gera o relatório "Tipo 60 analítico" exigido pelo convênio de ICMS 85/2001.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RelatorioTipo60AnaliticoMFD();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As informações serão geradas no arquivo "RETORNO.TXT", no diretório configurado no parâmetro "path" do arquivo .ini.

O diretório default configurado é o raiz (C:\).

O relatório gerado terá o seguinte layout:

Tipo do relatório.....:	60
Subtipo.....:	A
Data de emissão.....:	12/05/2003
Número de série.....:	5708990400028
Cancelamentos.....:	11,14
Descontos.....:	7,09
F.....:	18,55
I.....:	0,00
N.....:	1,95
1200.....:	0,00
1700.....:	0,00
ISS.....:	0,00

Após a linha "N..." serão gravadas as informações de vendas referentes às alíquotas tributárias. Portanto, o número de linhas irá variar de acordo com o número de alíquotas cadastradas na impressora. As informações contidas no relatório Tipo 60 Analítico são referentes aos dados da última Redução Z. Desta forma, para gerar o relatório referente ao movimento do dia você deve executar essa função após a Redução Z, caso contrário as informações geradas serão referentes ao movimento do dia anterior.

3.14.1.21 int Bematech_FI_RelatorioTipo60Mestre (void)

Relatório Tipo 60 Mestre.

Gera o relatório "Tipo 60 mestre" exigido pelo convênio de ICMS 85/2001.

```
// Exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_RelatorioTipo60Mestre();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 4 Arquivo de inicialização não encontrado ou inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao criar ou gravar arquivo STATUS.TXT ou RETORNO.TXT
- 27 Status da impressora diferente de 6,0,0,0 (ACK, ST1, ST2 e ST3)

Nota:

As informações serão geradas no arquivo "RETORNO.TXT", no diretório configurado no parâmetro "path" do arquivo .ini.

O diretório default configurado é o raiz (C:\).

O relatório gerado terá o seguinte layout:

Tipo do relatório.....:	60
Subtipo.....:	M
Data de emissão.....:	12/05/03
Número de série.....:	5708990400028
Número do equipamento.....:	0001
Modelo do documento fiscal:	2D
COO inicial.....:	000001
COO final.....:	000012
Contador de reduções.....:	0307
Reinício de Operacao.....:	0129
Venda Bruta.....:	38,73
Totalizador geral.....:	6.169,21

As informações contidas no relatório Tipo 60 Mestre são alimentadas pelas funções: [Bematech_FI_AberturaDoDia\(\)](#) e [Bematech_FI_FechamentoDoDia\(\)](#).

A função [Bematech_FI_RelatorioTipo60Mestre](#) deve ser executada após a função [Bematech_FI_FechamentoDoDia\(\)](#) ou após a Redução Z (caso a redução tenha sido executada automaticamente pela impressora).

3.14.1.22 void Bematech_FI_ReloadINIFile (void)

Reload INI File.

Permite a mudança dos parametros do arquivo .INI "on the fly".

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_ReloadINIFile();
```

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 1 - OK.
- 0 - Erro de comunicação.

3.14.1.23 int Bematech_FI_VerificaImpressoraLigada (void)

Verifica Impressora Ligada.

Verifica se a impressora está ligada ou conectada no computador.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_VerificaImpressoraLigada();
```

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Impressora Ligada
- 0 Erro de comunicação
- 4 O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows
- 5 Erro ao abrir a porta de comunicação
- 6 Impressora desligada ou cabo de comunicação desconectado

3.15 Download MFD

Funções

- int [Bematech_FI_DownloadMF](#) (const char *cArquivo)

- int `Bematech_FI_DownloadMFD` (const char *cArquivo, const char *cTipoDownload, const char *cDadoInicial, const char *cDadoFinal, const char *cUsuario)
- int `Bematech_FI_DownloadSB` (const char *cArquivo)
- int `Bematech_FI_FormatoDadosMFD` (const char *cArquivoMFD, const char *cDestino, const char *cFormato, const char *cTipoDownload, const char *cDadoInicial, const char *cDadoFinal, const char *cUsuario)

3.15.1 Documentação das funções

3.15.1.1 int Bematech_FI_DownloadMF (const char * cArquivo)

Download MF.

Fazer o Download da Memória Fiscal das Impressoras com CONV85.

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_DownloadMF("MFISCAL.MF");
```

Parâmetros:

cArquivo Nome do arquivo a ser gerado (Ex. "Download.mf")

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 2 Parâmetro inválido
- 5 Erro ao abrir a porta de comunicação
- 8 Erro ao gerar o arquivo

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.

3.15.1.2 int Bematech_FI_DownloadMFD (const char * cArquivo, const char * cTipoDownload, const char * cDadoInicial, const char * cDadoFinal, const char * cUsuario)

Download MFD.

Fazer o download da memória de Fita Detalhe das Impressoras Bematech

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_DownloadMFD("DOWNLOAD.MFD", "2", "000001", "000010", "1");
```

Parâmetros:

cArquivo Nome do arquivo a ser gerado (Ex: mp2100.mfd)

cTipoDownload Tipo do download a ser efetuado.

cDadoInicial Data ou COO inicial, sendo a data nos formatos DDMMAA ou DDMMAAA ou ainda o COO com no máximo 6 dígitos

cDadoFinal Data ou COO final, sendo a data nos formatos DDMMAA ou DDMMAAA ou ainda o COO com no máximo 6 dígitos

cUsuario Número de ordem do proprietário do ECF. Exemplo: primeiro proprietário "cUsuario = 1"

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-5 Erro ao abrir a porta de comunicação

-8 Erro ao gerar o arquivo

Nota:

cTipoDownload deve ser "0" para download total, "1" para download por data ou "2" para download por COO

A numeração dos COOs é diferente para cada proprietário.

Os parâmetros cDadoInicial e cDadoFinal são obrigatórios se o tipo de download for por data ou por COO.

O parâmetro cUsuario é obrigatório se o download for por COO.

Esta função é utilizada somente nas impressoras fiscais térmicas.

Esta função não retornará os status ACK, ST1 e ST2 da impressora, pois ela é utilizada, apenas, para o download da MFD.

3.15.1.3 int Bematech_FI_DownloadSB (const char * cArquivo)

Download SB.

Fazer o Download do Software Básico das Impressoras com CONV85

```
// exemplo em C/C++
int iRetorno;
iRetorno = Bematech_FI_DownloadSB("DOWNLOAD.SB");
```

Parâmetros:

cArquivo nome do arquivo a ser gerado (Ex. "Download.sb")

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-2 Parâmetro inválido

-5 Erro ao abrir a porta de comunicação

-8 Erro ao gerar o arquivo

Nota:

Esta função é utilizada somente nas impressoras fiscais térmicas.

3.15.1.4 int Bematech_FI_FormatoDadosMFD (const char * cArquivoMFD, const char * cDestino, const char * cFormato, const char * cTipoDownload, const char * cDadoInicial, const char * cDadoFinal, const char * cUsuario)

Formato Dados MFD.

Gerar os dados da MFD em arquivo no formato TXT, RTF ou MFD

```
// exemplo em C/C++
int iRetorno;
iRetorno =
Bematech_FI_FormatoDadosMFD("DOWNLOAD.MFD", "SAIDA.TXT", "2", "2",
"000001", "000010", "1");
```

Parâmetros:

cArquivoMFD Nome do arquivo MFD que contém a origem dos dados MFD, exemplo: "DOWNLOAD.MFD".

cDestino Nome do arquivo que será gerado, exemplo: "SAIDA.TXT".

cFormato Tipo do arquivo de saída desejado.

cTipoDownload Tipo do download, se é total, por data ou por COO.

cDadoInicial Data ou COO inicial, sendo a data no formato DDMMAA ou DDMMAAAA ou ainda o COO com no máximo 6 dígitos.

cDadoFinal Data ou COO final, sendo a data no formato DDMMAA ou DDMMAAAA ou ainda o COO com no máximo 6 dígitos.

cUsuario Número de ordem do proprietário do ECF, exemplo: primeiro proprietário "cUsuario = 1".

Nota:

cTipoDownload deve ser "0" para download total, "1" para download por data ou "2" para download por COO

A numeração dos COOs é diferente para cada proprietário.

Os parâmetros cDadoInicial e cDadoFinal são obrigatórios se o tipo de download for por data ou por COO.

O parâmetro cUsuario é obrigatório se o download for por COO.

Esta função é utilizada somente nas impressoras fiscais térmicas.

O arquivo MFD de origem deve ser gerado através da função Bematech_FI_DownloadMFD.

Esta função não retornará os status ACK, ST1 e ST2 da impressora, pois ela é utilizada, apenas, para gerar o download da MFD nos formatos citados acima.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

-1 Erro de execução da função

-2 Parâmetro inválido

-8 Erro ao criar ou gravar arquivo

3.16 Internas

Funções

- int [Bematech_FI_AbrePorta](#) (const char *Porta)
- int [Bematech_FI_AbrePortaSerial](#) (void)
- void [Bematech_FI_CodificaDataRFD](#) (int iValor, char *cValor)
- int [Bematech_FI_ComandoDriverRede](#) (char *cTexto)
- int [Bematech_FI_ControlePortaria40888RJ](#) (void)
- int [Bematech_FI_CriaArquivoRetorno](#) (const char *cRetorno)
- int [Bematech_FI_CriaArquivoStatus](#) (const char *cStatus)
- int [Bematech_FI_CriaCabecalho](#) (FILE *fp)
- int [Bematech_FI_DownloadMFDPorCOO](#) (const char *cArquivo, const char *cCOOInicial, const char *cCOOFinal, const char *cUsuario)
- int [Bematech_FI_DownloadMFDPorData](#) (const char *cArquivo, const char *tmpDataInicial, const char *tmpDataFinal)
- int [Bematech_FI_DownloadMFDTotal](#) (const char *cArquivo)
- int [Bematech_FI_FechaPortaSerial](#) (void)
- void [Bematech_FI_FormataCasasDecimais](#) (char *cValorBruto, int iCasasDecimais)
- void [Bematech_FI_FormataValores](#) (char *cValor, char *cValorFormatado)
- void [Bematech_FI_GetNomeArquivoCotepe](#) (char *cModeloImpressora, char *cNumeroSerie, char *cData, char *cNomeArquivo)
- int [Bematech_FI_GetTamanhoCodigoItem](#) (void)
- int [Bematech_FI_ObtemIndiceFormaPagamento](#) (const char *formaPagamento, char *indiceFormaPagamento)
- void [Bematech_FI_RetiraCaracter](#) (char *cDestino, const char *cOrigem)
- void [Bematech_FI_RetiraCaracter](#) (char *cString)
- int [Bematech_FI_ReturnaString](#) (const char *cStringParametro, int iPosicaoInicial, char cCaracter, char *cStringRetornada)
- int [Bematech_FI_VerificaDataCotepeTrataComando](#) (char *cData)
- void [EncryptRFD](#) (BYTE *buffer, DWORD size, DWORD key)

3.16.1 Documentação das funções

3.16.1.1 int Bematech_FI_AbrePorta (const char * Porta)

Abre Porta. Abrir sem usar o Arquivo INI.

Parâmetros:

Porta Porta

Retorna:

int com a informação sobre a execução do comando

3.16.1.2 int Bematech_FI_AbrePortaSerial (void)

Abre Porta Serial. Abrir e setar as configurações da porta de comunicação do micro.

Retorna:

int com a informação sobre a execução do comando

3.16.1.3 void Bematech_FI_CodificaDataRFD (int *iValor*, char * *cValor*)

Codifica Data RFD.

Codifica a data, conforme exigência do fisco para arquivos RFD.

Parâmetros:

iValor Data a ser convertida.

→ *cValor* Data convertida.

3.16.1.4 int Bematech_FI_ComandoDriverRede (char * *cTexto*)

Comando Driver Rede. Ler o comando do arquivo bemafi32.cmd e enviar para a impressora fiscal.

Parâmetros:

cTexto comando a ser enviado para o driver

Retorna:

int com a informação sobre a execução do comando

3.16.1.5 int Bematech_FI_ControlPortaria40888RJ (void)

Controle Portaria 40888RJ.

Lê os totalizadores parciais e grava no arquivo de configuração.

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

1 Ok

0 Erro de comunicação

-1 Erro de execução da função

-4 O arquivo de inicialização BemaFI32.ini não foi encontrado no diretório de sistema do Windows

-5 Erro ao abrir a porta de comunicação

-8 Erro ao criar ou gravar no arquivo retornoStatus STATUS.TXT ou RETORNO.TXT

Nota:

Esta função deve ser chamada no momento da abertura do cupom fiscal para gravar os valores atuais. Os totalizadores serão gravados na chave "TotalizadoresParciais" na seção "Info" do arquivo de configuração.

3.16.1.6 int Bematech_FI_CriaArquivoRetorno (const char * *cRetorno*)

Cria Arquivo Retorno. Criar o arquivo retorno.txt com o retorno de informação da impressora.

Parâmetros:

cRetorno retorno a ser gravado

Retorna:

int com a informação sobre a execução do comando

3.16.1.7 int Bematech_FI_CriaArquivoStatus (const char * *cStatus*)

Cria Arquivo Status. Criar o arquivo status.txt com o status de execução do comando.

Parâmetros:

cStatus status de execução do comando

Retorna:

int com a informação sobre a execução do comando

3.16.1.8 int Bematech_FI_CriaCabecalho (FILE * *fp*)

Cria Cabecalho.

Cria o cabecalho no arquivo RFD.

Parâmetros:

fp Arquivo a ser gerado o cabeçalho.

3.16.1.9 int Bematech_FI_DownloadMFDPorCOO (const char * *cArquivo*, const char * *cCOOInicial*, const char * *cCOOFinal*, const char * *cUsuario*)

Download MFD Por COO. Fazer o Download da Memória de Fita Detalhe por COO nas Impressoras MFD.

Parâmetros:

cArquivo nome do arquivo a ser gerado (Ex: "mp2000.mfd")

cCOOInicial COO inicial com até 6 dígitos

cCOOFinal COO final com até 6 dígitos

cUsuario número de ordem do proprietário do ECF

Retorna:

int com a informação sobre a execução do comando

3.16.1.10 int Bematech_FI_DownloadMFDPorData (const char * *cArquivo*, const char * *tmpDataInicial*, const char * *tmpDataFinal*)

Download MFD Por Data. Fazer o Download da Memória de Fita Detalhe das Impressoras Bematech.

Parâmetros:

cArquivo nome do arquivo a ser gerado (Ex: "mp2000.mfd")

tmpDataInicial data inicial no formato DDMMAA ou DDMMAAAA

tmpDataFinal data final no formato DDMMAA ou DDMMAAAA

Retorna:

int com a informação sobre a execução do comando

3.16.1.11 int Bematech_FI_DownloadMFDTotal (const char * *cArquivo*)

Download MFD Total. Fazer o Download total da Memória de Fita Detalhe das Impressoras MFD.

Parâmetros:

cArquivo nome do arquivo a ser gerado (Ex: "mp2000.mfd")

Retorna:

int com a informação sobre a execução do comando

3.16.1.12 int Bematech_FI_FechaPortaSerial (void)

Fecha Porta Serial. Fechar a porta de comunicação com o micro.

Retorna:

int com a informação sobre a execução do comando

3.16.1.13 void Bematech_FI_FormataCasasDecimais (char * *cValorBruto*, int *iCasasDecimais*)

Formata Casas Decimais.

Formata o valor para o número de casas decimais solicitadas pelo usuário.

Retorna:

void.

3.16.1.14 void Bematech_FI_FormataValores (char * *cValor*, char * *cValorFormatado*)

Formata Valores. Formatar um valor com a máscara "...#.#0,00".

Parâmetros:

cValor Valor a ser formatado, com no máximo 20 dígitos

→ *cValorFormatado* ponteiro para receber o valor formatado

3.16.1.15 void Bematech_FI_GetNomeArquivoCotepe (char * *cModeloImpressora*, char * *cNumeroSerie*, char * *cData*, char * *cNomeArquivo*)

Get Nome Arquivo Cotepe.

Gera o nome do arquivo cotepe, conforme determinacao do fisco.

Parâmetros:

cModeloImpressora Modelo da impressora.

cNumeroSerie Número de série da impressora.

cData Data que conterà no nome do arquivo.

→ *cNomeArquivo* Nome do arquivo Cotepe.

3.16.1.16 int Bematech_FI_GetTamanhoCodigoItem (void)

Get Tamanho Código Item.

Identifica se o código do produto na venda de item deve ter 13 ou 14 bytes.

Retorna:

int contendo a informação sobre a execução do comando.

Valores retornados:

- 0 Erro de comunicação
- >0 Tamanho do código do item (13 ou 14).

Nota:

Função destinada à compatibilização da venda de item com a impressora MP-7000 TH FI. Nas impressoras anteriores, o código do produto na venda de item possuía 13 bytes, a partir da MP-7000 TH FI este código passou para 14 bytes.

3.16.1.17 int Bematech_FI_ObtemIndiceFormaPagamento (const char * *formaPagamento*, char * *indiceFormaPagamento*)

Obtem Índice Forma Pagamento. Retornar o índice da forma de pagamento. Se a forma de pagamento não estiver cadastrada, ela será programada.

Parâmetros:

- formaPagamento* Forma de pagamento
- *indiceFormaPagamento* ponteiro para receber o índice da forma de pagamento

Retorna:

int com a informação sobre a execução do comando

Valores retornados:

- 1 Ok
- 0 Erro de comunicação
- 1 Comando não executado
- 2 Erro de parâmetros
- 4 Arquivo de inicialização não encontrado ou inválido
- 8 Erro ao criar ou gravar no arquivo texto
- 24 Forma de pagamento não programada e sem espaço para programá-la

Amendment:

Rodrigo R. Olimpio - 03/07/2001

Foi incluído o parametro "char * cIndiceFormaPagamento" e a função foi alterada de "char *" para "int". Essa alteração foi realizada para corrigir o uso do malloc. A variável estava sendo alocada mas não desalocada. Toda vez que a função era chamada o malloc alocava outra variável.

3.16.1.18 void Bematech_FI_RetiraCaracter (char * *cDestino*, const char * *cOrigem*)

Retira Caracter. Retirar o ponto "." e a "," dos valores numericos. Ex. Entrada: "1.500,00" - Saída: "150000". Retira qualquer caracter não numerico de uma string.

Parâmetros:

→ *cDestino* ponteiro para receber a string alterada
cOrigem string numérica de origem

3.16.1.19 void Bematech_FI_RetiraCaracter (char * *cString*)

Retirar Caracter. Retirar o ponto "." e a "," dos valores numericos Ex. Entrada: "1.500,00" - Saída: "150000". Retira qualquer caracter não numerico de um string.

Parâmetros:

cString Valor

3.16.1.20 int Bematech_FI_RetornaString (const char * *cStringParametro*, int *iPosicaoInicial*, char *cCaracter*, char * *cStringRetornada*)

Retorna String. Retornar parte de uma string. Essa função retorna parte de uma string armazenada em um variável ponteiro passada como parâmetro na função. Você indica a posição inicial do string e a posição final que corresponde a um caracter por exemplo ";". A função retorna ainda um valor inteiro com a posição final do string.

Parâmetros:

cStringParametro String passada como parâmetro
iPosicaoInicial Posição inicial da string a ser retornada
cCaracter Caracter para a posição final
→ *cStringRetornada* ponteiro para receber parte da string retornada

Retorna:

int com o final da string

3.16.1.21 int Bematech_FI_VerificaDataCotepeTrataComando (char * *cData*)

Verifica Data Cotepe Trata Comando.

Realiza verificações e define o dia aonde será gravado os dados no arquivo .RFD.

Parâmetros:

→ *cData* Data do arquivo .RFD.

3.16.1.22 void EncryptRFD (BYTE * *buffer*, DWORD *size*, DWORD *key*)

EncryptRFD.

Criptografa as informações para o arquivo RFD.

Parâmetros:

buffer Buffer contendo os dados a serem criptografados.

size Tamanho do buffer.

key Chave que auxiliará na criptografia.

3.17 Monitor

Para utilizar o Bematech Monitor é necessário instalar a biblioteca tanto no servidor quanto no cliente, assim como é necessário instalar o bematechmonitor no cliente e o bematechmonitorservice no servidor. Os arquivos leia-me.txt contém informações mais detalhadas sobre a instalação de cada módulo.

Além disso, a aplicação a ser utilizada também deve ser instalada no servidor.

3.17.1 Configuração da biblioteca (cliente e servidor)

Após instalar as bibliotecas, é necessário configurá-las para que trabalhem via rede e não localmente, como é o padrão. Para tal, basta configurar as seguintes variáveis:

```
<configuracoes>
<ConfigRede>0</ConfigRede>
</configuracoes>
<ModoRemoto>
<IP>10.0.1.100</IP>
<Porta>5001</Porta>
</ModoRemoto>
```

3.17.1.1 configuracoes.ConfigRede Modo de rede a ser utilizado. Preencha com 0 para modo local, 4 para modo cliente e 5 para modo servidor.

3.17.1.2 modoremoto.ip Indica o IP do servidor. Deve ser preenchido apenas no cliente. Será ignorado se presente no servidor.

3.17.1.3 modoremoto.porta Indica a porta em uso no servidor. Deve ser preenchido tanto no cliente quanto no servidor. Essa porta será utilizada para a conexão dos clientes e das aplicações e deve estar liberada por firewalls, caso existam.

3.17.2 Configuração exclusiva do servidor

Além de configurar a biblioteca no servidor, é necessário configurar o serviço bematechmonitorservice. Para tal, basta editar o arquivo /etc/bematechmonitorservice.conf e preencher a variável PORTA da seguinte maneira:

```
# Arquivo de configuração do Bematech Monitor Service.
# PORTA: porta na qual o servidor irá esperar conexões das impressoras e das aplicações.
PORTA=5001
```

Após configurado, é necessário reiniciar o serviço através de (como root):
`/etc/init.d/bematechmonitorservice restart`

3.17.3 Configuração exclusiva do cliente

Além da biblioteca, não é necessário configurar mais nada no cliente. Apenas é necessário garantir que o serviço bematechmonitor esteja rodando. O serviço pode ser iniciado através de (como root)
`/etc/init.d/bematechmonitor start`

4 Documentação do exemplo

4.1 fiscal.cpp

Exemplo, em C/C++ do uso de funções fiscais da biblioteca.

```
#include <bemafiscal/declares.h>
#include <stdio.h>

int main()
{
    int iRetorno, iCasasDecimais, iAltura, iLargura, iPosicaoCaracteres, iFonte, iMargem, iCorrecaoErros, i
    char *cCPF, *cNome, *cEndereco, *cCodigo, *cDescricao, *cAliquota, *cTipoQuantidade, *cQuantidade, *cUn
    *cAcrescimoDesconto, *cTipoAcrescimoDesconto, *cValorAcrescimoDesconto, *cValor, *cFormaPagamento, *cValor
    *cFormaOrigem, *cFormaDestino;

    // Abre Cupom MFD
    cCPF = "10.123.154-98";
    cNome = "Fulano de Tal";
    cEndereco = "Rua Sem Fim, 1000";
    iRetorno = Bematech_FI_AbreCupomMFD(cCPF, cNome, cEndereco);

    // Vende Item 1
    cCodigo = "123";
    cDescricao = "Caneta";
    cAliquota = "II";
    cTipoQuantidade = "I";
    cQuantidade = "100";
    iCasasDecimais = 2;
    cUnitario = "2,50";
    cTipoDesconto = "%";
    cDesconto = "0000";
    iRetorno = Bematech_FI_VendeItem(cCodigo, cDescricao, cAliquota, cTipoQuantidade, cQuantidade, iCasasDe

    // Vende Item 2
    cCodigo = "456";
    cDescricao = "Lapis";
    cAliquota = "NN";
    cTipoQuantidade = "I";
    cQuantidade = "100";
    iCasasDecimais = 2;
    cUnitario = "1,30";
    cTipoDesconto = "%";
    cDesconto = "0000";
    iRetorno = Bematech_FI_VendeItem(cCodigo, cDescricao, cAliquota, cTipoQuantidade, cQuantidade, iCasasDe

    // Vende Item 3
    cCodigo = "789";
    cDescricao = "Borracha";
    cAliquota = "FF";
    cTipoQuantidade = "I";
```

```
cQuantidade = "50";
iCasasDecimais = 2;
cUnitario = "2,00";
cTipoDesconto = "%";
cDesconto = "0000";
iRetorno = Bematech_FI_VendeItem(cCodigo, cDescricao, cAliquota, cTipoQuantidade, cQuantidade, iCasasDe

// Aumenta Descricao do Item
iRetorno = Bematech_FI_AumentaDescricaoItem("Produto com a descricao podendo chegar ate 200 caracteres"

// Vende Item 4
cCodigo = "258";
cAliquota = "II";
cTipoQuantidade = "I";
cQuantidade = "15";
iCasasDecimais = 2;
cUnitario = "5,00";
cTipoDesconto = "%";
cDesconto = "0000";
iRetorno = Bematech_FI_VendeItem(cCodigo, cDescricao, cAliquota, cTipoQuantidade, cQuantidade, iCasasDe

// Desconto no Item 1, por valor
cItem = "001";
cAcrescimoDesconto = "D";
cTipoAcrescimoDesconto = "$";
cValorAcrescimoDesconto = "1000";
iRetorno = Bematech_FI_AcrescimoDescontoItemMFD(cItem, cAcrescimoDesconto, cTipoAcrescimoDesconto, cVal

// Acrescimo no Item 3, por percentual
cItem = "003";
cAcrescimoDesconto = "A";
cTipoAcrescimoDesconto = "%";
cValorAcrescimoDesconto = "2500";
iRetorno = Bematech_FI_AcrescimoDescontoItemMFD(cItem, cAcrescimoDesconto, cTipoAcrescimoDesconto, cVal

// Cancelamento do Desconto no Item 1
cAcrescimoDesconto = "D";
cItem = "001";
iRetorno = Bematech_FI_CancelaAcrescimoDescontoItemMFD(cAcrescimoDesconto, cItem);

// Cancelamento da Venda do Item 4 (Item Anterior)
iRetorno = Bematech_FI_CancelaItemAnterior();

// Cancelamento da Venda do Item 1 (Item Generico)
cItem = "001";
iRetorno = Bematech_FI_CancelaItemGenerico(cItem);

// Subtotaliza Cupom MFD
iRetorno = Bematech_FI_SubTotalizaCupomMFD();

// Desconto Subtotal MFD
cAcrescimoDesconto = "D";
cTipoAcrescimoDesconto = "%";
cValor = "10,00";
iRetorno = Bematech_FI_AcrescimoDescontoSubtotalMFD(cAcrescimoDesconto, cTipoAcrescimoDesconto, cValor)

// Cancela Desconto Subtotal MFD
cAcrescimoDesconto = "D";
iRetorno = Bematech_FI_CancelaAcrescimoDescontoSubtotalMFD(cAcrescimoDesconto);

// Totaliza Cupom MFD
iRetorno = Bematech_FI_TotalizaCupomMFD();

// Efetua Forma de Pagamento
cFormaPagamento = "Cheque";
cValorPago = "300,00";
iRetorno = Bematech_FI_EfetuaFormaPagamento(cFormaPagamento, cValorPago);
```

```

// Termina Fechamento do Cupom
cMensagem = "Obrigado, volte sempre!";
cTipoCodigo = "EAN13";
cCodigo = "123456789012";
iAltura = 100;
iLargura = 1;
iPosicaoCaracteres = 3;
iFonte = 0;
iMargem = 5;
iCorrecaoErros = 4;
iColunas = 5;
iRetorno = Bematech_FI_TerminaFechamentoCupomCodigoBarrasMFD(cMensagem, cTipoCodigo, cCodigo, iAltura,
iMargem, iCorrecaoErros, iColunas);

// Cupom Adicional MFD
iRetorno = Bematech_FI_CupomAdicionalMFD();

// Estorno Forma de Pagamento
cFormaOrigem = "Cheque";
cFormaDestino = "Dinheiro";
cValor = "300,00";
iRetorno = Bematech_FI_EstornoFormasPagamento(cFormaOrigem, cFormaDestino, cValor);
}

```

4.2 gerencial.cpp

Exemplo, em C/C++ do uso de funções gerenciais da biblioteca.

```

#include <bemafiscal/declares.h>
#include <stdio.h>

int main()
{
    int iRetorno;
    char *cIndiceRelatorio, *cTexto;

    // Abre Relatorio Gerencial
    cIndiceRelatorio = "01";
    iRetorno = Bematech_FI_AbreRelatorioGerencialMFD(cIndiceRelatorio);

    // Usa Relatorio Gerencial
    cTexto = "Espaco destinado ao relatorio gerencial com ate 618 caracteres";
    iRetorno = Bematech_FI_UsaRelatorioGerencialMFD(cTexto);

    // Fecha Relatorio Gerencial
    iRetorno = Bematech_FI_FechaRelatorioGerencial();
}

```

4.3 nao_fiscal.cpp

Exemplo, em C/C++ do uso de funções não fiscais da biblioteca.

```

#include <bemafiscal/declares.h>
#include <stdio.h>
#include <string.h>

int main()
{
    int iRetorno;
    char *cCPF, *cNome, *cEndereco, *cIndiceTotalizador, *cValorRecebimento, *cNumeroItem, *cAcrescimoDesco
*cValorAcrescimoDesconto, *cFormaPagamento, *cValorPago, *cMensagem, *cValor, cNumeroCupom[7];

```

```
// Efetua Suprimento
cValor = "300,00";
cFormaPagamento = "Dinheiro";
iRetorno = Bematech_FI_Suprimento(cValor, cFormaPagamento);

// Abre Recebimento Nao Fiscal MFD
cCPF = "10.123.154-98";
cNome = "Fulano de Tal";
cEndereco = "Rua Sem Fim, 1000";
iRetorno = Bematech_FI_AbreRecebimentoNaoFiscalMFD(cCPF, cNome, cEndereco);

// Efetua Recebimento Nao Fiscal 1
cIndiceTotalizador = "01";
cValorRecebimento = "10,00";
iRetorno = Bematech_FI_EfetuaRecebimentoNaoFiscalMFD(cIndiceTotalizador, cValorRecebimento);

// Efetua Recebimento Nao Fiscal 2
cIndiceTotalizador = "03";
cValorRecebimento = "25,00";
iRetorno = Bematech_FI_EfetuaRecebimentoNaoFiscalMFD(cIndiceTotalizador, cValorRecebimento);

// Efetua Recebimento Nao Fiscal 3
cIndiceTotalizador = "02";
cValorRecebimento = "100,00";
iRetorno = Bematech_FI_EfetuaRecebimentoNaoFiscalMFD(cIndiceTotalizador, cValorRecebimento);

// Efetua Recebimento Nao Fiscal 4
cIndiceTotalizador = "05";
cValorRecebimento = "15,00";
iRetorno = Bematech_FI_EfetuaRecebimentoNaoFiscalMFD(cIndiceTotalizador, cValorRecebimento);

// Cancela Item Nao Fiscal MFD
cNumeroItem = "002";
iRetorno = Bematech_FI_CancelaItemNaoFiscalMFD(cNumeroItem);

// Subtotaliza Recebimento MFD
iRetorno = Bematech_FI_SubTotalizaRecebimentoMFD();

// Efetua Desconto no Subtotal do Recebimento MFD por valor
cAcrescimoDesconto = "D";
cTipoAcrescimoDesconto = "$";
cValorAcrescimoDesconto = "15,00";
iRetorno = Bematech_FI_AcrescimoDescontoSubtotalRecebimentoMFD(cAcrescimoDesconto, cTipoAcrescimoDesconto);

// Cancela Desconto no Subtotal do Recebimento MFD
cAcrescimoDesconto = "D";
iRetorno = Bematech_FI_CancelaAcrescimoDescontoSubtotalRecebimentoMFD(cAcrescimoDesconto);

// Totaliza Recebimento MFD
iRetorno = Bematech_FI_TotalizaRecebimentoMFD();

// Efetua Forma de Pagamento
cFormaPagamento = "Cartao";
cValorPago = "150,00";
iRetorno = Bematech_FI_EfetuaFormaPagamento(cFormaPagamento, cValorPago);

// Termina Recebimento MFD
cMensagem = "Obrigado, volte sempre!";
iRetorno = Bematech_FI_FechaRecebimentoNaoFiscalMFD(cMensagem);

// Abre Comprovante Nao Fiscal Vinculado MFD
cFormaPagamento = "Cartao";
cValorPago = "150,00";
memset(cNumeroCupom, '\0', sizeof(cNumeroCupom));
iRetorno = Bematech_FI_NumeroCupom((char*) cNumeroCupom);
cCPF = "10.123.154-98";
```

```
cNome = "Fulano de Tal";
cEndereco = "Rua Sem Fim, 1000";
iRetorno = Bematech_FI_AbreComprovanteNaoFiscalVinculadoMFD(cFormaPagamento, cValorPago, cNumeroCupom,

// Usa Comprovante Nao Fiscal Vinculado MFD
cMensagem = "Digite o texto a ser impresso aqui !!!";
iRetorno = Bematech_FI_UsaComprovanteNaoFiscalVinculado(cMensagem);

// Fecha Comprovante Nao Fiscal Vinculado MFD
iRetorno = Bematech_FI_FechaComprovanteNaoFiscalVinculado();

// Reimprime o Comprovante Nao Fiscal Vinculado MFD
iRetorno = Bematech_FI_ReimpressaoNaoFiscalVinculadoMFD();

// Segunda Via do Comprovante Nao Fiscal Vinculado MFD
iRetorno = Bematech_FI_SegundaViaNaoFiscalVinculadoMFD();

// Estorno Comprovante Nao Fiscal Vinculado MFD
cCPF = "10.123.154-98";
cNome = "Fulano de Tal";
cEndereco = "Rua Sem Fim, 1000";
iRetorno = Bematech_FI_EstornoNaoFiscalVinculadoMFD(cCPF, cNome, cEndereco);

// Efetua Sangria
cValor = "200,00";
iRetorno = Bematech_FI_Sangria(cValor);
}
```