

Impressoras Fiscais MECAF

DLL ECF32M

Especificação de Uso



Revisão 1.12

As informações contidas neste manual são de caráter técnico-informativo e são de propriedade da Mecaf Eletrônica Ltda., não podendo ser reproduzidas e/ou alteradas sem autorização por escrito da mesma.

A Mecaf reserva-se o direito de fazer as alterações concernentes ao manual e ao produto sem a necessidade de uma informação, ou aviso, prévio.

Criação e produção:
EDS – Engenharia de Desenvolvimento de Software

Revisado por:
AT – Suporte Técnico em Maio/2004

Diebold Procomp - Divisão de Mecânica Fina
Estrada da Aldeinha, 618 – Alphaville
CEP 06465-100 – Barueri – SP
Tel.: (11) 4191-2581
Fax: (11) 4195-1532
Site: www.mecaf.com.br
e-mail: suporte@mecaf.com.br

Índice Eletrônico

Índice Eletrônico	3
Introdução	4
Outras ferramentas de desenvolvimento	4
Programas Exemplo desta DLL	4
Instalação da DLL	4
Arquivo de Configuração	5
Arquivo de Log	6
Formato das Funções	7
Sintaxe da Mensagem de Retorno	7
Códigos de Retorno	8
Programação	11
Ferramentas e aplicativos úteis para a fase de desenvolvimento e testes	12
OpenCif()	13
CloseCif()	13
AbreCupomFiscal()	14
AbreCupomFiscalCPF_CNPJ()	14
VendaItemStr()	15
CancelamentoItem()	20
DescontoItemPorcentagem()	21
DescontoItemValor()	22
DescontoItem()	23
TotCupomSemDescAcres()	24
TotCupomAcresValor()	24
TotCupomAcresPorcentagem()	25
TotCupomDescValor()	26
TotCupomDescPorcentagem()	27
Pagamento()	28
PagamentoComTexto()	30
Troco()	31
FechaCupomFiscal()	32
CancelaCupomFiscal()	33
LeituraX()	33
LeituraXComRelGer()	34
ReducaoZ()	34
ReducaoZComRelGer()	35
LeMemFiscalData()	36
LeMemFiscaReducao()	36
ModoChequeValidacao()	37
ModoChequeValidacaoAvancoNV()	38
ImprimeCheque()	39
ImprimeValidacao()	43
CancelaChequeValidacao()	43
ProgramaLegenda()	44
AbreCupomVinculado()	45
AbreCupomNaoVinculado()	46
OperRegNaoVinculado()	46
ImprimeLinhaNaoFiscal()	48
ImprimeLinhaNaoFiscalTexto()	49
EncerraCupomNaoFiscal()	50
CancelaCupomNaoFiscal()	50
AcionarGaveta()	51
ProgramaHorarioVerao()	51
ProgramaHorarioVeraoStr()	52
ImprimeTotalizadores()	53
TransTabAliquotas()	53
TransTabAliquotasCasas()	54
TransTotCont()	55
TransStatus()	55

TransDataHora()	58
TransMemFiscalData()	58
TransMemFiscalDataArq()	59
TransMemFiscaRed()	60
TransMemFiscaRedArq()	61
EcfPar()	61
ProgLinhaAdicional()	66
AjusteHora()	67
EcfID()	67
ProgAliquotas()	68
ProgAliquotasICMS_ISS()	69
ProgSimbolo()	70
ProgArredondamento()	72
EsperaResposta()	73
ImprimeNaoFiscal()	76
Problemas na comunicação com a Impressora Fiscal	77
Configuração da Impressora Fiscal	78
Como obter o Número do Cupom Fiscal?	78
Como obter o Número do Caixa do ECF?	78
Como obter o Número de Série do ECF?	78
Como obter o Total do Cupom em andamento?	78
Como obter o Total a pagar do Cupom em andamento?	79
Tabela de caracteres ABICOMP	79
Histórico das Revisões	80
Observações Gerais	82

Introdução

Este documento descreve a forma de programação das Impressoras Fiscais MECAF, atendendo às novas exigências do Convênio 156/94.

As funções descritas a seguir foram elaboradas no formato DLL (Dynamic Link Library) para uso no ambiente de 32 bits (Windows 95, 98, Me, NT, 2000 e XP) permitindo o seu uso em aplicativos desenvolvidos em Visual Basic, Delphi, Visual C++ e outros.

Outras ferramentas de desenvolvimento

Consulte o nosso site para obter informações sobre outras ferramentas de desenvolvimento para as Impressoras Fiscais Mecaf, como por exemplo:

- 1) O Driver desenvolvido para DOS chamado DRVECF.SYS que pode ser acessado por Linguagens DOS como por exemplo Clipper e Cobol.
- 2) A interface de comunicação por troca de arquivos MEDRVECF.EXE que pode ser acessada por qualquer linguagem que tenha dificuldade em fazer uso de DLLs e rode em ambiente Windows 32bits.

Programas Exemplo desta DLL

Estão disponíveis em nosso site programas exemplo desta DLL ECF32M acessando as Impressoras Fiscais Mecaf em:

- 1) Visual Basic
- 2) Delphi

Instalação da DLL

A DLL ECF32M deve ser instalada (copiada) em uma pasta específica, conforme a plataforma utilizada (lembre-se que esta DLL deve estar em uma única pasta):

Arquivo: [ECF32M.DLL](#)

DLL ECF32M
Especificação de Uso

Plataforma (32 bits)	Diretório destino
Windows 95, 98, 98 SE e Me	\Windows\System
Windows NT Workstation, 2000 Professional	\WINNT\System32
Windows XP	\Windows\System32
Para todas versões do Windows	Mesma pasta do seu aplicativo.

Esta biblioteca faz uso de um arquivo de configuração, denominado **CIF.INI**; este arquivo também possui diretório destino específico:

Arquivo: CIF.INI	
Plataforma (32 bits)	Diretório destino
Windows 95, 98, 98 SE e Me	\Windows
Windows NT Workstation e 2000 Professional	\WINNT
Windows XP	\Windows

O arquivo de configuração será discutido no tópico seguinte.

ATENÇÃO!

Instale, ou copie, estes arquivos respeitando rigorosamente os diretórios destinos conforme a plataforma utilizada. A não observação destas definições implicará em mal funcionamento e resultados indesejáveis.

Arquivo de Configuração

A DLL ECF32M utiliza um arquivo de configuração denominado "**CIF.INI**".

O arquivo de configuração CIF.INI é um arquivo texto, no padrão de inicialização do Windows. Contendo:

Parâmetros	Descrição																				
[PORT]	Nome da seção.																				
COM	Determina a porta serial que será utilizada. São aceitos os parâmetros 1, 2 ,3 ou 4 para habilitar respectivamente as portas seriais COM1, COM2, COM3 e COM4.																				
<table><tr><th>Porta Serial</th><th>IRQ</th><th>Endereço I/O</th><th>No CIF.INI</th></tr><tr><td>COM1 (default)</td><td>4</td><td>03F8</td><td>COM=1</td></tr><tr><td>COM2</td><td>3</td><td>02F8</td><td>COM=2</td></tr><tr><td>COM3</td><td>4</td><td>03F0</td><td>COM=3</td></tr><tr><td>COM4</td><td>3</td><td>02F0</td><td>COM=4</td></tr></table>		Porta Serial	IRQ	Endereço I/O	No CIF.INI	COM1 (default)	4	03F8	COM=1	COM2	3	02F8	COM=2	COM3	4	03F0	COM=3	COM4	3	02F0	COM=4
Porta Serial	IRQ	Endereço I/O	No CIF.INI																		
COM1 (default)	4	03F8	COM=1																		
COM2	3	02F8	COM=2																		
COM3	4	03F0	COM=3																		
COM4	3	02F0	COM=4																		
DEPURA	Deve ser usado em ambiente de desenvolvimento para ajudar na depuração do seu aplicativo. Determina a geração de um arquivo de LOG, com todas as funções e parâmetros que o seu aplicativo está usando. As informações presentes no arquivo de LOG serão geradas conforme o nível de depuração determinado Básico, Médio ou Avançado. Recomendamos usar o nível Avançado. Será criado um arquivo de LOG por dia com o nome <AAAAMMDD.MLG>. OBS: Há uma descrição mais detalhada no tópico “ Arquivo de LOG ”.																				
<table><tr><th>Descrição</th><th>No CIF.INI</th></tr><tr><td>O arquivo de Log não será gerado (default).</td><td>Depura=0</td></tr><tr><td>Nível <u>Básico</u>.</td><td>Depura=1</td></tr><tr><td>Nível <u>Médio</u>.</td><td>Depura=2</td></tr><tr><td>Nível <u>Avançado</u></td><td>Depura=3</td></tr></table>		Descrição	No CIF.INI	O arquivo de Log não será gerado (default).	Depura=0	Nível <u>Básico</u> .	Depura=1	Nível <u>Médio</u> .	Depura=2	Nível <u>Avançado</u>	Depura=3										
Descrição	No CIF.INI																				
O arquivo de Log não será gerado (default).	Depura=0																				
Nível <u>Básico</u> .	Depura=1																				
Nível <u>Médio</u> .	Depura=2																				
Nível <u>Avançado</u>	Depura=3																				

ARQUIVO	Determina a geração de um arquivo de resposta do ecf. Ao enviar um comando ao ecf, a resposta será gravada pela dll em um arquivo denominado ECF32M_Retorno.TXT . Este arquivo só será gerado, caso a função EsperaResposta() seja utilizada. As demais funções não irão gerar este arquivo.						
	<table> <tr> <th>Descrição</th><th>No CIF.INI</th></tr> <tr> <td>NÃO gera o arquivo de resposta (default)</td><td>ARQUIVO=0</td></tr> <tr> <td>Gera o arquivo de resposta</td><td>ARQUIVO=1</td></tr> </table>	Descrição	No CIF.INI	NÃO gera o arquivo de resposta (default)	ARQUIVO=0	Gera o arquivo de resposta	ARQUIVO=1
Descrição	No CIF.INI						
NÃO gera o arquivo de resposta (default)	ARQUIVO=0						
Gera o arquivo de resposta	ARQUIVO=1						
'TIMEOUT	Determina o valor do timeout a ser adotado pela dll durante o uso da função EsperaResposta() . Este parâmetro deverá ser configurado entre 3 e 15 segundos. Caso o valor seja inferior a 3 segundos, o timeout será configurado com o valor de 3 segundos; caso o valor seja superior a 15 segundos, o timeout será configurado o com valor de 15 segundos. O valor default é 3 segundos.						
	<table> <tr> <th>Descrição</th><th>No CIF.INI</th></tr> <tr> <td>Timeout da função EsperaResposta().</td><td>TIMEOUT=5</td></tr> </table>	Descrição	No CIF.INI	Timeout da função EsperaResposta().	TIMEOUT=5		
Descrição	No CIF.INI						
Timeout da função EsperaResposta().	TIMEOUT=5						
MSG_SYS	Determina se a dll irá tratar as mensagens do sistema operacional. Estas mensagens são utilizadas e consumidas pelo sistema operacional. A dll com o objetivo de não causar nenhum processo de "blocagem" nos demais aplicativos, auxilia o sistema operacional na tarefa de processar suas mensagens.						
	<table> <tr> <th>Descrição</th><th>No CIF.INI</th></tr> <tr> <td>A DLL não ficará bloqueada (default)</td><td>MSG_SYS=0</td></tr> <tr> <td>A DLL ficará bloqueada</td><td>MSG_SYS=1</td></tr> </table>	Descrição	No CIF.INI	A DLL não ficará bloqueada (default)	MSG_SYS=0	A DLL ficará bloqueada	MSG_SYS=1
Descrição	No CIF.INI						
A DLL não ficará bloqueada (default)	MSG_SYS=0						
A DLL ficará bloqueada	MSG_SYS=1						

Exemplo: Se a impressora estiver conectada à porta COM1, o arquivo CIF.INI terá a seguinte configuração:

```
[PORT]
COM=1
DEPURA=0
```

ATENÇÃO!

- **Ao editar o arquivo CIF.INI não coloque espaços entre o parâmetro e o sinal de igual, e o sinal de igual e o valor do parâmetro; isto provocará um erro durante a leitura do parâmetro em questão.**
- **Evite o uso do parâmetro "MSG_SYS"; este deverá ser utilizado somente nos casos em que o desenvolvedor seja experiente e tenha pleno domínio do processo de controle das mensagens utilizadas pelo sistema operacional.**

[**Voltar ao Índice**](#)

Arquivo de Log

O objetivo do arquivo de log é fornecer uma ferramenta de auxílio ao desenvolvedor da aplicação que fará uso da dll. As funções, parâmetros e respostas trocadas entre o aplicativo e o ECF ficam registradas em um arquivo texto.

Este arquivo possui os comandos e funções executadas pela aplicação, ao estabelecer comunicação com o ECF. Através deste arquivo é possível observar quais foram os comandos enviados ao ECF, e quais foram as respostas enviadas pelo ECF. Conforme o nível de

DLL ECF32M

Especificação de Uso

depuração (valor do parâmetro "**Depura**") é possível saber quais foram as funções chamadas pela aplicação durante o processo de comunicação com o ECF.

O arquivo de Log será gerado no diretório da aplicação, ou diretório corrente da unidade em que a aplicação está sendo executada. Este arquivo é aberto e fechado a cada 10 segundos.

O nome do arquivo de log é formado da seguinte maneira:

<AAAAMMDD>.MLG

Onde:

- AAAA – Representa o ano corrente.
- MM – O mês corrente e
- DD – O dia.

Por exemplo:

O arquivo <20030812.MLG> foi gerado em 12 de Agosto de 2003.

Formato das Funções

A maioria das funções da DLL ECF32M, tem o seguinte formato:

int NOME_DA_FUNCAO(Par_1 ,Par_2 ,... ,Par_N);

Sintaxe da Mensagem de Retorno

O formato geral das mensagens de retorno obtida através da função **EsperaResposta()** é uma string conforme ilustrado abaixo:

Tipo	Nseq	Cod	Mensagem
------	------	-----	----------

Onde:

Tipo	caractere +, - ou "S" indicando:	
	+	Comando executado com sucesso .
	-	Comando não executado, ou executado com erro .
	S	Resposta de Status
Nseq	4 caracteres numéricos no padrão ASCII, formando um contador incrementado de uma unidade a cada comando recebido e processado pelo ECF.	
Cod	2 caracteres numéricos no padrão ASCII, indicando o código de retorno do comando executado. Pode indicar Sucesso ou o Erro ocorrido.	
Mensagem	String alfanumérica no padrão ASCII de tamanho variável contendo a resposta resumida do comando enviado.	

Exemplo:

- Mensagem de retorno de comando executado com Sucesso:

+	0001	00	COMANDO OK
---	------	----	------------

- Mensagem de retorno de comando de Transmissão de Data e Hora:

+	0002	00	01/01/01-12:00:00
---	------	----	-------------------

DLL ECF32M

Especificação de Uso

- Mensagem de Erro para um Comando Inválido:

-	0001	28	REDUCAO Z PENDENTE
---	------	----	--------------------

- Exemplo de retorno da função ECFPar("41"), para obter o número do COO:

+	0002	001200
---	------	--------

Onde "001200" é o número do COO.

Nota:

Esta sintaxe não é válida para a função TransStatus(); esta função possui outro formato para o retorno.

[Voltar ao Índice](#)

Códigos de Retorno

O retorno das funções indicam se o comando foi enviado ao ECF com sucesso.

Todas as funções retornam **CIF_OK** sempre que o comando foi enviado com sucesso ao ECF, significando que o mesmo reconheceu o comando.

Após enviar com Sucesso um comando, **é necessário** que o aplicativo obtenha a resposta do ECF através da função **EsperaResposta()**, que é uma função específica para aguardar o retorno do ECF. O objetivo é verificar se o comando foi executado corretamente, ou o motivo pelo qual não foi possível realizar a operação.

Quando o retorno da função EsperaResposta() for CIF_OK, CIF_OK_PPAPPEL, CIF_OK_CANCCUP ou CIF_OK_CUPNF, indica que o comando foi executado com sucesso. Se o retorno desta função for uma constante com valor negativo, indica que houve algum erro.

Os códigos de retorno das funções estão descritos abaixo:

Códigos de Sucesso gerados pela DLL ECF32M e ECF (0 a 3)		
Constante	Valor	Descrição
CIF_OK	0	Função executada com sucesso.
CIF_OK_PPAPPEL	1	Função executada com sucesso. OBS: Detectado pouco papel.
CIF_OK_CANCCUP	2	Função executada com sucesso. Mas, OBS: Cancelando Cupom.
CIF_OK_CUPNF	3	Função executada com sucesso. OBS: Abrindo Cupom Não com Relatório Gerencial.

Códigos de Erros gerados pela DLL ECF32M (-99 a -83)		
Constante	Valor	Descrição
CIF_TIMEOUT	-99	Estourou o tempo de espera da resposta do ECF ao comando enviado.
CIF_OVERFLOW	-98	Buffer Overflow. Tamanho da mensagem enviada pelo ECF é maior que o buffer fornecido pela aplicação.
CIF_SEMRETORNO	-97	Ainda não obteve retorno.
CIF_ERR_TABERTA	-96	Detectado tampa aberta.
CIF_ERR_MECANICO	-95	Erro mecânico.
CIF_ERR_IRRECUPERAVEL	-94	Erro irreversível.
CIF_CMDEXECUCAO	-93	Indica que o comando está em execução.
CIF_ERR_PPAPPEL	-92	Detectado pouco papel.
CIF_ERR_TEMP	-91	Temperatura da cabeça alta.

Códigos de Erros gerados pela DLL ECF32M (-99 a -83)		
Constante	Valor	Descrição
CIF_ERR_READSER	-90	Falha na leitura da serial.
CIF_ERR_ANSWER	-89	Retorno não reconhecido.
CIF_ERR_SYS	-88	Falha na alocação de recursos do Windows.
CIF_ERR_SERIAL	-87	Falha na abertura, ou escrita, do dispositivo serial.
CIF_ERR_CONFIG	-86	Erro no arquivo de configuração CIF.INI
CIF_EMEXECUÇÃO	-85	Erro de reconhecimento do comando por parte do ECF.
CIF_ERR	-84	Falha geral na execução da DLL.
CIF_ERR_FIMPAPEL	-83	Fim de Papel durante comando em execução. Este código de erro foi incorporado somente à função EsperaResposta() na ocorrência de Fim de Papel durante a execução de um comando.

Códigos de Erros gerados pelo ECF (-1 a -64):	
Código	Descrição
-1	O cabeçalho contém caracteres inválidos.
-2	Comando inexistente.
-3	Valor não numérico em campo numérico. Um ou mais parâmetros estão fora do formato indicado. Verifique no arquivo de Log desta DLL: 1) O tamanho de cada parâmetro; 2) O tipo de cada parâmetro (numérico ou alfanumérico).
-4	Valor fora da faixa entre 20H E 7FH. Nessa função executada, a impressora imprime somente os caracteres destacados na tabela ABICOMP no final deste manual.
-5	Campo deve iniciar com '@', '&' ou '%'.
-6	Troco já realizado.
-7	O intervalo é inconsistente. O primeiro valor deve ser menor que o segundo valor (no caso de datas, valores anteriores a 01/01/1995 serão consideradas como pertencentes ao intervalo de anos 2000 a 2094.
-9	A string "TOTAL" não é aceita.
-10	A sintaxe do comando está incorreta. Um ou mais parâmetros estão fora do formato indicado. Verifique no arquivo de Log desta DLL: 1) O tamanho de cada parâmetro; 2) O tipo de cada parâmetro (numérico ou alfanumérico).
-11	Excedeu o nº máximo de linhas permitidas pelo comando.
-12	O terminador enviado não está respeitando o protocolo de comunicação. Verifique se a configuração serial da Impressora Fiscal está conforme o indicado neste manual.
-13	O checksum enviado está incorreto. Verifique se a configuração serial da Impressora Fiscal está conforme o indicado neste manual, principalmente o protocolo deve estar <STX ETX>. Atenção! Se estiver ativado o protocolo <STX ETX BCC>, é necessário desativar o <BCC>
-15	A situação tributária deve iniciar com 'T', 'F', 'I' ou 'N'. A situação tributária deve seguir o formato abaixo: "Tnn": ICMS ou ISS, onde "nn" varia entre 00 e 15, indicando a sequência das tributações cadastradas na sua Impressora Fiscal.

Códigos de Erros gerados pelo ECF (-1 a -64):	
Código	Descrição
	<p>"F00": Substituição Tributária;</p> <p>"N00": Não Tributado;</p> <p>"I00": Isento.</p> <p>OBS: Ao se programar uma nova alíquota indica-se o tipo de tributação (ICMS ou ISS).</p>
-16	Data inválida.
-17	Hora inválida.
-18	Alíquota não programada ou fora do intervalo.
-19	O campo de sinal está incorreto.
-20	Comando só poder ser aceito em Intervenção Fiscal. Para executar esse comando, é necessário colocar a Impressora Fiscal em Intervenção Técnica.
-21	Comando só pode ser aceito em modo Normal. Sua Impressora Fiscal está em Intervenção Técnica. Para executar esse comando, é necessário colocá-la em Modo Normal de operação.
-22	Necessário abrir cupom fiscal. Sua Impressora Fiscal não está em operação fiscal. Esse comando necessita que a Impressora Fiscal já esteja em um Cupom Fiscal.
-23	Comando não pode ser aceito durante cupom fiscal. Sua Impressora Fiscal está em uma operação fiscal. Esse comando necessita que a Impressora Fiscal não esteja em um Cupom Fiscal.
-24	Necessário abrir cupom não fiscal. Sua Impressora Fiscal não está em operação não fiscal. Esse comando necessita que a Impressora Fiscal já esteja em um Cupom Não Fiscal.
-25	Comando não pode ser aceito durante cupom não fiscal. Sua Impressora Fiscal está em uma operação não fiscal. Esse comando necessita que a Impressora Fiscal não esteja em um Cupom Não Fiscal.
-26	O relógio já está em horário de verão.
-27	O relógio não está em horário de verão.
-28	Necessário realizar Redução Z.
-29	Fechamento do dia (Redução Z) já executado.
-30	Necessário programar legenda.
-31	Item inexistente ou já cancelado.
-32	O cupom anterior não pode ser cancelado.
-33	Detectado falta de papel.
-36	Necessário programar os dados do estabelecimento.
-37	Necessário realizar Intervenção Fiscal.
-38	A memória fiscal não permite mais realizar vendas. Só é possível executar as operações de Leitura X ou Leitura da Memória Fiscal.
-39	Problemas com a memória NOVRAM. Será necessário realizar uma Intervenção Técnica.
-40	Necessário programar a data do relógio.
-41	Número máximo de itens por cupom ultrapassado (Nº máximo = 450).
-42	Já foi realizado o ajuste de hora diário.
-43	Comando válido ainda em execução.

Códigos de Erros gerados pelo ECF (-1 a -64):	
Código	Descrição
-44	Esta em estado de impressão de cheque.
-45	Não está em estado de impressão de cheque.
-46	Necessário inserir o cheque.
-47	Necessário inserir nova bobina.
-48	Necessário executar uma Leitura X.
-49	Detectado algum problema na impressora (paper jam, sobre tensão, etc).
-50	Cupom já foi totalizado.
-51	Necessário totalizar cupom antes de fechar.
-52	Necessário finalizar cupom com comando correto.
-53	Ocorreu erro de gravação na memória fiscal.
-54	Excedeu número máximo de estabelecimentos.
-55	Memória Fiscal não inicializada.
-56	Ultrapassou valor do pagamento.
-57	Registrador não programado ou troco já realizado.
-58	Falta completar valor do pagamento.
-59	Campo somente de caracteres não numéricos (Alfabéticos).
-60	Excedeu campo máximo de caracteres.
-61	Troco não realizado.
-62	Comando desabilitado.
-63	Excedeu tempo
-64	Erro de Leitura de CMC-7 (somente 2 estações com CMC-7)

[**Voltar ao Índice**](#)

Programação

A programação da impressora fiscal é feita através de chamadas às funções da DLL ECF32M. Para descrever os parâmetros de entrada e saída das funções adotou-se a seguinte convenção:

Convenção utilizada

n	Caractere numérico em ASCII, onde: `0' (30h) =< n <= `9' (39h)
s	Caractere alfanumérico em ASCII, onde: 20h =< s <= 7Fh.
c	Caractere alfabético em ASCII, onde: (20h =< c <= 2Fh) e (3Ah =< c <= 7Fh)
b	Byte , onde : 00h =< b <= FFh.
g	Caractere alfanumérico de sinal em ASCII, onde: g= '+' ou g='-'
d	Caractere numérico de dia em ASCII, onde: `01' < dd <= `31'.
m	Caractere numérico de mês em ASCII, onde: `01' =< mm <= `12'
a	Caractere Numérico de ano em ASCII, onde: `00' =< aa <= `99'.
H	Caractere numérico de Hora em ASCII, onde: `00' =< HH <= `23'.
M	Caractere Numérico de Minuto em ASCII, onde: `00' =< MM <= `59'.
S	Caractere Numérico de Segundo em ASCII, onde: `00' =< SS <= `59'.

Nota:

O tamanho de cada parâmetro de envio, ou retorno, será indicado entre chaves.

Exemplo:

- | | |
|-------------|--|
| (15n) | Indica 15 caracteres do tipo numérico. |
| (2n(00~15)) | Indica 2 caracteres numéricos, porém com limites especificados entre 00 e 15. |
| (16/18n) | Indica 16 caracteres do tipo numérico.
No ECF de uma estação 48 colunas, são aceitos, opcionalmente, 18 caracteres numéricos. |

As funções da DLL são descritas a seguir.

[Voltar ao Índice](#)

Ferramentas e aplicativos úteis para a fase de desenvolvimento e testes

- 1) Utilize os programas exemplo em Visual Basic ou Delphi disponíveis em nosso site. Para fazer download acesse:
 - www.mecaf.com.br
 - Seção Download
 - Escolha "Impressora Fiscal IF113I"
 - Escolha o produto IF113I
 - Escolha o programa exemplo em Visual Basic ou Delphi.
- 2) Utilize o arquivo de Log desta DLL. Através do arquivo de Log, é possível analisar as funções e os parâmetros enviados para o ECF. Faça uso dessa ferramenta! Para ativar a gravação de Log, consulte a sessão "Arquivo de Log" nesse manual.
- 3) Se precisar fazer alguns testes da comunicação serial, recomendamos que use um utilitário disponível em nosso site chamado <MecafECF.exe>. Para fazer download acesse:
 - www.mecaf.com.br
 - Seção Download
 - Escolha "Impressora Fiscal IF113I"
 - Escolha o produto IF113I
 - Escolha MecafECF.EXE

Conhecendo o MecafECF.exe:

- Este utilitário não necessita de qualquer Driver ou DLL para se comunicar com o ECF;
- Deve ser usado em uma janela Prompt do DOS;
- A porta de comunicação default é a COM1;
- Para mudar a porta de comunicação para COM2, digite: MecafEcf P=2 <ENTER>

Testando a Transmissão de comandos ao ECF:

- Ligue o ECF, e conecte o cabo de comunicação padrão RS232.
- Para testar a transmissão de comandos ao ECF, solicite a emissão de uma Leitura X através do menu "Operações Fiscais"
 - Se a Leitura X não foi impressa, o sinal de transmissão está com problema;
 - Se a Leitura X foi impressa, o sinal de transmissão está Ok;

Testando a recepção das respostas do ECF:

- Ligue o ECF, e conecte o cabo de comunicação padrão RS232.
- Para testar a recepção de respostas do ECF, use a última opção do Menu: "Diagnóstico do ECF"
 - Se ocorrer o erro de Timeout, significa que pode haver algum problema no sinal de recepção da resposta do ECF.
 - Se a tela de diagnóstico de ECF, foi montada sem qualquer erro, o sinal de recepção está Ok;

Funções de Inicialização e Término de uso do ECF

Função **OpenCif()**

Função:	OpenCif
Descrição:	Esta função é responsável pela inicialização da dll, habilita a porta serial e outros processos internos e deve , necessariamente, ser a primeira função a ser utilizada pela aplicação.
Sintaxe C:	int FAR PASCAL OpenCif (void);
Sintaxe VB:	Declare Function OpenCif Lib "ECF32M.DLL" () As Long
Sintaxe Delphi:	Function OpenCif(): Integer; stdcall; external 'ECF32M.dll';
Parâmetros:	Nenhum.
Retorno:	CIF_OK CIF_IRRECUPERAVEL CIF_ERR_SERIAL CIF_ERR_SYS CIF_ERR_CONFIG CIF_ERR_TEMP CIF_ERR_PPAPEL CIF_ERR_MECANICO CIF_ERR_TABERTA
Exemplo:	OpenCif()
Exemplo em C:	<pre> typedef int (FAR PASCAL *OPENCIF) (void); int ret; OPENCIF pOpenCif; pOpenCif = (OPENCIF) GetProcAddress(h, "OpenCif") ; switch (pOpenCif()) { case CIF_ERR_CONFIG: // Erro no arquivo CIF.INI break; case CIF_ERR_SERIAL: // Falha na abertura da serial break; case CIF_ERR_SYS: // Erro na alocação de arquivos break; case CIF_ERR_TEMP: // Temperatura Alta break; case CIF_ERR_PPAPEL: // Pouco Papel break; case CIF_ERR_IRRECUPERAVEL: // Erro irrecuperavel no modulo impressor break; case CIF_ERR_MECANICO: // Erro mecânico no mecanismo impressor break; case CIF_ERR_TABERTA: // Tampa Aberta break; case CIF_OK: break; } </pre>

Função **CloseCif()**

Função: **CloseCif**

DLL ECF32M

Especificação de Uso

Descrição:	Esta função é responsável pela finalização do uso da dll. Libera a porta serial e outros processos internos. Esta função deve, necessariamente, ser a última função a ser utilizada pela aplicação.
Sintaxe C:	<code>void FAR PASCAL CloseCif(void);</code>
Sintaxe VB:	<code>Declare Function CloseCif Lib "ECF32M.DLL" ()</code>
Sintaxe Delphi:	<code>Procedure CloseCif(); stdcall; external 'ECF32M.dll';</code>
Parâmetros:	Nenhum.
Retorno:	Nenhum.
Exemplo:	<code>CloseCif()</code>
Exemplo em C:	<pre>typedef int (FAR PASCAL *CLOSECIF) (void); CLOSECIF pCloseCif; : : pCloseCif = (CLOSECIF) GetProcAddress(h, "CloseCif"); pCloseCif();</pre>

[Voltar ao Índice](#)

Funções de Operações Fiscais

Funções de Operações Fiscais *AbreCupomFiscal()*

Função:	AbreCupomFiscal
Descrição:	Abre cupom fiscal de venda
Sintaxe C:	<code>int FAR PASCAL AbreCupomFiscal(void);</code>
Sintaxe VB:	<code>Declare Function AbreCupomFiscal Lib "ECF32M.DLL" () As Long</code>
Sintaxe Delphi:	<code>function AbreCupomFiscal():Integer; stdcall; external 'ECF32M.dll';</code>
Parâmetros:	Nenhum.
Retorno:	CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEXECUÇÃO
Exemplo:	<code>AbreCupomFiscal()</code>
Exemplo em C:	<pre>typedef int (FAR PASCAL *ABRECUPOM) (void); int ret; ABRECUPOM pAbreCupomFiscal; : : pAbreCupomFiscal = (ABRECUPOM) GetProcAddress(h, "AbreCupomFiscal"); ret = pAbreCupomFiscal ();</pre>

[Voltar ao Índice](#)

Função *AbreCupomFiscalCPF_CNPJ()*

Função:	AbreCupomFiscalCPF_CNPJ
Descrição:	Abre um cupom fiscal de venda possibilitando a passagem do número do CPF/CNPJ do cliente para ser impresso.
Sintaxe C:	<code>int FAR PASCAL AbreCupomFiscalCPF_CNPJ(</code> <code>char *CPF_CNPJ</code> <code>);</code>
Sintaxe VB:	<code>Declare Function AbreCupomFiscalCPF_CNPJ Lib "ECF32M.DLL" (</code> <code>ByVal CPF_CNPJ as String</code>

DLL ECF32M

Especificação de Uso

Sintaxe Delphi:) As Long
function AbreCupomFiscalCPF_CNPJ(
CPF_CNPJ:Pchar
):Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
CPF_CNPJ Número do CPF/CNPJ do cliente.com 28 caracteres alfanuméricos {28s}.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo
Exemplo em C: AbreCupomFiscalCPF_CNPJ("1234567890123456789012345678").
typedef int (FAR PASCAL *ABRECUPOM) (char *CPJ_CNPJ);
int ret;

ABRECUPOM pAbreCupomFiscalCPF_CNPJ;
:
:
:
pAbreCupomFiscalCPF_CNPJ = (ABRECUPOM) GetProcAddress(h,
"AbreCupomFiscalCPF_CNPJ");
ret = pAbreCupomFiscalCPF_CNPJ (CPJ_CNPJ);

Atenção:

- Função disponível a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Função

VendaItemStr()

Função: **VendaItemStr**
Descrição: Realiza o registro de itens nas operações fiscais de venda.
Esta função é semelhante à função VendaItem(), a única diferença é que todos parâmetros dessa função são do tipo ponteiro para string para facilitar o seu uso por linguagens que tem dificuldade em passar parâmetros do tipo byte.

Sintaxe C: Int FAR PASCAL VendaItemStr(
char *fmt
, char *qtd
, char *punit
, char *trib
, char *tdesc
, char *valor
, char *unid
, char *cod
, char *ex
, char *descr
, char *legop
);

Sintaxe VB: Declare Function VendaItemStr Lib "ECF32M.DLL" (
ByVal fmt As String
, ByVal qtd As String
, ByVal punit As String
, ByVal trib As String
, ByVal tdesc As String
, ByVal valor As String
, ByVal unid As String
, ByVal cod As String
, ByVal ex As String
, ByVal descr As String

,ByVal legop As String
) As Long

Sintaxe Delphi: function VendaltemStr (
 fmt: Pchar
 ;qtd: Pchar
 ;punit: Pchar
 ;trib: Pchar
 ;Tdesc: Pchar
 ;valor: PChar
 ;unid: PChar
 ;cod: PChar
 ;ex: Pchar
 ;descr:Pchar
 ;legendaOP: Pchar
) :Integer;stdcall; external 'ECF32M.dll';

Parâmetros:

Fmt Formato de impressão do item.
Esse parâmetro interfere diretamente no formato do Valor do Item e da Quantidade, indicando o tamanho e número de dígitos de casas decimais. O registro de item em uma única linha, somente será possível se a quantidade for igual a 1.

Valor	Descrição
" " (00h – Nulo)	Linhas: Impressão do registro de item em duas linhas. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"-" (2Dh)	Linhas: Impressão do registro de item em uma linha. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"Z" (5Ah) (*)	Linhas: Impressão do registro de item em linha normal. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"A" (41h) (*)	Linhas: Impressão do registro de item em uma linha. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"B" (42h) (*)	Linhas: Impressão do registro de item em uma linha. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 8 dígitos na parte inteira e 3 decimais
"C" (43h) (*)	Linhas: Impressão do registro de item em duas linhas. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"D" (44h) (*)	Linhas: Impressão do registro de item em duas linhas. Qtd: 6 dígitos, com 3 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 8 dígitos na parte inteira e 3 decimais
"E" (45h) (*)	Linhas: Impressão do registro de item em uma linha. Qtd: 7 dígitos, com 4 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"F" (46h) (*)	Linhas: Impressão do registro de item em uma linha. Qtd: 7 dígitos, com 4 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 8 dígitos na parte inteira e 3 decimais
"G" (47h) (*)	Linhas: Impressão do registro de item em duas linhas. Qtd: 7 dígitos, com 4 dígitos na parte inteira e 3 decimais Punit: 11 dígitos, com 9 dígitos na parte inteira e 2 decimais
"H" (48h) (*)	Linhas: Impressão do registro de item em duas linhas. Qtd: 7 dígitos, com 4 parte inteira e 3 decimais Punit: 11 dígitos, com 8 dígitos na parte inteira e 3 decimais

ATENÇÃO:

(*) – Parâmetros disponíveis a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

qtd: Quantidade do item.

Conforme o parâmetro **fmt**, poderá ser composto por:

- 6 bytes no formato "nnnnnn" (6n), sendo 3 dígitos na parte inteira e 3 dígitos na parte decimal ou
- 7 bytes no formato "nnnnnnn" (7n), sendo 4 dígitos na parte inteira e 3 dígitos na parte decimal.

Exemplos :

1) Quantidade com 6 dígitos (3 dígitos na parte inteira e 3 dígitos na parte decimal).

1 unidade (com 6 dígitos) _____ "001000".
100 unidades (com 6 dígitos) _____ "100000".
0,345 unidades (com 6 dígitos) _____ "000345".

2) Quantidade com 7 dígitos (4 dígitos na parte inteira e 3 dígitos na parte decimal).

1 unidade (com 7 dígitos) _____ "0001000".
100 unidades (com 7 dígitos) _____ "0100000".
0,345 unidades (com 7 dígitos) _____ "0000345".

punit: Preço ou valor unitário do item.

Sempre composto de 11 bytes no formato "nnnnnnnnnnnn" (11n), poderá variar o número de dígitos da parte inteira e decimal:

- 11 bytes no formato "nnnnnnnnnnnn" (11n), sendo 9 dígitos na parte inteira e 2 dígitos na parte decimal ou
- 11 bytes no formato "nnnnnnnnnnnn" (11n), sendo 8 dígitos na parte inteira e 3 dígitos na parte decimal.

Exemplos:

1) Preço Unitário com 11 dígitos (9 dígitos na parte inteira e 2 dígitos na parte decimal).

\$1,00 _____ "00000000100".
\$1.000,00 _____ "00000100000".
\$0,99 _____ "00000000099".

2) Preço Unitário com 11 dígitos (8 dígitos na parte inteira e 3 dígitos na parte decimal).

\$1,00 _____ "00000001000".
\$1.000,00 _____ "00001000000".
\$0,998 _____ "00000000998".

trib: Situação tributária composto por 3 bytes obedecendo a seguinte codificação:

- **"Tnn"**: Tributado como ICMS ou ISS, com "nn" variando entre "00" e "15" conforme o índice da alíquota. O totalizador de situação tributária **"T00"** é específico para realizar operações sujeitas ao ISS (Imposto Sobre Serviços).
- **"F00"**: Substituição Tributária
- **"I00"**: Isenção Tributária
- **"N00"**: Não Tributado (não incidência)

Exemplos:

- 1) Para registrar um item na alíquota T01: "T01"
- 2) Para registrar um item como Substituição Tributária (F): "F00"
- 3) Para registrar um item como Isento (I): "I00"
- 4) Para registrar um item como Não Tributado (N): "N00"

tdesc: Tipo do desconto a ser enviado no próximo campo, no seguinte formato:

- "&" (26h) ou "1" (31h): Desconto por valor
- "%" (25h), "0" (30h), "" (00h – Null): Desconto por porcentagem

OBS: Mesmo quando não houver desconto em item, é necessário passar o tipo de desconto e preencher o valor com zeros.

valor: Valor do desconto do item. Pode ser informado por valor ou porcentagem, conforme escolhido no campo anterior tdsc.

- **Desconto por valor:** composto de 15 caracteres no formato "nnnnnnnnnnnnnnn" (15n). (13 dígitos para parte inteira e 2 dígitos para a parte decimal).

- **Desconto por porcentagem:** composto de 4 caracteres no formato "nnnn" (4n). (2 dígitos para a parte inteira e 2 dígitos para a parte decimal)

OBS: Mesmo quando não houver desconto em item, especifique o tipo de desconto (valor ou porcentagem) e preencha o campo valor com zeros, respeitando o número de caracteres para cada tipo, se: % (4 dígitos) ou & (15 dígitos).

Exemplos:

- 1) Para registrar um desconto em item de \$21,80: "0000000000002180"
- 2) Para registrar um desconto em item de 07,20%: "0720"

unid: Descrição da unidade da mercadoria ("pc", "kg", "mt", etc). Composto por 2 caracteres no formato "ss" (2s).

Exemplos:

- 1) Para registrar um item na unidade kilo: "kl"

cod: Código da mercadoria composto por 13 caracteres no formato "sssssssssssss". (13s).

Exemplos:

- 1) Para registrar um código de produto "1234567890123", informe: "1234567890123"
- 2) Para registrar um código de produto "AB-123", informe: " AB-123" ou "AB-123 "

ex: Tamanho da descrição da mercadoria (1n).

- "" (00h - Null): a descrição do item deverá ter 20 caracteres alfanuméricos (default).
- "0" (30h): descrição do item deverá ter 20 caracteres alfanuméricos.
- "1" (31h): descrição do item deverá ter 38 caracteres alfanuméricos.
- "2" (32h): descrição do item deverá ter 76 caracteres alfanuméricos (38 * 2).
- "3" (33h): descrição do item deverá ter 114 caracteres alfanuméricos (38 * 3).
- "4" (34h): descrição do item deverá ter 152 caracteres alfanuméricos (38 * 4).

- "5" (35h): descrição do item deverá ter 190 caracteres alfanuméricos (38 * 5)

descr: Descrição da mercadoria. Deve ser formatada com o número exato de caracteres indicado em "ex" (20, 38, 76, 114, 152 ou 190 caracteres). Se for necessário, complete a descrição com espaços em branco à direita ou esquerda.

Exemplos:

1) Para registrar um item cuja descrição é "Batata", se:

Se o campo "Ex" for "0", então: "Batata "

Se o campo "Ex" for "1", então: "Batata "

legop: Legenda opcional que será impressa somente quando houver desconto válido. Composto de 14 caracteres no formato "ssssssssssssss". (14s). Caso este parâmetro seja uma string nula (legop = ""), será impressa a legenda "DESCONTO" (default).

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: ("G","0000448","00000000109","T01","%","0000","kl","2008800000273","","BATA1 A", "")

Onde:

- **Fmt:** "G" (a quantidade deverá ter 7 dígitos, sendo 4 dígitos para parte inteira e 3 dígitos para parte decimal, e o preço unitário deverá ter 11 dígitos, sendo 9 dígitos para a parte inteira e 2 dígitos para parte decimal)
- **Qtd** (0,448 kl): "0000448" (7 dígitos: 4 para parte inteira e 3 para parte decimal)
- **Punit** (\$1,09): "00000000109" (11 dígitos: 9 para parte inteira e 2 para parte decimal)
- **Trib:** "T01" (3 dígitos: 1s e 2n)
- **Tdesc:** "%" (por porcentagem)
- **Valor (sem desconto):** "0000" (4 dígitos: 2 para parte inteira e 2 para parte decimal)
- **Und:** "kl" (unidade do produto)
- **Cod:** "2008800000273" (13s)
- **Ex:** "" (nulo, portanto o tamanho do campo descrição terá 20 caracteres)
- **Descr:** "BATATA " (20 caracteres).
- **LegOp:** "" (nulo, portanto se houvesse valor de desconto válido nesse registro de item, seria impresso a legenda "DESCONTO").

Exemplo em C:

```
typedef int (FAR PASCAL *VENDA) (
    char *
    ,char *
    ,char *
    ,char *
    ,char *
    ,char *
    ,char *
    ,char *
    ,char *
    ,char *
    );

char BufDesc[21];
char BufQtd[7];
char BufPunit[12];
char BufTrib[5];
char BufValor[16];
char BufUnid[3];
char BufCod[14];
```

VENDA pVenda;

```
pVenda = (VENDA) GetProcAddress(h, "VendaltemStr") ;
strcpy (BufQtd,"001000");
strcpy (BufPunit,"00000001000");
strcpy (BufTrib,"I00");
strcpy (BufValor,"0000");
strcpy (BufUnid,"pc");
strcpy (BufCod,"00000000000001");
strcpy (BufDesc,"abcdefghijklmnopqrst");
```

```
pVenda(
    "0"
    ,BufQtd
    ,BufPunit
    ,BufTrib
    ,"%0"
    ,BufValor
    ,BufUnid
    ,BufCod
    ,"0"
    ,BufDesc
    ,""
    );
```

Observações:

- Nos campos numéricos, os dígitos não significativos deverão ser preenchidos com zero ('0').

[Voltar ao Índice](#)

Funções de Operações Fiscais

CancelamentoItem()

Função:	CancelamentoItem
Descrição:	<p>Cancela um item já registrado no cupom.</p> <ul style="list-style-type: none"> - Deve-se informar o número sequencial do item a ser cancelado; - Lembre-se que mesmo após o cancelamento de um item, o número sequencial dos itens ainda válidos no cupom, não sofrem qualquer alteração. - Um item só poderá ser cancelado se estiver registrado no cupom.
Sintaxe C:	<pre>int FAR PASCAL CancelamentoItem(char * numitem);</pre>
Sintaxe VB:	<pre>Declare Function CancelamentoItem Lib "ECF32M.DLL" (ByVal numitem As String) As Long</pre>
Sintaxe Delphi:	<pre>function CancelamentoItem (numitem: Pchar) :Integer; stdcall; external 'ECF32M.dll';</pre>
Parâmetros:	
NumItem:	Número sequencial do item. Composto de 3 caracteres no formato "nnn" podendo variar entre "001" e "450" (3n).
Retorno:	<p>CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT</p>

Exemplo:
Exemplo em C:

```

CIF_EMEXEÇÃO
CancelamentoItem("001")
typedef int (FAR PASCAL *CANCELAMENTO) (char *);
int ret;
char numitem[4];
CANCELAMENTO pCancelaltem;
:
:
pCancelaltem = (CANCELAMENTO) GetProcAddress(h, "CancelamentoItem");
strcpy (numitem,"001");
ret = pCancelaltem (numitem);

```

[Voltar ao Índice](#)

Funções de Operações Fiscais

DescontoItemPorcentagem()

Função: **DescontoItemPorcentagem**
Descrição: Realiza a operação de desconto por porcentagem no último item registrado. Lembre-se que:
1) Este comando só é válido caso o item ainda não tenha sofrido nenhum desconto, seja através dessa função ou da função VendaItem() ou VendaItemStr().
2) Caso a operação de desconto seja aplicada em um item registrado em uma alíquota de ISS e o ECF esteja programado com "Desconto sobre ISS" desabilitado (Intervenção Técnica), esta função não será executada com sucesso. (o parâmetro "Desconto sobre ISS" está disponível a partir da versão FCP-500 do firmware da impressora fiscal)

Sintaxe C:

```

Int FAR PASCAL DescontoItemPorcentagem (
    char * porcentagem
    ,char *legop
);

```

Sintaxe VB:

```

Declare Function DescontoItemPorcentagem Lib "ECF32M.DLL" (
    ByVal porcentagem As String
    ,ByVal legop As String
) As Long

```

Sintaxe Delphi:

```

function DescontoItemPorcentagem (
    porcentagem: Pchar
    ;legop: Pchar
):Integer; stdcall; external 'ECF32M.dll';

```

Parâmetros:
Porcentagem: Porcentagem de Desconto. Composto de 4 caracteres no formato "nnnn" (2n para a parte inteira e 2n para a parte decimal) (4n)
LegOp: Legenda opcional do desconto. Composto de 14 caracteres no formato "ssssssssssssss" (14s). Impressa somente em caso de desconto válido.
 Obs: Caso esse parâmetro seja uma string nula (LegOp = Null), será impressa a legenda "DESCONTO" (default).

Retorno:
 CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXEÇÃO

Exemplo:
 1) Desconto de 0,50%:
 DescontoItemPorcentagem ("0050","SUPER DESCONTO")
 2) Desconto de 1,00%:
 DescontoItemPorcentagem ("0100","SUPER DESCONTO")

Exemplo em C:

```

typedef int (FAR PASCAL *DESCITEM) (char, char, char *);

```

```
char toper;
char BufDesc[16],BufLegOp[15];
int ret;
DESCITEM pDescontoItem;
:
:
pDescontoItem= (DESCITEM) GetProcAddress(h, " DescontoItemPorcentagem
");

strcpy (BufDesc,"1000");
strcpy (BufLegOp,"Promocao  ");

ret = pDescontoItem(BufValor,BufLegOp);
```

[Voltar ao
Índice](#)

**Funções de
Operações Fiscais**

DescontoItemValor()

Função: **DescontoItemValor**
Descrição: Realiza a operação de desconto por valor no último item registrado. Lembre-se que:
1) Este comando só é válido caso o item ainda não tenha sofrido nenhum desconto, seja através dessa função ou da função VendaItem() ou VendaItemStr().
2) Caso a operação de desconto seja aplicada em um item registrado em uma alíquota de ISS e o ECF esteja programado com “Desconto sobre ISS” desabilitado (Intervenção Técnica), esta função não será executada com sucesso. (o parâmetro “Desconto sobre ISS” está disponível a partir da versão FCP-500 do firmware da impressora fiscal)

Sintaxe C: Int FAR PASCAL DescontoItemValor (
char *valor
, char *legop
);

Sintaxe VB: Declare Function DescontoItemValor Lib "ECF32M.DLL" (
ByVal valor As String
, ByVal legop As String
) As Long

Sintaxe Delphi: function DescontoItemValor (
valor: Pchar
; legop: Pchar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
Valor: Valor de Desconto. Composto de 15 caracteres no formato “nnnnnnnnnnnnnnnn” (13 para a parte inteira e 2 para a parte decimal) (15n)
LegOp: Legenda opcional do desconto. Composto de 14 caracteres no formato “ssssssssssssss” (14s). Impressa somente em caso de desconto válido.
Obs: Caso esse parâmetro seja uma string nula (LegOp = Null), será impressa a legenda “DESCONTO” (default).

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo:
1) Desconto de \$0,50:
DescontoItemValor (“0000000000000050”, “SUPER DESCONTO”)
2) Desconto de \$1,00:

Exemplo em C:

```
DescontoItemValor ("000000000000100","SUPER DESCONTO")
typedef      int (FAR PASCAL *DESCITEM) (char, char, char *);
char BufDesc[16],BufLegOp[15];
int ret;
DESCITEM pDescontoItem;
:
:
pDescontoItem= (DESCITEM) GetProcAddress(h, " DescontoItemValor ");

strcpy (BufDesc," 000000000000100");
strcpy (BufLegOp,"Promocao  ");

ret = pDescontoItem(BufValor,BufLegOp);
```

[Voltar ao Índice](#)

Funções de Operações Fiscais

DescontoItem()

Função:

DescontoItem

Descrição:

Realiza a operação de desconto no último item registrado. Lembre-se que:

- 1) Este comando só é válido caso o item ainda não tenha sofrido nenhum desconto, seja através dessa função ou da função `VendaItem()` ou `VendaItemStr()`.
- 2) Caso a operação de desconto seja aplicada em um item registrado em uma alíquota de ISS e o ECF esteja programado com "Desconto sobre ISS" desabilitado (Intervenção Técnica), esta função não será executada com sucesso. (o parâmetro "Desconto sobre ISS" está disponível a partir da versão FCP-500 do firmware da impressora fiscal)

Sintaxe C:

```
Int FAR PASCAL DescontoItem (
    char toper
    ,char *valor
    ,char *legop
);
```

Sintaxe VB:

```
Declare Function DescontoItem Lib "ECF32M.DLL" (
    ByVal toper As Byte
    ,ByVal valor As String
    ,ByVal legop As String
) As Long
```

Sintaxe Delphi:

```
function DescontoItem (
    toper:Char
    ;valor: Pchar
    ;legop: Pchar
):Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

Toper:

Informa o tipo do desconto a ser enviado. Composto por um caracterer:

- '&' (26h) ou '1' (31h): por valor
- '%' (25h) ou '0' (30h) por porcentagem

Valor:

Valor do desconto.

- **Por valor:** Composto de 15 caracteres no formato "nnnnnnnnnnnnnnnn" (13 para a parte inteira e 2 para a parte decimal) (15n)
- **Por Porcentagem:** Composto de 4 caracteres no formato "nnnn" (2n para a parte inteira e 2n para a parte decimal) (4n)

LegOp:

Legenda opcional do desconto. Composto de 14 caracteres no formato "ssssssssssssss" (14s). Impressa somente em caso de desconto válido. Obs: Caso esse parâmetro seja uma string nula (LegOp = Null), será

Retorno: impressa a legenda "DESCONTO" (default).
 CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: 1) Desconto de 0,50%:
 DescontoItem ('%', "0050", "SUPER DESCONTO")

2) Desconto de \$0,50:
 DescontoItem ('&', "0000000000000050", "SUPER DESCONTO")

Exemplo em C:

```
typedef int (FAR PASCAL *DESCITEM) (char, char, char *);
char toper;
char BufDesc[16], BufLegOp[15];
int ret;
DESCITEM pDescontoItem;
:
:
pDescontoItem= (DESCITEM) GetProcAddress(h, "DescontoItem") ;

toper = '%';
strcpy (BufDesc, " 1000");
strcpy (BufLegOp, "Promocao  ");

ret = pDescontoItem(toper, BufValor, BufLegOp);
```

[Voltar ao Índice](#)

Funções de Operações Fiscais

TotCupomSemDescAcres()

Função: **TotCupomSemDescAcres**
Descrição: Realiza a totalização de um cupom de venda sem as operações de desconto/acrécimo.

- Após totalizar o Cupom, não é mais permitido registrar item, dar desconto em item, cancelar item, entre outras operações.

Sintaxe C: int FAR PASCAL TotCupomSemDescAcres ();
Sintaxe VB: Declare Function TotCupomSemDescAcres Lib "ECF32M.DLL" () As Long
Sintaxe Delphi: function TotCupomSemDescAcres ():Integer; stdcall; external 'ECF32M.dll'
Parâmetros: Nenhum.
Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: TotCupomSemDescAcres ()
Exemplo em C:

```
typedef int (FAR PASCAL *TOTALIZA) (char, char, char *);
int ret;
TOTALIZA pTotalizaCupom;

pTotalizaCupom = (TOTALIZA) GetProcAddress(h, " TotCupomSemDescAcres " ) ;

ret = pTotalizaCupom ();
```

[Voltar ao Índice](#)

Funções de

TotCupomAcresValor ()

Operações Fiscais

Função:	TotCupomAcrsValor
Descrição:	Realiza a totalização de um cupom de venda, com a operação de acréscimo por valor. <ul style="list-style-type: none">• Após totalizar o Cupom, não é mais permitido registrar item, dar desconto em item, cancelar item, entre outras operações.
Sintaxe C:	<pre>int FAR PASCAL TotCupomAcrsValor (char *valor ,char *legop);</pre>
Sintaxe VB:	<pre>Declare Function TotCupomAcrsValor Lib "ECF32M.DLL" (ByVal valor As String ,ByVal legop As String) As Long</pre>
Sintaxe Delphi:	<pre>function TotCupomAcrsValor (valor: PChar ;legop: PChar):Integer; stdcall; external 'ECF32M.dll'</pre>
Parâmetros:	
Valor	Valor do acréscimo. Composto de 15 caracteres no formato "nnnnnnnnnnnnnn" (13 para a parte inteira e 2 para a parte decimal) (15n)
legop:	Legenda opcional, impressa somente em caso de desconto/acrécimo válido, composto de 14 caracteres no formato "ssssssssssssss" (14s). Caso esse parâmetro seja uma string nula (legop = Null), será impressa a legenda "ACRÉSCIMO" (default).
Retorno:	CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEEXECUÇÃO
Exemplo:	1) Totalizar Cupom com Acrécimo de \$1,50 e legenda opcional. TotCupomAcrsValor ("0000000000000150","*VOLTE SEMPRE*")
Exemplo em C:	<pre>typedef int (FAR PASCAL *TOTALIZA) (char, char, char *); char BufValor[16],BufLegOp[15]; int ret; TOTALIZA pTotalizaCupom; pTotalizaCupom = (TOTALIZA) GetProcAddress(h, " TotCupomAcrsValor "); strcpy (BufValor,"0000000000001000"); strcpy (BufLegOp,"Servico "); ret = pTotalizaCupom (BufValor,BufLegOp);</pre>

[Voltar ao Índice](#)

Funções de Operações Fiscais

[TotCupomAcrsPorcentagem \(\)](#)

Função:	TotCupomAcrsPorcentagem
Descrição:	Realiza a totalização de um cupom de venda, com a operação de acréscimo por porcentagem.

- Após totalizar o Cupom, não é mais permitido registrar item, dar desconto em item, cancelar item, entre outras operações.

Sintaxe C:	int FAR PASCAL TotCupomAcresPorcentagem (char *porcentagem char *legop);
Sintaxe VB:	Declare Function TotCupomAcresPorcentagem Lib "ECF32M.DLL" (ByVal porcentagem As String ByVal legop As String) As Long
Sintaxe Delphi:	function TotCupomAcresPorcentagem (porcentagem: Pchar legop: PChar):Integer; stdcall; external 'ECF32M.dll'
Parâmetros:	
Porcentagem	Porcentagem do acréscimo. Composto de 4 caracteres no formato "nnnn" (2n para a parte inteira e 2n para a parte decimal) (4n)
legop:	Legenda opcional, impressa somente em caso de desconto/acrécimo válido, composto de 14 caracteres no formato "ssssssssssssss" (14s). Caso esse parâmetro seja uma string nula (legop = Null), será impressa a legenda "ACRÉSCIMO" (default).
Retorno:	CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEXECUÇÃO
Exemplo:	1) Totalizar Cupom com Acrécimo de 1,50% e legenda opcional. TotCupomAcresPorcentagem("0150", "*VOLTE SEMPRE*") 2) Totalizar Cupom com Acrécimo de 0,50% e legenda default. TotCupomAcresPorcentagem ("0050", "")
Exemplo em C:	typedef int (FAR PASCAL *TOTALIZA) (char, char, char *); char BufValor[16],BufLegOp[15]; int ret; TOTALIZA pTotalizaCupom; pTotalizaCupom = (TOTALIZA) GetProcAddress(h, " TotCupomAcresPorcentagem"); strcpy (BufValor,"1000"); strcpy (BufLegOp,"Servico"); ret = pTotalizaCupom (BufValor,BufLegOp);

[**Voltar ao Índice**](#)

Funções de Operações Fiscais

TotCupomDescValor ()

Função:	TotCupomDescValor
Descrição:	Realiza a totalização de um cupom de venda, com a operação de desconto por valor. <ul style="list-style-type: none"> • Após totalizar o Cupom, não é mais permitido registrar item, dar desconto em item, cancelar item, entre outras operações.

- Caso a operação de desconto seja aplicada em um cupom com itens registrados em alíquotas de ISS e o ECF esteja programado com o parâmetro "Desconto sobre ISS" (Intervenção Técnica) desabilitado, o valor do desconto não poderá ser maior, ou igual, ao valor total da venda sobre ICMS.

Sintaxe C: int FAR PASCAL TotCupomDescValor (
char *valor
, char *legop
);

Sintaxe VB: Declare Function TotCupomDescValor Lib "ECF32M.DLL" (
ByVal valor As String
, ByVal legop As String
) As Long

Sintaxe Delphi: function TotCupomDescValor (
valor: Pchar
; legop: PChar
): Integer; stdcall; external 'ECF32M.dll'

Parâmetros:

Valor Valor do desconto. Composto de 15 caracteres no formato "nnnnnnnnnnnnnn" (13 para a parte inteira e 2 para a parte decimal) (15n)

legop: Legenda opcional, impressa somente em caso de desconto/acrécimo válido, composto de 14 caracteres no formato "ssssssssssssss" (14s). Caso esse parâmetro seja uma string nula (legop = Null), será impressa a legenda "DESCONTO" ou "ACRÉSCIMO" (default).

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: 1) Totalizar Cupom com Desconto de \$1,50 e legenda opcional.

TotCupomDescValor ("0000000000000150", "*VOLTE SEMPRE*")

Exemplo em C: typedef int (FAR PASCAL *TOTALIZA) (char, char, char *);
char BufValor[16], BufLegOp[15];
int ret;
TOTALIZA pTotalizaCupom;

pTotalizaCupom = (TOTALIZA) GetProcAddress(h, "TotCupomDescValor");

strcpy (BufValor, "0000000000001000");
strcpy (BufLegOp, "Servico");

ret = pTotalizaCupom (BufValor, BufLegOp);

[**Voltar ao Índice**](#)

Funções de Operações Fiscais

[TotCupomDescPorcentagem](#) 

Função: TotCupomDescPorcentagem

Descrição: Realiza a totalização de um cupom de venda, com a operação de desconto por porcentagem.

Obs:

- Após totalizar o Cupom, não é mais permitido registrar item,

- dar desconto em item, cancelar item, entre outras operações.
- Caso a operação de desconto seja aplicada em um cupom com itens registrados em alíquotas de ISS e o ECF esteja programado com o parâmetro "Desconto sobre ISS" (Intervenção Técnica) desabilitado, o valor do desconto não poderá ser maior, ou igual, ao valor total da venda sobre ICMS.

Sintaxe C: int FAR PASCAL TotCupomDescPorcentagem (
char * porcentagem
, char * legop
);

Sintaxe VB: Declare Function TotCupomDescPorcentagem Lib "ECF32M.DLL" (
ByVal porcentagem As String
, ByVal legop As String
) As Long

Sintaxe Delphi: function TotCupomDescPorcentagem (
porcentagem: PChar
; legop: PChar
): Integer; stdcall; external 'ECF32M.dll'

Parâmetros:
Porcentagem Porcentagem do desconto. Composto de 4 caracteres no formato "nnnn" (2n para a parte inteira e 2n para a parte decimal) (4n)

legop: Legenda opcional, impressa somente em caso de desconto/acréscimo válido, composto de 14 caracteres no formato "ssssssssssssss" (14s). Caso esse parâmetro seja uma string nula (legop = Null), será impressa a legenda "DESCONTO" ou "ACRÉSCIMO" (default).

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: 1) Totalizar Cupom com Desconto de 1,50% e legenda opcional.
TotalizarCupom ('Z', '%', "0150", "*VOLTE SEMPRE*")

Exemplo em C: typedef int (FAR PASCAL *TOTALIZA) (char, char, char *);
char BufValor[16], BufLegOp[15];
int ret;
TOTALIZA pTotalizaCupom;

pTotalizaCupom = (TOTALIZA) GetProcAddress(h, "TotCupomDescPorcentagem");

strcpy (BufValor, "1000");
strcpy (BufLegOp, "Servico ");

ret = pTotalizaCupom (BufValor, BufLegOp);

[**Voltar ao Índice**](#)

Funções de Operações Fiscais

Pagamento()

Função: Pagamento

Descrição: Realiza o pagamento ou o troco do cupom de venda, imprimindo a(s) forma(s) de pagamento no seguinte formato:

PAGO EM:

<legenda programável> = nnnnnnnnnnnnn,nn

Observações:

- 1) Este comando só poderá ser utilizado após a totalização do cupom, através da função TotalizarCupom() ou TotalizarCupomParcial().
- 2) A função Troco() também pode realizar a operação de troco. Use a função que melhor se adaptar ao seu aplicativo.

Sintaxe C:

```
int FAR PASCAL Pagamento(
    char *reg
    ,char *vpcto
    ,char troco
);
```

Sintaxe VB:

```
Declare Function Pagamento Lib "ECF32M.DLL" (
    ByVal reg As String
    ,ByVal vpcto As String
    ,ByVal troco As Byte
) As Long
```

Sintaxe Delphi:

```
function Pagamento(
    reg: Pchar
    ;vpcto: Pchar
    ;troco: Char
):Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

reg: Índice do registrador não fiscal da forma de pagamento ("dinheiro", "cheque", "ticket", "a prazo" entre outras). Os registradores não fiscais ocupam posições de memória que vão de "00" a "15". Indicar a posição referente a forma de pagamento programada. (2n)

Pagamento:

Quando se tratar de um pagamento, indique a forma de pagamento cujo valor está sendo somado ao caixa.

Troco:

Quando se tratar de um troco, indique a forma de pagamento cujo valor está sendo retirado do caixa.

vpcto:

Indica o valor do Pagamento ou se é uma operação de Troco. Composto de 15 caracteres numéricos no formato "nnnnnnnnnnnnnn", sendo 13 dígitos para a parte inteira e 2 dígitos para a parte decimal (15n)

Pagamento:

Quando se tratar de um pagamento, indique o valor do pagamento (superior a zero). Para representar \$1,00 informe: "000000000000100"

Troco:

Quando se tratar de um troco, basta informar o valor zerado ("000000000000000"). O valor do troco será calculado automaticamente pela Impressora Fiscal.

troco:

Parâmetro que indica o modo do Troco.

Pagamento:

Quando se tratar de um pagamento (valor superior a zero) esse parâmetro poderá ser '0' ou '1'.

Troco:

Se o valor do troco será subtraído do registrador especificado ou não.

- '0' (30h) ou '' (00h – Nulo): será subtraído
- '1' (31h): não será subtraído

Retorno:

CIF_OK

CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEEXECUÇÃO

Exemplo:

- 1) Para registrar um pagamento de \$2,05 no registrador "01":
 Pagamento("01","0000000000000205",'1')
- 2) Para registrar um troco no registrador "01":
 Pagamento("01","0000000000000000",'1')

Exemplo em C:

```
typedef int (FAR PASCAL PGTO) (char, char *,char);
char BufValor[16];
int ret;
PGTO pPagamento;

pPagamento = (PGTO) GetProcAddress(h, "Pagamento") ;

strcpy (BufValor,"000000000012300");
ret = pPagamento ("02",BufValor,'0');
```

[Voltar ao Índice](#)

Funções de Operações Fiscais

PagamentoComTexto()

Função: **PagamentoComTexto**

Descrição: Realiza o pagamento do cupom de venda, permitindo a impressão de inserção de linhas de comentário (40s/80s). A forma de pagamento será impressa no seguinte formato:
 PAGO EM:
 <legenda programável> = nnnnnnnnnnnnnn,nn
 Observações:
 1) Este comando só poderá ser utilizado após a totalização do cupom, através da função TotalizarCupom.

Sintaxe C:

```
int FAR PASCAL PagamentoComTexto(
    char *reg
    ,char *vpgto
    ,char parm
    ,char *comentario
);
```

Sintaxe VB:

```
Declare Function PagamentoComTexto Lib "ECF32M.DLL" (
    ByVal reg As String
    ,ByVal vpgto As String
    ,ByVal parm As Byte
    ,ByVal comentario As String
    As Long
```

Sintaxe Delphi:

```
Function PagamentoComTexto(
    reg: PChar
    ;vpgto: PChar
    ;parm: Char
    ;comentario: PChar
):Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

reg: Identificação do registrador não fiscal (forma de pagamento, dinheiro, cheque, ticket, a prazo etc...).
 Os registradores não fiscais ocupam posições de memória que vão de

"00" á "15", indicar a posição referente ao registrador programado, (2n).

Vpgto: Valor do pagamento, programado no campo anterior, composto de 15 caracteres no formato "nnnnnnnnnnnnnn" (15n).

parm: Indica quantas linhas de comentário serão enviadas.

- '2' (32h): será enviada uma linha de comentário;
- '1' (34h): serão enviadas duas linhas de comentário.

Comentário: Linha de comentário opcional, composta de 40 (parm='1') ou 80 (parm='2') caracteres alfanuméricos, respeitando o que foi determinado através do parâmetro anterior. (40/80s)

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUCAO

Exemplo: PagamentoComTexto ("04","000000000009810",'1',
 "1234567890123456789012345678901234567890")

Exemplo em C:

```
typedef int (FAR PASCAL PGTOTEXTO) (char *, char *,char ,char *);
char BufValor[16];
int ret;
PGTOTEXTOpPagamentoComTexto;

pPagamentoComTexto = (PGTOTEXTO) GetProcAddress(h,
"PagamentoComTexto");

strcpy (BufValor,"0000000000012300");
ret = pPagamentoComTexto ("02",BufValor,'2' ,"Dolares");
```

Observações:

- Função disponível a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Funções de Operações Fiscais

Troco()

Função: **Troco**

Descrição: Realiza a operação de troco relativa a operação de pagamento do cupom.

Observações:

- A função Pagamento() também pode realizar operação de troco.
- Este comando só poderá ser utilizado após o pagamento do cupom, através da função Pagamento, ou PagamentoComTexto.
- O valor do troco será calculado pela ECF.

Sintaxe C: int FAR PASCAL Troco (
 char * reg
);

Sintaxe VB: Declare Function Troco Lib "ECF32M.DLL" (
 ByVal reg As String
) As Long

Sintaxe em Delphi: Function Troco:(
 reg: PChar
) Integer; stdcall;external 'ECF32M.dll';

Parâmetros:

reg: Identificação do registrador não fiscal (forma de pagamento, dinheiro, cheque, ticket, a prazo etc...).

Retorno: Os registradores não fiscais ocupam posições de memória que vão de "00" á "15", indicar a posição referente ao registrador programado, (2n).
 CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: Troco("02")

Exemplo em C:

```
typedef int (FAR PASCAL TROCO) (char *);
int ret;
TROCO pTroco;

pTroco = (TROCO) GetProcAddress(h,"Troco") ;
ret = pPagamentoComTexto ("02");
```

Observações:

- Função disponível a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Funções de Operações Fiscais

FechaCupomFiscal()

Função: **FechaCupomFiscal**

Descrição: Realiza o fechamento do cupom fiscal. Esta função deve ser usada após a totalização e o pagamento.

Sintaxe C:

```
int FAR PASCAL FechaCupomFiscal(
    char *tam_msg
    ,char *msg
);
```

Sintaxe VB:

```
Declare Function FechaCupomFiscal Lib "ECF32M.DLL" (
    ByVal tam_msg As String
    ,ByVal msg As String
) As Long
```

Sintaxe Delphi:

```
function FechaCupomFiscal (
    tam_msg: PChar
    ;msg: PChar
):Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

tam_msg: Identificador de campo de tamanho da mensagem, composto de quatro caracteres no formato Snnn, obedecendo a seguinte condição:

- s = 'S' (53h) - indica que o próximo campo é o tamanho da mensagem.
- nnn – Pode variar entre "000" e "999", indica o tamanho da mensagem a ser enviada.

OBS: Caso o campo "tam_msg" possua o valor "S000", nenhuma mensagem será enviada. O parâmetro "msg" deverá possuir uma string nula.

msg: Mensagem promocional de tamanho definido pelo parâmetro anterior: tam_msg.

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: FechaCupomFiscal("S012","VOLTE SEMPRE")

Exemplo em C:

```
typedef int (FAR PASCAL *FECHACUPOM) (char*, char *);
```

DLL ECF32M Especificação de Uso

```
int ret;
FECHACUPOM      pFechaCupomFiscal;

pFechaCupomFiscal = (FECHACUPOM) GetProcAddress(h,
"FechaCupomFiscal");
ret = pFechaCupomFiscal ("S000",""); // sem mensagem promocional
ret = pFechaCupomFiscal ("S010","123456789\x0a"); // com mensagem promocional
```

[Voltar ao
Índice](#)

Funções de Operações Fiscais

CancelaCupomFiscal()

Função: **CancelaCupomFiscal**

Descrição: Realiza a operação de cancelamento do cupom de venda anterior. Esta operação só é válida se executada imediatamente após o fechamento do cupom de venda ou no caso dele ainda estar em andamento.

Sintaxe C: int FAR PASCAL CancelaCupomFiscal (void);

Sintaxe VB: Declare Function CancelaCupomFiscal Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function CancelaCupomFiscal(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: CancelaCupomFiscal()

Exemplo em C: typedef int (FAR PASCAL *CANCELACUPOM) (void);
int ret;
CANCELACUPOM pCancelaCupomFiscal;

```
pCancelaCupomFiscal = (CANCELACUPOM) GetProcAddress(
                                                                    h
                                                                    ,"CancelaCupomFiscal"
                                                                    );
ret = pCancelaCupomFiscal ();
```

[Voltar ao
Índice](#)

Funções de Operações Fiscais

LeituraX()

Função: **LeituraX**

Descrição: Imprime a Leitura do Dia, informando a situação dos totalizadores e contadores naquele instante. Caso a opção de Relatório Gerencial estiver habilitada, o ECF entrará automaticamente em Modo Relatório Gerencial (modo não fiscal) após a emissão do cupom de Leitura X. A impressão do Relatório Gerencial é livre e limitada a 10 minutos, e deverá ser encerrado pela função EncerrraCupomNaoFiscal().

Sintaxe C: int FAR PASCAL LeituraX(
char RelGer
);

Sintaxe VB: Declare Function LeituraX Lib "ECF32M.DLL" (
ByVal RelGer As Byte
) As Long

DLL ECF32M

Especificação de Uso

Sintaxe Delphi: `function LeituraX(
 relGer: Char
): Integer; stdcall; external 'ECF32M.dll';`

Parâmetros:

RelGer: Indica se o cupom de Leitura X conterá ou não o Relatório Gerencial.

- '0' (30h) ou "" (00h – Null): Sem relatório gerencial;
- '1' (31h): Com relatório gerencial

Retorno:

CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXEÇÃO

Exemplo: `LeituraX('0')`

Exemplo em C: `typedef int (FAR PASCAL *CANCELACUPOM) (void);
int ret;
CANCELACUPOM pCancelaCupomFiscal;

pCancelaCupomFiscal = (CANCELACUPOM) GetProcAddress(
 h
 , "CancelaCupomFiscal"
);

ret = pCancelaCupomFiscal ();`

[Voltar ao Índice](#)

Funções de Operações Fiscais

[LeituraXComRelGer\(\)](#)

Função: **LeituraXComRelGer**

Descrição: Imprime a Leitura do Dia e coloca o ECF no Modo Relatório Gerencial. A impressão do Relatório Gerencial é livre e limitada a 10 minutos, e deverá ser encerrado pela função `EncerraCupomNaoFiscal()`.

Sintaxe C: `int FAR PASCAL LeituraXComRelGer();`

Sintaxe VB: `Declare Function LeituraXComRelGer Lib "ECF32M.DLL" () As Long`

Sintaxe Delphi: `function LeituraXComRelGer (): Integer; stdcall; external 'ECF32M.dll';`

Parâmetros: Nenhum.

Retorno:

CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXEÇÃO

Exemplo: `LeituraXComRelGer()`

Exemplo em C: `typedef int (FAR PASCAL * LEITURAX) (void);
int ret;
LEITURAX pLeituraX;

pLeituraX = (CANCELACUPOM) GetProcAddress(
 h
 , "LeituraXComRelGer"
);

ret = pLeituraX ();`

[Voltar ao Índice](#)

Funções de Operações Fiscais

[ReducaoZ\(\)](#)

Função: **ReduçãoZ**

Descrição: Realiza o fechamento do dia e gera um registro na Memória Fiscal.

Caso a opção de Relatório Gerencial estiver habilitada, o ECF entrará automaticamente em Modo Relatório Gerencial (modo não fiscal) após a emissão do cupom de Redução Z. A impressão do Relatório Gerencial é livre e limitada a 10 minutos, e deverá ser encerrado pela função EncerraCupomNaoFiscal().

Sintaxe C: int FAR PASCAL ReducaoZ(
char RelGer
);

Sintaxe VB: Declare Function ReducaoZ Lib "ECF32M.DLL" (
ByVal RelGer As Byte
) As Long

Sintaxe Delphi: function ReducaoZ (
relGer: Char
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
RelGer: Indica se o cupom de Redução Z conterá ou não o Relatório Gerencial.

- '0' (30h) ou "" (00h – Null): Sem relatório gerencial;
- '1' (31h): Com relatório gerencial

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ReducaoZ('0')

Exemplo em C: typedef int (FAR PASCAL *REDUCAOZ) (void);
char RelGer;
int ret;
REDUCAOZ pReducaoZ;

RelGer='1';
pReducaoZ = (REDUCAOZ) GetProcAddress(h, "ReducaoZ");
ret = pReducaoZ (RelGer);

[Voltar ao Índice](#)

Funções de Operações Fiscais ***ReducaoZComRelGer()***

Função: **ReduçãoZComRelGer**

Descrição: Emite uma Redução Z e coloca o ECF no Modo Relatório Gerencial. A impressão do Relatório Gerencial é livre e limitada a 10 minutos, e deverá ser encerrado pela função EncerraCupomNaoFiscal().

Sintaxe C: int FAR PASCAL ReducaoZComRelGer();

Sintaxe VB: Declare Function ReducaoZComRelGer Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function ReducaoZComRelGer (): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ReducaoZComRelGer('0')

Exemplo em C: typedef int (FAR PASCAL *REDUCAOZ) (void);
int ret;
REDUCAOZ pReducaoZ;

pReducaoZ = (REDUCAOZ) GetProcAddress(h, "ReducaoZComRelGer");

ret = pReducaoZ ();

[Voltar ao Índice](#)

Funções de Operações Fiscais

LeMemFiscalData()

Função: **LeMemFiscalData**
Descrição: Imprime um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por data da redução.
Sintaxe C: int FAR PASCAL LeMemFiscalData(
char *datai
, char *dataf
, char Res
);
Sintaxe VB: Declare Function LeMemFiscalData Lib "ECF32M.DLL" (
ByVal datai As String
, ByVal dataf As String
, ByVal Res As Byte
) As Long
Sintaxe Delphi: function LeMemFiscalData (
datai:PChar
; dataf: PChar
; res: Char
): Integer; stdcall; external 'ECF32M.dll';
Parâmetros:
datai: Data inicial. Composto de 6 caracteres no formato "ddmmaa" (2d2m2a).
dataf: Data final. Composto de 6 caracteres no formato "ddmmaa" (2d2m2a).
Res: Indica se a Leitura será resumida ou não, obedecendo a seguinte condição:

- '0' (30h) ou '' (00h – Null) - Leitura completa.
- '1' (31h) - Leitura resumida.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO
Exemplo: LeMemFiscalData ("130503", "130603", '1')
Exemplo em C: typedef int (FAR PASCAL *LEMEMDATA) (char *, char*, char);
int ret;
char datai[7], dataf[7];
LEMEMDATA pLeMemFiscalData;

pLeMemFiscalData = (LEMEMDATA) GetProcAddress(h, "LeMemFiscalData");
strcpy (datai, "010297");
strcpy (dataf, "020297");
ret = pLeMemFiscalData (datai, dataf ,0);

[Voltar ao Índice](#)

Funções de Operações Fiscais

LeMemFiscalReducao()

Função: **LeMemFiscalReducao**
Descrição: Imprime um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por número da redução.

Sintaxe C: `int FAR PASCAL LeMemFiscalReducao (
char *redi
,char *redf
,char Res
);`

Sintaxe VB: `Declare Function LeMemFiscalReducao Lib "ECF32M.DLL" (
ByVal redi As String
,ByVal redf As String
,ByVal Res As Byte
) As Long`

Sintaxe Delphi: `function LeMemFiscalReducao(
redi:PChar
;redf: PChar
;res: Char
):Integer; stdcall; external 'ECF32M.dll';`

Parâmetros:

Redi: Redução inicial. Composto de 4 caracteres no formato "nnnn" (4n).

Redf: Redução final. Composto de 4 caracteres no formato "nnnn" (4n).

Res: Indica se a Leitura será resumida ou não, obedecendo a seguinte condição:

- '0' (30h) ou "" (00h - Null)- leitura completa.
- '1' (31h) - leitura resumida.

Retorno: `CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO`

Exemplo: `LeMemFiscalReducao("0123","0321",'1')`

Exemplo em C: `typedef int (FAR PASCAL *LEMEMRED) (char *, char*);
int ret;
char redi[5],redf[5];
LEMEMRED pLeMemFiscalReducao;
:
:
pLeMemFiscalReducao = (LEMEMRED) GetProcAddress(h
,"LEMEMFISCALREDUCAO");
strcpy (redi,"0010");
strcpy (redf,"0020");
ret = pLeMemFiscalReducao (redi,redf);`

[Voltar ao Índice](#)

Funções de Impressão de Cheque e Validação de Documentos

Funções de Impressão de Cheque e Validação de Documentos

ModoChequeValidacao()

Função: **ModoChequeValidacao**

Descrição: Inicializa a impressão de cheque ou validação. O ECF passará então para o estado de inserção de documento. No caso de validação, este comando somente estará habilitado se o número máximo de autenticações **não** for excedido (1 + 4 repetições da mesma autenticação) e o registro do valor a ser validado estiver ativo (operação imediatamente anterior com valor. Ex: pagamento, venda de item).

Sintaxe C: `int FAR PASCAL ModoChequeValidacao(`

DLL ECF32M

Especificação de Uso

```

char tipo
, char Load
);
Sintaxe VB: Declare Function ModoChequeValidacao Lib "ECF32M.DLL" (
    ByVal tipo As Byte
    , ByVal Load As Byte
) As Long
Sintaxe Delphi: function ModoChequeValidacao(
    tipo: Char
    ; load: Char
): Integer; stdcall; external 'ECF32M.dll';
Parâmetros:
Tipo: Seleciona modo cheque ou validação:
    • '0' (30h) - modo validação.
    • '1' (31h) - modo cheque (somente ECF 3002).

Load: Seleciona entrada do papel de validação (1s)
Para ECF 3002 / ECF3001:
    • '0'(30h) - carga pela entrada superior (modo default).
    • '1'(31h) - carga pelo slip (inferior).

Para Compact Fiscal (uma estação):
Seleciona entrada do papel de validação.
    • '0'(30h) - posiciona cabeça lado direito (modo default).
    • '1'(31h) - posiciona cabeça lado esquerdo.
    • '3'(31h) - posiciona cabeça no centro.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEEXECUÇÃO
Exemplo: ModoChequeValidacao('0', '0')
Exemplo em C: typedef int (FAR PASCAL *MODOCQUEUEVALID)( char ,char );
int ret;

MODOCQUEUEVALID pModoChequeValid;

pModoChequeValid = (MODOCQUEUEVALID) GetProcAddress( h
, "ModoChequeValidacao" );
ret = pModoChequeValid( '1' , '0' );

```

[Voltar ao Índice](#)

Funções de Impressão de Cheque e Validação de Documentos

ModoChequeValidacaoAvancoNV()

Função: **ModoChequeValidacaoAvancoNV**
Descrição: Inicializa a impressão de cheque ou validação. O ECF passará então para o estado de inserção de documento. No caso de validação: Este comando somente estará habilitado se o número máximo de autenticações **não** for excedido (1 + 4 repetições da mesma autenticação) e o registro do valor a ser validado estiver ativo (operação imediatamente anterior com valor. Ex.: pagamento, venda de item). Esta função permite inserir a quantidade de avanços de linha antes da validação (somente para ECFs duas estações), e o número do registrador não sujeito ao ICMS empregado na operação do cupom não fiscal não vinculado, para

Sintaxe C: autenticação.

```
int FAR PASCAL ModoChequeValidacaoAvancoNV(
    char tipo
    ,char Load
    ,char Avanco
    ,char *Item
);
```

Sintaxe VB: Declare Function ModoChequeValidacaoAvancoNV Lib "ECF32M.DLL" (
 ByVal tipo As Byte
 ,ByVal Load As Byte
) As Long

Sintaxe Delphi: function ModoChequeValidacaoAvancoNV(
 tipo: Char
 ;load: Char
):Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

Tipo: Seleciona modo cheque ou validação:

- '0' (30h) – modo validação.
- '1' (31h) – modo cheque (somente ECF 3002).

Load: Seleciona entrada do papel de validação (1s)
(somente ECF 3002 / ECF3001):

- '0'(30h) - carga pela entrada superior (modo default).
- '1'(31h) - carga pelo slip (inferior).

Seleciona entrada do papel de validação.
(somente Compact Fiscal):

- '0'(30h) - posiciona cabeça lado direito (modo default).
- '1'(31h) - posiciona cabeça lado esquerdo.
- '3'(33h) - posiciona cabeça no centro.

Avanço: Quantidade de avanços de linha antes da validação (1c).
 Este valor deve estar na faixa compreendida entre '0' (30h) a '9' (39h).

Item: Número do registrador não sujeito ao ICMS empregado do cupom não fiscal não vinculado (2n). Este valor deve estar na faixa compreendida entre "16" a "31".

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXEÇÃO

Exemplo: ModoChequeValidAvancoNV ('1' , '0' , '1' , "16")

Exemplo em C: typedef int (FAR PASCAL *MODOCHEQUEVALIDAVANCONV)(char ,char ,char ,char *);
 int ret;
 MODOCHEQUEVALIDAVANCONV pModoChequeValidAvancoNV;

pModoChequeValidAvancoNV = (MODOCHEQUEVALIDAVANCONV)
 GetProcAddress(h,"ModoChequeValidacaoAvancoNV");
 ret = pModoChequeValidAvancoNV ('1' , '0' , '1' , "16");

[Voltar ao Índice](#)

**Funções de
 Impressão de Cheque e
 Validação de Documentos**

ImprimeCheque()

Função: ImprimeCheque

Revisão 1.12 39 83

Descrição: Imprime o cheque de acordo com o parâmetro valor. Os campos do extenso do valor do cheque e o extenso do mês serão preenchidos automaticamente pelo ECF.

Cada linha será impressa imediatamente “após” o avanço do papel, programado através do parâmetro lx, onde:

- **avanço = avanço x 1/160 polegadas**

O primeiro avanço é calculado a partir do topo da área útil de impressão do cheque. As dimensões máximas estão especificadas no Apêndice.

Nota:

- É permitido o cancelamento da impressão de cheque, porém o mesmo deve estar posicionado para impressão.
- Caso o nome da moeda corrente não tenha sido programada pela função ProgMoeda(), Prog_Moeda() ou ProgramaMoeda() , se assumirá a string default “REAL” e “REAIS”.

Sintaxe C: int FAR PASCAL ImprimeCheque(
char l1 ,char c1
, char l2 ,char c2
, char l3 ,char c3
, char l4 ,char l5
, char c5
, char l6
, char l7
, char c8
, char *Valor
, char *Favorecido
, char *Local
, char SetAno
, char *Data
, char *Com1
, char *Com2
);

Sintaxe VB Declare Function ImprimeCheque Lib “ECF32M”.DLL (
ByVal l1 As Byte ,ByVal c1 As Byte
, ByVal l2 As Byte ,ByVal c2 As Byte
, ByVal l3 As Byte ,ByVal c3 As Byte
, ByVal l4 As Byte ,ByVal l5 As Byte
, ByVal c5 As Byte
, ByVal l6 As Byte
, ByVal l7 As Byte
, ByVal c8 As Byte
, ByVal *Valor As String
, ByVal *Favorecido As String
, ByVal *Local As String
, ByVal SetAno As Byte
, ByVal *Data As String
, ByVal *Com1 As String
, ByVal *Com2 As String
);

Sintaxe Delphi: function ImprimeCheque(
l1: Char
; c1: Char
; l2: Char
; c2: Char
; l3: Char
; c3: Char

```
;l4: Char
;l5: Char
;c5: Char
;l6: Char
;l7: Char
;c8: Char
;valor: PChar
;favorecido: PChar
;local: PChar
;setano: Char
;data: PChar
;coment1: PChar
;coment2: PChar
);Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

Ln e Cn: Com **n** podendo variar entre 1 e 8, temos:

Ln: Número de avanços de linha antes da impressão do parâmetro **n**.
Cn: Coluna de impressão do parâmetro **n**.

Onde **n** representa:

- 1 - Valor do cheque
- 2 - 1a linha do extenso do valor do cheque;
- 3 - 2a linha do extenso do valor do cheque;
- 4 - Favorecido;
- 5 - Localização e Data;
- 6 - 1a linha de comentário;
- 7 - 2a linha de comentário;
- 8 - Ano;

Lembre-se que:

- Para **n** igual a 4 (Favorecido) , 6 (1a linha de comentário) e 7 (2a linha de comentário) podemos indicar somente a linha
- Para **n** igual a 8 (Ano) podemos indicar somente a coluna.

Ex: l2,c8 o valor numérico do cheque será impresso na linha 2 e na coluna 30

Valor: Valor do cheque a ser impresso, no formato "nnnnnnnnnnnnnnnn" sendo 12 para a parte inteira e 2 decimais (14n).

Favorecido: Linha do campo do favorecido (80s).

Local: Localização (cidade) (20s).

SetAno: Parâmetro de impressão do ano (1s):

- '0' (30h) - o ano será impresso com apenas o último dígito ('8');
- '1' (31h) - o ano será impresso com dois dígitos ('98');
- '2' (32h) - o ano será impresso com quatro dígitos ('1998');

Data: Data atual (2d2m2a).

Com1: Primeira linha de comentário (80s).

Com2: Segunda linha de comentário (80s).

Retorno: CIF_OK

CIF_ERR_SERIAL

CIF_TIMEOUT

CIF_EMEXECUÇÃO

Exemplo: ImprimeCheque(0x28,0x3d, // l1 e c1: valor do cheque

```

0x34,0x0a,    // l2 e c2: 1a linha extenso
0x30,0x06,    // l3 e c3: 2a linha extenso
0x30,         // l4: favorecido
0x30,0x22,    // l5 e c5: localizacao e data
0x10,         // l6: 1a linha comentario
0x10,         // l7: 2a linha comentario
0x4c,         // c8: campo do ano

"01004501501290",    // Valor (14 dig)

"favorecido12345678901234567890123456789012345678901234567890
234567fim",    // Favorecido (80 dig)

"SAO PAULO      ",    // Cidade (20 dig)

'2',    // SetAno: Ano com 4 dígitos

"110100",    // Data (6 dígitos): "11 de Janeiro de 2000"

"comentario12345678901234567890123456789012345678901234567890
1234567fim",    // 1ª Linha de Comentário (80 dig)

"comentario12345678901234567890123456789012345678901234567890
1234567fim"    // 2ª Linha de Comentário (80 dig)

)

```

Exemplo C:

```

typedef int (FAR PASCAL *IMPCHEQUE) (unsigned char, char*);
int      ret;
char     linha[41];
IMPCHEQUE  plmprimeCheque;

plmprimeCheque = (IMPCHEQUE) GetProcAddress(h,
                                             "ImprimeCheque ");
ret=plmprimeCheque(
    0x28,0x3d,    // l1 e c1      valor do cheque
    0x34,0x0a,    // l2 e c2      1a linha extenso
    0x30,0x06,    // l3 e c3      2a linha extenso
    0x30,         // l4           favorecido

    0x30,0x22,    // l5 e c5 localizacao e data
    0x10,         // l6           1a linha comentario
    0x10,         // l7           2a linha comentario
    0x4c,         // c8           campo do ano
    "01004501501290",    // Valor (14 dig)
    "favorecido12345678901234567890123456789012345678901234567
8901234      567fim",
    "SAO PAULO      ",
    '2',
    "110100",
    "comentario12345678901234567890123456789012345678901234567890
890123      4567fim",
    "comentario12345678901234567890123456789012345678901234567890
890123      4567fim"
);

```

Funções de Impressão de Cheque e Validação de Documentos

ImprimeValidacao()

Função: **ImprimeValidacao**
Descrição: Imprime uma linha de validação, sendo a segunda linha opcional. Lembre-se:
 1) Este comando só será habilitado caso um valor tenha sido previamente registrado.
 2) Pode-se repetir até quatro autenticações para um mesmo valor, com os campos de comentário da primeira e segunda de forma livre.

Sintaxe C: Int FAR PASCAL ImprimeValidacao (char *Leg ,char *LinhaOp);

Sintaxe VB: Declare Function ImprimeValidacao Lib "ECF32M.DLL" (ByVal Leg As String ,ByVal LinhaOp As String) As Long

Sintaxe Delphi: function ImprimeValidacao(leg: Pchar ;LinhaOp: PChar):Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
Leg: Campo de comentário da 1a linha de autenticação (5s).
LinhaOp: Segunda linha de comentário (40/48s). Caso o parâmetro seja uma string nula (LinhaOp = Null), não será impressa a segunda linha opcional.

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUCAO

Exemplo: ImprimeValidacao ("PAGO ", "2a linha de autenticao livre ")
Exemplo em C: typedef int (FAR PASCAL *IMPRESSAO) (void);
 int ret;
 IMPRESSAO plmprimeValidacao;

 plmprimeValidacao = (IMPRESSAO) GetProcAddress(h, "ImprimeValidacao");
 ret = plmprimeValidacao ("PAGO ", "2a linha de autenticao livre ");

Funções de Impressão de Cheque e Validação de Documentos

CancelaChequeValidacao()

Função: **CancelaChequeValidacao**
Descrição: Cancela a impressão de cheque/validação pendente, retornando ao modo normal de operação.
Observação:
 • Para que este comando seja executado, é necessário que o documento (cheque ou validação) esteja posicionado.

Sintaxe C: int FAR PASCAL CancelaChequeValidacao(void);

DLL ECF32M Especificação de Uso

Sintaxe VB: Declare Function CancelaChequeValidacao Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function CancelaChequeValidacao(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: CancelaChequeValidacao()

Exemplo em C: typedef int (FAR PASCAL *CANCELAIMPRESSAO) (void);
int ret;
CANCELAIMPRESSAO pCancelaImpressao;
:
:
pCancelaImpressao = (CANCELAIMPRESSAO) GetProcAddress(h,
"CancelaChequeValidacao");
ret = pCancelaImpressao ();

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

Funções de Operações Não Sujeitas ao ICMS

[ProgramaLegenda\(\)](#)

Função: ProgramaLegenda

Descrição: Programa a legenda dos Registradores Não Fiscais de Formas de Pagamento e Operações Não Fiscais.

Lembre-se:

- 1) Após a programação de uma Forma de Pagamento, será impresso na Leitura X ou Redução Z:
#nn: <legenda programável> = vvvvvvvvvvvv,vv , onde:
nn: número do totalizador programado.
vv...vv,vv: valor do totalizador
- 2) Um Registrador Não Fiscal programado será impressos nos cupons de Leitura X e Redução Z somente após o uso
- 3) Para alterar um Registrador Não Fiscal de Formas de Pagamento ou de Operações Não Fiscais, é necessário colocar o ECF em Intervenção Técnica.
- 4) Para incluir um Registrador Não Fiscal de Formas de Pagamento ou de Operações Não Fiscais, NÃO é necessário colocar o ECF em Intervenção Técnica.

Sintaxe C: int FAR PASCAL ProgramaLegenda (
char *reg
,char *leg
);

Sintaxe VB: Declare Function ProgramaLegenda Lib "ECF32M.DLL" (
ByVal reg As String
,ByVal leg As String
) As Long

Sintaxe Delphi: function ProgramaLegenda (
reg: PChar
;leg: PChar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

reg: Identificação do registrador não fiscal a ser programado. Composto de 2 caracteres podendo variar entre "00" e "31" (2n).

Indique "**reg**" variando entre:

- "00" a "15" para programar uma Forma de Pagamento (Dinheiro, Cheque, Cartão, ...)
- "16" a "31" para programar uma Operação Não Fiscal (Reforço de Caixa, Sangria, ...).

leg: Legenda do respectivo registrador não fiscal, composto de 16 caracteres no formato "cccccccccccccccc". (16c)

Retorno:

CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT

Exemplo:

Exemplo em C:

```
CIF_EMEXECUÇÃO
ProgLegenda("16","VALE FUNCIONARIO")
typedef int (FAR PASCAL *PRGLEG) (char *,char *);
int          ret;
char  reg[3],leg[17];
PRGLEG      pProgLegenda;
:
:
pProgLegenda = (PRGLEG) GetProcAddress(h, "ProgramaLegenda");

strcpy (reg,"00");
strcpy (leg,"AAAAAAAAAAAAAAAA");
ret = pProgLegenda ( reg,leg);
```

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

AbreCupomVinculado()

Função:

AbreCupomVinculado

Descrição:

Pode-se abrir um Comprovante Não Fiscal Vinculado a:

- Cupom de Fiscal de Venda
- Comprovante Não Fiscal Não Vinculado

Lembre-se:

- 1) A impressão é livre e limitada a **2 minutos**. A cada 10 linhas será impresso a frase "NÃO É DOCUMENTO FISCAL".
- 2) Um Comprovante Não Fiscal Vinculado só será habilitado se requisitado imediatamente após a emissão dos mesmos.
- 3) Um Comprovante Não Fiscal Vinculado poderá ser utilizado para emissão de comprovante de TEF (Transação Eletrônica de Fundos), emissão de prestações, etc.

Sintaxe C:

int FAR PASCAL AbreCupomVinculado(void);

Sintaxe VB:

Declare Function AbreCupomVinculado Lib "ECF32M.DLL" () As Long

Sintaxe Delphi:

function AbreCupomVinculado(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

Nenhum.

Retorno:

CIF_OK
 CIF_ERR_SERIAL

Exemplo:

Exemplo em C:

```
CIF_TIMEOUT
CIF_EMEXECUÇÃO
AbreCupomVinculado()
typedef int (FAR PASCAL *ABRECUPV) (void);
int      ret;
ABRECUPV  pAbreCupomVinculado;
:
:
pAbreCupomVinculado = (ABRECUPV) GetProcAddress(h, "AbreCupomVinculado");
ret = pAbreCupomVinculado ( );
```

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

AbreCupomNaoVinculado()

Função:

AbreCupomNaoVinculado

Descrição:

Abre um Cupom Não Fiscal Não Vinculado. Neste cupom é possível realizar operações não fiscais em registradores através da função **OperRegNaoVinculado()**. Este cupom habilita a emissão do comprovante não fiscal VINCULADO, imediatamente após o fechamento deste.

Sintaxe C:

```
int FAR PASCAL AbreCupomNaoVinculado (void);
```

Sintaxe VB:

```
Declare Function AbreCupomNaoVinculado Lib "ECF32M.DLL" ( ) As Long
```

Sintaxe Delphi:

```
function AbreCupomNaoVinculado(): Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

Nenhum.

Retorno:

```
CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO
```

Exemplo:

Exemplo em C:

```
AbreCupomNaoVinculado()
typedef int (FAR PASCAL *ABRECUPNV) (void);
int      ret;
ABRECUPNV  pAbreCupomNaoVinculado;
```

```
pAbreCupomVinculado = (ABRECUPNV) GetProcAddress(h,
"AbreCupomNaoVinculado");
ret = pAbreCupomNaoVinculado ( );
```

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

OperRegNaoVinculado()

Função:

OperRegNaoVinculado

Descrição:

Realiza a operação nos registradores não vinculados (programados na posições de 16 a 31).

É permitido realizar o pagamento da operação através da função Pagamento(), ou PagamentoComTexto(), que caracteriza a finalização da operação nos registradores não fiscais, sendo então somente permitido o fechamento do comprovante não fiscal não vinculado.

Lembre-se:

1) Para saber quais registradores estão cadastrados no ECF, use a função

ECFPar() passando como parâmetro 66 a 81, Exemplo:
 ECFPar("66"): irá transmitir as informações do registrador 16, como a legenda, o valor e um sinal + (valor positivo) ou – (valor negativo)

Sintaxe C:

```
int FAR PASCAL OperRegNaoVinculado (
    char *reg
    ,char *valor
    ,char oper
    ,char toper
    ,char *valorop
    ,char *legop
);
```

Sintaxe VB:

```
Declare Function OperRegNaoVinculado Lib "ECF32M.DLL" (
    ByVal reg As String
    ,ByVal valor As String
    ,ByVal oper As Byte
    ,ByVal toper As Byte
    ,ByVal valorop As String
    ,Byval legop As String
) As Long
```

Sintaxe Delphi:

```
function OperRegNaoVinculadoA (
    reg: Pchar
    ;valor: Pchar
    ;oper: Char
    ;toper: Char
    ;valorOp: Pchar
    ;legOp: Pchar
): Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

reg: Identificação do Registrador Não Fiscal, podendo variar entre "16" e "31". Composto de 2 caracteres no formato nn (2n).

valor: Valor referente à operação. Composto de 15 caracteres no formato nn...nn (13 para a parte inteira e 2 para a parte decimal) (15n). Lembre-se que, os dígitos não significativos deverão ser preenchidos com zero ('0').

oper: Indica se o próximo campo é operação de desconto ou acréscimo. Composto de 1 caracter, no seguinte formato:

- '@' (40h): Acréscimo.
- 'Z' (5Ah) ou '' (00h – Nulo): Desconto.

toper: Tipo do desconto/acréscimo a ser enviado no próximo campo, no seguinte formato:

- '&' (26h) ou '1' (31h) - por valor.
- '%" (25h) ou '0' (30h) - por porcentagem.

valorop: Valor do desconto/acréscimo

- Por Valor: composto de 15 caracteres no formato nn...nn (13 para a parte inteira e 2 para a parte decimal) (15n)
- Por porcentagem: composto de 4 caracteres no formato nnnn (2 para a parte inteira e 2 para a parte decimal).(4n)

legop: Legenda opcional, impressa somente em caso de desconto/acréscimo válido, composto de 14 caracteres no formato ss...ss. (14n). Caso o parâmetro seja uma string nula (legop = Null), será impressa a legenda default: "DESCONTO" ou "ACRÉSCIMO".

Retorno:

```
CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUCAO
```

Exemplo: OpRegNaoVinculado ("16", "000000000100000", '@', '%', , "1000", "");

Exemplo em C:

```
typedef int (FAR PASCAL *OPTOT) (char *, char *, char, char, char *, char
*);
int ret;
char op, toper, reg[3], valor[16], valorop[16];
OPTOT pOpRegNaoVinculado;
:
:
pOpRegNaoVinculado = (OPTOT) GetProcAddress(h, "OperRegNaoVinculado");

strcpy (reg, "16");
op = '@' ;
toper = '%';
strcpy (valor, "000000000100000");
strcpy (valorop, "1000");
ret = pOpRegNaoVinculado ( reg, valor, op, toper, valorop, "");
```

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

ImprimeLinhaNaoFiscal()

Função: **ImprimeLinhaNaoFiscal**

Descrição: Imprime uma linha não fiscal com avanço de linha nos cupons de Relatório Gerencial e Comprovante Não Fiscal Vinculado. Ao contrário da função ImprimeNaoFiscal(), esta função retorna a resposta ao ser executado com sucesso ou não.

Sintaxe C: Int FAR PASCAL ImprimeLinhaNaoFiscal (char par ,char *str);

Sintaxe VB: Declare Function ImprimeLinhaNaoFiscal Lib "ECF32M.DLL" (ByVal par As Byte ,ByVal str As String) As Long

Sintaxe Delphi: function ImprimeLinhaNaoFiscal(par: Char ;msg: PChar): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

par: Atributo da linha a ser impressa:

- '0' (30h): Impressão Modo Normal.
- '1' (31h): Impressão Modo Expandida.

str: Linha a ser impressa no formato "ss..ss":

- Impressão Modo Expandida: 20 ou 24 caracteres (20/24s)
- Impressão Modo Normal: 40 ou 48 caracteres (40/48s)

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: Poderá ser usado:

1) Impressão Modo Normal com 40 caracteres:

ImprimeLinhaNaoFiscal('0', "1234567890123456789012345678901234567890")

- 2) Impressão Modo Normal com 48 caracteres:

```
ImprimeLinhaNaoFiscal('0','123456789012345678901234567890123456789012345678')
```

- 3) Impressão Modo Expandido com 20 caracteres:

```
ImprimeLinhaNaoFiscal('1','12345678901234567890')
```

- 4) Impressão Modo Expandido com 24 caracteres:

```
ImprimeLinhaNaoFiscal('1','123456789012345678901234')
```

```
Exemplo em C: typedef int (FAR PASCAL *IMPLIN) (char ,char *);
int      ret;
char      par, BufStr[49];
IMPLIN    plmprimeLinhaNaoFiscal:
```

```
pImprimeLinhaNaoFiscal = (PRGLEG) GetProcAddress(h, "ImprimeLinhaNaoFiscal ");
```

```
par = '1';
strcpy (BufStr"12345678901234567890");
ret = plmprimeLinhaNaoFiscal ( par.BufStr):
```

Voltar ao Índice

Funções de Operações Não Sujeitas ao ICMS

```
ImprimeLinhaNaoFiscalTexto()
```

Função: **ImprimeLinhaNaoFiscalTexto**

Descrição:

Imprime uma linha não fiscal com avanço de linha nos cupons de Relatório Gerencial e Comprovante Não Fiscal Vinculado. Esta função permite a passagem de uma string de tamanho máximo igual a 999.

Ao contrário da função `ImprimeNaoFiscal()`, esta função retorna a resposta ao ser executado com sucesso ou não.

Sintaxe C:

```
Int FAR PASCAL ImprimeLinhaNaoFiscalTexto (
    char *par
    ,char *str
);
```

Sintaxe VB:

```

7) Declare Function ImprimeLinhaNaoFiscalTexto Lib "ECF32M.DLL" (
    ByVal par As String
    ,ByVal str As String
) As Long

```

Sintaxe Delphi:

```
function ImprimeLinhaNaoFiscalTexto (
    par: Pchar
    ;str: Pchar
); Integer; stdcall; external 'ECF32M.dll';
```

Parâmetros:

par: Atributo da linha a ser impressa:

- '0' (30h) – Impressão Modo Normal
- '1' (31h) – Impressão Modo Expandido
- 'Snnn' (53h) – Tamanho do string (máximo de 999 caracteres). Por exemplo: "S011".

str: Linha a ser impressa no formato "ss..ss":

- Impressão Modo Expandida: 20 ou 24 caracteres (20/24s)
- Impressão Modo Normal: 40 ou 48 caracteres (40/48s)

- String normal a ser impresso. Este parâmetro só é válido caso o valor de "par" seja "Snnn".

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: ImprimeLinhaNaoFiscalTexto("S020","12345678901234567890")

Exemplo em C:

```
typedef int (FAR PASCAL *IMPLINTEXTO ( char*,char *);
int          et;
char          par[5], BufStr[49];
IMPLINTEXTO  ImprimeLinhaNaoFiscalTexto;
:
:
pImprimeLinhaNaoFiscalTexto = (PRGLEGTEXT0) GetProcAddress(h,
"ImprimeLinhaNaoFiscalTexto");

strcpy( par ,"S020" );
strcpy(BufStr ,"12345678901234567890" );
ret = pImprimeLinhaNaoFiscalTexto( par ,BufStr );
```

Observações:

- Função disponível a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Funções de Operações Não Sujeitas ao ICMS

EncerraCupomNaoFiscal()

Função: EncerraCupomNaoFiscal

Descrição: Encerra os seguintes cupons de operações não fiscais:

- Leitura X com Relatório Gerencial;
- Cupom Não Fiscal Vinculado e
- Cupom Não Fiscal Não Vinculado.

Sintaxe C: Int FAR PASCAL EncerraCupomNaoFiscal (void);

Sintaxe VB: Declare Function EncerraCupomNaoFiscal Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function EncerraCupomNaoFiscal(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO

Exemplo: EncerraCupomNaoFiscal()

Exemplo em C:

```
typedef int (FAR PASCAL *ENCCUPOM) (void);
int          ret;
ENCCUPOM    pEncerraCupom;

pEncerraCupom = (ENCCUPOM) GetProcAddress(h, "EncerraCupomNaoFiscal");
ret = pEncerraCupom ();
```

[Voltar ao Índice](#)

Funções de

CancelaCupomNaoFiscal()

**Operações Não
 Sujeitas ao ICMS**

Função: **CancelaCupomNaoFiscal**
Descrição: Cancela um comprovante não fiscal vinculado ou um comprovante não fiscal não vinculado.
Sintaxe C: int FAR PASCAL CancelaCupomNaoFiscal(void);
Sintaxe VB: Declare Function CancelaCupomNaoFiscal Lib "ECF32M.DLL" () As Long
Sintaxe Delphi: function CancelaCupomNaoFiscal(): Integer; stdcall; external 'ECF32M.dll';
Parâmetros: Nenhum.
Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO
Exemplo: CancelaCupomNaoFiscal ()
Exemplo em C: typedefint (FAR PASCAL *CANCUPNF) (void);
 int ret;
 CANCECUPNF pCancelaCupomNaoFiscal;

 pCancelaCupomNaoFiscal = (CANCECUPNF)GetProcAddress(h
 ,"CancelaCupomNaoFiscal");
 ret = pCancelaCupomNaoFiscal ();

[Voltar ao
 Índice](#)

Funções Diversas

Funções Diversas **[AcionarGaveta\(\)](#)**

Função: **AcionarGaveta**
Descrição: Gera um pulso para acionar a gaveta.
Sintaxe C: Int FAR PASCAL AcionarrGaveta (void);
Sintaxe VB: Declare Function AcionarGaveta Lib "ECF32M.DLL" () As Long
Sintaxe Delphi: function AcionarGaveta (): Integer; stdcall; external 'ECF32M.dll';
Parâmetros: Nenhum.
Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO
Exemplo: AcionarGaveta ()
Exemplo em C: typedef int (FAR PASCAL *GAVETA) ();
 int ret;
 GAVETA pAbrirGaveta;

 pAbrirGaveta = (GAVETA) GetProcAddress(h, "AcionarGaveta");
 ret = pAcionarGaveta ();

[Voltar ao Índice](#)

Funções Diversas **[ProgramaHorarioVerao\(\)](#)**

Função: **ProgramaHorarioVerao**
Descrição: Controla a programação do horário de verão, adiantando ou atrasando uma hora.
Sintaxe C: int FAR PASCAL ProgramaHorarioVerao (char hv);
Sintaxe VB: Declare Function ProgramaHorarioVerao Lib "ECF32M.DLL" (

Sintaxe Delphi: ByVal hv As Byte
) As Long
function ProgramaHorarioVerao(
 hv: Char
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
hv: Controle do horário de verão, composto de 1 caracter no seguinte formato:

- '+' (2Bh) - Entra no horário de verão (adianta uma hora){1g}.
- '-' (2Dh) - Sai do horário de verão (atrasa uma hora){1g}.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ProgramaHorarioVerao('+')

Exemplo em C: typedef int (FAR PASCAL *PROGHV) (char);

```
PROGHV    pProgramaHv;
int       ret;

pProgramaHv = (PROGHV) GetProcAddress(h, "ProgramaHorarioVerao");
ret = pProgramaHv ( 0x2B);
```

[Voltar ao Índice](#)

Funções Diversas *ProgramaHorarioVeraoStr()*

Função: ProgramaHorarioVeraoStr

Descrição: Controla a programação do horário de verão, adiantando ou atrasando uma hora.

Sintaxe C: int FAR PASCAL ProgramaHorarioVeraoStr (char *hv);

Sintaxe VB: Declare Function ProgramaHorarioVeraoStr Lib "ECF32M.DLL" (
 ByVal hv As String
) As Long

Sintaxe Delphi: function ProgramaHorarioVeraoStr(
 hv: pChar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
hv: Controle do horário de verão, composto de 1 caracter no seguinte formato:

- "+" (2Bh) - Entra no horário de verão (adianta uma hora){1g}.
- "-" (2Dh) - Sai do horário de verão (atrasa uma hora){1g}.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: 1) ProgramaHorarioVeraoStr("+")
2) ProgramaHorarioVeraoStr("-")

Exemplo em C: typedef int (FAR PASCAL *PROGHV) (char);

```
PROGHV    pProgramaHv;
int       ret;

pProgramaHv = (PROGHV) GetProcAddress(h, "ProgramaHorarioVeraoStr");
```

ret = pProgramaHv ("+");

[Voltar ao
Índice](#)

Funções Diversas *ImprimeTotalizadores()*

Função: **ImprimeTotalizadores**

Descrição: Imprime uma linha contendo as informações do registrador de operações não sujeitas ao ICMS especificado, no seguinte formato:
#nn: <legenda programável> = vvvvvvvvvvvv,vv , onde:
nn: número do totalizador programado.
vv...vv,vv: valor do totalizador

Este comando também é aceito no cupom de operações não sujeitas ao ICMS.

Sintaxe C: Int FAR PASCAL ImprimeTotalizadores (char *reg);

Sintaxe VB: Declare Function ImprimeTotalizadores Lib "ECF32M.DLL" (ByVal reg As String) As Long

Sintaxe Delphi: function ImprimeTotalizadores(reg: PChar): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
reg: Identificação do registrador não fiscal, composto de 2 caracteres no formato nn {2n("00" =< nn <= "15")}

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ImpTotalizadores ("01");
Exemplo em C: typedef int (FAR PASCAL *IMPTOT) (char *);

```
IMPTOT      plmpTotalizadores;
int          ret;
```

```
plmpTotalizadores = (IMPTOT) GetProcAddress(h, "ImprimeTotalizadores");
ret = plmpTotalizadores ( "01");
```

[Voltar ao
Índice](#)

Funções Diversas *TransTabAliquotas()*

Função: **TransTabAliquotas**

Descrição: Transmissão da tabela de alíquotas. A tabela de alíquotas será enviada da seguinte maneira: "trib0 trib1 trib2 ... trib15", onde:

- tribn: Corresponde às alíquotas efetivas programadas nos totalizadores parciais T00 a T15, composto de 4 caracteres numéricos por alíquota, no formato nnnn, sendo 2 para a parte inteira e 2 para a parte decimal.(4n)
- A string "0000" será enviada caso o respectivo totalizador não tenha sido programado.

DLL ECF32M

Especificação de Uso

Sintaxe C: int FAR PASCAL TransTabAliquotas (void);

Sintaxe VB: Declare Function TransTabAliquotas Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function TransTabAliquotas(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: TransTabAliquotas()

Exemplo em C: typedef int (FAR PASCAL *TABALIQ) (void);

```
TABALIQ      pTabAliq;
int          ret;

pTabAliq = (TABALIQ) GetProcAddress(h, "TransTabAliquotas");
ret = pTabAliq ();
```

[Voltar ao Índice](#)

Funções Diversas *TransTabAliquotasCasas()*

Função: **TransTabAliquotasCasas**

Descrição: Transmissão da tabela de alíquotas. Esta transmissão ocorrerá conforme o formato determinado através do parâmetro "Len". Esta tabela será enviada da seguinte maneira: "trib0 trib1 trib2 ... trib15", onde:

- tribn: Corresponde às alíquotas efetivas programadas nos totalizadores parciais T00 a T15. Será composto por 4 ou 6 caracteres conforme o parâmetro Len.
- A string "0000", ou "000000" será enviada caso o respectivo totalizador não tenha sido programado.
- Função disponível a partir da versão 2.1 desta DLL, e versão FCP-500 do firmware da impressora fiscal.

Sintaxe C: int FAR PASCAL TransTabAliquotasCasas (char Len);

Sintaxe VB: Declare Function TransTabAliquotasCasas Lib "ECF32M.DLL" (ByVal Len As Byte) As Long

Sintaxe Delphi: function TransTabAliquotasCasas (Len: Char): Integer;

Parâmetros:

Len: Determina o número de caracteres que serão utilizados na transmissão dos valores das alíquotas:

- '1' (31h): O valor será transmitido com 6 caracteres, sendo 2 caracteres para parte inteira e 4 caracteres para parte decimal, no formato "nnnnnn". (6n)
- '0' (30h): O valor será transmitido com 4 caracteres, sendo 2 caracteres para parte inteira e 2 caracteres para parte decimal, no formato "nnnn". (4n)

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: TabAliqCasas('1')

Exemplo em C: typedef int (FAR PASCAL *TABALIQCASAS) (char);

TABALIQCASAS
int

pTabAliqCasas;
ret;

pTabAliqCasas = (TABALIQCASAS) GetProcAddress(h, "TransTabAliquotasCasas");
ret = pTabAliqCasas('1');

[Voltar ao
Índice](#)

Funções Diversas **TransTotCont()**

Função:
Descrição:

TransTotCont

Transmissão de totalizadores e contadores. Os totalizadores e contadores serão enviados da seguinte maneira através da função EsperaResposta():
"nseq nop gt tot00 tot01 ... tot15 f i n desc can acres c_nsi c_can c_reinic c_red", onde:

- nseq: Número sequencial do ECF atribuído pelo estabelecimento {6n}.
- nop: Número de ordem de operação {6n}.
- gt: Totalizador Geral {19n}.
- tot xx: Totalizador Parcial de Situação Tributária. Serão enviados os totalizadores `T00' até `T15' {15n,...15n}.
- f: Totalizador Parcial de Substituição Tributária {15n},
- i: Totalizador Parcial de Isenção {15n}.
- n: Totalizador Parcial de Não Incidência {15n},
- desc: Totalizador Parcial de Descontos {15n}.
- can: Totalizador Parcial de Cancelamentos {15n}.
- c_nsi: Contador de operações não sujeitas ao ICMS {4n}.
- c_can: Contador de Cancelamentos {4n}.
- c_reinic: Contador de Reinício de Operação {4n}.
- c_red: Contador de Reduções {4n}.

Sintaxe C:

int FAR PASCAL TransTotCont (void);

Sintaxe VB:

Declare Function TransTotCont Lib "ECF32M.DLL" () As Long

Sintaxe Delphi:

function TransTotCont(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

Nenhum.

Retorno:

CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo:

TransTotCont ()

Exemplo em C:

typedef int (FAR PASCAL *TXTOTCONT) (void);

TXTOTCONT pTxTotCont;
int ret;

pTxTotCont = (TXTOTCONT) GetProcAddress(h , "TransTotCont");
ret = pTxTotCont ();

[Voltar ao
Índice](#)

Funções Diversas **TransStatus()**

Função: TransStatus

Descrição: Transmissão de Status.
 Através desta função é possível identificar o estado atual da ECF (Em Cupom Fiscal, Em Cupom Não Fiscal e etc), o estado dos sensores (tampa aberta, pouco papel e etc) e verificar possíveis estados de erro (erro mecânico, erro irreversível e etc). É possível testar um único bit de status ou receber todos os bits em um buffer. Os bits de status são numerados de 1 a 40. Para testar um bit qualquer, basta chamar a função passando o número desse bit. A função retornará o estado desse bit (1 = ligado; 0 = desligado). Se a função for chamada e o número do bit for zero, todos os bits serão retornados em um buffer, contendo '1' (31h) se estiver ligado ou '0' (30h) se estiver desligado. É importante salientar que para qualquer uma das maneiras de obter o estado dos bits, sempre é necessário passar, como parâmetro, um buffer com o tamanho mínimo de 41 posições.

Sintaxe C: Int FAR PASCAL TransStatus (
 int BitTest
 , char *BufStat
);

Sintaxe VB: Declare Function TransStatus Lib "ECF32M.DLL" (
 ByVal BitTest As Long
 , ByVal BufStat As String
) As Long

Sintaxe Delphi: Function TransStatus(
 bittest: Integer
 ; bufStat: PChar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

BitTest: Número do bit a ser testado entre 1 e 40.
 Se BitTest = 0, a função retorna todos os bits de status em BufStat.

BufStat: Buffer de retorno de todos os bits de Status (41 posições).
 BufStat deve ser sempre passado como parâmetro, mesmo se não for pedido o retorno de todos os bits de status.

Retorno: **Se BitTest > 0**, esta função retorna o estado do bit de status solicitado:
 0 (0h) = Desligado
 1 (1h) = Ligado

Se BitTest = 0, os bits serão passados em BufStat e o retorno será CIF_OK.
 Se ocorrer perda de comunicação com a impressora, a função retornará CIF_TIMEOUT ou CIF_ERR_SERIAL.

Os significados dos bits de status estão descritos abaixo:

Bit	Estado do Bit	Descrição
1	1	Cancelamento habilitado.
2	1	Em horário de verão.
4	1	Em impressão fiscal.
3	1	Redução Z pendente.
5	1	Em intervenção fiscal.
6	1	Redução Z do dia feita.
7	1	Cupom não fiscal aberto.
8	1	Cupom fiscal aberto.
9	1	Em processo de Leitura da Memória Fiscal.
10	1	Em processo de Leitura X.
11	1	Em processo de Redução Z.
12	1	Em processo de fechamento do cupom.
13	1	Em processo de cancelamento do cupom.
14	1	Em processo de cancelamento de item.
15	1	Em processo de venda de item.

16	1	Em início de cupom de venda.
17	1	Comando em execução.
18	1	Impressora com Pouco Papel.
19	1	Impressora apresenta Temperatura Alta na cabeça de impressão.
20	1	Impressora apresenta Erro Mecânico.
21	1	Impressora apresenta Erro Irrecuperável.
22	1	Impressora com Buffer de recepção cheio.
23	X	Estado da Gaveta de Valores. (Este estado depende do modelo da gaveta utilizada)
24	1	Impressora com Tampa Aberta.
25	1	Leitura X pendente.
26	1	Documento para validação foi detectado.
27	1	Impressora está esperando inserção do documento a ser validado.
28	1	Modo Validação selecionado.
29	1	Impressora apresenta Fim de Papel.
30	1	Cheque detectado.
31	1	Impressora está esperando inserção do cheque.
32	1	Modo Cheque selecionado.
33	1	Processo de Pagamento completado.
34	1	Processo de Troco realizado.
35	1	Processo de Pagamento iniciado.
36	1	Totalização do cupom realizada.
37	1	Totalização Parcial do cupom realizada.
38	1	Arredondamento ativado.
39	1	Em processo de pagamento.
40	1	Em processo de totalização do cupom.

Exemplo:
Exemplo em C:

```
TransStatus(22,Buffer)

typedef int (FAR PASCAL *TXSTAT) (int,char*);
:
:
TXTSTAT      pTxStatus;
int          ret,BitTest;
char  BufStat[41]

// Obtem o endereço da função na DLL
pTxStatus = (TXSTAT) GetProcAddress(h, "TransStatus");

// Pede o estado de todos os bits de Status

ret = pTxStatus (0,BufStat);
if (ret == CIF_OK)
{
    // Verifica se a Tampa está aberta
    if ( BufStat[24] == '1' )
        return TAMPA_ABERTA;

    // Verifica se tem cupom fiscal aberto
    if ( BufStat[8] == '1' )
    {
        FechaCupom();
        return 0;
    }
}

// Verifica se a totalização foi realizada
if (pTxStatus (36,BufStat))
{
    return TOTALIZA_OK;
```

}

Observações:

- É importante salientar que para qualquer uma das maneiras de obter o estado dos bits, sempre é necessário passar, como parâmetro, um buffer com o tamanho mínimo de 41 posições.

[Voltar ao Índice](#)

Funções Diversas [TransDataHora\(\)](#)

Função: TransDataHora

Descrição: Transmissão de data e hora.

A resposta será enviada da seguinte maneira: "*data-hora*", onde:

- *data*: Data atual - Composto de 8 caracteres no formato dd/mm/aa{2d2m2a}.
- *hora*: Hora Atual - Composto de 8 caracteres no formato hh:mm:ss{2h2m2s}.

Por exemplo: "25/07/01-11:48:40"

Sintaxe C: int FAR PASCAL TransDataHora (void);

Sintaxe VB: Declare Function TransDataHora Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function TransDataHora(): Integer; stdcall; external 'ECF32M.dll';

Parâmetros: Nenhum.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: TransDataHora()

Exemplo em C: typedef int (FAR PASCAL *TXDATAHORA) (void);

TXDATAHORA pTxDataHora;

int ret;

pTxDataHora = (TXDATAHORA) GetProcAddress(h, "TransDataHora");
ret = pTxDataHora ();

[Voltar ao Índice](#)

Funções Diversas [TransMemFiscalData\(\)](#)

Função: TransMemFiscalData

Descrição: Transmite, através da porta serial, um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por data da redução. Estes dados são gravados em um arquivo, no diretório corrente, denominado LMF_Datas.TXT.

Sintaxe C: Int FAR PASCAL TransMemFiscalData(
char *datai
,char *dataf
);

Sintaxe VB: Declare Function TransMemFiscalData Lib "ECF32M.DLL" (
ByVal datai As String

Sintaxe Delphi: `,ByVal dataf As String
As Long
Function TransMemFiscalData (
 datai: Pchar
 ; dataf: Pchar
): Integer; stdcall; external 'ECF32M.dll';`

Parâmetros:
 datai: Data inicial, composto de 6 caracteres no formato ddmmaa (2d2m2a).
 dataf: Data final, composto de 6 caracteres no formato ddmmaa (2d2m2a).

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEEXECUÇÃO

Exemplo: TransMemFiscalData("020902","020803")

Exemplo em C: `typedef int (FAR PASCAL TRANSMEMFISCALDATA) (char *, char*);
int ret;
char datai[7], dataf[7];
TRANSMEMFISCALDATA pTransMemFiscalData;
:
:
pLeMemFiscalData = (TRANSMEMFISCALDATA) GetProcAddress(h,
"TransMemFiscalData");
strcpy (datai, "010297");
strcpy (dataf, "020297");
ret = pTransMemFiscalData (datai, dataf);`

Observações:

- Esta função foi incluída a partir da versão 2.4 desta biblioteca.

[Voltar ao Índice](#)

Funções Diversas ***TransMemFiscalDataArqO***

Função: TransMemFiscalDataArq

Descrição: Transmite, através da porta serial, um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por data da redução. Estes dados são gravados em um arquivo cujo nome, e localização, são fornecidos pelo usuário.

Sintaxe C: `int FAR PASCAL TransMemFiscxalDataArq(
 char *datai
 , char *dataf
 , char *pathNomeArq
);`

Sintaxe VB: `Declare Function TransMemFiscalDataArq Lib "ECF32M.DLL" (
 ByVal datai As String
 , ByVal dataf As String
 , ByVal PathNomeArq As String
As Long`

Sintaxe Delphi: `Function TransMemFiscalDataArq (
 datai: Pchar
 ; dataf: Pchar
 ; PathNomeArq: Pchar
): Integer; stdcall; external 'ECF32M.dll';`

Parâmetros:
 datai: Data inicial, composto de 6 caracteres no formato ddmmaa (2d2m2a).
 dataf: Data final, composto de 6 caracteres no formato ddmmaa (2d2m2a).
 PathNomeArq: Localização e nome do arquivo. Por exemplo:
 "C:\MEU_LOCAL\LMFD.TXT"

Retorno: CIF_OK

Exemplo:
Exemplo em C:

```

CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO
TransMemFiscalArq ("010297","011297","C:\\TEMP\\LMFD.TXT")
typedef int (FAR PASCAL TRANSMEMFISCALDATAARQ) (char *, char*,char* );
int      ret;
char
TRANSMEMFISCALDATAARQ      datai[7],dataf[7];
                             pTransMemFiscalDataArq;

pLeMemFiscalDataArq = (TRANSMEMFISCALDATAARQ) GetProcAddress(h,
"TransMemFiscalDataArq") ;
strcpy (datai,"010297");
strcpy (dataf,"020297");
ret = pTransMemFiscalArq (datai ,dataf ,"C:\\TEMP\\LMFD.TXT" );

```

[Voltar ao Índice](#)

**Funções de
Operações Fiscais**

TransMemFiscaRed()

Função: **TransMemFiscalRed**
Descrição: Transmite um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por número da redução. Estes dados são gravados em um arquivo, no diretório corrente, denominado LMF_Reducoes.TXT.

Sintaxe C: Int FAR PASCAL TransMemFiscalRed(
char *redi
, char *redf
);

Sintaxe VB: Declare Function TransMemFiscalRed Lib "ECF32M.DLL" (
ByVal redi As String
, ByVal redf As String
) As Long

Sintaxe Delphi: Function TransMemFiscalRed (
redi: Pchar
; redf: Pchar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:
Redi: Redução inicial, composto de 4 caracteres no formato nnnn (4n).
Redf: Redução final, composto de 4 caracteres no formato nnnn (4n).

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo:
Exemplo em C:

```

TransMemFiscaRed("0001","0099")
typedef int (FAR PASCAL *TRANSMEMFISCALRED) (char *, char*);
int      ret;
char
TRANSMEMFISCALRED      redi[5],redf[5];
                             pTransMemFiscalRed;

pTranMemFiscalRed = (TRANSMEMFISCALRED) GetProcAddress( h
,"TransMemFiscalRed" );
strcpy (redi,"0010");
strcpy (redf,"0020");
ret = pTransMemFiscalRed( redi ,redf );

```

Observações:

- Esta função foi incluída a partir da versão 2.4 desta biblioteca.

Funções de Operações Fiscais

TransMemFiscalRedArq()

Função:	TransMemFiscalRedArq
Descrição:	Transmite um relatório das reduções armazenadas na memória fiscal no período relativo à leitura solicitada por número da redução. Estes dados são gravados em um arquivo cujo nome, e localização, são fornecidos pelo usuário.
Sintaxe C:	<pre>Int FAR PASCAL TransMemFiscalRedArq(char *redi ,char *redf ,char *pathNomeArq);</pre>
Sintaxe VB:	<pre>Declare Function TransMemFiscalRedArq Lib "ECF32M.DLL" (ByVal redi As String ,ByVal redf As String ,ByVal PathNomeArq As String) As Long</pre>
Sintaxe Delphi:	<pre>Function TransMemFiscalRedArq (redi: Pchar ;redf: Pchar ;PathNomeArq: Pchar): Integer; stdcall; external 'ECF32M.dll';</pre>
Parâmetros:	
Redi:	Redução inicial, composto de 4 caracteres no formato nnnn (4n).
Redf:	Redução final, composto de 4 caracteres no formato nnnn (4n).
PathNomeArq:	Localização e nome do arquivo. Por exemplo: "C:\MEU_LOCAL\LMFR.TXT"
Retorno:	<pre>CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEXECUÇÃO</pre>
Exemplo:	TransMemFiscalRedArq("0010", "0020", "C:\MEU_LOCAL\LMFR.TXT")
Exemplo C:	<pre>typedef int (FAR PASCAL *TRANSMEMFISCALREDARQ) (char *, char*,char *); int ret; char redi[5],redf[5]; TRANSMEMFISCALREDARQ pTransMemFiscalRedArq; pTranMemFiscalRedArq = (TRANSMEMFISCALREDARQ) GetProcAddress(h ,"TransMemFiscalRedArq"); strcpy (redi,"0010"); strcpy (redf,"0020"); ret = pTransMemFiscalRedArq(redi ,redf , "C:\TEMP\LMFR.TXT");</pre>

Observações:

- Esta função foi incluída a partir da versão 2.4 desta biblioteca.

Funções Diversas

EcfPar()

Função:	EcfPar
Descrição:	Permite a leitura de contadores, totalizadores, registradores etc... diretamente da base fiscal. Os parâmetros do ECF serão enviados pela interface serial, obedecendo à sintaxe das mensagens de retorno. Caso o

parâmetro **"par"** seja uma string nula, será enviada uma sequência completa de dados. Caso contrário, será enviado um dado específico indicado no parâmetro.

Sintaxe C: Int FAR PASCAL EcfPar (char *Par);
Sintaxe VB: Declare Function EcfPar Lib "ECF32M.DLL"(
 ByVal Par As String
) As Long

Sintaxe Delphi: function EcfPar(
 par: PChar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

- par:** Se o parâmetro "" (Null): A resposta será enviada com o seguinte formato:
- Número de série, formado por 10 caracteres numéricos no formato nn...nn{10s}.
 - Tabela de alíquotas no formato "trib0 trib1 trib2...trib15", onde:
 - tribn – corresponde as alíquotas efetivas programadas nos totalizadores parciais T00 a T15, composto de 4 caracteres numéricos por alíquota, no formato nnnn, sendo 2 para a parte inteira e 2 para parte decimal{4n}.
 - Legendas e valores dos totalizadores não fiscais no formato "leg00 val00 sg00 leg01 val01 sg01 ... leg15 val 31 sg31", onde :
 - legXX – corresponde à legenda programada ao registrador XX pela função ProgramaLegenda(),composto de 16 caracteres alfabéticos no formato ss...ss {16s}.
 - valXX – corresponde ao valor do registrador XX, composto de 15 caracteres numéricos, no formato nn...nn.{15n}.
 - sgXX – corresponde ao sinal do valor existente no registrador XX, composto de 1caracter alfabético no formato g ("+" ou "-") {1g}.
 - Linha adicional de cupom, composto de 40 caracteres alfanumérico no formato ss...ss.{40/48s}
 - Versão fiscal, composto de 7 caracteres alfanumérico no formato ss...ss.{7s}

Número sequencial do caixa, composto de 6 caracteres numéricos no formato nn...nn.{6n}

Se par for entre "00" e "99", a resposta será enviada conforme parâmetros da tabela abaixo:

Par = "00" a "99" – A resposta será enviada com o seguinte formato:

Par	Descrição
00 a 15:	Totalizador parcial de alíquota sujeita ao ICMS, de T00 a T15 respectivamente {15n}.

- 16 a 31: Alíquota do totalizadores parciais sujeitas ao ICMS, de T00 a T15 respectivamente {4n}.
- 32: Totalizador parcial de isenção tributária {15n}.
- 33: Totalizador parcial de alíquota não tributada {15n}.
- 34: Totalizador parcial de tributação na fonte {15n}.
- 36: Totalizador parcial de descontos {15n}.
- 37: Totalizador parcial de acréscimos {15n}.
- 38: Totalizador parcial de cancelamento de itens {15n}.
- 39: Totalizador Geral {19n}.
- 40: Venda Bruta do Dia {19n}.
- 41: Contador de Operações (COO) {6n}.
 Número do última operação executada pelo ECF: LeituraX, ReduçãoZ, Cupom Fiscal, Comprovante Não Fiscal Vinculado ou Não Vinculado, etc.
- 42: Contador de Reduções {4n}.
- 43: Contador de Reinício de Operação {4n}.
- 44: Contador de Cupons Cancelados {4n}.
- 45: Contador de Operações não sujeita ao ICMS {4n}.
- 46: Contador de Estabelecimentos cadastrados (Modo Treinamento = 1) {2n}.
- 47: Versão do Firmware {7^a}.
- 48: Número sequencial atribuído pelo estabelecimento (número do caixa) {6n}.
- 49: Número de Série do ECF {10n}.
- 50 a 81: Legenda + valor dos totalizadores parciais não sujeitas ao ICMS {16s15n1g}.
- Lembre-se:
- 1) Os registradores programados na posições de 00 a 15 realizam a operação de pagamento dos Cupons Fiscais e Comprovantes Não Fiscais Não Vinculados. Por exemplo: Dinheiro, Cheque, Cartão, entre outros.
- 2) Os registradores programados na posições de 16 a 31 realizam a operação nos Comprovantes Não Fiscais Não Vinculados. Por exemplo: Sangria, Reforço de Caixa, Vale Funcionário, entre outros.
- Para obter a legenda do registrador 00: ECFPar("50")
 Para obter a legenda do registrador 16: ECFPar("66")
 Para obter a legenda do registrador 31: ECFPar("81")
- 82: No do cupom de venda final {6n} + N°. do cupom venda inicial {6n};
- 83: Data referente a redução Z {ddmmaa};
- 84: Status comprovante não fiscal:

Bit	Estado do Bit	Descrição
0	1	Em comprovante não fiscal vinculado.
1	1	Em comprovante não fiscal não vinculado.
2	1	Em Relatório Gerencial.
3	1	Relatório Gerencial em Leitura X
	0	Relatório Gerencial em Redução Z
4	1	Cancelamento de um comprovante não fiscal vinculado habilitado
5	1	Cancelamento de um comprovante não fiscal não vinculado habilitado.
6	1	Relatório Gerencial requisitado
7	1	Em processo de cancelamento de um comprovante não fiscal.

85: Marca e modelo do fabricante {25s}; (*)

86: Status de inicialização {2s} + Status de vinculação de totalizadores parciais de ISS {4s}, conforme sintaxe do comando de Status (ESC \$ 32): (*)

Bit	Estado do Bit	Descrição
0	1	Desconto de ISS habilitado.
1	1	Reservado.
2	1	Reservado.
3	1	Reservado.
4	1	CMC7 habilitado (válido somente para ECF duas estações).
5	1	Linha id cadastrado.
6	1	Arredondamento conforme ABNT.
7	1	Memória fiscal esgotada.

Status de vinculação dos totalizadores parciais de ISS:

inscrição dos totalizadores para o ICMS.

Bit	Estado do Bit	Descrição
0	Se 1: Totalizador 00 vinculado ao ISS. Se 0: Totalizador 00 vinculado ao ICMS.	
1	Se 1: Totalizador 01 vinculado ao ISS. Se 0: Totalizador 01 vinculado ao ICMS.	
2	Se 1: Totalizador 02 vinculado ao ISS. Se 0: Totalizador 02 vinculado ao ICMS.	
3	Se 1: Totalizador 03 vinculado ao ISS. Se 0: Totalizador 03 vinculado ao ICMS.	
4	Se 1: Totalizador 04 vinculado ao ISS. Se 0: Totalizador 04 vinculado ao ICMS.	
5	Se 1: Totalizador 05 vinculado ao ISS. Se 0: Totalizador 05 vinculado ao ICMS.	
6	Se 1: Totalizador 06 vinculado ao ISS. Se 0: Totalizador 06 vinculado ao ICMS.	
7	Se 1: Totalizador 07 vinculado ao ISS. Se 0: Totalizador 07 vinculado ao ICMS.	
8	Se 1: Totalizador 08 vinculado ao ISS. Se 0: Totalizador 08 vinculado ao ICMS.	
9	Se 1: Totalizador 09 vinculado ao ISS. Se 0: Totalizador 09 vinculado ao ICMS.	
10	Se 1: Totalizador 10 vinculado ao ISS. Se 0: Totalizador 10 vinculado ao ICMS.	
11	Se 1: Totalizador 11 vinculado ao ISS. Se 0: Totalizador 11 vinculado ao ICMS.	
12	Se 1: Totalizador 12 vinculado ao ISS. Se 0: Totalizador 12 vinculado ao ICMS.	
13	Se 1: Totalizador 13 vinculado ao ISS. Se 0: Totalizador 13 vinculado ao ICMS.	
14	Se 1: Totalizador 14 vinculado ao ISS. Se 0: Totalizador 14 vinculado ao ICMS.	
15	Se 1: Totalizador 15 vinculado ao ISS. Se 0: Totalizador 15 vinculado ao ICMS.	

87: Totalizador de descontos ISS {15n} + Totalizador de acréscimos ISS {15n} + Totalizador de Cancelamentos ISS {15n}. (*)

88: CNPJ {18s} + IE {15s} + CCM {15s}. (*)

89: Razão Social {201s}. (*)

90 a 94: Reservado.

95: Modelo do ECF {4s}:

"375P"	Duas estações com 40 colunas.
"300P"	Uma estação com 40 colunas.
"100M"	Uma estação com 48 colunas.

	96:	Total do cupom em andamento ou do último cupom finalizado. {15n}
	97:	Total dos pagamentos realizados – Total do cupom. {15n}
	98 a 99:	Reservado.
Retorno:		CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEXECUÇÃO
Exemplo:		EcfPar("41")
Exemplo em Delphi:		<pre> Procedure NomeDaProcedure(); var retorno: Integer; strMsg: String; RespostaECF: array [0..2000] of Char; QtosNParametro: Integer; begin // Para obter (por exemplo): // // COO: ECFPar('41') - QtosNParametro = 6 // Número do Caixa: ECFPar('48') - QtosNParametro = 6 // Número de Série do ECF: ECFPar('49') - QtosNParametro = 10 // Consulte nesse manual outros parâmetros parâmetros retorno := ECFPar('41'); // Obtendo o COO QtosNParametro := 6; // a resposta do COO possui 6 caracteres if retorno = CIF_OK then retorno := EsperaResposta(RespostaECF); if retorno = CIF_OK then Begin strMsg:= copy(RespostaECF, 6, QtosNParametro); MessageDlg('Parametro lido do ECF: [' + strMsg + ']' , mtInformation, [mbOk],0); end else MessageDlg('Erro ao transmittir ECFPar()' , mtInformation, [mbOk],0); end; end; </pre>
Exemplo em Visual Basic:		<pre> Private Sub NomeDaSub() Const MaxBuf = 2000 Dim RespostaECF As String * MaxBuf Dim QtosNParametros As Integer Dim Retorno As Integer // Para obter (por exemplo): // // COO: ECFPar("41") - QtosNParametro = 6 // Número do Caixa: ECFPar("48") - QtosNParametro = 6 // Número de Série do ECF: ECFPar("49") - QtosNParametro = 10 Retorno = ECFPar("41") QtosNParametros = 6 'a resposta do COO possui 6 caracteres 'consulte o manual de programação da DLL para outros parâmetros e tipos de respostas If Retorno = CIF_OK Then Retorno = EsperaResposta(RespostaECF) End If </pre>

```
If Retorno = CIF_OK Then
    MsgBox "Parâmetro lido do ECFPar: " + Mid(RespostaECF, 6,
    QtosNParametros)
Else
    MsgBox ("Erro ao transmitir ECFPAR()")
End If

End Sub
```

Exemplo em C: typedef int (FAR PASCAL *TXPARECF) (void);

```
TXPARECF    pTxParECF;
int          ret;

pTxParECF = (TXPARECF) GetProcAddress(h, "EcfPar");
ret = pTxParECF ( );
```

Observações:

- (*) – Parâmetros disponíveis a partir da versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Funções Diversas ***ProgLinhaAdicional()***

Função: **ProgLinhaAdicional**
Descrição: Programa a linha de identificação adicional, impressa no final dos cupons. Uma vez programada, esta linha será impressa em todos os cupons fiscais e não fiscais.
Sintaxe C: int FAR PASCAL ProgLinhaAdicional (char *leg);
Sintaxe VB: Declare Function ProgLinhaAdicional Lib "ECF32M.DLL" (ByVal leg As String) As Long
Sintaxe Delphi: function ProgLinhaAdicional (leg: PChar): Integer; stdcall; external 'ECF32M.dll';
Parâmetros:
 leg: Legenda usada para imprimir informações adicionais como loja, vendedor, etc., impresso no fechamento dos cupons fiscais, composto de 40/48 caracteres no formato ss...ss.{ 40/48s}
Retorno: CIF_OK
 CIF_ERR_SERIAL
 CIF_TIMEOUT
 CIF_EMEXECUÇÃO
Exemplo: ProgLinhaAdicional("Reserva de no maximo quarenta caracteres")
Exemplo em C: typedef int (FAR PASCAL *PRGLINHA) (char *);

```
PRGLINHA    pLinhaAdicional;
int          ret;

pLinhaAdicional = (PRGLINHA) GetProcAddress(h, "ProgLinhaAdicional");
```


DLL ECF32M

Especificação de Uso

- *nserie*: Número de série de fabricação { 10n}
- *nseq*: Número sequencial atribuído ao equipamento pelo estabelecimento { 6n}.

Parâmetros: Nenhum.

Sintaxe C: Int FAR PASCAL EcfID (void);

Sintaxe VB: Declare Function EcfID Lib "ECF32M.DLL" () As Long

Sintaxe Delphi: function EcfID(): Integer; stdcall; external 'ECF32M.dll';

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: EcfID()

Exemplo em C: typedef int (FAR PASCAL *TXECFID) (void);

```
TXECFID    pTxId;
int        ret;

pTxId = (TXECFID) GetProcAddress(h, "EcfID");
ret = pTxId ( );
```

[Voltar ao Índice](#)

Funções Diversas *ProgAliquotas()*

Função: **ProgAliquotas**

Descrição: Programas as alíquotas fiscais.

Sintaxe C: int FAR PASCAL ProgAliquotas (char *trib, char *valor);

Sintaxe VB: Declare Function ProgAliquotas Lib "ECF32M.DLL" (ByVal trib As String, ByVal valor As String) As Long

Sintaxe Delphi: function ProgAliquotasICMS (trib: Pchar; valor: Pchar): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

trib: Indica o tipo e a posição do totalizador parcial a ser programado. Este parâmetro deve respeitar o seguinte formato: <"Xnn">, onde:

- X – Representa o tipo do totalizador: (1c)
 - T – Tributado;
 - I – Isento;
 - F – Substituição tributária e
 - N – Não incidência.
- nn – Representa a posição do totalizador, ou alíquota, "00" a "15". (2n)

valor: Valor da alíquota; este parâmetro deve respeitar o seguinte formato: nnnn. (4n) Por exemplo: "1000" – alíquota de dez por cento.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ProgAliquotas("T07","1500")

Exemplo em C: typedef int (FAR PASCAL *PRGALQUOTA) (char *, char *);

```
PRGALIQOTApAliquota;
int ret;

pAliquota = (PRGALIQOTA) GetProcAddress(h, "ProgAliquotas");
ret = pAliquota ("T01","0700");
```

Observações:

- O tipo e posição, representados pelo parâmetro "trib" igual a "T00" é reservada, no ECF, para alíquotas de ISS, ou seja, a alíquota programada com este parâmetro será considerada uma alíquota de ISS.
- As alíquotas programadas nas demais posições ("Txx"), poderão ser gravadas no ECF como alíquotas de ICMS ou ISS.
- Apesar da alíquota ter sido cadastrada no ECF como ICMS ou ISS, nessa essa DLL deverá sempre ser enviado "Txx". Exemplo: "T00", "T01", "T02", "T03", ..., "T15".
- **Esta função só poderá ser executada para adicionar novas alíquotas ao ECF. Para alterar as alíquotas já programadas, o ECF deverá ser colocado em estado de Intervenção Técnica.**

[Voltar ao Índice](#)

Funções Diversas

ProgAliquotasICMS_ISS()

Função: **ProgAliquotasICMS_ISS**

Descrição: Programa as alíquotas fiscais, permitido, através do parâmetro "Tipo", que se determine se a alíquota, de uma determinada posição, seja de ISS, ou ICMS.

Sintaxe C: int FAR PASCAL ProgAliquotasICMS_ISS(
char *trib
, char *valor
, char Tipo
);

Sintaxe VB: Declare Function ProgAliquotasICMS_ISS Lib "ECF32M.DLL" (
ByVal trib As String
, ByVal valor As String
, ByVal Tipo As Byte
) As Long

Sintaxe Delphi: function ProgAliquotasICMS (
trib: Pchar;
valor: Pchar ;
Tipo: Char
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

- trib:** Indica o tipo e a posição do totalizador parcial a ser programado. Este parâmetro deve respeitar o seguinte formato: <"Xnn">, onde:
- X – Representa o tipo do totalizador: (1c)
 - T – Tributado;
 - I – Isento;
 - F – Substituição tributária e
 - N – Não incidência.
 - nn – Representa a posição do totalizador, ou alíquota, "00" a "15". (2n)

valor: Valor da alíquota; este parâmetro deve respeitar o seguinte formato: nnnn. (4n) Por exemplo: "1000" – alíquota de dez por cento.

Tipo: Determina o tipo da alíquota (Tributado):
'0' (30h) – A alíquota será vinculada ao ICMS.
'1' (31h) – A alíquota será vinculada ao ISS.

Retorno: CIF_OK
CIF_ERR_SERIAL
CIF_TIMEOUT
CIF_EMEXECUÇÃO

Exemplo: ProgAliquotasICMS_ISS("T01","0700", '1')

Exemplo em C: typedef int (FAR PASCAL *PRGALIQOTAICMS_ISS) (char *, char *, char);

```
PRGALIQOTAICMS_ISS      pAliquotasICMS_ISS;
int                      ret;

pAliquotasICMS_ISS = (PRGALIQOTAICMS_ISS) GetProcAddress(h,
"ProgAliquotasICMS_ISS");
ret = pAliquotasICMS_ISS ("T01","0700", '1');
```

Observações:

- O tipo e posição, representados pelo parâmetro "trib" igual a "T00" é reservada, no ECF, para alíquotas de ISS, ou seja, a alíquota programada com este parâmetro será considerada uma alíquota de ISS.
- **Esta função só poderá ser executada para adicionar novas alíquotas ao ECF. Para alterar as alíquotas já programadas, o ECF deverá ser colocado em estado de Intervenção Técnica.**
- Esta função está disponível a partir da versão 2.1 desta dll, e versão FCP-500 do firmware da impressora fiscal.

[Voltar ao Índice](#)

Funções Diversas [ProgSimbolo\(\)](#)

Função: ProgSimbolo

Descrição: Programa o caracter a ser impresso na validação de um documento. Este caracter é definido através do envio de dez bytes que representam as colunas gráficas a serem impressas.

Sintaxe C: int FAR PASCAL ProgSimbolo (
char B1
, char B2
, char B3
, char B4
, char B5
, char B6
, char B7
, char B8
, char B9
, char B10
, char Hab
);

Sintaxe VB: Declare Function ProgSimbolo Lib "ECF32M.DLL" (
ByVal B1 As Byte
, ByVal B2 As Byte
, ByVal B3 As Byte
, ByVal B4 As Byte
, ByVal B5 As Byte
, ByVal B6 As Byte

);

PRGSIMBOLO pSimbolo;
int ret;

pSimbolo = (PRGSIMBOLO) GetProcAddress(h, "ProgSimbolo");
ret = pSimbolo (0xC3 ,0x3C ,0x42 ,0x3C ,0x42 ,0x3C ,0x42 ,0x3C ,0x42 ,0x3C , '1');

[Voltar ao Índice](#)

Funções Diversas **ProgArredondamento()**

Função:	ProgArredondamento
Descrição:	<p>Programa o tipo de tratamento a ser dado aos valores utilizados nas operações aritméticas realizadas pelo ecf.</p> <p>Os valores poderão ser truncados ou arredondados.</p> <p>O parâmetro "Par" indica o tipo de tratamento a ser aplicado.</p> <p>OBS:</p> <ul style="list-style-type: none"> • O ecf, por default, realiza o tratamento de truncamento dos valores durante a execução das operações aritméticas. • Para as versões de firmware, do ecf, inferiores a versão FCP-500, esta função só poderá ser utilizada caso o ecf se encontre no estado de Intervenção Técnica. Para as versões iguais, ou superiores, esta função deverá ser chamada após a última Redução Z, e antes da emissão de um cupom fiscal de venda.
Sintaxe C:	<pre>int FAR PASCAL ProgArredondamento (char Par);</pre>
Sintaxe VB:	<pre>Declare Function ProgArredondamento Lib "ECF32M.DLL" (Par ByVal As Byte) As Long</pre>
Sintaxe Delphi:	<pre>function ProgArredondamento (Par: Char): Integer; stdcall; external 'ECF32M.dll';</pre>
Parâmetros:	<p>Parâmetro Par que indica o tipo de tratamento a ser aplicado:</p> <ul style="list-style-type: none"> ▪ '0' (30h) ou 0 (00h) – Truncamento. ▪ '1' (31h) ou 1 (01h) – Arredondamento. ▪ '2' (32h) ou 2 (02h) – Arredondamento ABNT.
Retorno:	<p>CIF_OK CIF_ERR_SERIAL CIF_TIMEOUT CIF_EMEXECUÇÃO</p>
Exemplo:	ProgArredondamento('1')
Exemplo em C:	typedef int (FAR PASCAL *PRGARREND) (void);

PRGARREND pPrgArred;
int ret;

pPrgArred = (PRGARREND) GetProcAddress(h, "ProgArredondamento");
ret = pPrgArred('1'); // Programa arredondamento.
If(ret != CIF_OK) { // Erro na programacao do parametro.
 MessageBox(
 NULL
 , "Não conseguiu programar o tipo de arredondamento"
 , "ERRO"
 , MB_ICONSTOP
);
}

}

[Voltar ao
Índice](#)

Funções Diversas **[EsperaResposta\(\)](#)**

Função: **EsperaResposta**

Descrição: **Recomendamos o uso desta função para a realização do tratamento dos códigos de retornos do ecf.**

Obtém a mensagem e o código de retorno de um comando enviado ao ECF. Após o envio de um comando, o ECF envia uma resposta que poderá ser obtida através desta função. Para isso é preciso passar como parâmetro um buffer para recepção da mensagem de retorno e, opcionalmente, configurar um tempo máximo de espera desta resposta através do arquivo CIF.INI – [Arquivo de Configuração](#).

O timeout deverá ser configurado entre os seguintes valores 3 e 30 segundos. Caso o valor seja inferior a 3 segundos, o timeout será configurado com o valor de 3 segundos; caso o valor seja superior a 30 segundos, o timeout será configurado com o valor de 30 segundos. O valor default é 3 segundos.

A função EsperaResposta permanece “bloqueada” a espera de um retorno até que o valor de timeout seja alcançado. Este valor será contabilizado a partir do momento em que o ecf não responder a solicitação de status que poderão ocorrer durante o período em que se aguarda uma resposta.

Sintaxe C: int FAR PASCAL EsperaResposta (char *buf_ret);

Sintaxe VB: Declare Function EsperaResposta Lib "ECF32M.DLL" (
ByVal buf_ret As String
) As Long

Sintaxe Delphi: function EsperaResposta (
buf_ret: Pchar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

Buf_ret: Buffer para recepção da mensagem de retorno de um comando. Este buffer deve ser grande o suficiente para receber todos os caracteres da resposta.

Retorno: CIF_OK
 CIF_OK_PPAPPEL
 CIF_TIMEOUT
 CIF_OVERFLOW
 CIF_ERR_SYS
 CIF_ERR_READSER
 CIF_ERR_ANSWER
 CIF_ERR_TEMP
 CIF_ERR_PPAPPEL
 CIF_ERR_IRRECUPERAVEL
 CIF_ERR_MECANICO
 CIF_ERR_TABERTA

Códigos de erro do ECF, que se encontram na faixa compreendida entre -1 a -64 obtidos através da string retornada através do buffer "buf_ret", conforme o tópico "Códigos de Retorno".

Além dos códigos de retorno mencionados, a dll retornará, através da geração do arquivo ECF32M_Retorno.TXT as seguintes mensagens:

Mensagem	Descrição
"-111188Erro alloc. recursos."	Erro na alocação de recursos do sistema operacional.
"-111189Resposta invalida."	Resposta do ecf inválida. (problemas na comunicação).
"-111190Erro com. serial."	Erro na comunicação serial.
"-111191Temperatura alta."	A cabeça de impressão apresenta temperatura alta. (válido somente para impressora de duas estações).
"-111192Pouco papel."	A impressora apresenta pouco papel.
"-111194Erro irrecoverável."	A impressora apresenta erro irrecoverável.
"-111195Erro mecânico."	A impressora apresenta erro mecânico.
"-111196Tampa aberta."	A impressora apresenta tampa aberta.

OBS: O arquivo de retorno só será gerado caso a opção **ARQUIVO** esteja presente e configurada no arquivo de configuração de operação (CIF.INI) da dll.

Exemplo: EsperaResposta(Buffer)

Exemplo em C:

```
typedef int (FAR PASCAL *ESPERARESPOSTA) (char *);

int TrataRetorno ( void )
{
    int          Ret;
    char          Buff[2048];
    ESPERARESPOSTA pEsperaResposta;

    // Obtém o endereço da função na DLL
    pEsperaResposta = (ESPERARESPOSTA) GetProcAddress(h, "EsperaResposta") ;

    // sai por TimeOut.
    Rer=pEsperaResposta( Buff );
    switch( Ret )
    {
        case CIF_TIMEOUT:
```

```

        MessageBox( NULL , "TimeOut" , "ERRO" , MB_ICONSTOP );
        return 0;

    case CIF_ERR_ANSWER:
        MessageBox( NULL , Buff , "RESPOSTA DESCONHECIDA" , MB_ICONSTOP );
        return 0;

    case CIF_ERR_READSER:
        MessageBox( NULL , "" , "ERRO LEITURA SERIAL" , MB_ICONSTOP );
        return 0;

    case CIF_OVERFLOW:
        MessageBox( NULL , Buff , "OVERFLOW BUFFER RECEPÇÃO"
,MB_ICONSTOP );
        return 0;

    case CIF_ERR_TABERTA:
        MessageBox( NULL , Buff , "TAMPA ABERTA " , MB_ICONSTOP );
        return 0;
    case CIF_IRRECUPERAVEL:
        MessageBox( NULL , Buff , "ERRO IRRECUPERÁVEL " , MB_ICONSTOP );
        return 0;
    case CIF_ERR_MECANICO:
        MessageBox( NULL , Buff , "ERRO MECÂNICO " , MB_ICONSTOP );
        return 0;
    case CIF_ERR_TEMP:
        MessageBox(
            NULL
            ,Buff
            , "TEMPERATURA DA CABEÇA ALTA "
            ,MB_ICONSTOP
        );
        return 0;
    case CIF_ERR_PPAPEL:
        MessageBox( NULL , Buff , "POUCO PAPEL " , MB_ICONSTOP );
        return 0;
    case CIF_ERR_SYS:
        MessageBox(
            NULL
            ,Buff
            , "ERRO NA ALOCAÇÃO DE RECURSOS"
            ,MB_ICONSTOP
        );
        return 0;
    default:
        if(Ret >=0)
        {
            MessageBox (
                NULL
                ,Buff
                , "Comando executado com sucesso "
                ,MB_OK
            );
        }
        else
        {
            MessageBox (
                NULL
                ,Buff
                , "Comando não executado "
                ,MB_OK
            );
        }
}

```

```

        return 1;
    }
}

```

Observações:

- Com o objetivo de facilitar o trabalho do desenvolvedor e agilizar o processo de resposta do ECF foi introduzida, a partir da **versão 2.5** da dll a função **EsperaResposta()**, que dispensa a implementação do "loop" de espera da função recomendada anteriormente **ObtemRetorno()**.
- Além dos códigos de retorno mencionados, a dll retornará, através da geração do arquivo ECF32M_Retorno.TXT as seguintes mensagens:

Mensagem	Descrição
"-111188Erro aloc. recursos."	Erro na alocação de recursos do sistema operacional.
"-111189Resposta invalida."	Resposta do ecf inválida. (problemas na comunicação).
"-111190Erro com. serial."	Erro na comunicação serial.
"-111191Temperatura alta."	A cabeça de impressão apresenta temperatura alta. (válido somente para impressora de duas estações).
"-111192Pouco papel."	A impressora apresenta pouco papel.
"-111194Erro irrecuperável."	A impressora apresenta erro irrecuperável.
"-111195Erro mecânico."	A impressora apresenta erro mecânico.
"-111196Tampa aberta."	A impressora apresenta tampa aberta.

O arquivo de retorno só será gerado caso a opção **ARQUIVO** esteja presente e configurada no arquivo de configuração de operação (CIF.INI) da dll.

[**Voltar ao Índice**](#)

Funções de Impressão Livre

Funções de Impressão Livre ***ImprimeNaoFiscal()***

Função: **ImprimeNaoFiscal**

Descrição: Envia o buffer de impressão em cupons não fiscais de impressão livre (Relatório Gerencial e Comprovante Não Fiscal Vinculado). Neste modo livre não há o retorno do ECF, ou seja, não é permitido o uso da função ObtemRetorno(). Caso se deseje a impressão não fiscal livre com retorno, deve ser usada a função ImprimeLinhaNaoFiscal(). Esta função é válida nos seguintes cupons não fiscais:
 Relatório Gerencial e
 Comprovante Não Fiscal Vinculado
 Se for enviado o caracter 10 (0Ah), os dados do buffer de impressão serão impressos e a impressora avançará uma linha.

Sintaxe C: int FAR PASCAL ImprimeNaoFiscal (char NoImpr ,char *Buf_Imp);

Sintaxe VB: Declare Function ImprimeNaoFiscal Lib "ECF32M.DLL" (
 ByVal NoImpr As Byte
 ,ByVal Buf_Imp As String
) As Long

Sintaxe Delphi: function ImprimeNaoFiscal (
 NoImpr: Char;
 Buf_Imp: Pchar
): Integer; stdcall; external 'ECF32M.dll';

Parâmetros:

Nolmpr: Número de vezes que Buf_Imp será impresso.

Buf_Imp: Buffer com os dados a serem impressos { 2040s}.
 Atenção! Esta função suporta no máximo 2040 {?s}.

Retorno: CIF_OK
 CIF_ERR_SERIAL

Exemplo: ImprimeNaoFiscal ('4', "Teste de Impressao com AVANCALINHA \0x0a")

Exemplo em C:

```
typedefint (FAR PASCAL *IMPRIME) (char *);
char Buff[41];
char Nolmpr;
IMPRIME plmprime;
.
.
// Obtem o endereço da função
plmprime = (IMPRIME) GetProcAddress(h, "ImprimeNaoFiscal");
Nolmpr=2;

strcpy (Buff, "Teste de Impressao com AVANCALINHA \0x0a");
ret=plmprime(Nolmpr,Buff);
```

[Voltar ao Índice](#)

Problemas na comunicação com a Impressora Fiscal

Caso não esteja conseguindo se comunicar com o ECF, siga esse roteiro de testes para descobrir onde está o problema:

- Verifique se o ECF está ligado (com o Led verde aceso);
- Verifique se o cabo de comunicação serial está bem conectado ao micro e ao ECF;
- Verifique se a versão da sua DLL ECF32M é a mais atual.
- Verifique se a sua DLL ECF32M está em apenas uma única pasta.
- Verifique se a pasta onde está instalada a DLL ECF32M é uma das indicadas na sessão de Instalação deste manual.
- Verifique se o arquivo CIF.INI está na pasta indicada na sessão de Instalação deste manual.
- Verifique no arquivo CIF.INI se a porta COM está correta;
- Utilize um dos programas exemplo (Visual Basic ou Delphi) para testar a comunicação com o ECF;
- Se possível, utilize o MecafECF.exe para testar a comunicação serial (descrição, instruções para download e uso constam desse manual);
- Execute o auto-teste (veja as instruções na FAQ da IF113 disponível em nosso site), e veja a configuração do ECF. Deverá estar conforme abaixo:
 - Protocolo: STX/ETX
 - Velocidade: 9600 bps
 - Paridade: Nenhuma (Sem)
 - Bits de Parada: 1
 - Bits de Dados: 8
 - ACK/NACK: Desabilitado
- Verifique se a configuração (IRQ e endereço de I/O) da porta COM de micro está conforme indicado neste manual.
- Refaça os testes em um outro micro. O problema pode estar na saída serial do micro.
- Verifique se o cabo de comunicação está no padrão RS232. Veja a especificação (pinagem) desse cabo na FAQ disponível em nosso site.
- Verifique a saída serial do ECF;

[**Voltar ao Índice**](#)

Configuração da Impressora Fiscal

Execute o auto-teste para verificar e alterar a configuração da sua Impressora Fiscal. Para saber como utilizar o auto-teste, consulte na FAQ disponível em nosso site.

Exemplo de uma configuração válida para o ECF:

```
*****
**                                     **
**      Interface      : SERIAL      **
**      Protocolo      : STX ETX     **
**      Baud Rate      : 9600        **
**      Bits de Dados  : 8 Bits      **
**      Paridade       : SEM         **
**      Modo ACK/NACK  : Desabilitado**
**      Desconto ISS   : Habilitado  **
**                                     **
*****
```

[**Voltar ao Índice**](#)

Como obter o Número do Cupom Fiscal?

Considere o COO (Contador de Ordem de Operações) como sendo o Número do Cupom Fiscal ou Número do Comprovante Não Fiscal.

Para capturar o COO do último cupom ou do cupom em andamento, use a função EcfPar("41").

A resposta será enviada através do buffer da função EsperaResposta(buffer) no formato: "+9999123456", onde "123456" é o COO do cupom em andamento.

[**Voltar ao Índice**](#)

Como obter o Número do Caixa do ECF?

Considere o Número de Caixa do ECF como sendo o Número sequencial atribuído pelo Estabelecimento.

Para capturar o Número de Caixa do ECF, use a função EcfPar("48").

A resposta será enviada através do buffer da função EsperaResposta(buffer) no formato: "+9999000001", onde "000001" é o Número de Caixa do ECF.

[**Voltar ao Índice**](#)

Como obter o Número de Série do ECF?

Para capturar o Número de Série do ECF, use a função EcfPar("49") (passando como parâmetro "49").

A resposta será enviada através do buffer da função EsperaResposta(buffer) no formato: "+99990000018903", onde "0000018903" é o Número de Série do ECF.

[**Voltar ao Índice**](#)

Como obter o Total do Cupom em andamento?

Para capturar o total do cupom em andamento ou do último cupom finalizado do ECF, use a função EcfPar("96").

DLL ECF32M
Especificação de Uso

A resposta será enviada através do buffer da função EsperaResposta(buffer) no formato: "+9999000000000100000", onde "000000000100000" é o total do cupom em andamento.

[Voltar ao Índice](#)

Como obter o Total a pagar do Cupom em andamento?

Para capturar o total a pagar do cupom em andamento ou do último cupom finalizado do ECF, use a função EcfPar("97").

A resposta será enviada através do buffer da função EsperaResposta(buffer) no formato: "+9999000000000000100", onde "000000000000100" é o total a pagar do cupom.

[Voltar ao Índice](#)

Tabela de caracteres ABICOMP

DEC.	HEX	CHR	DEC.	HEX	CHR	DEC.	HEX	CHR	DEC.	HEX	CHR
32	20		88	58	X	144	90	n.d.	200	C8	É
33	21	!	89	59	Y	145	91	n.d.	201	C9	Ê
34	22	"	90	5A	Z	146	92	n.d.	202	CA	Ë
35	23	#	91	5B	[147	93	n.d.	203	CB	ì
36	24	\$	92	5C	\	148	94	n.d.	204	CC	í
37	25	%	93	5D]	149	95	n.d.	205	CD	î
38	26	&	94	5E	^	150	96	n.d.	206	CE	ï
39	27	'	95	5F	_	151	97	n.d.	207	CF	ñ
40	28	(96	60	`	152	98	n.d.	208	D0	ò
41	29)	97	61	a	153	99	n.d.	209	D1	ó
42	2A	*	98	62	b	154	9A	n.d.	210	D2	ô
43	2B	+	99	63	c	155	9B	n.d.	211	D3	õ
44	2C	,	100	64	d	156	9C	n.d.	212	D4	ö
45	2D	-	101	65	e	157	9D	n.d.	213	D5	œ
46	2E	.	102	66	f	158	9E	n.d.	214	D6	ù
47	2F	/	103	67	g	159	9F	n.d.	215	D7	ú
48	30	0	104	68	h	160	A0		216	D8	û
49	31	1	105	69	i	161	A1	À	217	D9	ü
50	32	2	106	6A	j	162	A2	Á	218	DA	ÿ
51	33	3	107	6B	k	163	A3	Â	219	DB	ß
52	34	4	108	6C	l	164	A4	Ã	220	DC	à
53	35	5	109	6D	m	165	A5	Ä	221	DD	á
54	36	6	110	6E	n	166	A6	Ç	222	DE	â
55	37	7	111	6F	o	167	A7	È	223	DF	±
56	38	8	112	70	p	168	A8	É	224	E0	n.d.
57	39	9	113	71	q	169	A9	Ê	225	E1	n.d.
58	3A	:	114	72	r	170	AA	Ë	226	E2	n.d.
59	3B	;	115	73	s	171	AB	Ì	227	E3	n.d.
60	3C	<	116	74	t	172	AC	Í	228	E4	n.d.
61	3D	=	117	75	u	173	AD	Î	229	E5	n.d.
62	3E	>	118	76	v	174	AE	Ï	230	E6	n.d.
63	3F	?	119	77	w	175	AF	Ñ	231	E7	n.d.
64	40	@	120	78	x	176	B0	Ò	232	E8	n.d.
65	41	A	121	79	y	177	B1	Ó	233	E9	n.d.
66	42	B	122	7A	z	178	B2	Ô	234	EA	n.d.
67	43	C	123	7B	{	179	B3	Õ	235	EB	n.d.
68	44	D	124	7C		180	B4	Ö	236	EC	n.d.
69	45	E	125	7D	}	181	B5	Œ	237	ED	n.d.

DLL ECF32M
Especificação de Uso

70	46	F	126	7E	~	182	B6	Û	238	EE	n.d.
71	47	G	127	7F		183	B7	Ü	239	EF	n.d.
72	48	H	128	80	n.d.	184	B8	Ů	240	F0	n.d.
73	49	I	129	81	n.d.	185	B9	Ű	241	F1	n.d.
74	4A	J	130	82	n.d.	186	BA	Ý	242	F2	n.d.
75	4B	K	131	83	n.d.	187	BB	ÿ	243	F3	n.d.
76	4C	L	132	84	n.d.	188	BC	ē	244	F4	n.d.
77	4D	M	133	85	n.d.	189	BD	'	245	F5	n.d.
78	4E	N	134	86	n.d.	190	BE	§	246	F6	n.d.
79	4F	O	135	87	n.d.	191	BF	°	247	F7	n.d.
80	50	P	136	88	n.d.	192	C0	ı	248	F8	n.d.
81	51	Q	137	89	n.d.	193	C1	à	249	F9	n.d.
82	52	R	138	8A	n.d.	194	C2	á	250	FA	n.d.
83	53	S	139	8B	n.d.	195	C3	â	251	FB	n.d.
84	54	T	140	8C	n.d.	196	C4	ã	252	FC	n.d.
85	55	U	141	8D	n.d.	197	C5	ä	253	FD	n.d.
86	56	V	142	8E	n.d.	198	C6	ç	254	FE	n.d.
87	57	W	143	8F	n.d.	199	C7	È	255	FF	n.d.

[Voltar ao Índice](#)

Histórico das Revisões

Este tópico apresenta o histórico das revisões do manual e das versões da dll.

Revisões do manual

Revisão	Data	Descrição
1.00	Agosto/2001	<ul style="list-style-type: none"> Primeira versão do manual.
1.01	Setembro/2001	<ul style="list-style-type: none"> Revisão do manual da dll, incluindo: <ul style="list-style-type: none"> Nova formatação Novas funções que contemplam a versão de firmware FCP-500.
1.02	Setembro/2001	<ul style="list-style-type: none"> Correção da descrição dos valores do parâmetro troco, presente na função Pagamento. Correção do exemplo de uso da função ImprimeLinhaNaoFiscal. Correção da sintaxe da declaração dos parâmetros da função <code>ImprimeLinhaNaoFiscalTexto</code>. (Linguagens C e Visual Basic) .
1.03	Outubro/2001	<ul style="list-style-type: none"> Introdução do item "Instalação da DLL". Alterações no item "Arquivo de Configuração"
1.04	Dezembro/2001	<ul style="list-style-type: none"> Introdução das funções: <ul style="list-style-type: none"> TransMemFiscalData TransMemFiscalRed TransMemFiscalDataArg TransMemFiscalRedArg
1.05	Maio/2002	<ul style="list-style-type: none"> Introdução dos parâmetros ARQUIVO, TIMEOUT e STATUS no arquivo de

Revisão 1.12 80 83

		configuração – “ Arquivo de Configuração ”.
		<ul style="list-style-type: none"> • Introdução da função EsperaResposta(). • Alteração no item “Observações Gerais”.
1.06	Junho/2002	Introdução do descritivo da função ProgArredondamento .
1.07	Agosto/2002	Alteração dos itens: <ul style="list-style-type: none"> ▪ Histórico das Revisões ▪ Observações Gerais
1.08	Janeiro/2004	<ul style="list-style-type: none"> ▪ Reformulação do layout deste manual
1.09	Janeiro/2004	<ul style="list-style-type: none"> ▪ Inserção das funções: <ul style="list-style-type: none"> ○ VendaItemStr() ○ ProgramaHorarioVeraoStr() ○ AcionarGaveta() ▪ Funções consideradas obsoletas: <ul style="list-style-type: none"> ○ AbrirGaveta() ○ VendaItem() ○ ObtemRetorno() <p>OBS: Estas funções continuam fazendo parte da DLL ECF32M, mas deixam de estar descritas nesse manual.</p>
1.10	Fevereiro/2004	<ul style="list-style-type: none"> ▪ Inserção de um código exemplo em Delphi e Visual Basic da Função ECFPar() associada à função EsperaResposta (); ▪ Inserção e algumas Perguntas e Respostas mais frequentes; ▪ Inserção da tabela de caracteres ABICOMP. ▪ Funções consideradas obsoletas: <ul style="list-style-type: none"> ○ TotalizarCupomParcial() ○ TotalizarCupom() <p>OBS: Estas funções continuam fazendo parte da DLL ECF32M, mas deixam de estar descritas nesse manual.</p> <ul style="list-style-type: none"> ▪ Inserção das funções: <ul style="list-style-type: none"> ○ TotCupomSemDescAcres() ○ TotCupomAcresValor() ○ TotCupomAcresPorcentagem() ○ TotCupomDescValor() ○ TotCupomDescPorcentagem() ○ DescontoItemPorcentagem() ○ DescontoItemValor() ○ LeituraXComRelGer() ○ ReducaoZComRelGer()
1.20	Maio/2004	<ul style="list-style-type: none"> ▪ Melhora na descrição do ECFPar() parâmetros 50 a 81. ▪ Melhora na descrição do OperRegNaoFiscal(). ▪ Inclusão do código de erro CIF_ERR_FIMPAPEL (-83), para indicar Fim de Papel. ▪ Correção do erro -6 para “Troco já realizado.”

Revisões da DLL		
Versão	Data	Descrição
2.1	Agosto/2001	Última versão da dll.
2.2	Outubro/2001	Introduzido o controle de tratamento das mensagens do sistema operacional através do uso do parâmetro "MSG_SYS" presente no arquivo CIF.INI
2.3	Novembro/2001	Introdução de melhorias internas no processo de depuração via geração de arquivos (" Arquivo de Log ")
2.4	Dezembro/2001	Introdução das funções: <ul style="list-style-type: none"> ▪ TransMemFiscalData ▪ TransMemFiscalRed ▪ TransMemFiscalDataArq ▪ TransMemFiscalRedArq
2.5	Maio/2002	<ul style="list-style-type: none"> • Introdução dos parâmetros ARQUIVO, TIMEOUT e STATUS no arquivo de configuração – "Arquivo de Configuração". • Alteração nos Códigos de Retorno. – Introdução do código CIF_CMDEXECUCAO. • Introdução da função EsperaRespostaO.
2.6	Junho/2002	Melhorias no sistema de depuração da dll (geração do arquivo de LOG).
2.0.0.7	Agosto/2002	<ul style="list-style-type: none"> ▪ Correções internas da função OpenCif. ▪ Introdução de melhorias internas na função OpenCif. ▪ Correção interna do dispositivo de Timeout. ▪ Correção na sistemática de envio de comandos para o ecf. A solicitação de status, antes do envio de um comando, não é mais uma situação default.
2.0.0.8	Setembro/2003	Corrigido problema de timeout nos sistemas operacionais Windows 2000 e XP.
2.0.0.9	Outubro/2003	Implementada a Função VendalItemStr().
2.0.1.0	Novembro/2003	Corrigido o número da versão no Log, pois estava 2.0.0.7
2.0.1.1	Janeiro/2004	Implementadas as Funções: <ul style="list-style-type: none"> ▪ ProgramaHorarioVeraoStr() ▪ AcionarGaveta()
2.0.1.2	Fevereiro/2004	Implementadas as Funções: <ul style="list-style-type: none"> ▪ LeituraXComRelGer() ▪ ReducaoZComRelGer() ▪ TotCupomSemDescAcres() ▪ TotCupomAcresValor() ▪ TotCupomAcresPorcentagem() ▪ TotCupomDescValor() ▪ TotCupomDescPorcentagem() ▪ DescontoItemPorcentagem() ▪ DescontoItemValor()
2.0.1.3	Maio/2004	Corrigido o número da versão no Log para 2.0.1.3, pois estava 2.0.1.1
2.0.1.4	Maio/2004	Implementado o tratamento de Fim de Papel através do código de erro CIF_ERR_FIMPAPEL (-83).

[Voltar ao Índice](#)

Observações Gerais

Esta dll é uma ferrameta que visa proporcionar, ao desenvolvedor, um meio fácil, rápido e confiável de desenvolvimento de sua aplicação fiscal.

- **É importante destacar que algumas funções, descritas neste manual, são válidas a partir da versão 2.1 da dll ECF32M, e versão FCP-500 do firmware da impressora fiscal.**
- **Consulte, também o manual de programação da impressora fiscal; neste pode-se observar mais detalhes sobre o comportamento da impressora fiscal.**

Além desta ferramenta, a MECAF dispõe de exemplos desenvolvidos em **Visual Basic** e **Delphi**, que exemplificam o uso desta dll.

Com o objetivo de melhor atender ao clientes e desenvolvedores, solicitamos sua opinião sobre esta ferramenta; entre em contato conosco através do seguinte endereço:

E-mail: suporte@mecaf.com.br

Para obter maiores informações sobre os nossos produtos, consulte nosso site www.mecaf.com.br, procure pelo produto **IF113I**.

Desde de já agradecemos sua atenção.