

# Manual de programação ECD

Terminais Verificadores de Preços – ECD-1200 / ECD1200M / ECD-2000 / ECD-2500



Versão 01.03  
Edição: outubro de 2016

Este manual tem caráter técnico-informativo, sendo propriedade da SWEDA. Todos os direitos reservados. Nenhuma parte deste manual pode ser reproduzida ou armazenada por processo mecânico, eletrônico, de fotocópia, de gravação ou de qualquer outro tipo, sem autorização prévia e por escrito da SWEDA.

Todos os cuidados foram tomados na preparação deste manual. Entretanto, a SWEDA não assume nenhuma responsabilidade por erros, imprecisões ou omissões em seu conteúdo, assim como por quaisquer tipos de danos, perdas ou lucros cessantes decorrentes do uso da informação aqui contida.

As informações contidas neste manual estão sujeitas a alteração sem aviso prévio.

## Conteúdo

1. Introdução .....	4
2. Documentação .....	4
2.1 Documentação de Referência .....	4
2.2 Descrição Funcional .....	5
2.2.1 Interface .....	5
3. Protocolo de Mensagens do ECD1200 .....	8
3.1 Mensagens .....	8
3.2 Mensagens de Operação .....	8
3.3 Mensagens de Atividade .....	9
3.4 Mensagens de Configuração .....	9

## Lista de figuras

Figura 1: ECD-1200 .....	4
Figura 2: ECD-2000 .....	4
Figura 3: ECD-2500 .....	4

## Lista de tabelas

Tabela 1: Resumo das Funções da DLL .....	5
Tabela 2: Estrutura de Dados stAddress .....	6
Tabela 3: Mensagens de Operação .....	8
Tabela 4: Mensagens de Atividade .....	9
Tabela 5: Mensagens de Configuração .....	10
Tabela 6: Exemplo de Configuração .....	10

## 1. Introdução

Este manual descreve a DLL (*Dynamic Link Library*) responsável pela configuração e controle de acesso aos Terminais de Consulta de Preço da linha ECD. Esta DLL pode ser utilizada em qualquer linguagem de programação em 32 bits, compatível com Windows. Adicionalmente, o manual descreve o seu protocolo de mensagens via TCP/IP.

## 2. Documentação

### 2.1 Documentação de Referência

Os Terminais de Consulta de Preço Sweda da linha ECD são dispositivos com scanner a laser, por onde se reconhecem os códigos de barras dos produtos. Conecta-se ao Servidor da base de dados por uma conexão Ethernet (TCP/IP). O Servidor responde ao ECD mandando a descrição e o preço do produto, previamente cadastrados em sua base de dados. Estas informações são apresentadas pelo *display* de cristal líquido. Para maiores referências às características do Terminal de Consulta de Preço Sweda, consulte o nosso site ([www.sweda.com.br](http://www.sweda.com.br)).



Figura 1: ECD-1200



Figura 2: ECD-2000



Figura 3: ECD-2500

## 2.2 Descrição Funcional

A DLL funciona como uma ponte entre os protocolos dos ECDs e a base de dados onde os produtos se encontram cadastrados. Deste modo, a aplicação do cliente ganha mais funcionalidade e flexibilidade, pois seu funcionamento é independente do tipo da base de dados (Access, Paradox, etc.) e da linguagem de programação utilizada (Delphi, C++ Builder, Visual Basic, Visual C++, C#, etc.).

### 2.2.1 Interface

A DLL de comunicação entre o Servidor e o ECD é a “*ecdSweda.dll*”, desta forma, o aplicativo deverá chamar as funções desta DLL. No pacote da Interface também contém a DLL “*pthreadVC2.dll*”, mas o aplicativo não necessita chamar nenhuma função desta dll, precisando apenas deixá-la no mesmo diretório da DLL principal “*ecdSweda.dll*”.

Na “*tabela 1*” são listadas as funções contidas na DLL, é importante observar que o programador deve seguir uma seqüência para que seu aplicativo tenha acesso completo à DLL e aos ECDs. Onde nesta sequência deverá estar: a chamada da função *bInitialize()* logo após carregar a DLL e a chamada da *bTerminate()* antes de sair da aplicação. A porta 6500 é utilizada pelo Servidor para comunicação com o ECD, devido a isso, a mesma deverá estar liberada para o uso.

Funções fornecidas pela <i>ecdSweda.dll</i>
<b>BOOL</b> __stdcall <b>bInitialize</b> (HWND Handle);
<b>BOOL</b> __stdcall <b>bTerminate</b> (void);
<b>void</b> __stdcall <b>vGetMsgWinReceiveDataTC</b> (LPUINT MsgWinReg);
<b>BOOL</b> __stdcall <b>bConnect</b> (stAddress *ID);
<b>BOOL</b> __stdcall <b>bDisconnect</b> (stAddress *ID);
<b>BOOL</b> __stdcall <b>bReceiveBarcode</b> (stAddress *ID, char *barcode);
<b>BOOL</b> __stdcall <b>bSendProdNotFound</b> (stAddress *ID);
<b>BOOL</b> __stdcall <b>bSendProdPriceTC50X</b> (stAddress *ID , char *NameProd, char *PriceProd);
<b>void</b> __stdcall <b>Inet_NtoA</b> (DWORD nIP , char *sIP);
<b>DWORD</b> __stdcall <b>Inet_Addr</b> (char *sIP);
<b>BOOL</b> __stdcall <b>bShowWindow</b> (void);
<b>BOOL</b> __stdcall <b>SendConfiguration</b> (char *ip, char *ip_cliente, char *ip_server, char *mascara, char *linha1, char *linha2, char *linha3, char *linha4, char *tempo);
<b>BOOL</b> __stdcall <b>GetConfiguration</b> (char *ip, char *data);
<b>BOOL</b> __stdcall <b>GetConfigurationNew</b> (char *ip, char *ip_cliente, char *ip_server, char *mascara, char *linha1, char *linha2, char *linha3, char *linha4, char *tempo, char *gateway, char *linha5, char *linha6, char *linha7, char *linha8);
<b>BOOL</b> __stdcall <b>SendPrice</b> (char *ip , char *NameProd, char *PriceProd);
<b>BOOL</b> __stdcall <b>CloseTerminal</b> (stAddress *ID);

Tabela 1: Resumo das Funções da DLL

Campo	Tipo	Definição
IP	DWORD	Endereço IP do ECD
Port	WORD	Porta do <i>Socket</i> conectado no ECD

Tabela 2: Estrutura de Dados stAddress

A DLL (*ecdSweda.dll*) é compatível com outras DLLs (*Sc50X.dll* e *tcd1200i.dll*) utilizadas anteriormente para conexão com os Terminais de Consulta de Preço.

- **BOOL \_\_stdcall bInitialize (HWND Handle):** Inicia o Servidor dos ECDs. Deve ser chamada no início do aplicativo e antes de qualquer outra função da DLL. O *Handle* é o manipulador da janela do aplicativo, que responde as mensagens dos ECDs, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com as DLLs (*Sc50X.dll* e *tcd1200i.dll*);
- **BOOL \_\_stdcall bTerminate (void):** Finaliza o Servidor dos ECDs. Deve ser chamada imediatamente antes do término do aplicativo, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com a DLL (*Sc50X.dll* e *tcd1200i.dll*);
- **void \_\_stdcall vGetMsgWinReceiveDataTC (LPUINT MsgWinReg):** Permite ao aplicativo obter o número da mensagem registrada no Windows, que deve interceptar em seu loop de mensagens do Windows no *handle* da janela cadastrada em *bInitialize*, de forma, a saber, quando há dados para ler dos ECDs. Esta função é compatível com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall bConnect (stAddress\* ID):** Esta rotina verifica na lista de eventos se houve a conexão de algum ECD ao Servidor. Caso algum ECD tenha se conectado, a rotina retorna *TRUE*, e coloca o endereço do ECD na estrutura *ID*. Esta função é compatível com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall bDisconnect (stAddress\* ID):** Esta rotina verifica na lista de eventos se algum ECD se desconectou ao Servidor. Caso algum ECD tenha se desconectado, a rotina retorna *TRUE*, e coloca o endereço do ECD desconectado na estrutura *ID*. Esta função é compatível com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall bReceiveBarcode (stAddress \*ID, char \* barcode):** Esta rotina verifica na lista de eventos se há alguma solicitação de código de barras (*barcode*) pendente. Caso possua esta rotina retorna *TRUE*, e coloca o endereço do ECD na estrutura *ID* e o código de barras em *barcode*. Esta função é compatível com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall bSendProdNotFound (stAddress\* ID):** Esta rotina envia uma mensagem para o ECD identificado pela estrutura *ID*, indicando que o produto não está cadastrado atualmente na base de dados, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com a DLL (*Sc50X.dll*);

- **BOOL \_\_stdcall bSendProdPriceTC50X (stAddress\* ID , char \*NameProd, char \*PriceProd):** Esta rotina envia uma mensagem para o ECD identificado pela estrutura *ID*, indicando o nome e preço do produto, que serão exibidos respectivamente na primeira e segunda linha do *display* do ECD, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com a DLL (*Sc50X.dll*);
- **void \_\_stdcall Inet\_NtoA (DWORD nIP , char \*sIP):** Esta rotina converte um número de IP no formato *DWORD* em um número IP no formato *string* (*XXX.XXX.XXX.XXX*). Esta função é compatível com a DLL (*Sc50X.dll*);
- **DWORD \_\_stdcall Inet\_Addr (char \*sIP):** Esta rotina retorna um IP no formato *DWORD* convertido a partir de um IP no formato *string*. Esta função é compatível com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall bShowWindow (void):** Esta rotina sempre retornará *TRUE*, e a DLL não abrirá uma tela para configuração do ECD, devendo este processo ser realizado pelo aplicativo. Onde esta função está contida na DLL apenas para manter compatibilidade com a DLL (*Sc50X.dll*);
- **BOOL \_\_stdcall SendConfiguration (char \*ip, char \*ip\_cliente, char \*ip\_server, char \*mascara, char \*linha1, char \*linha2, char \*linha3, char \*linha4, char \*tempo):** Através desta rotina o Servidor envia os parâmetros de configuração para um determinado ECD, identificado pelo número de ip no formato *string*, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com a DLL (*tcd1200i.dll*);
- **BOOL \_\_stdcall GetConfiguration (char \*ip, char \*data):** Através desta rotina o Servidor captura os parâmetros de configuração de um determinado ECD, identificado pelo número de ip no formato *string*. Todos os parâmetros serão concatenados em uma única *string*, de acordo com protocolo de mensagem do ECD. Para maiores informações, verifique o item 3 (*Protocolo de Mensagem do ECD*). Esta função é compatível com a DLL (*tcd1200i.dll*);
- **BOOL \_\_stdcall GetConfigurationNew (char \*ip, char \*ip\_cliente, char \*ip\_server, char \*mascara, char \*linha1, char \*linha2, char \*linha3, char \*linha4, char \*tempo, char \*gateway, char \*linha5, char \*linha6, char \*linha7, char \*linha8):** Através desta rotina o Servidor captura os parâmetros de configuração de um determinado ECD, identificado pelo número de ip no formato *string*, havendo sucesso, esta rotina retorna *TRUE*. Os parâmetros (gateway, linha5, linha6, linha7, linha8) não retornarão dados significativos, pois são destinados a implementações futuras. Esta função é fornecida apenas pela DLL (*ecdSweda.dll*);



- **BOOL \_\_stdcall SendPrice (char \*ip , char \*NameProd, char \*PriceProd):** Esta rotina envia uma mensagem para um determinado ECD, identificado pelo número de ip no formato *string*, indicando o nome e preço do produto, que serão exibidos respectivamente, na primeira e segunda linha do *display* do ECD, havendo sucesso, esta rotina retorna *TRUE*. Esta função é compatível com a DLL (*tcd1200i.dll*);
- **BOOL \_\_stdcall CloseTerminal (stAddress \*ID):** Esta rotina desconecta um determinado ECD, identificado pela estrutura *ID*, havendo sucesso na desconexão, retornará *TRUE*. Esta função é fornecida apenas pela DLL (*ecdSweda.dll*);

### 3. Protocolo de Mensagens do ECD1200

O uso da DLL permite a utilização dos ECDs sem a necessidade de dominar o protocolo, porém conhecê-lo ajuda na compreensão das mensagens enviadas pelo ECD.

#### 3.1 Mensagens

A mensagem é qualquer informação lida ou escrita pelo ECD no "Socket" aberto para o Servidor. Na versão atual, todas as mensagens são ASCII iniciadas com caractere *tralha* (" #") enviadas ao *Socket* de comunicação (porta 6500). Os tipos de mensagens são:

- Mensagem de Operação;
- Mensagens de Atividade;
- Mensagens de Configuração.

#### 3.2 Mensagens de Operação

Estas mensagens destinam-se a consulta de preço, segue abaixo na "tabela 3" a descrição destas mensagens:

Mensagem	Objetivo	Origem
#999999999999	Enviar o código de barra	ECD1200
#nfound	Avisar que código de barra não foi cadastrado	Servidor
#Nome   Preço	Enviar o nome e preço do produto	Servidor

Tabela 3: Mensagens de Operação

Na mensagem #999999999999, o valor 999999999999 representa o código de barra lido pelo scanner. Quando esta mensagem chega ao Servidor, o caractere *tralha* é removido e feito uma busca na base de dados. Caso não encontre o produto, é emitida a mensagem #nfound, quando o encontra, é enviada a mensagem composta pelo nome e preço do



produto separado pelo caractere *pipe* ("*|*") que estão cadastrados na base de dados, colocando o nome do produto na primeira linha do *display* do ECD e o preço na segunda.

### 3.3 Mensagens de Atividade

Estas mensagens são referentes à conectividade entre o Servidor e os ECDs, segue abaixo na “*tabela 4*” a descrição destas mensagens:

Mensagem	Objetivo	Origem
#ok	Resposta a uma conexão aberta	Servidor
#ecd-1200 versão	Identifica o tipo de ECD e sua versão	ECD1200
#live?	Pergunta ao ECD se está ativo	Servidor
#live	Avisar que se encontra ativo	ECD1200

Tabela 4: Mensagens de Atividade

Assim que a conexão entre o ECD e o Servidor é realizada, o Servidor manda uma mensagem *#ok*, para que confirme ao ECD que sua conexão foi aceita. Neste ponto, o ECD identifica-se pela mensagem *#ecd-1200*.

A mensagem *#live?* é enviada pelo Servidor para cada ECD de 10 em 10 segundos, registrando sempre sua hora de envio. Em resposta a esta mensagem, o ECD responde com a mensagem *#live*. Ao receber esta última mensagem o Servidor armazena a hora da resposta, se este intervalo de tempo exceder a 50 segundos a conexão é fechada automaticamente pelo Servidor.

### 3.4 Mensagens de Configuração

Estas mensagens destinam-se a realizar a configuração *online* dos ECDs conectados ao Servidor, segue abaixo na “*tabela 5*” a descrição destas mensagens:

Mensagem	Objetivo	Origem
#config?	Solicita ao ECD a configuração atual	Servidor
#configTam_ipsX_ipsTam_ipcX_ipc Tam_msnX_msnTam_linha1X_linha1 Tam_linha2X_linha2Tam_linha3X_linha4 Tam_linha4X_linha4TempExibicao	Envio da sua configuração atual ao Servidor	ECD1200
#rconfigTam_ipsX_ipsTam_ipcX_ipc Tam_msnX_msnTam_linha1X_linha1 Tam_linha2X_linha2Tam_linha3X_linha4	Envio de uma nova configuração ao ECD	Servidor

<b>Tam_linha4X_linha4TempExibicao</b>		
---------------------------------------	--	--

Tabela 5: Mensagens de Configuração

Quando ECD recebe a mensagem *#config?*, ele envia uma mensagem de sua configuração. Onde esta mensagem é formada por uma única *string* que concatena todos os parâmetros de configuração, sendo da seguinte forma:

- Possui um cabeçalho *#config*;
- A separação dos dados na *string* é dada por um caractere que representa o tamanho do dado;
- A sequência dos dados será:
  - Endereço IP do Servidor;
  - Endereço IP do ECD;
  - Máscara de rede do ECD;
  - Texto da linha 1 e texto da linha 2 (*Mensagem Promocional 1*);
  - Texto da linha 3 e linha 4 (*Mensagem Publicitária 2*);
  - E por último, vem um caractere que representa o tempo de exibição em segundos, referente à apresentação do nome do produto e o seu preço no *display* do ECD.

Para exemplificar isto, considere a seguinte configuração:

Configuração	Tamanho da String
Endereço IP do Servidor: <b>192.168.1.8</b>	11
Endereço IP do ECD: <b>192.168.1.12</b>	12
Mascara da Rede: <b>255.255.255.0</b>	13
Linha1: <b>Sweda</b>	5
Linha2: <b>Passe o produto...</b>	18
Linha3: <b>Obrigado</b>	8
Linha4: <b>Volte Sempre...</b>	15
Tempo de Exibição: <b>6 segundos</b>	-

Tabela 6: Exemplo de Configuração

Como o tamanho do dado deve ser armazenando dentro de um caractere (*byte*), e longe da faixa dos caracteres especiais como "\r", "\n" ou "\0", é adicionado o código ASCII do caractere zero ("0").

Desse modo, os tamanhos dos dados da "tabela 6" (11, 12, 13, 5, 18, 8, 15) e o tempo de exibição (6) passam a ser representados pelos respectivos caracteres: ";", "<", "=", "5", "B", "8", "?" e "6", que são respectivamente "0"+11, "0"+12, "0"+13, "0"+5, "0"+18, "0"+8, "0"+15 e "0"+6.

Assim a mensagem de configuração seria a seguinte:

```
"#config;192.168.1.8<192.168.1.12=255.255.255.05SwedaBPasse o produto...8Obrigado?Volte Sempre...6"
```

Para reconfigurar o ECD, o Servidor envia a mensagem com cabeçalho *#rconfig* com dados escritos de maneira semelhante à mensagem *#config*, mostrada anteriormente. Por exemplo:

```
"#rconfig;192.168.1.8<192.168.1.12=255.255.255.05SwedaBPasse o produto...8Obrigado?Volte Sempre...6"
```

Após receber a nova configuração, o ECD atualiza seus arquivos de configuração, fecha a conexão atual, e efetua uma nova conexão ao Servidor com os novos parâmetros.