Manual da DLL – Protocolo Companytec

Manual da DLL - protocolo Companytec

DT 433

1 Introdução

Caro desenvolvedor. Essa biblioteca foi desenvolvida para facilitar a implementação dos equipamentos Companytec em seu software. Disponibilizando todas as funções necessárias para a integração, nossa DLL já está presente em muitos sistemas compatíveis com nossa solução.

A fim de atender as linguagens mais utilizadas na atualidade, nossa DLL possui funções que se adaptam à sua linguagem, para que isso seja possível, existem várias funções que executam o mesmo trabalho, diferenciando-se apenas na sua maneira de passagem de parâmetros e retornos. Entre as linguagens que já utilizam a DLL podemos citar:

- Delphi;
- Visual Basic 6;
- Visual Basic .NET;
- C#;
- C/C++;
- Fox Pro;
- COBOL for Windows.

2 Funções gerais

O conjunto de funções apresentadas neste capitulo representam as primeiras e principais funções criadas nesta DLL, caso o desenvolvedor tenha problemas em tratar o parâmetros passados pelas funções e problemas de tipagem de dados, como a dificuldade em criar as estruturas de dados definidas na DLL, o desenvolvedor poderá optar por outras soluções apresentadas nos capítulos abaixo.

As funções definidas na DLL não são especificas por linguagem, logo o desenvolvedor pode utilizar algumas funções deste capitulo, e quaisquer outra função que ele tiver dificuldade, poderá optar por escolher a função de outros capítulos.

As funções definidas nos próximos capítulos, por exemplo VB6, podem ser utilizados por outras linguagens de programação sem qualquer problema, a criação destes conjuntos de funções especificas se deu em função de restrições impostas pela linguagem alvo.

Abaixo seguem as funções padrões da DLL Companytec:

2.1 Comunicação

2.1.1 Abrir comunicação serial

Função que abre a porta de comunicação serial para envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function InicializaSerial(np:byte): Boolean;
- **Parâmetros de entrada:**
  - Np: Número da porta serial.
- **Retorno da função:**
  - True: conexão aberta com sucesso;
  - False: não foi possível se conectar.

2.1.2 Fechar comunicação serial

Função chamada para finalizar a conexão serial.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0
- **Estrutura:** Function FechaSerial: DWORD;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - 1: conexão fechada com sucesso;
  - 0: não foi possível fechar a comunicação.

2.1.3 Abrir comunicação ethernet (IP)

Função que abre a comunicação ethernet para o envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function InicializaSocket (ip:pchar): Boolean;
- **Parâmetros de entrada:**
  - Ip: Endereço na rede em que será feita a conexão.
- **Retorno da função:**
  - True: conexão aberta com sucesso;
  - False: não foi possível se conectar.

Atenção: Quando utilizada esta função, a porta que irá se conectar à automação é a 1771. Caso seja necessária alguma manutenção utilizando o software HRS Console, é aconselhável fechar o software gerencial.

2.1.4 Abrir comunicação ethernet (IP: porta)

Função que abre a comunicação ethernet para o envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function InicializaSocket2(ip:pchar;porta:integer):Boolean;
- **Parâmetros de entrada:**
  - Ip: Endereço na rede em que será feita a conexão.
  - Porta: Porta em que será aberta a comunicação, por padrão, a porta 2001 da automação é destinada ao sistema gerencial.
- **Retorno da função:**
  - True: conexão aberta com sucesso;
  - False: não foi possível se conectar.

2.1.5 Fechar a comunicação ethernet

Função chamada para finalizar uma conexão do tipo ethernet.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function FechaSocket: boolean;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - True: conexão fechada com sucesso;
  - False: não foi possível fechar a comunicação.

2.2 Abastecimento

2.2.1 Ler abastecimento

Função que lê o abastecimento atual da memória da automação e incrementa o ponteiro de leitura de abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.
- **Estrutura:** Function LeAbastecimento:abast;
- **Parâmetros de entrada:** nenhum.

- **Retorno da função:**
  - Abast: retorna os dados do abastecimento como, preço total, volume abastecido, número do bico e etc, em uma estrutura definida na DLL (veja a definição do tipo “abast” no final deste documento).

2.2.2 Ler abastecimento PAF1

Função utilizada para ler os abastecimentos com identificação de frentista e cliente. Este comando, quando não passado nenhum parâmetro de entrada, executa automaticamente o comando de incremento.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:**
  Function LeAbastecimentoPAF1(autoIncrement:boolean=true):AbastPAF1;
- **Parâmetros de entrada:** nenhum ou false para não enviar o auto incremento.
- **Retorno da função:**
  - AbastPAF1: Retorna os dados do abastecimento como, preço total, volume, número do bico, informações de identfid e etc, em uma estrutura definida na DLL (veja a definição do tipo “abastPAF1” no final deste documento).

2.2.3 Ler abastecimento por registro

Função que tem por finalidade ler um registro (local na memória) específico.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function LeStRegistroFid(NumReg:integer):ShortString;
- **Parâmetros de entrada:**
  - NumReg: número do registro desejado.
- **Retorno da função:**
  - Sucesso: String do abastecimento de acordo com o protocolo Companytec;
  - Falha: falha na execução do comando.

2.2.4 Incrementar ponteiro

Função que tem por finalidade informar ao concentrador que o abastecimento atual já foi lido e armazenado, podendo então assim colocar o ponteiro de leitura no próximo registro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Procedure Incrementa;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** nenhum.

2.2.5 Leitura dos ponteiros de abastecimento

Função utilizada para coletar a posição dos ponteiros de escrita e leitura dos abastecimentos na memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.4.0.
- **Estrutura:** function GetMemoryPointers:MemoryPointers;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - MemoryPointers: retorna a posição dos ponteiros de escrita e leitura em uma estrutura definida na DLL (veja a definição do tipo “MemoryPointers” no final deste documento).

2.3 Visualização

2.3.1 Ler volume ou valor do abastecimento em andamento

Função que tem por finalidade ler os abastecimentos que estão em andamento.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function LeVisualizacao( ):OnLine;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - Online: estrutura composta por 48 posições contendo código de bico e o volume que o mesmo está abastecendo. As posições não utilizadas retornarão no campo “bico”, o valor “00” (veja definição do tipo “online” no final deste documento).

2.3.2 Ler volume ou valor do abastecimento em andamento com identificação

Função utilizada para ler o andamento de um abastecimento com o respectivo código identfid que o autorizou.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function GetVisualizacaoId:shortstring;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - Shortstring: código de bico junto com o código do identificador que o autorizou.

2.4 Identfid

2.4.1 Ler identificador

Função utilizada para coletar o identificador que foi lido pela automação, mas não está gravado em sua memória, neste caso a automação envia o código do identificador ao computador quando este solicitar.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function FidIdent:IFid;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - Ifid: armazena as informações do cartão retornado pelo concentrador.

2.4.2 Incrementar identificador

Função utilizada para incrementar o ponteiro de identificador e passar para o próximo lido que não esteja registrado no concentrador.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Procedure FidIncrementa;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** nenhum.

2.4.3 Gravar identificador

Função utilizada para gravar um identificador na memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:**
  Function saveTagFid (controle1, controle2:char; tag,t1in,t1out,t2in,t2out:string): integer;
- **Parâmetros de entrada:**
  - Controle 1: nível de usuário (verificar tabela de permissões no final deste documento);
  - Controle 2:nível de permissão (verificar tabela de permissões no final deste documento);
  - Tag: código do identificador;
  - T1in: início do turno A (hh:mm);
  - T1out: término do turno A (hh:mm);
  - T2in: início do turno B (hh:mm);
  - T2out: término do turno B (hh:mm);

- **Retorno da função:**
  - INTEGER: posição onde foi armazenado o identificador;
  - 0: falha na execução do comando.

2.4.4 Excluir identificador

Função para deletar um identificador da memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function deleteTagFid (posicao:integer;tag:string): integer;
- **Parâmetros de entrada:**
  - Posição: posição na memória onde se encontra o identificador;
  - Tag: código de identificação do cartão a ser excluído.
- **Retorno da função:**
  - Integer: posição deletada;
  - 0: falha na execução do comando.

2.4.5 Limpar lista de identificadores

Função utilizada para limpar a lista de cartões identificadores da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** procedure clearTagFid;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** nenhum.

2.4.6 Ler identificadores na memória

Função para ler os identificadores da memória da automação através de seu registro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function FidLeRegistro (nro:integer): ShortString;
- **Parâmetros de entrada:**
  - Nro: posição de memória que deseja ser lida.
- **Retorno da função:**
  - Shorstring: retorna a string completa conforme protocolo de comunicação Companytec.

2.4.7 Colocar cartão na lista negra

Esta função tem por finalidade colocar identificadores na lista negra, que seria uma lista onde o concentrador não reconhece os identificadores temporariamente, ou seja, um cartão que está na lista negra não pode desbloquear um bico. Esta lista possuí 20 posições.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function PushBlackList(tag:pchar):boolean;
- **Parâmetros de entrada:**
  - TAG: código de identificador para ser colocado na lista negra.
- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

2.4.8 Remover cartão da lista negra

Esta função tem por finalidade de retirar um cartão da lista negra.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function PopBlackList(tag:pchar):boolean;
- **Parâmetros de entrada:**
  - TAG: código de identificador a ser removido da lista negra.

- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

2.4.9 Limpar lista negra

Esta função tem por finalidade excluir todos os cartões da lista negra, ou seja, limpá-la.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function ClearBlackList():boolean;
- **Parâmetros de entrada:** nenhum;
- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

2.5 Status

2.5.1 Ler status dos bicos

Função utilizada para ler a situação de cada bomba conectada ao equipamento.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function LeStatus( ):multistatus;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**

- Multistatus: estrutura de dados de 48 posições que contém o atual estado das bombas conectadas ao concentrador (veja a definição do tipo "multistatus" no final deste documento).

2.6 Gerenciamento de bombas

2.6.1 Liberar bico

Esta função libera a bomba para abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function AutoLibera (bico:string): Error;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Error: tipo de dados definido pelo usuário;
  - None: comando executado com sucesso (veja definição do tipo ERROR no final deste documento).

2.6.2 Autorizar bico

Esta função autoriza a bomba a realizar apenas um abastecimento (funcional quando a bomba foi colocada anteriormente em modo de bloqueio, após o término do abastecimento, a mesma voltará para o status de bloqueada).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function AutorizaAbast (bico:string): Error;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Error: tipo de dados definido pelo usuário;
  - None: comando executado com sucesso (veja definição do tipo ERROR no final deste documento).

2.6.3 Bloquear bico

Função utilizada para bloquear a bomba para não realizar abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function BloqueiaBico (bico:string): Error;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Error: tipo de dados definido pelo usuário;

- None: comando executado com sucesso (veja definição do tipo ERROR no final deste documento).

2.6.4 Parar abastecimento

Função utilizada para parar o abastecimento, está funcionalidade não está disponível em todas as bombas. Para verificar esta informação, favor consultar o manual da bomba desejada.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function ParaBomba (bico:string): Error;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Error: tipo de dados definido pelo usuário;
  - None: comando executado com sucesso (veja definição do tipo ERROR no final deste documento).

2.6.5 Alterar preço por litro

Função utilizada para alterar o preço da bomba pela automação (o valor será alterado no display somente quando for iniciado um novo abastecimento).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:**

Function AlteraPrecoNivel(bico:string;preco:double;decimais:byte;nivel:integer):error;

- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Preço: preço a ser alterado;
  - Decimais: quantidade de casas decimais;
  - Nível: nível de preço a ser alterado (0, 1 ou 2)
- **Retorno da função:**
  - Error: tipo de dados definido pelo usuário;
  - None: comando executado com sucesso (veja definição do tipo ERROR no final deste documento).

2.6.6 Ler preço por litro

Função utilizada para a leitura de preços e seus níveis.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function LePPLNivel(bico:string; niveis: integer):PPLNivel;
- **Parâmetros de entrada:**
  - Bico: código do bico;

- Níveis: quantidade dos níveis que deseja ler, poder ler somente o nível 0, 1 ou 2.

- **Retorno da função:**
  - Valor dos preços solicitados em uma estrutura PPLNível (veja informação no final deste manual);
  - -1: falha na execução do comando.

2.6.7 Predeterminar/ Presetar

Função utilizada para predeterminar o valor do abastecimento para um referido bico.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function Preset(bico:byte;valor:PChar):ShortString;
- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Valor: total a ser pré-determinado.
- **Retorno da função:**
  - PXX: onde XX é o número do bico;
  - FALHA: falha na execução do comando.

2.6.8 Predeterminar/ Presetar com identificação

Função que executa o comando de predeterminação para liberar o bico com código de identificador.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:**
  function presetIdf(bico: Byte; tag:AnsiString; frentista, autoriza, dinheiro: Boolean; valor: PChar; tempo: Integer):ShortString;
- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Tag: código do identificador;
  - Frentista: flag para configurar o preset como cliente ou frentista;
  - Autoriza: flag para liberar a bomba para abastecimento ou não;
  - Dinheiro: flag para escolher o tipo de preset como dinheiro ou volume;
  - Valor: valor a ser presetado;
  - Tempo: tempo até começar o abastecimento ou que o frentista aproxime o cartão do sensor identfid.
- **Retorno da função:**
  - FXX: onde XX é o número do bico;
  - FALHA: falha na execução do comando.

2.6.9 Ler totalizador em volume

Função utilizada para coletar o totalizador em volume do bico desejado.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function GetEncerranteVolume(bicoInt:byte):ShortString;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Totalizador: comando executado com sucesso;
  - FALHA: falha na execução do comando.

2.6.10 Ler totalizador em valor

Função utilizada para coletar o totalizador em valor do bico desejado.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function GetEncerranteValor(bicoInt:byte):ShortString;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - Totalizador: comando executado com sucesso;
  - FALHA: falha na execução do comando.

2.7 Relógio

2.7.1 Consultar calendário

Esta função retorna o valor que está no relógio do concentrador.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function getclock:string;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - Data no formato DD/MM/AA HH:MM.

2.7.2 Ajustar calendário automaticamente

Função para ajustar o relógio do concentrador utilizando a data e hora do PC.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function SetClock(par:string):boolean;
- **Parâmetros de entrada:**
  - Par: 'AUTO' para o concentrador sincronizar com o relógio do computador.
- **Retorno da função:**

- True: comando executado com sucesso;
- False: falha na execução do comando.

2.7.3 Ajustar calendário para um horário específico

Função utilizada para sincronizar o calendário para um horário escolhido pelo usuário.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function SetIntClock(dia,hora,minuto:byte):Boolean
- **Parâmetros de entrada:**
  - Dia, hora e minuto.
- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

2.7.4 Ajustar calendário de forma estendida

Função utilizada para sincronizar o calendário para um data e horário escolhido pelo usuário.

- **Compatibilidade:** Função compatível a partir da versão 5.1.0.
- **Estrutura:**function setExtendedWatch(ano,mes,dia,diaSemana,hora,minuto,segundo: string):Boolean
- **Parâmetros de entrada:**
  - Ano, mês, dia, dia da semana, hora, minuto e segundo.
- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

3 Funções compatíveis com VB6

As funções compatíveis com VB6 tem o formate VB_nomedafunção. Estas funções podem ser utilizadas pelos usuários de C# passando os parâmetros como referência.

Sistemas que antes foram implementados com VB6 e que agora migraram para o VB NET perceberão que algumas funções não vão funcionar como o desejado, para solucionar este problema, deve-se utilizar as funções compatíveis com .NET.

3.1 Comunicação

3.1.1 Abrir comunicação serial

Esta função abre a porta de comunicações serial para envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_OpenSerial(np: integer):integer;
- **Parâmetros de entrada:**
  - Np: Número da porta serial.
- **Retorno da função:**
  - 1: conexão aberta com sucesso;
  - 0: não foi possível se conectar.

3.1.2 Fechar comunicação serial

Função chamada para finalizar a conexão serial.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_CloseSerial:integer;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - 1: conexão fechada com sucesso;
  - 0: não foi possível fechar a comunicação.

3.1.3 Abrir comunicação ethernet (IP)

Função que abre a porta de comunicação ethernet para o envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_OpenSocket(ip: string):integer;

- **Parâmetros de entrada:**
  - Ip: Endereço na rede em que será feita a conexão.
- **Retorno da função:**
  - 1: conexão aberta com sucesso;
  - 0: não foi possível se conectar.

Atenção: Quando utilizada esta função, a porta que irá se conectar à automação é a 1771. Caso seja necessária alguma manutenção utilizando o software HRS Console, é aconselhável fechar o software gerencial.

3.1.4 Fechar comunicação ethernet

Função chamada para finalizar uma conexão do tipo ethernet.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.

- **Estrutura:** Function VB_CloseSerial:integer;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - 1: conexão fechada com sucesso;
  - 0: não foi possível fechar a comunicação.

3.2 Abastecimento

3.2.1 Ler abastecimento

Função que lê o abastecimento atual da memória da automação e incrementa o ponteiro de leitura de abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_ReadSale(var st:string):integer;
- **Parâmetros de entrada:**
  - St: string que por referência é passada vazia para retornar o abastecimento.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.2.2 Ler abastecimento PAF1

Função utilizada para ler os abastecimentos com identificação de frentista e cliente.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_ReadSalePAF(var st:string):integer;

- **Parâmetros de entrada:**
  - St: string que por referência é passada vazia para retornar o abastecimento.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.2.3 Incrementar ponteiro

Função que tem por finalidade informar ao concentrador que o abastecimento atual já foi lido e armazenado, podendo então assim colocar o ponteiro de leitura no próximo registro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** function VB_NextSale:integer;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** retorna um valor inteiro que pode ser desprezado.

3.3 Status

3.3.1 Ler status dos bicos

Função utilizada para ler a situação de cada bomba conectada ao equipamento.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** function VB_ReadState(var st:string):integer;
- **Parâmetros de entrada:**
  - St: string passada por referência que retorna o estado de cada lado da bomba no formato do protocolo de comunicação Companytec.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4 Gerenciamento de bombas

3.4.1 Liberar bico

Esta função libera a bomba para abastecimentos, mantendo a mesma em status de livre.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_FreePump(bico:string):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.2 Autorizar bico

Esta função autoriza a bomba a realizar apenas um abastecimento (funcional quando a bomba foi colocada anteriormente em modo de bloqueio, após o término do abastecimento, a mesma voltará para o status de bloqueada).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_AutPump(bico:string):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.3 Bloquear bico

Função utilizada para bloquear a bomba para não realizar abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_BlockPump(bico:string):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.4 Parar abastecimento

Função utilizada para parar o abastecimento. Esta funcionalidade não está disponível em todas as bombas. Para verificar esta informação, favor consultar o manual da bomba desejada.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_StopPump(bico: String):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.5 Alterar preço por litro

Função utilizada para alterar o preço da bomba pela automação (o valor será alterado no display somente quando iniciado um novo abastecimento).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** function VB_SetPriceLevel(bico,price,level:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Preço: o preço a ser alterado;
  - Level: nível de preço desejado.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.6 Predeterminar/ Presetar

Função utilizada para predeterminar o valor do abastecimento para um referido bico.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_PresetPump(bico,cash:string):integer;

- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Cash: total a ser pré-determinado com 6 caracteres.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

3.4.7 Ler totalizador em volume

Função que retorna o totalizador em volume do bico que foi passado como parâmetro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_ReadTotalsVolume(bico:string):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
- **Retorno da função:**
  - Totalizador: comando executado com sucesso;
  - -1: falha na execução do comando.

3.4.8 Ler totalizador em valor

Função que retorna o totalizador em valor do bico que foi passado como parâmetro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_ReadTotalsCash(bico:pchar):integer;

- **Parâmetros de entrada:**
  - Bico: código do bico;
- **Retorno da função:**
  - Totalizador: comando executado com sucesso;
  - -1: falha na execução do comando.

3.5 Envio de comando

3.5.1 Enviar comando nativo

Função para realizar a comunicação com o concentrador utilizando comandos do protocolo.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function VB_SendReceiveText(var st:string;timeout:integer):integer;

- **Parâmetros de entrada:**
  - St: comando no formato do protocolo Companytec, por referência;
  - Timeout: tempo para esperar a resposta do concentrador.
- **Retorno da função:**
  - St: retorna a string com a resposta do comando em caso de sucesso;
  - 0: falha na execução do comando.

4 Funções compatíveis com C/C++/C#/VB.NET

As funções compatíveis com C/C++, C# e VB.NET são utilizadas no exemplo de comunicação disponível no kit de desenvolvimento.

O tipo string retornado na DLL é composto de uma sequência de caracteres com o caractere '\0' no final, indicando o final da string, este tipo é compatível com o tipo char.

Quando utilizar as funções deste capítulo para C# e VB.Net, siga o método abaixo:

```
//declaração na DLL

Function C_GetSalePAF():pchar; stdcall;

//declaração

[DllImport("companytec.dll")]

public static extern System.IntPtr C_GetSalePAF() ;

//chamada no código

IntPtr data = C_GetSalePAF();

string str  = Marshal.PtrToStringAnsi(data);
```

4.1 Comunicação

4.1.1 Abrir comunicação serial

Esta função abre a porta de comunicações serial para envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_OpenSerial(np:integer):integer;
- **Parâmetros de entrada:**

- Np: Número da porta serial.

- **Retorno da função:**
  - 1: conexão aberta com sucesso;
  - 0: não foi possível se conectar.

4.1.2 Fechar comunicação serial

Função chamada para finalizar a conexão serial.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_CloseSerial:integer;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - 1: conexão fechada com sucesso;
  - 0: não foi possível fechar a comunicação.

4.1.3 Abrir comunicação ethernet (IP)

Função que abre a porta de comunicação ethernet para o envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_OpenSocket(ip:pchar):integer;
- **Parâmetros de entrada:**
  - Ip: Endereço na rede em que será feita a conexão.
- **Retorno da função:**
  - 1: conexão aberta com sucesso;
  - 0: não foi possível se conectar.

Atenção: Quando utilizada esta função, a porta que irá se conectar à automação é a 1771. Caso seja necessária alguma manutenção utilizando o software HRS Console, é aconselhável fechar o software gerencial.

4.1.4 Abrir comunicação ethernet (IP: porta)

Função que abre a porta de comunicação ethernet para o envio e recebimento de comandos, ela precisa ser chamada somente uma vez, no início da aplicação, somente para dar o start da comunicação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** function C_OpenSocket2(ip:pchar; port:integer):integer;
- **Parâmetros de entrada:**

- Ip: Endereço na rede em que será feita a conexão;
- Porta: porta que deseja abrir a comunicação.

- **Retorno da função:**
  - 1: conexão aberta com sucesso;
  - 0: não foi possível se conectar.

4.1.5 Fechar a comunicação ethernet

Função chamada para finalizar uma conexão do tipo ethernet.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_CloseSocket:integer;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - 1: conexão fechada com sucesso;
  - 0: não foi possível fechar a comunicação.

4.2 Abastecimento

4.2.1 Ler abastecimento

Função que lê o abastecimento atual da memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_GetSale():pchar;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** retorna os dados do abastecimento como preço total, volume abastecido, número do bico e etc, em uma estrutura igual ao do protocolo Companytec.

4.2.2 Ler abastecimento PAF1

Função utilizada para ler os abastecimentos com identificação de frentista e cliente.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_GetSalePAF():pchar;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** retorna os dados do abastecimento como preço total, volume abastecido, número do bico e etc, em uma estrutura igual ao do protocolo Companytec.

4.2.3 Incrementar ponteiro

Função que tem por finalidade informar ao concentrador que o abastecimento atual já foi lido e armazenado, podendo então assim colocar o ponteiro de leitura no próximo registro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Procedure C_NextSale;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** nenhum.

4.2.4 Ler abastecimento por registro

Função que tem por finalidade ler um registro (local na memória) específico.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** function C_ReadRegister(reg:pchar):pchar;
- **Parâmetros de entrada:**
  - reg: número do registro a ser lido.
- **Retorno da função:** resposta de acordo com o comando de leitura de registro de abastecimento do protocolo Companytec.

4.3 Visualização

4.3.1 Ler volume ou valor do abastecimento em andamento

Função que tem por finalidade ler os abastecimentos que estão em andamento, retornando código de bico e o valor ou volume em tempo real.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_Visualize:pchar;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** Para cada abastecimento retorna código de bico e valor ou volume atual.

4.4 Identfid

4.4.1 Ler de identificador

Função utilizada para coletar o identificador que foi lido pela automação, mas não está gravado em sua memória, neste caso a automação envia o código do identificador ao computador quando este solicitar.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** function C_ReadIdf():pChar;
- **Parâmetros de entrada:** nenhum
- **Retorno da função:** retornar a string completa com informações do identificador passado conforme protocolo de comunicação Companytec.

4.4.2 Incrementar identificador

Função utilizada para incrementar o ponteiro de identificador e passar para o próximo lido que não esteja registrado no concentrador.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Procedure C_IncrementIdf;
- **Parâmetros de entrada:** nenhum
- **Retorno da função:** nenhum

4.4.3 Gravar identificador

Função utilizada para gravar um identificador na memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Function C_SaveTagIdf(control1,control2:integer;tag:pchar):integer;
- **Parâmetros de entrada:**
  - Control 1: nível de usuário (verificar tabela de permissões no final deste documento);
  - Control 2: nível de permissão (verificar tabela de permissões no final deste documento);
  - Tag: código do identificador.
- **Retorno da função:**
  - Número do registro: sucesso na exclusão;
  - 0: falha na exclusão

4.4.4 Excluir identificador

Função utilizada para deletar um identificador da memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Function C_DeleteTagIdf(control,position,tag:pchar):integer;
- **Parâmetros de entrada:**
  - Control: nível de funcionário e permissão que o identificador foi cadastrado;
  - Position: posição do identificador na memória da automação;
  - Tag: código do identificador.
- **Retorno da função:**
  - Número do registro: sucesso na exclusão;
  - 0: falha na exclusão.

4.4.5 Limpar lista de identificadores

Função utilizada para limpar a lista de cartões identificadores da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** procedure C_ClearMemoryIdf;
- **Parâmetros de entrada:** nenhum
- **Retorno da função:** nenhum

4.4.6 Colocar cartão na lista negra

Esta função tem por finalidade colocar identificadores na lista negra, que seria uma lista onde o concentrador não reconhece os identificadores temporariamente, ou seja, um cartão que está na lista negra não pode desbloquear um bico. Esta lista possuí 20 posições.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Function C_PushIdfBlackList(tag:pchar):integer;
- **Parâmetros de entrada:**
  - Tag: código de identificador a ser colocado na lista negra.
- **Retorno da função:**
  - 1: sucesso na execução do comando;
  - 0: falha na execução do comando.

4.4.7 Remover cartão da lista negra

Esta função tem por finalidade de retirar um cartão da lista negra.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Function C_RemoveIdfBlackList(tag:pchar):integer;
- **Parâmetros de entrada:**
  - Tag: código de identificador a ser removido da lista negra.
- **Retorno da função:**
  - 1: sucesso na execução do comando;
  - 0: falha na execução do comando.

4.4.8 Ler identificadores da memória

Função para ler os identificadores da memória da automação.

- **Compatibilidade:** Função compatível a partir da versão 5.3.0.
- **Estrutura:** Function C_ReadRegisterIdf(nro:integer):pchar;
- **Parâmetros de entrada:**
  - nro: posição de memória que deseja ser lida.
- **Retorno da função:** retornar a string completa conforme protocolo de comunicação Companytec.

4.5 Status

4.5.1 Ler status dos bicos

Função utilizada para ler a situação de cada bomba conectada ao equipamento.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_readState ():pchar;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:** retorno no mesmo formato da leitura de status presente no protocolo Companytec.

4.6 Gerenciamento de bombas

4.6.1 Liberar bico

Esta função libera a bomba para abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_FreePump(bico:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

4.6.2 Autorizar bico

Função autoriza a bomba a realizar apenas um abastecimento (funcional quando a bomba foi colocada anteriormente em modo de bloqueio, após o término do abastecimento, a mesma voltará para o status de bloqueada).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_AutoPump(bico:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: em caso de sucesso no comando;
  - 0: em caso de falha no comando.

4.6.3 Bloquear bico

Função utilizada para bloquear a bomba para não realizar abastecimentos.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_BlockPump(bico:pchar):integer;

- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

4.6.4 Parar abastecimento

Função utilizada para parar o abastecimento, essa funcionalidade não está disponível em todas as bombas. Para verificar esta informação, favor consultar o manual da bomba desejada.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_StopPump(bico:pchar):pchar;
- **Parâmetros de entrada:**
  - Bico: código do bico.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

4.6.5 Alterar preço por litro

Função utilizada para alterar o preço da bomba pela automação (o valor será alterado no display somente quando iniciado um novo abastecimento).

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_SetPrice(bico,preco:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Preço: preço a ser alterado.
- **Retorno da função:**
  - 1: comando executado com sucesso;
  - 0: falha na execução do comando.

4.6.6 Predeterminar/ Presetar

Função utilizada para predeterminar o valor do abastecimento para um referido bico.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_PresetPump(bico,cash:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
  - Cash: total a ser pré-determinado.
- **Retorno da função:**

- 1: comando executado com sucesso;
- 0: falha na execução do comando.

4.6.7 Ler totalizador em volume

Função que retorna o totalizador em volume do bico que foi passado como parâmetro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_ReadTotalsVolume(bico:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
- **Retorno da função:**
  - Encerrante: comando executado com sucesso;
  - -1: falha na execução do comando.

4.6.8 Ler totalizador em valor

Função que retorna o totalizador em valor do bico que foi passado como parâmetro.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_ReadTotalsCash(bico:pchar):integer;
- **Parâmetros de entrada:**
  - Bico: código do bico;
- **Retorno da função:**
  - Encerrante: comando executado com sucesso;
  - -1: falha na execução do comando.

4.7 Relógio

4.7.1 Consultar calendário

Esta função retorna o valor que está no relógio do concentrador.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_GetClock:pchar;
- **Parâmetros de entrada:** nenhum.
- **Retorno da função:**
  - Data no formato DD/MM/AA HH:MM.

4.7.2 Ajustar calendário automaticamente

Função para ajustar o relógio do concentrador utilizando a data e hora do PC.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_SetClock(par:pchar):boolean;

- **Parâmetros de entrada:**
  - Par: 'AUTO' para o concentrador sincronizar com o relógio do computador ou uma strinf no formato DDHHMM.
- **Retorno da função:**
  - True: comando executado com sucesso;
  - False: falha na execução do comando.

4.8 Envio de comando

4.8.1 Enviar comando nativo

Função para realizar a comunicação com o concentrador utilizando comandos do protocolo.

- **Compatibilidade:** Função compatível a partir da versão 5.0.0.
- **Estrutura:** Function C_SendReceiveText(comando:shortstring):shortstring;
- **Parâmetros de entrada:**
  - Comando: contém o comando no formato Companytec.
- **Retorno da função:**
  - Retorna uma string no formato do protocolo de comunicação Companytec.

5 Códigos de controle para identificadores

5.1 Níveis de usuários

O caractere X nas relações abaixo é apresentado como os níveis de usuários que podem ser dados aos tipos de cartões.

- X1: Tag veículo;
- X2: Tag máquina de lavar;
- X3: Reservado;
- X4: Cliente nível 1;
- X5: Cliente nível 2;
- X6: Cliente nível 3;
- X7: Funcionário nível 1;
- X8: Funcionário nível 2;
- X9: Funcionário nível 3;
- XA: Funcionário nível 4;
- XB: Funcionário nível 5;
- XC: Funcionário nível 6;
- XD: Gerente nível 1;
- XE: Gerente nível 2;

- XF: Controle total.

5.2 Níveis de permissão

Estas permissões representam o caractere “X” dos níveis de usuários.

- 1: Reservado;
- 2: Libera bomba;
- 4: Respeita turno;
- 6: Libera bomba / respeita turno;
- 8: Libera máquina de lavar;
- A: Libera bombas / libera máquina de lavar.

6 Estrutura das respostas

Abast
• value: boolean; • total_dinheiro: currency; • total_litros: double; • PU: currency; • tempo: string[8]; • canal: string[2]; • data: string[10]; • hora: string[5]; • st_full: string[55]; • registro: integer; • encerrante: real; • integridade: boolean; • checksun: boolean.

AbastPAF1

- value: boolean;
- total_dinheiro: currency;
- total_litros: double;
- PU: currency;
- tempo: string[8];
- codbico: string[2];
- numbico: integer;
- numtanque: integer;
- voltanque: integer;
- codcombustivel: integer;
- seriecbc: integer;
- tipocbc: char;
- data: string[10];
- hora: string[5];
- registro: integer;
- encerranteI: double;
- encerranteF: double;
- integridade: boolean;
- checksun: boolean;
- tag1: string[16];
- tag2: string[16].

Online

- litragem: array[1...48] of real;
- bico: array[1…48] of string[2].

Multistatus

- status: array[1…48] of StOptions.

Encerrante

- bico: string[2];
- valor: real.

MemoryPointers

- writePointer: string[4];
- readPointer: string[4].

Manual de funções

DLL - Protocolo Companytec

Companytec Automação e Controle Ltda.

Av. Ferreira Viana, 1421 - Areal - 96080-000 - Pelotas - RS

www.companytec.com.br

Fone: (53) 3284-8129

desenvolvimento@companytec.com.br