IDSysR30 SDK for OS Win32

ID Data Tecnologia Ltda.

Programmer's Documentation

Conteúdo

Introdução

A IDSysR30.dll é uma DLL - Dinamic Link Library do Windows utilizada para interface entre uma aplicações usuária e os equipamentos da ID Data.

Esta DLL foi escrita para permitir que o usuário desenvolva sua própria aplicação, onde possa configurar, enviar e receber dados de um equipamento da ID Data.

Funções

RequestOldestEvent

Descrição:

Solicita o evento mais antigo do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall RequestOldestEvent(unsigned char Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas - Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

MRPPointerIncrement

Descrição:

Solicita o incremento do ponteiro da memória MRP.
O ponteiro da memória MRP deverá apontar para o NSR seguinte ao último NSR lido, quando forem executados os comandos RequestOldestEvent e RequestEventBlock.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
MRPPointerIncrement(unsigned char Address, unsigned char Product, unsigned
char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

AddUser

Descrição:

Solicita inclusão dos dados do usuário.
Esta função envia ao equipamento todos os dados referentes a um novo usuário.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall AddUser(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS,
unsigned char *Username, unsigned long Keycode, unsigned char *BarCode,
unsigned char FacilityCode, unsigned long ProxCode, unsigned char Status,
unsigned char UserType, unsigned long Password, unsigned char DayBirthday,
unsigned char MonthBirthday, unsigned int PhotoSize, unsigned char *Photo,
unsigned int BiometricsSize, unsigned char QuantitySamples, unsigned char
*Biometrics)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (variável) – Ponteiro para o vetor que contém o buffer a ser transmitido

Informações	*Tamanho do Buffer*
Sem foto e sem biometria	104 bytes
Com foto e sem biometria	104 bytes + tamanho da foto
Sem foto e com biometria	104 bytes + (tamanho da biometria x quantidade de amostra)
Com foto e com biometria	104 bytes + (tamanho da foto) + (tamanho da biometria x quantidade de amostra)

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS

Username (52 bytes) - Ponteiro para o vetor que contém o Nome do usuário

Keycode (4 bytes) – Código individual do usuário

BarCode (10 bytes) – Ponteiro para o vetor que contém o valor do Código de Barras do Cartão de Código de Barras do usuário

FacilityCode (1 byte) – Código do Facility Code do Cartão de Proximidade do usuário

ProxCode (4 bytes) – Código do Cartão de Proximidade do usuário

Tipo do Cartão	*ProxCode (4 bytes)*			
Proximidade	00	00	*CC	*CC
Smartcard	*CC	*CC	*CC	*CC

*CC – Código do Cartão

Status (1 bytes) – Tipo de marcação de ponto do usuário (ver em Tabelas – Tabela 4)

UserType (1 byte) – Tipo de usuário (ver em Tabelas – Tabela 5)

Password (4 bytes) – Senha para marcação de ponto

DayBirthday (1 byte) – Dia do aniversário do usuário

MonthBirthday (1 byte) – Mês de aniversário do usuário

PhotoSize (2 bytes) – Tamanho em bytes da foto do usuário (máximo 30,720 bytes)

Photo (variável) – Ponteiro para um vetor em bytes que contém a foto

BiometricSize (2 bytes) – Tamanho em bytes da amostra da biometria (máximo 404 bytes)

QuantitySamples (1 byte) – Quantidade de amostras da biometria

Biometrics (variável) – Ponteiro para o vetor que contém a primeira amostra da biometria, caso existam mais de uma amostra concatenar os vetores.

Valor de Retorno:

Nenhum (void)

ChangeUserData

Descrição:

Solicita alteração de todos os dados do usuário.
Esta função envia ao equipamento todos os dados referentes a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ChangeUserData(unsigned
char Address, unsigned char Product, unsigned char *Buffer, unsigned char
*PIS , unsigned char *PIS_New, unsigned char *Username, unsigned long
Keycode, unsigned char *BarCode, unsigned char FacilityCode, unsigned long
ProxCode, unsigned char Status, unsigned char UserType, unsigned long
Password, unsigned char DayBirthday, unsigned char MonthBirthday, unsigned
int PhotoSize, unsigned char *Photo, unsigned int BiometricsSize, unsigned
char QuantitySamples, unsigned char *Biometrics)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (variável) – Ponteiro para o vetor que contém o buffer a ser transmitido

Informações	*Tamanho do Buffer*
Sem foto e sem biometria	115 bytes
Com foto e sem biometria	115 bytes + tamanho da foto
Sem foto e com biometria	115 bytes + (tamanho da biometria x quantidade de amostra)
Com foto e com biometria	115 bytes + (tamanho da foto) + (tamanho da biometria x quantidade de amostra)

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

PIS_New (5 bytes) – Ponteiro para o vetor que contém o novo valor do PIS

Username (52 bytes) - Ponteiro para o vetor que contém o novo Nome do usuário

Keycode (4 bytes) – Novo código individual do usuário

BarCode (10 bytes) – Ponteiro para o vetor que contém o novo valor do Código de Barras do Cartão de Código de Barras do usuário

FacilityCode (1 byte) – Novo código do Facility Code do Cartão de Proximidade do usuário

ProxCode (4 bytes) – Novo código do Cartão de Proximidade do usuário

Tipo do Cartão	*ProxCode (4 bytes)*

Proximidade	00	00	*CC	*CC
Smartcard	*CC	*CC	*CC	*CC

*CC – Código do Cartão

Status (1 byte) – Novo tipo de marcação de ponto do usuário (ver em Tabelas – Tabela 4)

UserType (1 byte) – Novo tipo de usuário (ver em Tabelas – Tabela 5)

Password (4 bytes) – Nova senha para marcação de ponto

DayBirthday (1 byte) – Novo dia do aniversário do usuário

MonthBirthday (1 byte) – Novo mês de aniversário do usuário

PhotoSize (2 bytes) – Tamanho em bytes da nova foto do usuário (máximo 30,720 bytes)

Photo (variável) – Ponteiro para um vetor em bytes que contém a nova foto do usuário

BiometricSize (2 bytes) – Tamanho em bytes da amostra da nova biometria (máximo 404 bytes)

QuantitySamples (1 byte) – Quantidade de amostras da biometria

Biometrics (variável) – Ponteiro para o vetor que contém a primeira nova amostra da biometria, caso existam mais de uma amostra concatenar os vetores.

Valor de Retorno:

Nenhum (void)

ChangeUserDataPIS

Descrição:

Solicita alteração do PIS do usuário.
Esta função envia ao equipamento o novo PIS referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ChangeUserDataPIS(unsigned char Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS , unsigned char *PIS_New)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (27 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

PIS_New (5 bytes) – Ponteiro para o vetor que contém o novo valor do PIS

Valor de Retorno:

Nenhum (void)

ChangeUserDataName

Descrição:

Solicita alteração do Nome do usuário.
Esta função envia ao equipamento o novo Nome referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ChangeUserDataName(unsigned char Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS, unsigned char *Username)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (75) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

Username (52 bytes) - Ponteiro para o vetor que contém o novo Nome do usuário

Valor de Retorno:

Nenhum (void)

ChangeUserDataBarCode

Descrição:

Solicita alteração do Cartão de Código de Barras do usuário.
Esta função envia ao equipamento o novo Código de Barras referentes a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataBarCode(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned char *BarCode)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – ***Tabela 1)***

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (33 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

BarCode (10 bytes) – Ponteiro para o vetor que contém o novo valor do Código de Barras do Cartão de Código de Barras do usuário

Valor de Retorno:

Nenhum (void)

ChangeUserDataProxCode

Descrição:

Solicita alteração das informações do Cartão de Proximidade ou Smartcard do usuário. Esta função envia ao equipamento dados do Cartão de Proximidade ou Smartcard referentes a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataProxCode(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned char FacilityCode,
unsigned long ProxCode)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (28 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

FacilityCode (1 byte) – Novo código do Facility Code do Cartão de Proximidade do usuário

ProxCode (4 bytes) – Novo código do Cartão de Proximidade do usuário

Tipo do Cartão	***ProxCode (4 bytes)***			
Proximidade	00	00	*CC	*CC
Smartcard	*CC	*CC	*CC	*CC

*CC – Código do Cartão

Valor de Retorno:

Nenhum (void)

ChangeUserDataStatus

Descrição:

Solicita alteração do Status do usuário.
Esta função envia ao equipamento o Novo status referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataStatus(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned char Status)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (24 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

Status (1 byte) – Novo tipo de marcação de ponto do usuário (ver em Tabelas – Tabela 4)

Valor de Retorno:

Nenhum (void)

ChangeUserDataUserType

Descrição:

Solicita alteração do Tipo do usuário.
Esta função envia ao equipamento o novo Tipo referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ChangeUserData(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS,
unsigned char UserType)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (24 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

UserType (1 byte) – Novo tipo de usuário (ver em Tabelas – Tabela 5)

Valor de Retorno:

Nenhum (void)

ChangeUserData

Descrição:

Solicita alteração da Senha do usuário.
Esta função envia ao equipamento a nova Senha referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataPassword(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned long Password)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (27 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

Password (4 bytes) – Nova senha para marcação de ponto

Valor de Retorno:

Nenhum (void)

ChangeUserDataBirthday

Descrição:

Solicita alteração da Data de Aniversário do usuário.
Esta função envia ao equipamento uma nova Data de Aniversário referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataBirthday(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned char DayBirthday,
unsigned char MonthBirthday)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (25 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

*DayBirthday (1 byte) – N**ovo d**ia do aniversário do usuário*

MonthBirthday (1 byte) – Novo mês de aniversário do usuário

Valor de Retorno:

Nenhum (void)

ChangeUserDataPhoto

Descrição:

Solicita alteração da Foto do usuário.
Esta função envia ao equipamento a nova Foto referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataPhoto(unsigned char Address, unsigned char Product, unsigned
char *Buffer, unsigned char *PIS, unsigned int PhotoSize, unsigned char
*Photo)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (25 bytes + tamanho da foto) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

PhotoSize (2 bytes) – Tamanho em bytes da nova foto do usuário (máximo 30,720 bytes)

Photo (variável) – Ponteiro para um vetor em bytes que contém a nova foto do usuário

Valor de Retorno:

Nenhum (void)

ChangeUserDataBiometric

Descrição:

Solicita alteração da Biometria do usuário.
Esta função envia ao equipamento a nova Biometria referente a um usuário já existente para alteração.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ChangeUserDataBimetric(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned char *PIS, unsigned int BiometricsSize,
unsigned char QuantitySamples, unsigned char *Biometrics)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (25 bytes + (quantidade de amostras x tamanho das amostras)) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser alterado

BiometricSize (2 bytes) – Tamanho em bytes da amostra da nova biometria (máximo 404 bytes)

QuantitySamples (1 byte) – Quantidade de amostras da biometria

Biometrics (variável) – Ponteiro para o vetor que contém a primeira nova amostra da biometria, caso existam mais de uma amostra concatenar os vetores.

Valor de Retorno:

Nenhum (void)

DeleteUser

Descrição:

Solicita exclusão do cadastro do usuário.
Esta função exclui o cadastro do usuário informado através do PIS.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall DeleteUser(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (21) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser excluído

Valor de Retorno:

Nenhum (void)

ReadUserData

Descrição:

Solicita dados de um usuário cadastrado.
Esta função envia ao equipamento uma solicitação de leitura de dados referente a um usuário já existente.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ReadUserData(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char *PIS)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (21 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

PIS (5 bytes) – Ponteiro para o vetor que contém o valor do PIS do usuário a ser lido

Valor de Retorno:

Nenhum (void)

SetEmployer

Descrição:

Solicita inclusão/alteração de cadastro de empresa/empregador.
Esta função envia ao equipamento todos os dados referentes a empresa/empregador para inclusão.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall SetEmployer(unsigned char
Address, unsigned char Product, unsigned char *Buffer,     unsigned char
IdentifyType, unsigned char *Identify, unsigned char *CEI, unsigned char
*EmployerName, unsigned char *EmployerAddress)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (278 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

IdentifyType (1 byte) – Tipo de identificador da empresa/empregador (ver em Tabelas – Tabela 6)

Identify (6 bytes) – Identificador da empresa/empregador (CNPJ ou CPF)

CEI (5 bytes) – CEI – Código de Cadastro Específico do INSS

EmployerName (150 bytes) – Ponteiro para o vetor que contém a Razão Social ou Nome do Empregador

EmployerAddress (100 bytes) – Ponteiro para o vetor que contém o Local da Prestação de Serviços

Valor de Retorno:

Nenhum (void)

ReadEmployerData

Descrição:

Solicita os dados da empresa/empregador cadastrados no equipamento.
Esta função envia ao equipamento uma solicitação de leitura dos dados referente à empresa/empregador.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ReadEmployerData(unsigned
char Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

ReadREPStatus

Descrição:

Solicita as informações sobre o Status atual do equipamento.
Esta função envia ao equipamento uma solicitação de leitura dos dados referente à empresa/empregador.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ReadREPStatus(unsigned char
Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

RebootREP

Descrição:

Solicita a reinicialização do equipamento.
Esta função força a reinicialização do equipamento sem afetar os dados previamente gravados no equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall RebootREP(unsigned char
Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

SetDateTime

Descrição:

Solicita alteração da data e hora do equipamento.
Esta função realiza a reconfiguração do relógio interno do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall SetDateTime(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char Day,
unsigned char Month, unsigned int Year, unsigned char Hour, unsigned char
Minute, unsigned char Second)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (23 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Day (1 byte) – Novo valor do Dia do equipamento

Month (1 byte) – Novo valor do Mês do equipamento

Year (2 bytes) – Novo valor do Ano do equipamento

Hour (1 byte) – Novo valor da Hora do equipamento

Minute (1 byte) – Novo valor do Minuto do equipamento

Second (1 byte) – Novo valor do Segundo do equipamento

Valor de Retorno:

Nenhum (void)

SetSavingTime

Descrição:

Solicita a inclusão de data inicial e final do Horário de Verão.
Esta função envia ao equipamento as datas de início de término do Horário de Verão para controle do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall SetSavingTime(unsigned char
Address, unsigned char Product, unsigned char *Buffer, unsigned char
StartDay, unsigned char StartMonth, unsigned char EndDay, unsigned char
EndMonth)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (20 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

StartDay (1 byte) – Dia de início do Horário de Verão

StartMonth (1 byte) – Mês de início do Horário de Verão

EndDay (1 byte) – Dia de término do Horário de Verão

EndMonth (1 byte) – Mês de término do Horário de Verão

Valor de Retorno:

Nenhum (void)

ReadREPConfig

Descrição:

Solicita a configuração atual do equipamento.
Esta função envia uma solicitação de leitura de todos os dados referentes às configurações internas do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ReadREPConfig(unsigned char
Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

SetupREPConfig

Descrição:

Solicita a alteração da configuração do equipamento.
Esta função envia todos os dados referentes às alterações de configuração interna do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall SetupREPConfig(unsigned
char Address, unsigned char Product, unsigned char *Buffer, unsigned char
RollSize, unsigned char EnableBuzzer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (18 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

RollSize (1 byte) – Tamanho (em metros) do rolo de papel para impressão de tickets

EnableBuzzer (1 byte) – Estado do Buzzer (ver em Tabelas – Tabela 7)

Valor de Retorno:

Nenhum (void)

SetupREPCommunication

Descrição:

Solicita a alteração da configuração da comunicação do equipamento.
Esta função envia todos os dados referentes às alterações de configuração de comunicação do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
SetupREPCommunication(unsigned char Address, unsigned char Product,
unsigned char *Buffer, unsigned long IPEquipment, unsigned long
SubnetMask, unsigned long IPGateway, unsigned long IPServer, unsigned int
OfflinePort, unsigned int OnlinePort, unsigned char OperationMode,
unsigned char Baudrate)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (38 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

IPEquipment (4 byte) – Endereço IP do equipamento

SubnetMask (4 byte) – Máscara de Sub-Rede

IPGateway (4 byte) – Endereço IP do Gateway

IPServer (4 byte) – Endereço IP do Servidor

OfflinePort (2 byte) – Número da porta para utilização no modo Off-Line

OnlinePort (2 byte) – Número da porta para utilização no modo On-Line

OperationMode (1 byte) – Modo de operação do equipamento (ver em Tabelas – Tabela 8)

Baudrate (1 byte) – Velocidade de transmissão de dados (ver em Tabelas – Tabela 9)

Valor de Retorno:

Nenhum (void)

RequestAlarm

Descrição:

Solicita a verificação de alguma situação crítica no equipamento.
Esta função envia uma solicitação de verificação de ocorrência de algum alarme, referente aos estados da memória, estados do rolo de papel, violação do equipamento, estado das baterias e impressão de relatórios.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall RequestAlarm(unsigned char
Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

SetupReaderProx

Descrição:

Solicita a alteração da configuração de leitura do Cartão de Proximidade do equipamento.
Esta função envia todos os dados referentes às alterações de configuração de leitura do Cartão de Proximidade interna do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall SetupReaderProx(unsigned
char Address, unsigned char Product, unsigned char *Buffer, unsigned char
QuantBitsWiegand, unsigned char StartBitFacility, unsigned char
DirReadFacility, unsigned char StartBitCode, unsigned char DirReadCode)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (21 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

QuantBitsWiegand (1 byte) – Quantidade de bits Wiegand

StarBitFacility (1 byte) – Bit inicial de leitura do Facility Code

DirReadFacility (1 byte) – Direção de leitura do Facility Code

StarBitCode (1 byte) – Bit inicial de leitura do Código do Cartão

DirReadCode (1 byte) – Direção de leitura do Código do Cartão

Valor de Retorno:

Nenhum (void)

ReadConfigProx

Descrição:

Solicita a configuração de leitura do Cartão de Proximidade atual do equipamento. Esta função envia uma solicitação de leitura de todos os dados referentes às configurações de leitura do Cartão de Proximidade internas do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall ReadConfigProx(unsigned
char Address, unsigned char Product, unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

ReadREPCommunication

Descrição:

Solicita a configuração de comunicação atual do equipamento.
Esta função envia uma solicitação de leitura de todos os dados referentes às configurações de comunicação internas do equipamento.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
ReadREPCommunication(unsigned char Address, unsigned char Product,
unsigned char *Buffer)
```

Parâmetros:

Address (1 byte) – Endereço do Equipamento (ver em Tabelas – Tabela 1)

Product (1 byte) – Código do Produto (ver em Tabelas – Tabela 2)

Buffer (15 bytes) – Ponteiro para o vetor que contém o buffer a ser transmitido

Valor de Retorno:

Nenhum (void)

PacketAvail

Descrição:

Solicita a avaliação do Buffer recebido do equipamento.
Esta função avalia o Buffer recebido do equipamento verificando erros e a execução do comando solicitado.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall PacketAvail(unsigned char
*Buffer)
```

Parâmetros:

Buffer (variável) – Ponteiro para o vetor que contém o buffer recebido e que será analisado

Valor de Retorno – SUCESSO:

Valores	*Descrição*
0	Função executada com sucesso
1	Função executada com sucesso e não há registros a serem lidos
2	Função executada com sucesso e chamar a função GetLogType2 para obtenção dos dados
3	Função executada com sucesso e chamar a função GetLogType3 para obtenção dos dados
4	Função executada com sucesso e chamar a função GetLogType4 para obtenção dos dados
5	Função executada com sucesso e chamar a função GetLogType5 para obtenção dos dados

Valor de Retorno – ERROS:

Valores	*Descrição*
-1	Erro de checksum do cabeçalho
-2	Erro de checksum da área de dados
-3	Comando inválido
-4	Comando não suportado pelo modelo de dispositivo utilizado
-5	Não existem mais eventos
-6	Dados não gravados por memória insuficiente na MT
-7	Dados não gravados por memória insuficiente na MRP
-8	Dados não gravados por memória insuficiente no módulo biométrico
-9	Dados não gravados por ausência da memória MT
-10	Dados não gravados por ausência da memória MRP
-11	Dados não gravados por ausência de módulo biométrico
-12	PIS já existente na memória MT

-13	PIS não existente na memória MT
-14	Falta de papel para impressão
-15	Nível crítico da bateria
-16	Violação do equipamento
-17	Memória MT cheia
-18	Memória MRP cheia
-19	Relatório 24h gerado
-100	Pacote inválido
-101	Buffer vazio
-102	Erro no tamanho da área de dados

GetDataUser

Descrição:

Solicita a os dados do usuário cadastrado no equipamento requisitado na função ReadDataUser.
Esta função deverá ser executada após a função ReadDataUser e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetDataUser(unsigned char
*PIS, unsigned char *Username, unsigned long *Keycode, unsigned char
*BarCode, unsigned char *FacilityCode, unsigned long *ProxCode, unsigned
char *Status, unsigned char *UserType, unsigned long *Password, unsigned
char *DayBirthday, unsigned char *MonthBirthday, unsigned int *PhotoSize,
unsigned char *Photo, unsigned int *BiometricSize, unsigned char
*QuantitySamples, unsigned char *Biometric_Sample1, unsigned char
*Biometric_Sample2)
```

Parâmetros:

PIS (5 bytes) – Ponteiro para o vetor que receberá o valor do PIS

Username (52 bytes) - Ponteiro para o vetor que receberá o Nome do usuário

Keycode (4 bytes) – Ponteiro para a variável que receberá o Código individual do usuário

BarCode (10 bytes) – Ponteiro para o vetor que receberá o valor do Código de Barras do Cartão de Código de Barras do usuário

FacilityCode (1 byte) – Ponteiro para a variável que receberá o Código do Facility Code do Cartão de Proximidade do usuário

ProxCode (4 bytes) – Ponteiro para a variável que receberá o Código do Cartão de Proximidade do usuário

Status (1 bytes) – Ponteiro para a variável que receberá o Tipo de marcação de ponto do usuário

UserType (1 byte) – Ponteiro para a variável que receberá o Tipo de usuário

Password (4 bytes) – Ponteiro para a variável que receberá a Senha para marcação de ponto

DayBirthday (1 byte) – Ponteiro para a variável que receberá o Dia do aniversário do usuário

MonthBirthday (1 byte) – Ponteiro para a variável que receberá o Mês de aniversário do usuário

PhotoSize (2 bytes) – Ponteiro para a variável que receberá o Tamanho em bytes da foto do usuário

Photo (variável) – Ponteiro para o vetor que receberá o valor da Foto em bytes

BiometricSize (2 bytes) – Ponteiro para a variável que receberá o Tamanho em bytes da amostra da biometria

QuantitySamples (1 byte) – Ponteiro para a variável que receberá a Quantidade de amostras da biometria

Biometric_Sample1 (variável) – Ponteiro para o vetor que receberá o primeira amostra da biometria, caso exista.

Biometric_Sample2 (variável) – Ponteiro para o vetor que receberá o segunda amostra da biometria, caso exista.

Valor de Retorno:

Nenhum (void)

GetDataEmployer

Descrição:

Solicita a os dados da empresa cadastrada no equipamento requisitada na função ReadDataEmployer.
Esta função deverá ser executada após a função ReadDataEmployer e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetDataEmployer(unsigned
char *IdentifyType, unsigned char *Identify, unsigned char *CEI, unsigned
char *EmployerName, unsigned char *EmployerAddress)
```

Parâmetros:

IdentifyType (1 byte) – Ponteiro para o variável que receberá o Tipo de identificador da empresa/empregador (ver em Tabelas – Tabela 6)

Identify (6 bytes) – Ponteiro para o vetor que receberá o Identificador da empresa/empregador

CEI (5 bytes) – Ponteiro para o vetor que receberá o CEI - Código de Cadastro Específico do INSS

EmployerName (150 bytes) – Ponteiro para o vetor que receberá a Razão Social ou Nome do Empregador

EmployerAddress (100 bytes) – Ponteiro para o vetor que receberá o Local da Prestação de Serviços

Valor de Retorno:

Nenhum (void)

GetLogType2

Descrição:

Solicita a os dados de registros do tipo 2.
Esta função deverá ser executada após a função RequestOldestEvent e a função PacketAvail com retorno 2.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetLogType2(unsigned long
*NSR, unsigned char *RegType, unsigned char *RegDateDay, unsigned char
*RegDateMonth, unsigned int *RegDateYear, unsigned char *RegTimeHour,
unsigned char *RegTimeMin, unsigned char *IdentifyType, unsigned char
*Identify, unsigned char *CEI, unsigned char *EmployerName, unsigned char
*EmployerAddress)
```

Parâmetros:

NSR (4 bytes) – Ponteiro para variável que vai receber o valor do NSR[1]

RegType (1 byte) – Ponteiro para a variável que vai receber o Tipo de registro

RegDateDay (1 byte) – Ponteiro para a variável que vai receber o Dia do registro

RegDateMonth (1 byte) – Ponteiro para a variável que vai receber o Mês do registro

RegDateYear (2 bytes) – Ponteiro para a variável que vai receber o Ano do registro

RegTimeHour (1 byte) – Ponteiro para a variável que vai receber a Hora do registro

RegTimeMin (1 byte) – Ponteiro para a variável que vai receber o Minuto do registro

IdentifyType (1 byte) – Ponteiro para o variável que receberá o Tipo de identificador da empresa/empregador

Identify (6 bytes) – Ponteiro para o vetor que receberá o Identificador da empresa/empregador

CEI (5 bytes) – Ponteiro para o vetor que receberá o CEI[1]

EmployerName (150 bytes) – Ponteiro para o vetor que receberá a Razão Social ou Nome do Empregador

EmployerAddress (100 bytes) – Ponteiro para o vetor que receberá o Local da Prestação de Serviços

[1] Ver descrição em Nomenclaturas

Valor de Retorno:

Nenhum (void)

GetLogType3

Descrição:

Solicita a os dados de registros do tipo 3.
Esta função deverá ser executada após a função RequestOldestEvent e a função PacketAvail com retorno 3.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetLogType3(unsigned long
*NSR, unsigned char *RegType, unsigned char *RegDateDay, unsigned char
*RegDateMonth, unsigned int *RegDateYear, unsigned char *RegTimeHour,
unsigned char *RegTimeMin, unsigned char *PIS)
```

Parâmetros:

NSR (4 bytes) – Ponteiro para variável que vai receber o valor do NSR[1]

RegType (1 byte) – Ponteiro para a variável que vai receber o Tipo de registro

RegDateDay (1 byte) – Ponteiro para a variável que vai receber o Dia do registro

RegDateMonth (1 byte) – Ponteiro para a variável que vai receber o Mês do registro

RegDateYear (2 bytes) – Ponteiro para a variável que vai receber o Ano do registro

RegTimeHour (1 byte) – Ponteiro para a variável que vai receber a Hora do registro

RegTimeMin (1 byte) – Ponteiro para a variável que vai receber o Minuto do registro

PIS (4 bytes) – Ponteiro para o vetor que receberá o PIS[1]

[1] Ver descrição em Nomenclaturas

Valor de Retorno:

Nenhum (void)

GetLogType4

Descrição:

Solicita a os dados de registros do tipo 4.
Esta função deverá ser executada após a função RequestOldestEvent e a função PacketAvail com retorno 4.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetLogType4(unsigned long
*NSR, unsigned char *RegType, unsigned char *DayBeforeAdjust, unsigned
char *MonthBeforeAdjust, unsigned int *YearBeforeAdjust, unsigned char
*HourBeforeAdjust, unsigned char *MinuteBeforeAdjust, unsigned char
*DayAfterAdjust, unsigned char *MonthAfterAdjust, unsigned int
*YearAfterAdjust, unsigned char *HourAfterAdjust, unsigned char
*MinuteAfterAdjust)
```

Parâmetros:

NSR (4 bytes) – Ponteiro para variável que vai receber o valor do NSR[1]

RegType (1 byte) – Ponteiro para a variável que vai receber o Tipo de registro

DayBeforeAdjust (1 byte) – Ponteiro para a variável que vai receber o valor do Dia antes do ajuste de data e hora

MonthBeforeAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Mês antes do ajuste de data e hora

YeayBeforeAdjust (2 bytes) – Ponteiro para a variável que vai receber o valor do Ano antes do ajuste de data e hora

HourBeforeAdjust (1 byte) - Ponteiro para a variável que vai receber o valor da Hora antes do ajuste de data e hora

MinuteBeforeAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Minuto antes do ajuste de data e hora

DayAfterAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Dia depois do ajuste de data e hora

MonthAfterAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Mês depois do ajuste de data e hora

YearAfterAdjust (2 byte) – Ponteiro para a variável que vai receber o valor do Ano depois do ajuste de data e hora

HourAfterAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Hora depois do ajuste de data e hora

MinuteAfterAdjust (1 byte) - Ponteiro para a variável que vai receber o valor do Minuto depois do ajuste de data e hora

[1] Ver descrição em Nomenclaturas

Valor de Retorno:

Nenhum (void)

GetLogType5

Descrição:

Solicita a os dados de registros do tipo 5.
Esta função deverá ser executada após a função RequestOldestEvent e a função PacketAvail com retorno 5.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetLogType5(unsigned long
*NSR, unsigned char *RegType, unsigned char *RegDateDay, unsigned char
*RegDateMonth, unsigned int *RegDateYear, unsigned char *RegTimeHour,
unsigned char *RegTimeMin, unsigned char *OpType, unsigned char *PIS,
unsigned char *Username)
```

Parâmetros:

NSR (4 bytes) – Ponteiro para variável que vai receber o valor do NSR[1]

RegType (1 byte) – Ponteiro para a variável que vai receber o Tipo de registro

RegDateDay (1 byte) – Ponteiro para a variável que vai receber o Dia do registro

RegDateMonth (1 byte) – Ponteiro para a variável que vai receber o Mês do registro

RegDateYear (2 bytes) – Ponteiro para a variável que vai receber o Ano do registro

RegTimeHour (1 byte) – Ponteiro para a variável que vai receber a Hora do registro

RegTimeMin (1 byte) – Ponteiro para a variável que vai receber o Minuto do registro

OpType (1 byte) – Ponteiro para a variável que vai receber o Tipo de Operação (ver em Tabelas – Tabela 10)

PIS (4 bytes) – Ponteiro para o vetor que receberá o PIS[1]

Username (52 bytes) – Ponteiro para o vetor que vai receber o Nome do usuário

[1] Ver descrição em Nomenclaturas

Valor de Retorno:

Nenhum (void)

GetREPStatus

Descrição:

Solicita os dados de Status do equipamento.
Esta função deverá ser executada após a função ReadREPStatus e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetREPStatus(unsigned long
*MT_Occupied, unsigned long *MT_Free, unsigned long *MRP_Occupied,
unsigned long *MRP_Free, unsigned long *TotalTicketsPrinted, unsigned int
*ActualTicketsPrinted, unsigned int *ActualTicketsToPrint, unsigned int
*KM_Printer, unsigned int *AutonomyPrinter, unsigned int *BatteyVoltage)
```

Parâmetros:

MT_Occupied (4 bytes) – Ponteiro para variável que vai receber o valor da MT[1] ocupada

MT_Free (4 bytes) – Ponteiro para variável que vai receber o valor da MT[1] livre

MRP_Occupied (4 bytes) – Ponteiro para variável que vai receber o valor da MRP[1] ocupada

MRP_Free (4 bytes) – Ponteiro para variável que vai receber o valor da MRP[1] livre

TotalTicketsPrinted (4 bytes) – Ponteiro para variável que vai receber o valor Total de Tickets impressos

ActualTicketsPrinted (2 bytes) – Ponteiro para variável que vai receber o valor Total de Tickets impressos no rolo de papel atual

ActualTicketsToPrint (2 bytes) – Ponteiro para variável que vai receber o valor da Previsão de Tickets a serem impressos no rolo de papel atual

KM_Printer (2 bytes) - Ponteiro para variável que vai receber o valor da Quilometragem da Impressora (em metros)

AutonomyPrinter (2 bytes) - Ponteiro para variável que vai receber o valor da Autonomia da Impressora (em metros)

BatteryVoltage (2 bytes) - Ponteiro para variável que vai receber o valor da Tensão atual da Impressora (em milivolts)

[1] Ver descrição em Nomenclaturas

Valor de Retorno:

Nenhum (void)

GetREPConfig

Descrição:

Solicita os dados de configuração interna do equipamento.
Esta função deverá ser executada após a função ReadREPConfig e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetREPConfig(unsigned char
*RollSize, unsigned char *EnableBuzzer)
```

Parâmetros:

RollSize (1 byte) – Ponteiro para variavel que vai receber o Tamanho (em metros) do rolo de papel para impressão de tickets

EnableBuzzer (1 byte) – Ponteiro para variavel que vai receber o Estado do Buzzer (ver em Tabelas – Tabela 7)

Valor de Retorno:

Nenhum (void)

GetREPCommunication

Descrição:

Solicita os dados de configuração de comunicação atual do equipamento.
Esta função deverá ser executada após a função ReadREPCommunication e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall
GetREPCommunication(unsigned long IPEquipment, unsigned long SubnetMask,
unsigned long IPGateway, unsigned long IPServer, unsigned int OfflinePort,
unsigned int OnlinePort, unsigned char OperationMode, unsigned char
Baudrate)
```

Parâmetros:

IPEquipment (4 byte) – Ponteiro para a variável que vai receber o Endereço IP do equipamento

SubnetMask (4 byte) – Ponteiro para a variável que vai receber a Máscara de Sub-Rede

IPGateway (4 byte) – Ponteiro para a variável que vai receber o Endereço IP do Gateway

IPServer (4 byte) – Ponteiro para a variável que vai receber o Endereço IP do Servidor

OfflinePort (2 byte) – Ponteiro para a variável que vai receber o Número da porta para utilização no modo Off-Line

OnlinePort (2 byte) – Ponteiro para a variável que vai receber o Número da porta para utilização no modo On-Line

OperationMode (1 byte) – Ponteiro para a variável que vai receber o Modo de operação do equipamento (ver em Tabelas – Tabela 8)

Baudrate (1 byte) – Ponteiro para a variável que vai receber o Velocidade de transmissão de dados (ver em Tabelas – Tabela 9)

Valor de Retorno:

Nenhum (void)

GetConfigProx

Descrição:

Solicita os dados de configuração de leitura do cartão de Proximidade do equipamento. Esta função deverá ser executada após a função ReadConfigProx e a função PacketAvail com retorno 0.

Sintaxe C++:

```
extern "C"__declspec(dllimport) void __stdcall GetConfigProx(unsigned char
QuantBitsWiegand, unsigned char StartBitFacility, unsigned char
DirReadFacility, unsigned char StartBitCode, unsigned char DirReadCode)
```

Parâmetros:

QuantBitsWiegand (1 byte) – Ponteiro para a variável que vai receber o valor Quantidade de bits Wiegand

StarBitFacility (1 byte) – para a variável que vai receber o valor Bit inicial de leitura do Facility Code

DirReadFacility (1 byte) – para a variável que vai receber o valor Direção de leitura do Facility Code

StarBitCode (1 byte) – para a variável que vai receber o valor Bit inicial de leitura do Código do Cartão

DirReadCode (1 byte) – para a variável que vai receber o valor Direção de leitura do Código do Cartão

Valor de Retorno:

Nenhum (void)

Nomenclaturas

NSR – Número Sequencial de Registro

Numeração seqüencial gerada pelo equipamento a cada evento gerado.

MT – Memória de Trabalho

Memória física do equipamento para registros de manipulação.

MRP – Memória de Registro de Ponto

Memória física do equipamento para registros permanentes.

CEI – Cadastro Específico do INSS

PIS – Programa de Integração Social

Tabelas

Tabela 1 – Valores para Endereço

Valores		Descrição
HEX	DEC	
0x00	0	Comunicação Ethernet, GPRS ou WiFi
0x01 ~ 0xEF	001 ~ 239	Endereço do equipamento em rede serial
0xF0 ~ 0xFE	240 ~ 254	Endereço multicast (grupo de equipamentos) em rede serial
0xFF	255	Endereço broadcast em rede serial

Tabela 2 – Valores para Código do Produto

Valores		Descrição
HEX	DEC	
0x01	1	ID REP

Tabela 3 – Valores para Tipo de Registro

Valores		Descrição
HEX	DEC	
0x00	0	Todos os registros
0x02	2	Registros do tipo Inclusão/Alteração Empregador
0x03	3	Registros do tipo Marcação de Ponto
0x04	4	Registros do tipo de Ajuste de Data/Hora
0x05	5	Registros do tipo Inclusão/Alteração/Exclusão Empregado

Tabela 4 – Valores para Status

Valores		Descrição (Tipos de Marcação de Ponto)
HEX	DEC	
0x00	0	Digitação do Código
0x01	1	Cartão (Código de Barras, Proximidade ou Smartcard)
0x02	2	Biometria
0x03	3	Código e Biometria
0x04	4	Cartão e Biometria
0x05	5	Código e Senha
0x06	6	Cartão e Senha
0x07	7	Código ou Cartão ou Biometria
0x08	8	Biometria ou Código e biometria
0x09	9	Biometria ou Cartão e biometria
0x0A	10	Código e Biometria e Senha
0x0B	11	Cartão e Biometria e Senha

Tabela 5 – Valores para Tipo de Usuário

Valores		Descrição
HEX	DEC	
0x00	0	Administrador
0x01	1	Usuário normal
0x02	2	Administrador e usuário normal

Tabela 6 – Valores para Tipo de Identificador da Empresa

Valores		Descrição
HEX	DEC	
0x00	0	CNPJ
0x01	1	CPF

Tabela 7 – Valores para Estado do Buzzer

Valores		Descrição
HEX	DEC	
0x00	0	Desabilitado
0x01	1	Habilitado

Tabela 8 – Valores para Modo de Operação

Valores		Descrição
HEX	DEC	
0x00	0	Server
0x01	1	Client/Server

Tabela 9 – Valores para Baudrate

Valores		Descrição
HEX	DEC	
0x00	0	9600
0x01	1	19200
0x02	2	38400
0x03	3	57600
0x04	4	115200
0x05	5	230400

Tabela 10 – Valores para Tipo de Operação

Valores	Descrição
I	Inclusão de usuário no equipamento
A	Alteração de usuário no equipamento
E	Exclusão de usuário no equipamento