Driver Ansaldo STS MicrolokII

Nome do Arquivo	MicrolokII.dll
Fabricante	Ansaldo STS
Equipamentos	Microlok II
Protocolo	Peer
Versão	1.0.10
Última Atualização	05/09/2025
Plataforma	Win32
Dependências	N/A
Leitura com Superblocos	Não
Nível	31315

Introdução

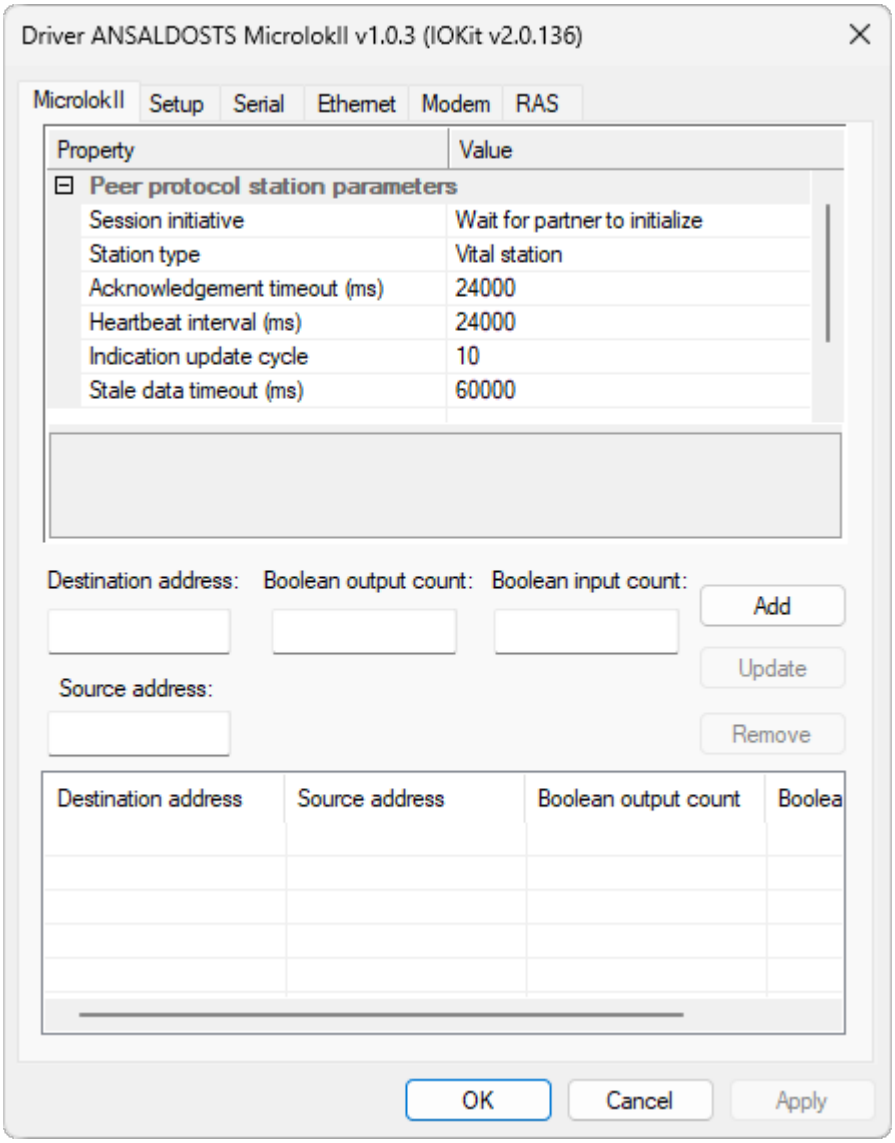
Este Driver implementa o protocolo Peer, permitindo a uma aplicação desenvolvida pela **Elipse Software** comunicar com os equipamentos Microlok II da Ansaldo STS.

Configuração do Driver

Os parâmetros **[P]** de configuração deste Driver não são utilizados. Todas as configurações devem ser realizadas na caixa de diálogo de propriedades deste Driver. Para mais informações sobre as abas de configuração, consulte o tópico **Documentação das Interfaces de Comunicação**.

Configuração das Propriedades

A aba **MicrolokII** contém configurações específicas para este Driver.



Aba MicrolokII

As opções disponíveis nesta aba estão descritas na tabela a seguir.

Opções disponíveis na aba MicrolokII

OPÇÃO	DESCRIÇÃO
Session initiative	Seleciona o modo de inicialização que este Driver tem na rede. Os valores possíveis para esta opção são **Wait for partner to initialize**: Permanece à espera da inicialização de sessão por outro equipamento ou **Initialize proactively**: Este Driver inicializa a sessão com outro equipamento. Selecionar uma destas opções é o comportamento a ser adotado por este Driver com todos os equipamentos da rede
Station type	Tipo de Estação que este Driver se apresenta para a rede. Os valores possíveis para esta opção são **Vital station** ou **Non-vital station**
Acknowledgment timeout (ms)	Esta opção define o tempo, em milissegundos, que uma Estação *peer* emissora aguarda depois de enviar uma mensagem que requer confirmação. O valor ideal para esta opção depende das latências médias e piores latências do *link* de comunicação física e do volume de

OPÇÃO	DESCRIÇÃO
	tráfego de mensagens que o *link* físico transporta. Os valores suportados para esta opção variam entre 50 e 60000 milissegundos. Em muitos casos, pode ser necessário definir esta opção por tentativa e erro. Esta opção deve ser definida para o valor mais baixo, em que os tempos limite de reconhecimento são minimizados e não completamente eliminados. O valor padrão desta opção é o valor definido na opção **Heartbeat interval (ms)**
Heartbeat interval (ms)	Esta opção define a taxa em que uma Estação *peer* tenta se comunicar com um par quando nenhuma alteração está ocorrendo nos bits da saída serial. O intervalo de *heartbeat* deve ser definido como um valor que não seja maior que metade do valor da opção **Stale data timeout (ms)**. Normalmente, esta opção deve ser definida como um valor que não seja inferior à opção **Acknowledgment timeout (ms)**. Esta opção, para uma Estação *peer*, pode ser definida para valores entre 100 e 600000 milissegundos. O valor padrão desta opção é definido como 40% do valor da opção **Stale data timeout (ms)**
Indication update cycle	Esta opção é usada em conjunto com a opção **Heartbeat interval (ms)** para controlar a taxa em que uma Estação *peer* envia mensagens de atualização de indicação para um par quando não há alterações nas saídas seriais para esta Estação. Um ciclo de atualização com o valor 1 (um) permite que uma mensagem de dados de indicação completa seja enviada a cada intervalo de *heartbeat*. Um ciclo de atualização com o valor 2 (dois) permite que uma mensagem de dados de indicação completa seja enviada em todos os outros intervalos de *heartbeat*, e assim por diante. Nos intervalos de *heartbeat* para os quais nenhuma mensagem de dados de indicação é enviada, uma mensagem de status da Estação que não contém dados é enviada. O uso periódico da mensagem de status ao invés da mensagem de dados de indicação reduz o carregamento em um canal de comunicação muito carregado, reduzindo o número de bytes transmitidos por este canal. Esta opção pode ser definida para valores entre 1 (um) e 100 e o valor padrão é 10
Stale data timeout (ms)	Esta opção define o intervalo de tempo em que uma falha de comunicação é declarada quando uma Estação *peer* não recebe dados de indicação ou mensagens de status do par. Este intervalo de tempo, para uma Estação *peer*, é reinicializado para o valor especificado por esta opção cada vez que um dado de indicação ou mensagem de status é recebido da Estação *peer*. Se o temporizador de dados obsoleto de uma Estação *peer* de mesmo nível expirar, o status da Estação é definido para *falhado* e a Estação é definida em estado de inicialização. Uma vez declarado como *falhado*, o *link* de comunicação da Estação não é novamente declarado *ativo* até que a sequência de inicialização da comunicação tenha sido concluída e uma mensagem de dados válida tenha sido recebida. O tempo limite de dados obsoletos para uma

OPÇÃO	DESCRIÇÃO
	Estação *peer* de mesmo nível pode ser definido para valores entre 500 e 600000 milissegundos. Para Estações não vitais, o valor padrão é 60000 ms

Separadamente, há parâmetros para o gerenciamento de conexão na camada de Sessão caso seja utilizada a camada física Ethernet.

Parâmetros para gerenciamento de conexão na camada de Sessão

OPÇÃO	DESCRIÇÃO
Performs an IP switch in case of timeout	Um valor **True** seleciona o comportamento de realizar uma troca pelo endereço IP de *backup*, ou vice-versa, após *n* expirações de tempo. O tempo de espera utilizado é igual ao valor da opção **Acknowledgment timeout** para o caso de este Driver realizar o papel de inicializador da Sessão e é igual ao valor da opção **Stale data timeout** quando este Driver está configurado com a opção **Wait for partner to initialize**. Um valor **False** não realiza troca pelo endereço IP de *backup*, ou vice-versa
IP switch after 'n' timeout(s)	Número de *time-outs* que devem ocorrer para que este Driver realize a troca pelo endereço IP de *backup*, ou vice-versa. O valor mínimo desta opção é 1 (um)
Wait time for new IP switch (ms)	Tempo de espera, em milissegundos, para permitir uma nova requisição de troca de endereço IP. Este tempo de espera impede que várias requisições vindas de múltiplas Sessões sejam chamadas em intervalos de tempo muito curtos. Os valores suportados para esta opção variam entre 0 e 60000 milissegundos. Esta configuração não tem impacto sobre os comandos dos Tags **IO.Ethernet.IPSelect** e **IO.Ethernet.IPSwitch**

Também é necessário configurar um cadastro para cada Estação de destino que se deseja comunicar. O cadastro de Estações é realizado em uma lista com os parâmetros informados na tabela a seguir.

Parâmetros para cadastro de Estações

PARÂMETRO	DESCRIÇÃO
Destination address	Endereço da Estação *peer* de destino
Source address	Endereço da Estação *peer* que este Driver usa para se identificar com um destinatário
Boolean output count	Número de saídas Booleanas. Esta opção refere-se à informação que este Driver comunica sobre si próprio na inicialização e deve coincidir com o valor esperado pelas Estações *peer*
Boolean input count	Número de entradas Booleanas. Esta opção refere-se à informação que a respectiva Estação *peer* comunica sobre si própria na inicialização e deve coincidir com este valor

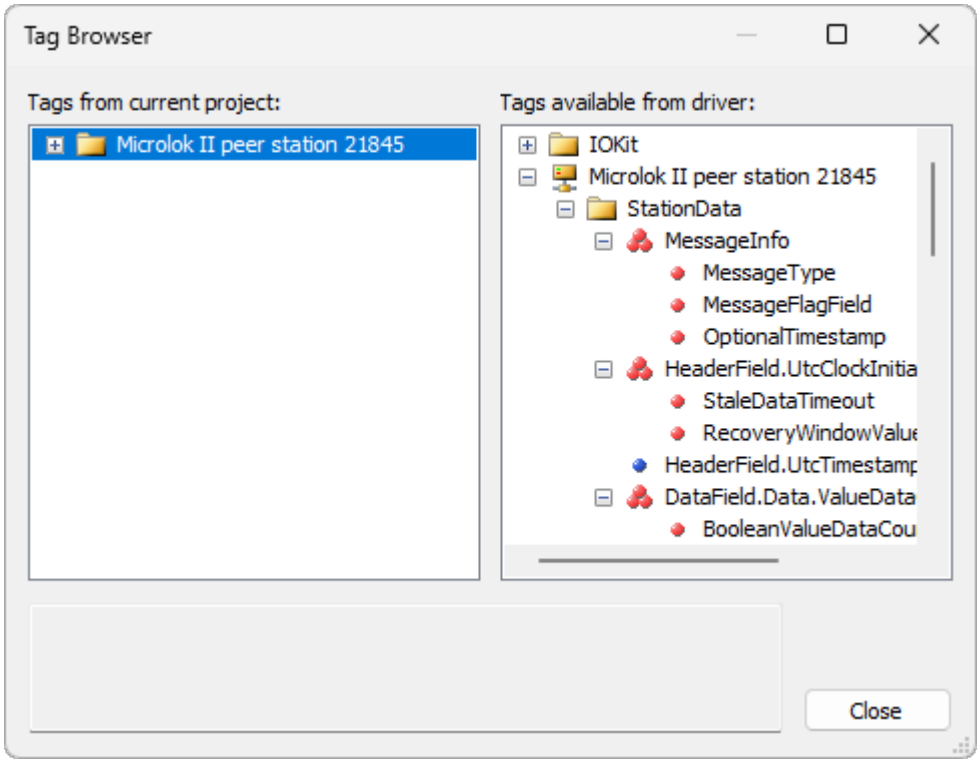
Utilize as opções **Add**, **Update** e **Remove** para adicionar parâmetros, alterar parâmetros já listados ou remover totalmente uma linha de parâmetros, respectivamente.

Referência de Tags

Esta seção contém informações sobre a configuração dos Tags deste Driver.

Utilizando o Tag Browser

O **Elipse E3**, a partir da versão **2.0**, o **Elipse Power** e o **Elipse Water** possuem uma ferramenta chamada **Tag Browser**, que permite que um Driver auxilie o usuário na criação e configuração de Tags.



Tag Browser

A listagem **Tags do projeto corrente** (*Tags from current project*) mostra os Tags e pastas que existem no projeto corrente. A listagem **Tags disponibilizados pelo driver** (*Tags available from driver*) mostra uma árvore com os Tags disponíveis em um Driver. Para criar um Tag novo em uma aplicação, arraste um dos Tags definidos pelo Driver para a pasta desejada no diretório corrente. O Driver Ansaldo STS MicrolokII contém os seguintes nós nesta árvore:

- O nó **IOKit** mostra os Tags disponíveis na biblioteca **IOKit**, divididos nas seguintes categorias:
  - **General**: Tags de uso geral
  - **Modem**: Tags de manipulação da comunicação via modem
  - **Ethernet**: Tags de manipulação da comunicação via rede Ethernet
  - **Parameters**: Tags para configuração dos parâmetros da biblioteca **IOKit**
- O nó **Microlok II peer station #** possui as pastas **StationData**, **MyData** e **Control**, com os respectivos Tags. Em cada uma destas pastas há Tags Bloco ou PLC. Há um nó destes para cada Estação *peer* cadastrada na janela de propriedades deste Driver. Para mais informações, consulte o tópico **Configuração das Propriedades**

O Tag Browser para o Driver Ansaldo STS MicrolokII visa automatizar a configuração dos Tags, auxiliando a carga e configuração de Tags disponíveis para um equipamento, e podem ser carregados tantos equipamentos quantos sejam necessários a um projeto, de forma que cada um destes equipamentos seja configurado pelo respectivo valor de endereço de destino.

Leitura de Seções StationData

Somente Leitura

Dispositivo	Endereço de destino, no formato **<Endereço>:**
Item	Nome da Seção, no formato **<Nome da seção>**

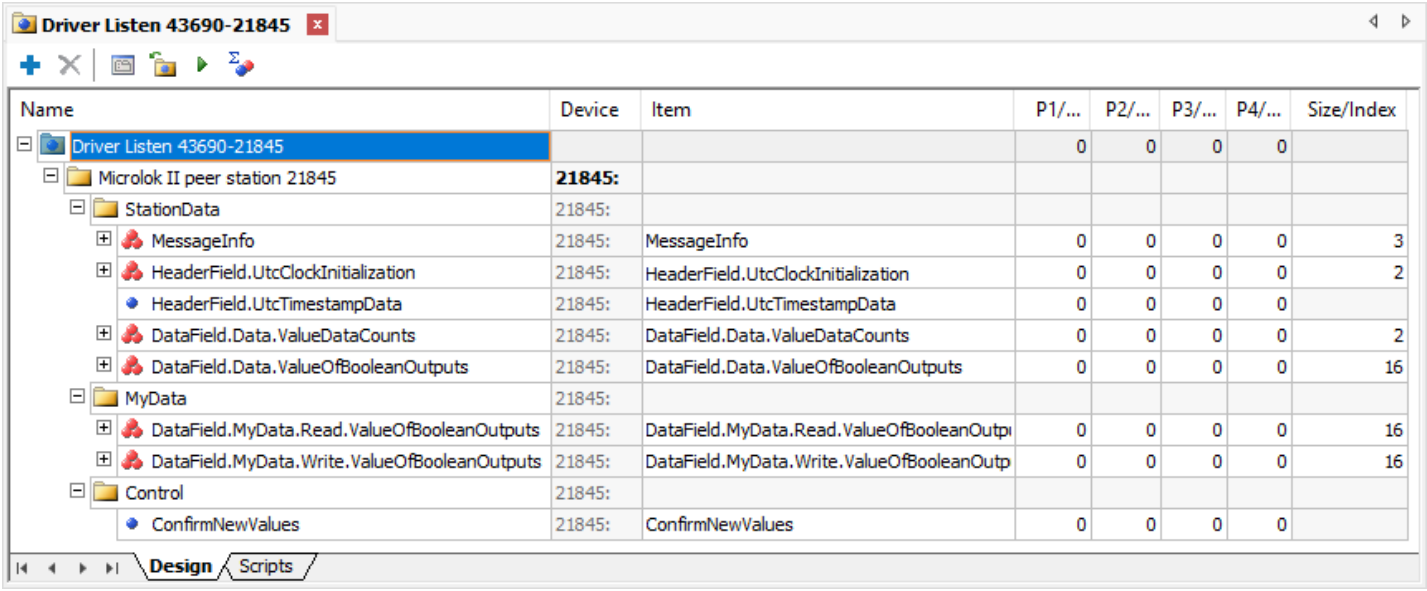
A pasta **StationData** possui um conjunto de Tags cuja origem das informações são as mensagens de dados recebidas da Estação *peer* endereçada. Por isto, estas mensagens são divididas em Seções para cada um dos Tags se encarregar de exibir a respectiva parte das informações. Todas as mensagens de dados vitais ou não vitais, assim que recebidas, são imediatamente disponibilizadas nestes Tags. As Seções **HeaderField.UtcClockInitialization** e **HeaderField.UtcTimestampData** somente existem em mensagens que possuem este campo de cabeçalho. Caso estas Seções não existam nas mensagens, são exibidas vazias.

NOTA

Dada a característica deste Driver de mostrar os valores assim que são recebidos, os tempos de varredura em todos estes Tags são desconsiderados. Desta forma, recomenda-se configurar a propriedade **AdviseType** de cada um dos Tags da pasta **StationData** com o valor 0 (zero, **AlwaysInAdvise**).

Se preferir omitir o campo **Dispositivo**, a sintaxe a seguir é a esperada para o campo **Item**:

```
<Endereço>:<Nome da Seção>
```



Exemplo de Seções

As Seções disponíveis estão descritas nos tópicos a seguir.

MessageInfo

Elementos do Tag Bloco MessageInfo

ELEMENTO	NOME	SIGNIFICADO
1	MessageType	Tipo de mensagem. Os valores possíveis para este Elemento são **1**: Inicialização, **2**: Dados vitais ou **3**: Dados não vitais
2	MessageFlagField	Campo de *flags* no formato **Word**. Os valores possíveis para este Elemento são **Bit 0**: Mensagem *Ack*, **Bit 1**: Mensagem *NAck*, **Bit 2**: *Ack* requerido, **Bit 3**: *Link* estabelecido, **Bit 4**: *Broadcast*, **Bit 5**: Mensagem com estampa de tempo, **Bit 6**: *Reset* de estampa de tempo, **Bit 7**: Requerimento de inicialização e **Bit 8**: Informação de cabeçalho opcional
3	OptionalTimestamp	Estampa de tempo da mensagem, no formato **Date**

NOTA

Utilize as propriedades de bits de 0 (zero) a 8 (oito) no Elemento **MessageFlagField** para acessar individualmente cada *flag*.

HeaderField.UtcClockInitialization

Elementos do Tag Bloco HeaderField.UtcClockInitialization

ELEMENTO	NOME	SIGNIFICADO
1	StaleDataTimeout	**DWord** com o valor de tempo limite de dados obsoletos
2	RecoveryWindowValue	**DWord** com o valor de janela de recuperação

HeaderField.UtcTimestampData

Contém um valor com o tipo de dados **DWord** contendo a estampa de tempo no formato **UTC** (*Coordinated Universal Time*).

DataField.Data.ValueDataCounts

Elementos do Tag Bloco DataField.Data.ValueDataCounts

ELEMENTO	NOME	SIGNIFICADO
1	BooleanValueDataCount	Número de dados de valores Booleanos
2	NumericValueDataCount	Número de dados de valores numéricos

NOTA

O valor do Elemento **BooleanValueDataCount** dividido por 8 (oito) determina o número de Elementos do Tag Bloco **DataField.Data.ValueOfBooleanOutputs**.

DataField.Data.ValueOfBooleanOutputs

Elementos do Tag Bloco DataField.Data.ValueOfBooleanOutputs

ELEMENTO	NOME	SIGNIFICADO
1	BooleanOutputValue0	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 0 (zero) até 7 (sete)
2	BooleanOutputValue1	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 8 (oito) até 15
3	BooleanOutputValue2	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 16 até 23
4	BooleanOutputValue3	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 24 até 31
5	BooleanOutputValue4	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 32 até 39
6	BooleanOutputValue5	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 40 até 47
7	BooleanOutputValue6	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 48 até 55
8	BooleanOutputValue7	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 56 até 63
9	BooleanOutputValue8	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 64 até 71
10	BooleanOutputValue9	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da

ELEMENTO	NOME	SIGNIFICADO
		posição 72 até 79
11	BooleanOutputValue10	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 80 até 87
12	BooleanOutputValue11	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 88 até 95
13	BooleanOutputValue12	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 96 até 103
14	BooleanOutputValue13	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 104 até 111
15	BooleanOutputValue14	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 112 até 119
16	BooleanOutputValue15	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 120 até 127

NOTAS

- Utilize as propriedades de bits de 0 (zero) a 7 (sete) para acessar individualmente cada saída Booleana, no respectivo Elemento deste Tag Bloco.
- O número de saídas Booleanas pode variar de 0 (zero) a 128, assim este Tag Bloco pode variar de tamanho útil de 1 (um) a 16 Elementos, em que cada Elemento comporta de 8 (oito) a 128 saídas. Portanto, pode-se alterar o tamanho do Tag Bloco e remover Elementos para adequar-se ao tamanho verdadeiro.
- Se o número de saídas é igual a 0 (zero), este Tag Bloco é vazio.

Leitura e Escrita MyData

A pasta **MyData** contém os Tags Bloco a seguir, dedicados às saídas Booleanas que este Driver deve informar como Estação *peer*.

- **DataField.MyData.Read.ValueOfBooleanOutputs**: para a leitura dos valores que estão sendo enviados atualmente
- **DataField.MyData.Write.ValueOfBooleanOutputs**: para a escrita dos valores que são atualizados no envio de dados vitais ou não vitais após a confirmação

Enquanto não há escrita de valores pelo Tag Bloco **DataField.MyData.Write.ValueOfBooleanOutputs** e após a confirmação pela escrita do Tag **ConfirmNewValues**, este Driver envia dados vitais ou não vitais para a respectiva Estação *peer* endereçada com todos os valores em 0 (zero).

A escrita do Tag Bloco **DataField.MyData.Write.ValueOfBooleanOutputs** pode ser realizada por Bloco inteiro, por cada Elemento de Bloco separadamente, ou por cada bit de Elemento de Bloco separadamente, ou seja, com a propriedade **UseBitFields** configurada em Verdadeiro.

Desta forma, a sequência de ações deve ser a seguinte:

1. Escrever valores no Tag Bloco **DataField.MyData.Write.ValueOfBooleanOutputs** com uma ou mais operações de escrita.
2. Confirmar os valores com a escrita do Tag **ConfirmNewValues**.
3. Este Driver passa a considerar os valores confirmados para serem enviados ciclicamente à Estação *peer* endereçada.
4. Se necessário, os valores podem ser consultados via Tag Bloco **DataField.MyData.Read.ValueOfBooleanOutputs**.

DataField.MyData.Read.ValueOfBooleanOutputs

Somente Leitura

Dispositivo	Endereço de destino, no formato **<Endereço>:**
Item	**DataField.MyData.Read.ValueOfBooleanOutputs**

Se preferir omitir o campo **Dispositivo**, a sintaxe a seguir é a esperada para o campo **Item**:

```
<Endereço>:DataField.MyData.Read.ValueOfBooleanOutputs
```

Este Tag Bloco exibe todos os valores de saídas Booleanas enviados à Estação *peer* correspondente. Os valores iniciais de todos os Elementos são iguais a 0 (zero).

Elementos do Tag Bloco DataField.MyData.Read.ValueOfBooleanOutputs

ELEMENTO	NOME	SIGNIFICADO
1	BooleanOutputValue0	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 0 (zero) até 7 (sete)
2	BooleanOutputValue1	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 8 (oito) até 15
3	BooleanOutputValue2	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 16 até 23
4	BooleanOutputValue3	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 24 até 31
5	BooleanOutputValue4	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 32 até 39
6	BooleanOutputValue5	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 40 até 47
7	BooleanOutputValue6	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 48 até 55
8	BooleanOutputValue7	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da

ELEMENTO	NOME	SIGNIFICADO
		posição 56 até 63
9	BooleanOutputValue8	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 64 até 71
10	BooleanOutputValue9	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 72 até 79
11	BooleanOutputValue10	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 80 até 87
12	BooleanOutputValue11	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 88 até 95
13	BooleanOutputValue12	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 96 até 103
14	BooleanOutputValue13	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 104 até 111
15	BooleanOutputValue14	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 112 até 119
16	BooleanOutputValue15	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 120 até 127

NOTAS

- Utilize as propriedades de bits de 0 (zero) a 7 (sete) para acessar individualmente cada saída Booleana, no respectivo Elemento deste Tag Bloco.
- O número de saídas Booleanas pode variar de 0 (zero) a 128, assim este Tag Bloco pode variar de tamanho útil de 1 (um) a 16 Elementos, em que cada Elemento comporta de 8 (oito) a 128 saídas. Portanto, pode-se alterar o tamanho do Tag Bloco e remover Elementos para adequar-se ao tamanho verdadeiro.
- Se o número de saídas é igual a 0 (zero), este Tag Bloco é vazio.

DataField.MyData.Write.ValueOfBooleanOutputs

Somente Escrita

Dispositivo	Endereço de destino, no formato **<Endereço>:**
Item	**DataField.MyData.Write.ValueOfBooleanOutputs**

Se preferir omitir o campo **Dispositivo**, a sintaxe a seguir é a esperada para o campo **Item**:

```
<Endereço>:DataField.MyData.Write.ValueOfBooleanOutputs
```

Inicialmente este Driver envia dados, vitais ou não, para a Estação *peer* correspondente endereçada com todos os valores zerados. Novos valores passam a ser enviados após a confirmação do Tag **ConfirmNewValues** com a escrita prévia de valores pelo Tag Bloco **DataField.MyData.Write.ValueOfBooleanOutputs**.

Elementos do Tag Bloco DataField.MyData.Write.ValueOfBooleanOutputs

ELEMENTO	NOME	SIGNIFICADO
1	BooleanOutputValue0	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 0 (zero) até 7 (sete)
2	BooleanOutputValue1	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 8 (oito) até 15
3	BooleanOutputValue2	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 16 até 23
4	BooleanOutputValue3	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 24 até 31
5	BooleanOutputValue4	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 32 até 39
6	BooleanOutputValue5	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 40 até 47
7	BooleanOutputValue6	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 48 até 55
8	BooleanOutputValue7	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 56 até 63
9	BooleanOutputValue8	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 64 até 71
10	BooleanOutputValue9	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 72 até 79
11	BooleanOutputValue10	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 80 até 87
12	BooleanOutputValue11	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 88 até 95
13	BooleanOutputValue12	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 96 até 103
14	BooleanOutputValue13	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da

ELEMENTO	NOME	SIGNIFICADO
		posição 104 até 111
15	BooleanOutputValue14	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 112 até 119
16	BooleanOutputValue15	Elemento de tipo de dados **Byte** contendo saídas Booleanas, da posição 120 até 127

NOTAS

- É possível utilizar as propriedades de bits de 0 (zero) a 7 (sete) para escrever individualmente cada saída Booleana, no respectivo Elemento deste Tag Bloco.
- O número de saídas Booleanas pode variar de 0 (zero) a 128, assim este Tag Bloco pode variar de tamanho útil de 1 (um) a 16 Elementos, em que cada Elemento comporta de 8 (oito) a 128 saídas. Portanto, pode-se alterar o tamanho do Tag Bloco e remover Elementos para adequar-se ao tamanho verdadeiro.
- Se o número de saídas é igual a 0 (zero), este Tag Bloco é vazio.

ConfirmNewValues

Somente Escrita

Dispositivo	Endereço de destino, no formato **<Endereço>:**
Item	**ConfirmNewValues**

Se preferir omitir o campo **Dispositivo**, a sintaxe a seguir é a esperada para o campo **Item**:

```
<Endereço>:ConfirmNewValues
```

Utilize a escrita de qualquer valor neste Tag PLC para confirmar os novos valores escritos em **DataField.MyData.Write.ValueOfBooleanOutputs**. Ao escrever neste Tag PLC, os valores passam a ser utilizados para enviar à Estação *peer* correspondente, e assim o conteúdo do Tag Bloco **DataField.MyData.Read.ValueOfBooleanOutputs** passa a exibir os valores escritos. Esta escrita também provoca o envio imediato dos novos dados para o dispositivo de destino, antecipando o próximo *heartbeat*.

Documentação das Interfaces de Comunicação

Esta seção contém a documentação das Interfaces de Comunicação referente ao Driver **MicrolokII**.

Configurações de um Driver

A configuração das Interfaces de Comunicação é realizada na caixa de diálogo de configuração de um Driver. Para acessar a configuração da caixa de diálogo no **Elipse E3** na versão 1.0, siga estes passos:

1. Clique com o botão direito do mouse em um objeto Driver (IODriver).
2. Selecione o item **Propriedades** no menu contextual.
3. Selecione a aba **Driver**.

4. Clique em **Outros parâmetros**.

No **Elipse E3** versão 2.0 ou posterior, clique em **Configurar o driver** na barra de ferramentas de um Driver. No **Elipse SCADA**, siga estes passos:

1. Abra o Organizer.
2. Selecione um Driver na árvore do Organizer.
3. Clique em **Extras** na aba **Driver**.

Atualmente, as Interfaces de Comunicação permitem que apenas uma conexão seja aberta para cada Driver. Isto significa que, no caso de acesso a duas portas seriais, é preciso adicionar dois Drivers em um aplicação e configurar cada um destes Drivers para cada porta serial.

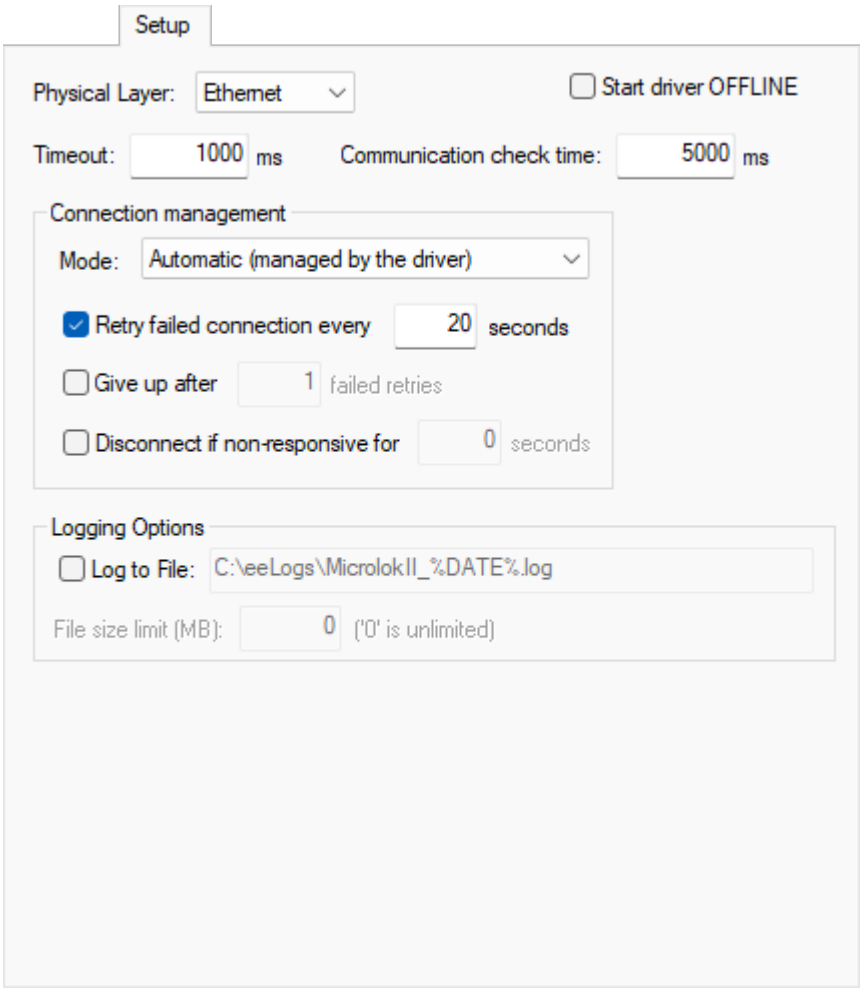
Caixa de Diálogo de Configuração

A caixa de diálogo das Interfaces de Configuração permite configurar a conexão de I/O que é utilizada por um Driver. Esta caixa de diálogo contém as abas **Setup**, **Serial**, **Ethernet**, **Modem** e **RAS** descritas nos tópicos a seguir. Se um Driver não implementa uma conexão de I/O específica, a respectiva aba não está disponível para configuração. Alguns Drivers podem conter abas adicionais, específicas para aquele Driver, na caixa de diálogo de configuração.

Aba Setup

A aba **Setup** contém a configuração geral de um Driver. Esta aba é dividida nos seguintes grupos:

- **Configurações gerais**: Configurações da camada física de um Driver, *time-out* e modo de inicialização
- **Connection management**: Configurações de como a Interface de Comunicação mantém a conexão e qual a política de recuperação em caso de falha
- **Logging options**: Controla a geração dos arquivos de log



Aba Setup

Opções gerais da aba Setup

OPÇÃO	DESCRIÇÃO
Physical Layer	Selecione a interface física em uma lista. As opções disponíveis são **Serial**, **Ethernet**, **Modem** e **RAS**. A interface selecionada deve ser configurada na aba específica
Timeout	Configure o *time-out*, em milissegundos, para a camada física. Esta é a medida de tempo que a interface de I/O aguarda para a recepção de um byte qualquer do *buffer* de recepção
Communication check time	Configure o tempo, em milissegundos, para definir o intervalo em que a comunicação é considerada em estado inativo. Enquanto um Driver de Comunicação receber dados válidos, o estado de comunicação é considerado ativo. Porém, se durante o funcionamento um Driver de Comunicação não receber dados válidos neste período de tempo, o estado é considerado inativo. O estado de comunicação é mostrado no Tag **IO.CommunicationStatus**
Start driver OFFLINE	Selecione esta opção para que um Driver inicie em modo **Offline** ou parado. Isto significa que a interface de I/O não é criada até que se configure um Driver em modo **Online** utilizando-se um Tag em uma aplicação. Este modo possibilita a configuração dinâmica da interface de I/O em tempo de execução

Opções para o grupo Connection management

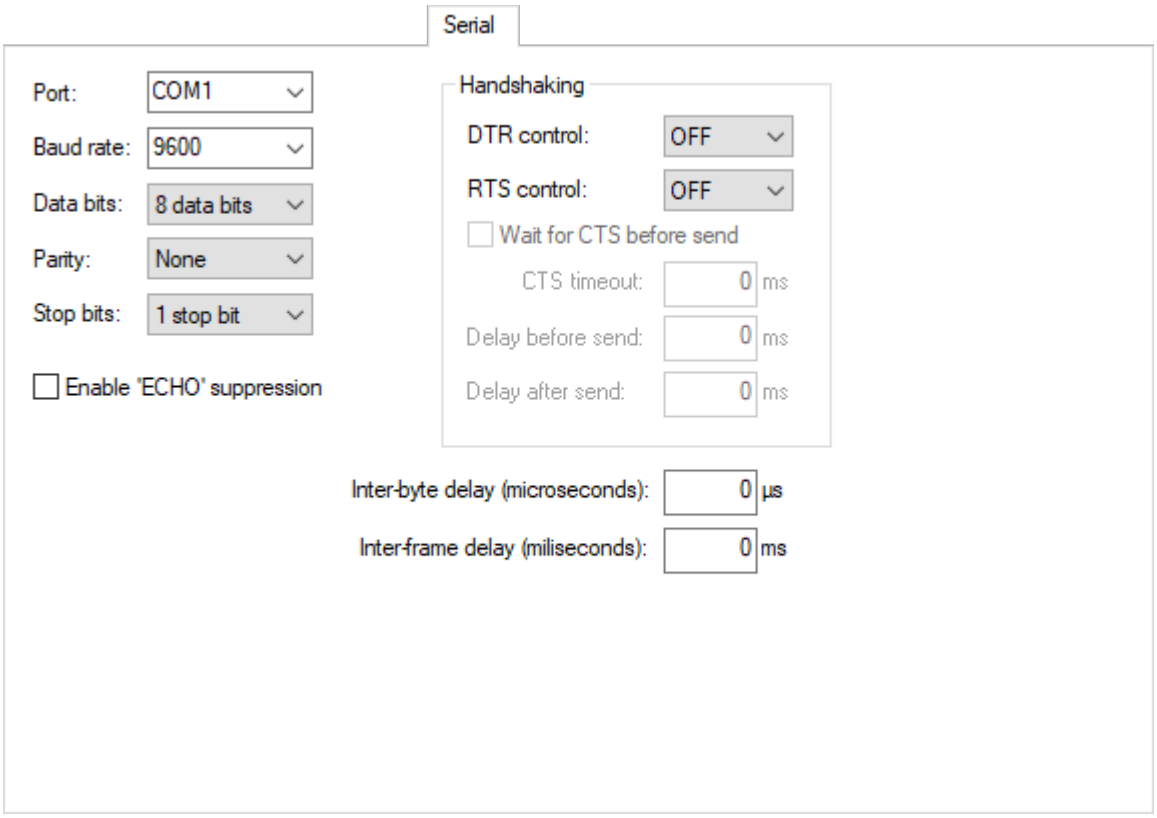
OPÇÃO	DESCRIÇÃO
Mode	Seleciona o modo de gerenciamento de conexão. Selecionar a opção **Automatic** permite que um Driver gerencie a conexão automaticamente, como especificado nas opções seguintes. Selecionar a opção **Manual** permite que uma aplicação gerencie a conexão completamente
Retry failed connection every ... seconds	Selecione esta opção para habilitar a retentativa de conexão de um Driver em um determinado intervalo, em segundos. Se a opção **Give up after failed retries** não está selecionada, este Driver continua retentando até que a conexão seja efetuada, ou que a aplicação seja parada
Give up after ... failed retries	Habilite esta opção para definir um número máximo de retentativas de conexão. Quando o número especificado de tentativas consecutivas de reconexão é atingido, um Driver vai para o modo **Offline**, assumindo que um problema de hardware foi detectado. Se um Driver estabelece uma conexão com sucesso, o número de retentativas sem sucesso é zerado. Se esta nova conexão é perdida, então o contador de retentativas inicia do zero
Disconnect if non-responsive for ... seconds	Habilite esta opção para forçar um Driver a se desconectar se nenhum byte chegou à interface de I/O no *time-out* especificado, em segundos. Este *time-out* deve ser maior que o *time-out* configurado na opção **Timeout**

Opções para o grupo Logging Options

OPÇÃO	DESCRIÇÃO
Log to File	Habilite esta opção e configure o nome do arquivo onde o log é escrito. Arquivos de log podem ser bem extensos, portanto utilize esta opção por curtos períodos de tempo, apenas para o propósito de testes e depurações. Caso se utilize a macro **%PROCESS%** no nome do arquivo de log, esta é substituída pelo identificador do processo atual. Esta opção é particularmente útil ao se utilizar várias instâncias de um mesmo Driver no **Elipse E3**, permitindo assim que cada instância gere um arquivo separado de log. Por exemplo, ao configurar esta opção com o valor "c:\e3logs\drivers\sim_%PROCESS%.log", gera-se um arquivo c:\e3logs\drivers\sim_00000FDA.log para o processo **0FDAh**. Pode-se também utilizar a macro **%DATE%** no nome do arquivo. Neste caso é gerado um arquivo de log por dia, no formato **aaaa_mm_dd**. Por exemplo, ao configurar esta opção com o valor "c:\e3logs\drivers\sim_%DATE%.log", gera-se o arquivo c:\e3logs\drivers\sim_2005_12_31.log em 31/12/2005 e o arquivo c:\e3logs\drivers\sim_2006_01_01.log em 01/01/2006. De forma semelhante, a macro **%DATE_HOUR%** gera um arquivo de log por hora, no formato **aaaa_mm_dd_hh**
File size limit (MB)	Configure o limite de tamanho do arquivo de log, em megabytes. Um valor igual a 0 (zero) significa que não há limite de tamanho para o arquivo de log

Aba Serial

Utilize esta aba para configurar os parâmetros da Interface **Serial**.



Aba Serial

Opções gerais da aba Serial

OPÇÃO	DESCRIÇÃO
Port	Selecione uma porta serial a partir da lista, de **COM1** até **COM4**, ou digite o nome de uma porta serial no formato **COM*n***, como por exemplo "COM15". Ao digitar o nome de uma porta serial manualmente, a caixa de diálogo aceita apenas nomes de portas seriais começando com a expressão "COM"
Baud rate	Selecione um *baud rate* a partir da lista (**1200**, **2400**, **4800**, **9600**, **19200**, **38400**, **57600** ou **115200**) ou digite um *baud rate*, como por exemplo 600
Data bits	Selecione 7 (sete) ou 8 (oito) bits de dados a partir da lista
Parity	Selecione uma paridade a partir da lista. As opções disponíveis são **None**, **Even**, **Odd**, **Mark** ou **List**
Stop bits	Selecione o número de stop bits a partir da lista. As opções disponíveis são **1**, **1.5** ou **2** stop bits
Enable 'ECHO' supression	Habilite esta opção para remover o eco recebido após a Interface de Comunicação enviar dados por uma porta serial. Se o eco não é igual aos bytes recém enviados, a Interface de Comunicação aborta a comunicação
Inter-byte delay (microseconds)	Defina uma espera entre cada byte transmitido pela Interface de Comunicação, em milionésimos de segundo, ou seja, 1000000 é igual a um segundo. Esta opção deve ser utilizada com esperas pequenas de menos de um milissegundo
Inter-frame delay (milliseconds)	Defina uma espera entre pacotes enviados ou recebidos pela Interface de Comunicação, em milésimos de segundo,

OPÇÃO	DESCRIÇÃO
	ou seja, 1000 é igual a um segundo. Esta espera é aplicada caso a Interface de Comunicação envie dois pacotes consecutivos, ou entre um pacote recebido e o próximo envio

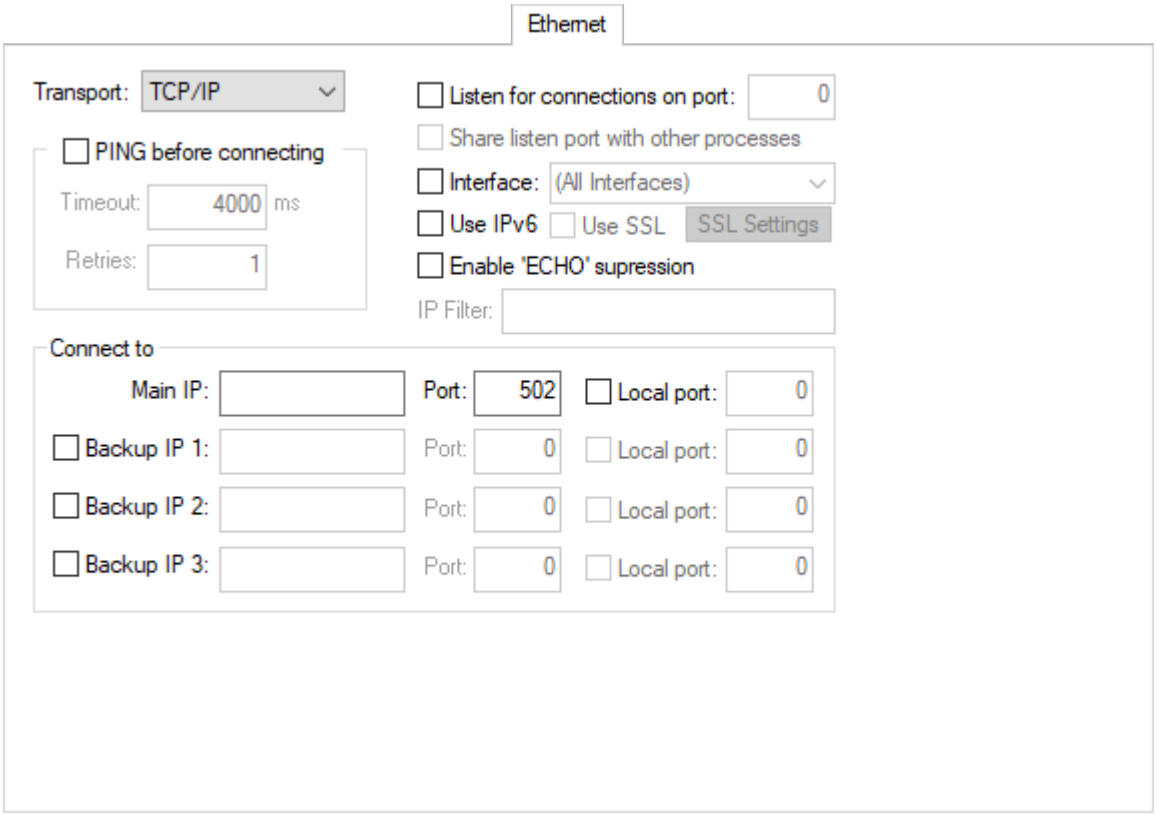
O grupo **Handshaking** configura o uso dos sinais **RTS**, **CTS** e **DTR** no processo de *handshaking* ou seja, controla quando um dado pode ser enviado ou recebido através de uma linha serial. Na maioria das vezes, configurar a opção **DTR control** para **ON** e a opção **RTS control** para **Toggle** funciona tanto com linhas seriais do tipo **RS232** quanto com linhas seriais do tipo **RS485**.

Opções disponíveis no grupo Handshaking

OPÇÃO	DESCRIÇÃO
DTR control	Selecione o valor **ON** para deixar o sinal **DTR** sempre ligado enquanto a porta serial está aberta. Selecione o valor **OFF** para desligar o sinal **DTR** enquanto a porta serial está aberta. Alguns equipamentos exigem que o sinal **DTR** esteja ligado para permitir a comunicação
RTS control	Selecione o valor **ON** para deixar o sinal **RTS** sempre ligado enquanto a porta serial está aberta. Selecione o valor **OFF** para desligar o sinal **RTS** enquanto a porta serial está aberta. Selecione o valor **Toggle** para ligar o sinal **RTS** enquanto se envia os bytes através da porta serial, e desligá-lo quando não se está enviando bytes e, portanto, habilitando a recepção
Wait for CTS before send	Disponível apenas quando a opção **RTS control** está configurada com o valor **Toggle**. Utilize esta opção para forçar um Driver a verificar o sinal **CTS** antes de enviar os bytes através da porta serial, após ligar o sinal de **RTS**. Neste modo o sinal **CTS** é tratado como um *flag* de permissão para envio
CTS timeout	Determina o tempo máximo, em milissegundos, que um Driver aguarda pelo sinal de **CTS** depois de ligar o sinal de **RTS**. Se o sinal de **CTS** não é levantado dentro deste *time-out*, este Driver falha a comunicação atual e retorna erro
Delay before send	Alguns equipamentos de porta serial demoram a habilitar o circuito de envio de dados depois que o sinal **RTS** é ligado. Configure esta opção para aguardar uma determinada quantidade de milissegundos depois de ligar o sinal **RTS** e antes de enviar o primeiro byte. **IMPORTANTE**: Esta espera deve ser utilizada com muito cuidado, pois consome 100% dos recursos de CPU enquanto aguarda. A performance geral do sistema se degrada conforme este valor aumenta
Delay after send	Tem o mesmo efeito que a opção **Delay before send**, mas neste caso a espera é efetuada depois que o último byte é enviado, antes de desligar o sinal **RTS**

Aba Ethernet

Utilize esta aba para configurar os parâmetros da Interface **Ethernet**. Estes parâmetros, exceto as configurações de porta, devem ser também configurados para uso na Interface **RAS**.



Aba Ethernet

Opções disponíveis na aba Ethernet

OPÇÃO	DESCRIÇÃO
Transport	Selecione o valor **TCP/IP** para um *socket* TCP (*stream*) ou selecione o valor **UDP/IP** para utilizar um *socket* UDP (*connectionless datagram*)
Listen for connections on port	Utilize esta opção para aguardar por novas conexões em uma porta IP específica, comum em Drivers Escravos. Caso esta opção permaneça desmarcada, um Driver se conecta ao endereço e porta especificados no grupo **Connect to**
Share listen port with other processes	Selecione esta opção para compartilhar a porta de escuta com outros Drivers e processos
Interface	Selecione a interface de rede local, identificada pelo endereço IP, que um Driver utiliza para efetuar e receber conexões, ou selecione o valor **(All Interfaces)** para permitir conexões em qualquer interface de rede
Use IPv6	Selecione esta opção para forçar um Driver a utilizar endereços no formato **IPv6** em todas as conexões Ethernet. Deixe esta opção desmarcada para utilizar o formato **IPv4**
Enable 'ECHO' suppression	Habilite esta opção para eliminar o *eco* dos dados recebidos. O *eco* é uma cópia dos dados enviados, que pode ser retornada antes da mensagem de resposta

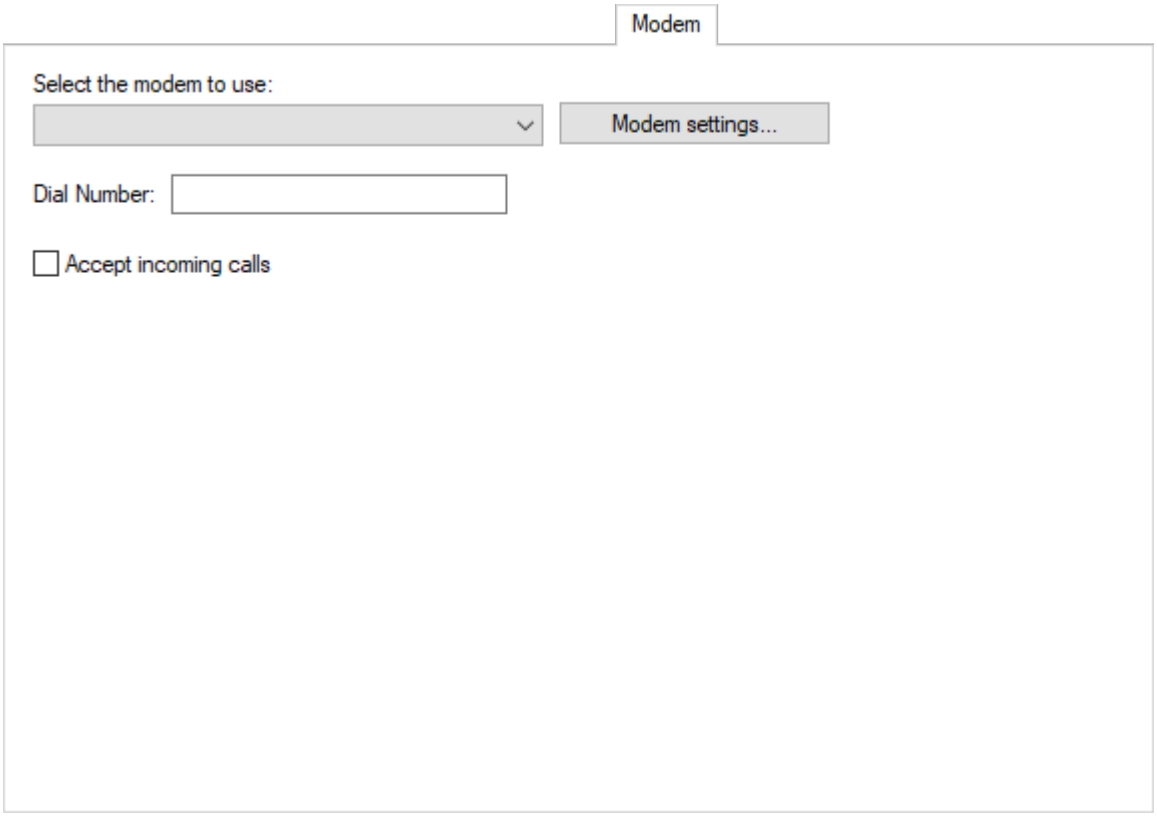
OPÇÃO	DESCRIÇÃO
IP Filter	Lista de endereços IP restringidos ou permitidos de onde um Driver aceita conexões (*Firewall*). Consulte a propriedade **IO.Ethernet.IPFilter** para mais informações
PING before connecting	Habilite esta opção para executar um comando **ping**, ou seja, para verificar se um dispositivo pode ser encontrado na rede, em um dispositivo antes de tentar uma conexão com o *socket*. Esta é uma maneira rápida de determinar uma conexão bem sucedida antes de tentar abrir um *socket* com um dispositivo. O *time-out* de uma conexão com um *socket* pode ser bem alto. As opções disponíveis são: • **Timeout**: Especifique o número de milissegundos de espera por uma resposta de um comando **ping**. Deve-se usar um comando **ping** para verificar o tempo normal de resposta, configurando esta opção para um valor acima desta média. Normalmente pode-se configurar um valor entre 1000 e 4000 milissegundos, ou seja entre 1 (um) e 4 (quatro) segundos • **Retries**: Número de retentativas de um comando **ping**, sem contar a tentativa inicial. Se todas as tentativas falharem, então a conexão com o *socket* é abortada

Opções disponíveis no grupo Connect to

OPÇÃO	DESCRIÇÃO
Main IP	Digite o endereço IP de um dispositivo remoto. Pode-se usar tanto o endereço IP separado por pontos quanto uma URL. No caso de uma URL, um Driver usa o serviço de DNS disponível para mapear a URL para um endereço IP, como por exemplo "192.168.0.13" ou "Server1"
Port	Digite a porta IP de um dispositivo remoto, entre 0 (zero) e 65535
Local port	Selecione esta opção para utilizar uma porta IP local fixa ao conectar a um dispositivo remoto
Backup IP 1, 2 e 3	Indique o endereço IP, a porta IP e a porta IP local fixa de até 3 (três) endereços de *backup* de um dispositivo remoto

Aba Modem

Utilize esta aba para configurar os parâmetros da Interface **Modem**. Algumas opções da aba **Serial** afetam a configuração de um modem, portanto é interessante não esquecer de configurar a Interface **Serial**.



Aba Modem

A Interface **Modem** utiliza os modems TAPI instalados no computador.

Opções disponíveis na aba Modem

OPÇÃO	DESCRIÇÃO
Select the modem to use	Selecione um modem a partir da lista de modems disponíveis no computador. Caso selecione-se o valor **Default modem**, então o primeiro modem disponível é utilizado. Selecionar este valor é recomendado especialmente quando uma aplicação é utilizada em outro computador
Modem settings	Clique para abrir a janela de configuração do modem selecionado
Dial Number	Digite o número padrão para discagem. Este valor pode ser modificado em tempo de execução. Pode-se utilizar o caractere **w** para representar uma pausa ou espera pelo tom de discagem. Por exemplo, "0w33313456" disca o número 0 (zero), espera e então disca o número "33313456"
Accept incoming calls	Habilite esta opção para que um Driver atenda o telefone quando receber uma chamada externa. Para utilizar esta opção é necessário configurar a opção **Connection management** na aba **Setup** para o valor **Manual**

Aba RAS

Use esta aba para configurar os parâmetros da Interface **RAS**. É necessário também configurar a aba **Ethernet**.

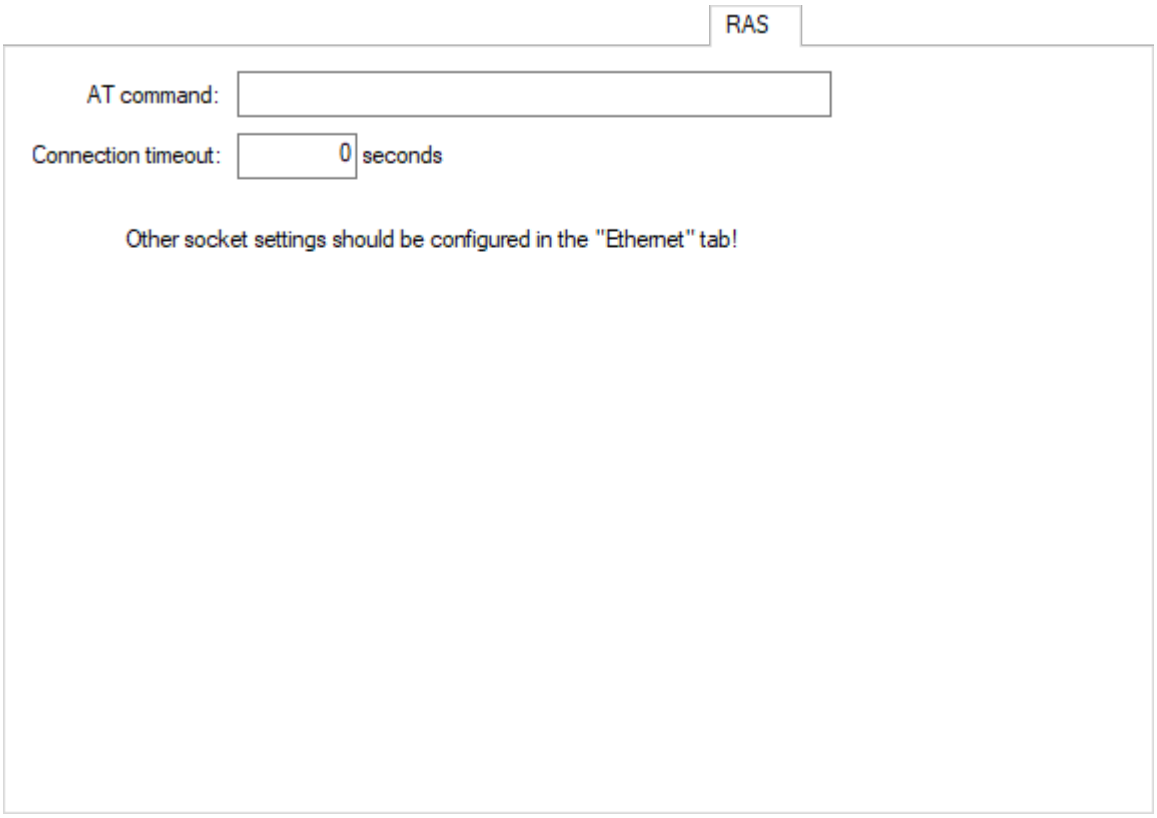
A Interface **RAS** abre uma conexão *socket* com um dispositivo RAS. Um dispositivo RAS é um servidor de modems acessível através de TCP/IP, aguardando por conexões *socket* em uma porta IP. Para cada conexão aceita nesta porta tem-se acesso a um modem.

Ao conectar-se a um dispositivo RAS, primeiramente a Interface de Comunicação conecta ao *socket* no endereço IP e na porta configurados na aba **Ethernet**. Depois que o *socket* é aberto, os passos de inicialização ou de conexão a seguir são efetuados:

1. Limpeza do *socket*, ou seja, remove qualquer mensagem de saudação **TELNET** recebida de um dispositivo RAS.
2. Envio de um comando de discagem **AT**, no formato **ASCII**, no *socket*.
3. Aguarda pela recepção de uma resposta **CONNECT**.
4. Caso o *time-out* expire, a conexão é abortada.
5. Se a resposta **CONNECT** é recebida dentro do *time-out*, o *socket* está disponível para comunicação com um dispositivo, ou seja, a conexão foi estabelecida.

Se o passo 5 (cinco) é efetuado com sucesso, então o *socket* comporta-se como um *socket* normal, com o dispositivo RAS funcionando como um roteador entre um Driver e o dispositivo. Os bytes enviados por um Driver são recebidos pelo dispositivo RAS e enviados para o dispositivo de destino utilizando um modem. Os bytes recebidos pelo dispositivo RAS do modem são enviados de volta a um Driver utilizando o mesmo *socket*.

Depois que a conexão é estabelecida, a Interface **RAS** monitora os dados recebidos por um Driver. Caso uma **String** "NO CARRIER" seja encontrada, o *socket* é fechado. Se o dispositivo RAS não envia o sinal **NO CARRIER**, a Interface **RAS** não consegue detectar quando a conexão modem entre o dispositivo RAS e o dispositivo final de I/O falha. Para recuperação de tal falha é fortemente recomendado que seja habilitada a opção **Disconnect if non-responsive** na aba **Setup**.



Aba RAS

Opções disponíveis na aba RAS

OPÇÃO	DESCRIÇÃO
AT command	Uma **String** com o comando **AT** completo usado para discar para um dispositivo de destino. Por exemplo, "ATDT33313456" disca por tom para o número "33313456"
Connection timeout	Número de segundos a aguardar por uma resposta **CONNECT** do modem, após o envio de um comando **AT**

Configurações Gerais

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** gerais das Interfaces de Comunicação.

Tags de Comunicação

Tags Gerais das Interfaces de Comunicação (N2/B2 = 0)

Os Tags descritos a seguir são fornecidos para todas as Interfaces de I/O suportadas.

IO.CommunicationStatus

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	6 (seis)
Configuração por String	IO.CommunicationStatus

Este Tag informa o estado da comunicação de um Driver. Indica o funcionamento da comunicação em função do recebimento de dados válidos dentro de um período de tempo arbitrado na configuração. Para mais informações, consulte o tópico **Aba Setup**. Os valores possíveis são **0 - Comunicação inativa**: O Driver não recebeu dados válidos ou deixou de receber dados depois de *n* milissegundos, conforme configurado na janela de propriedades, ou **1 - Comunicação ativa**: O Driver está recebendo dados válidos.

IO.IOKitEvent

Tipo de Tag	Tag Bloco
Tipo de Acesso	Somente Leitura
Parâmetro B1	-1 (menos um)
Parâmetro B2	0 (zero)
Parâmetro B3	0 (zero)
Parâmetro B4	1 (um)
Propriedade Size	4 (quatro)
Propriedade ParamItem	IO.IOKitEvent

Este Bloco retorna eventos de Driver gerados por várias fontes nas Interfaces de Comunicação. A propriedade **TimeStamp** de um Bloco representa o momento em que um evento ocorre. Os Elementos de Bloco são os seguintes:

- **Elemento 0**: Tipo de evento. Os valores possíveis são **0**: Informação, **1**: Advertência ou **2**: Erro
- **Elemento 1**: Fonte de um evento. Os valores possíveis são **0**: Driver (específico de um Driver), **-1**: IOKit (eventos genéricos da Interface de Comunicação), **-2**: Interface **Serial**, **-3**: Interface **Modem**, **-4**: Interface **Ethernet** ou **-5**: Interface **RAS**
- **Elemento 2**: Número do erro, específico de cada fonte de evento
- **Elemento 3**: Mensagem de um evento, uma **String** específica de cada evento

NOTA

Um Driver mantém um número máximo de 100 eventos internamente. Se eventos adicionais são reportados, os eventos mais antigos são descartados.

IO.PhysicalLayerStatus

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	2 (dois)
Configuração por String	IO.PhysicalLayerStatus

Este Tag indica o estado da camada física. Os valores possíveis são os seguintes:

- **0**: Camada física parada, ou seja, um Driver está em modo **Offline**, a camada física falhou ao inicializar ou excedeu o número máximo de tentativas de reconexão
- **1**: Camada física iniciada mas não conectada, ou seja, um Driver está em modo **Online**, mas a camada física não está conectada. Se a opção **Connection management** está configurada com o valor **Automatic**, a camada física pode

estar conectando, desconectando ou esperando por uma tentativa de reconexão. Se a opção **Connection management** está configurada com o valor **Manual**, então a camada física permanece neste estado até ser forçada a conectar

- **2**: Camada física conectada, ou seja, a camada física está pronta para ser usada. Isto **NÃO** significa que um equipamento esteja conectado, apenas que a camada de acesso está funcionando

IO.SetConfigurationParameters

Tipo de Tag	Tag Bloco
Tipo de Acesso	Somente Leitura
Parâmetro B1	-1 (menos um)
Parâmetro B2	0 (zero)
Parâmetro B3	0 (zero)
Parâmetro B4	3 (três)
Propriedade Size	2 (dois)
Propriedade ParamItem	IO.SetConfigurationParameters

Use este Tag para modificar qualquer propriedade da caixa de diálogo de configuração de um Driver em tempo de execução.

Este Tag funciona somente enquanto um Driver está em modo **Offline**. Para iniciar um Driver em modo **Offline**, selecione a opção **Start driver OFFLINE** na caixa de diálogo de configuração deste Driver. Pode-se tanto escrever em um Tag PLC ou em um Tag Bloco contendo os parâmetros a serem modificados. As escritas de Elementos de Bloco individuais não são suportadas, um Bloco inteiro precisa ser escrito de uma vez só.

No **Elipse SCADA** é necessário usar um Tag Bloco. Cada parâmetro a ser configurado utiliza dois Elementos de Bloco. Por exemplo, caso seja necessário configurar 3 (três) parâmetros, então o tamanho do Bloco deve ser 6 (seis, 3 × 2). O primeiro Elemento é o nome da propriedade, como uma **String**, e o segundo Elemento é o valor desta propriedade, conforme o exemplo a seguir.

```
// 'Block' deve ser um Tag Bloco com leitura automática,
// leitura por varredura e escrita automática desabilitadas.
// Configura os parâmetros
Block.element001 = "IO.Type" // Parâmetro 1
Block.element002 = "Serial"
Block.element003 = "IO.Serial.Port" // Parâmetro 2
Block.element004 = 1
Block.element005 = "IO.Serial.BaudRate" // Parâmetro 3
Block.element006 = 19200
// Escreve o Bloco inteiro
Block.Write()
```

Ao usar o **Elipse E3**, a habilidade de criar *arrays* em tempo de execução permite o uso tanto de um Tag de Comunicação quanto de um Tag Bloco. Pode-se utilizar o método **Write** de um Driver para enviar os parâmetros diretamente para este Driver, sem a necessidade de criar um Tag, conforme o exemplo a seguir.

```
Dim arr(6)
' Configura os elementos do array
arr(1) = "IO.Type"
arr(2) = "Serial"
arr(3) = "IO.Serial.Port"
arr(4) = 1
arr(5) = "IO.Serial.BaudRate"
arr(6) = 19200
' Há dois métodos de enviar os parâmetros
' Método 1: Usando um Tag de Comunicação
tag.WriteEx arr
' Método 2: Sem utilizar um Tag
Driver.Write -1, 0, 0, 3, arr
```

Uma variação do exemplo anterior usa um *array* bidimensional.

```
Dim arr(10)
' Configura os elementos do array. Note que o array foi redimensionado
' para 10 elementos. Elementos vazios são ignorados pelo Driver
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
Driver.Write -1, 0, 0, 3, arr
```

Um Driver não valida nomes de parâmetros ou valores passados, por isto tenha cuidado ao escrever parâmetros e valores. O método **Write** falha se o *array* de configuração é criado incorretamente. Pode-se consultar o log de um Driver ou usar o parâmetro *writeStatus* do método **WriteEx** para descobrir a causa exata de um erro.

```
Dim arr(10), strError
arr(1) = Array("IO.Type", "Serial")
arr(2) = Array("IO.Serial.Port", 1)
arr(3) = Array("IO.Serial.BaudRate", 19200)
If Not Driver.WriteEx -1, 0, 0, 3, arr, , , strError Then
  MsgBox "Falha ao configurar os parâmetros do Driver: " + strError
End If
```

IO.WorkOnline

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Leitura ou Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	4 (quatro)
Configuração por String	IO.WorkOnline

Este Tag informa o estado atual de um Driver e permite iniciar ou parar a camada física. Os valores possíveis são os seguintes:

- **0 - Driver Offline**: A camada física está fechada ou parada. Este modo permite uma configuração dinâmica dos parâmetros de um Driver através do Tag **IO.SetConfigurationParameters**
- **1 - Driver Online**: A camada física está aberta ou em execução. Enquanto está em modo **Online**, a camada física pode ser conectada ou desconectada e o estado atual pode ser conferido no Tag **IO.PhysicalLayerStatus**

No exemplo a seguir, utilizando o **Elipse E3**, um Driver é colocado em modo **Offline**, a porta COM é modificada e então é colocado em modo **Online** novamente.

```
'Configura o Driver em modo Offline
Driver.Write -1, 0, 0, 4, 0
'Muda a porta para COM2
Driver.Write -1, 0, 0, 3, Array("IO.Serial.Port", 2)
'Configura o Driver em modo Online
Driver.Write -1, 0, 0, 4, 1
```

O método **Write** pode falhar ao configurar um Driver em modo **Online**, ou seja, escrevendo o valor 1 (um). Neste caso, este Driver permanece em modo **Offline**. A causa da falha pode ser:

- Tipo de camada física configurada incorretamente, provavelmente um valor inválido foi configurado para a propriedade **IO.Type**
- Este Driver pode ter ficado sem memória
- A camada física pode ter deixado de criar a *thread* de trabalho. Procure no arquivo de log pela mensagem "Failed to create physical layer thread!"
- A camada física não conseguiu inicializar. A causa da falha depende do tipo de camada física. Pode ser um número de porta serial inválida, falha ao inicializar o Windows Sockets ou falha ao inicializar o TAPI (modem), entre outras. A causa é gravada no arquivo de log

IMPORTANTE

Mesmo que a configuração de um Driver para o modo **Online** seja bem-sucedida, isto não significa necessariamente que a camada física esteja pronta para uso, ou seja, pronta para executar operações de entrada e saída com um equipamento externo. O Tag **IO.PhysicalLayerStatus** deve ser verificado para assegurar que a camada física esteja conectada e preparada para a comunicação.

Propriedades

Estas são as propriedades gerais de todas as Interfaces de I/O suportadas.

IO.ConnectionMode

9 Controla o modo de gerenciamento da Conexão. Os valores possíveis são **0**: Modo automático, em que um Driver gerencia a conexão ou **1**: Modo manual, em que uma aplicação gerencia a conexão.

IO.GiveUpEnable

◩ Quando configurada para Verdadeiro, define um número máximo de tentativas de reconexão. Se todas as reconexões falharem, um Driver entra em modo **Offline**. Se configurada para Falso, um Driver tenta até que uma reconexão seja bem-sucedida.

IO.GiveUpTries

9 Número de tentativas de reconexão antes que esta seja abortada. Por exemplo, se o valor desta propriedade é igual a 1 (um), um Driver tenta apenas uma reconexão quando a conexão é perdida. Se esta falhar, este Driver entra em modo **Offline**.

IO.InactivityEnable

◩ Configure em Verdadeiro para habilitar e em Falso para desabilitar a detecção de inatividade. A camada física é desconectada se está inativa por um certo período de tempo. A camada física é considerada inativa apenas se é capaz de enviar dados mas não de recebê-los de volta.

IO.InactivityPeriodSec

Número de segundos para a verificação de inatividade. Se a camada física está inativa por este período de tempo, então é desconectada.

IO.RecoverEnable

Configure em Verdadeiro para habilitar um Driver a recuperar conexões perdidas e em Falso para deixar um Driver em modo **Offline** quando uma conexão é perdida.

IO.RecoverPeriodSec

Tempo de espera entre duas tentativas de conexão, em segundos.

NOTA

A primeira reconexão é executada imediatamente após a conexão ser perdida.

IO.StartOffline

Configure em Verdadeiro para iniciar um Driver em modo **Offline** e em Falso para iniciar um Driver em modo **Online**.

NOTA

Não faz sentido modificar esta propriedade em tempo de execução, já que esta só pode ser modificada quando um Driver já está em modo **Offline**. Para configurar um Driver em modo **Online** em tempo de execução, escreva o valor 1 (um) no Tag **IO.WorkOnline**.

IO.TimeoutMs

Define o *time-out* da camada física, em milissegundos. Um segundo equivale a 1000 milissegundos.

IO.Type

Define o tipo de interface física utilizada por um Driver. Os valores possíveis são os seguintes:

- **N ou None**: Não utiliza uma interface física, ou seja, um Driver deve fornecer uma interface personalizada
- **S ou Serial**: Utiliza uma porta serial local (COM*n*)
- **M ou Modem**: Utiliza um modem local, interno ou externo, acessado via TAPI (*Telephony Application Programming Interface*)
- **E ou Ethernet**: Utiliza um *socket* TCP/IP ou UDP/IP
- **R ou RAS**: Utiliza uma Interface **RAS** (*Remote Access Server*). Um Driver conecta-se a um equipamento RAS através da Interface **Ethernet** e então emite um comando **AT** (*dial*)

Configuração de Estatísticas

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** das estatísticas das Interfaces de Comunicação.

Tags de Comunicação

Tags de Estatísticas das Interfaces de Comunicação (N2/B2 = 0)

Os Tags descritos a seguir mostram estatísticas para todas as Interfaces de Comunicação.

IO.Stats.Partial.BytesRecv

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1101
Configuração por String	IO.Stats.Partial.BytesRecv

Este Tag retorna a quantidade de bytes recebidos na conexão atual.

IO.Stats.Partial.BytesSent

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1100
Configuração por String	IO.Stats.Partial.BytesSent

Este Tag retorna a quantidade de bytes enviados na conexão atual.

IO.Stats.Partial.TimeConnectedSeconds

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1102
Configuração por String	IO.Stats.Partial.TimeConnectedSeconds

Este Tag retorna o número de segundos que um Driver está conectado na conexão atual ou 0 (zero) se um Driver está desconectado.

IO.Stats.Partial.TimeDisconnectedSeconds

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1103
Configuração por String	IO.Stats.Partial.TimeDisconnectedSeconds

Este Tag retorna o número de segundos que um Driver está desconectado desde o término da última conexão ou 0 (zero) se um Driver está conectado.

IO.Stats.Total.BytesRecv

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1001
Configuração por String	IO.Stats.Total.BytesRecv

Este Tag retorna a quantidade de bytes recebidos desde que um Driver foi carregado.

IO.Stats.Total.BytesSent

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1000
Configuração por String	IO.Stats.Total.BytesSent

Este Tag retorna a quantidade de bytes enviados desde que um Driver foi carregado.

IO.Stats.Total.ConnectionCount

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1004
Configuração por String	IO.Stats.Total.ConnectionCount

Este Tag retorna a quantidade de conexões que um Driver já estabeleceu, com sucesso, desde que foi carregado.

IO.Stats.Total.TimeConnectedSeconds

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1002
Configuração por String	IO.Stats.Total.TimeConnectedSeconds

Este Tag retorna o número de segundos que um Driver permaneceu conectado desde que foi carregado.

IO.Stats.Total.TimeDisconnectedSeconds

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	0 (zero)
Parâmetro N4	1003
Configuração por String	IO.Stats.Total.TimeDisconnectedSeconds

Este Tag retorna o número de segundos que um Driver permaneceu desconectado desde que foi carregado.

Propriedades

Atualmente, não existem propriedades definidas especificamente para mostrar as estatísticas das Interfaces de Comunicação em tempo de execução.

Configuração da Interface Ethernet

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** da Interface **Ethernet**.

Tags de Comunicação

Tags da Interface Ethernet (N2/B2 = 4)

Os Tags descritos a seguir permitem controlar e identificar a Interface **Ethernet** em tempo de execução e também são válidos quando a Interface **RAS** está selecionada.

IMPORTANTE
Estes Tags estão disponíveis **SOMENTE** enquanto um Driver está em modo **Online**.

IO.Ethernet.IPSelect

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Leitura ou Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	4 (quatro)
Parâmetro N4	0 (zero)
Configuração por String	IO.Ethernet.IPSelect

Indica o endereço IP ativo. Os valores possíveis são **0**: O endereço principal de IP está selecionado, **1**: O primeiro endereço IP alternativo ou de *backup* está selecionado, **2**: O segundo endereço IP alternativo ou de *backup* está selecionado ou **3**: O terceiro endereço IP alternativo ou de *backup* está selecionado.

Se a Interface **Ethernet** ou **RAS** está conectada, este Tag indica qual dos quatro endereços IP configurados está em uso. Se a Interface está desconectada, este Tag indica qual endereço IP é usado primeiro na próxima tentativa de conexão.

Durante o processo de conexão, se o endereço IP ativo não está disponível, a Interface de Comunicação tenta conectar-se usando o outro endereço IP. Se a conexão com o endereço IP alternativo funcionar, este é configurado como o endereço IP ativo (*switchover* automático).

Para forçar um *switchover* manual, escreva valores de 0 (zero) a três (3) neste Tag. Isto força a reconexão com o endereço IP especificado (**0**: Endereço principal, **1, 2, 3**: Endereços alternativos) se um Driver está atualmente conectado. Se um Driver está desconectado, este Tag configura o endereço IP ativo para a próxima tentativa de conexão.

IO.Ethernet.IPSwitch

Tipo de Tag	Tag de Comunicação

Tipo de Acesso	Somente Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	4 (quatro)
Parâmetro N4	1 (um)
Configuração por String	IO.Ethernet.IPSwitch

Qualquer valor escrito neste Tag força um *switchover* manual. Se o endereço principal de IP está ativo, então o primeiro endereço IP alternativo ou de *backup* é ativado, e assim por diante passando por todos os endereços IP alternativos e voltando para o endereço principal até estabelecer uma conexão.

Se um Driver está desconectado, este Tag configura o endereço IP ativo para a próxima tentativa de conexão.

IO.Ethernet.SocketState

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	4 (quatro)
Parâmetro N4	2 (dois)
Configuração por String	IO.Ethernet.SocketState

A propriedade **Value** deste Tag corresponde a estados do *socket* em um mapa de bits:

- **Bit 0**: 0 (zero, não está em escuta) ou 1 (um, em escuta)
- **Bit 1**: 0 (zero, desconectado) ou 1 (um, conectado)

Propriedades

Estas propriedades controlam a configuração da Interface **Ethernet**.

NOTA

A Interface **Ethernet** também é usada pela Interface **RAS**.

IO.Ethernet.AcceptConnection

◩ Configure em Falso se um Driver não deve aceitar conexões externas, ou seja, se um Driver se comporta como mestre, ou configure em Verdadeiro para habilitar a recepção de conexões, ou seja, se um Driver se comporta como escravo.

IO.Ethernet.BackupEnable[2,3]

◩ Configure em Verdadeiro para habilitar o endereço IP alternativo ou de *backup*. Se a tentativa de reconectar com o endereço IP principal falhar, um Driver tenta utilizar um endereço IP alternativo ou de *backup*. Configure em Falso para

desabilitar a utilização.

IO.Ethernet.BackupIP[2,3]

Endereço IP alternativo ou de *backup* de um equipamento remoto. Pode-se utilizar tanto o endereço numérico como o nome de *host* de um equipamento, como por exemplo "192.168.0.7" ou "SERVER2".

IO.Ethernet.BackupLocalPort[2,3]

Número da porta local a ser utilizada na conexão ao endereço IP alternativo ou de *backup* de um equipamento remoto. Usado apenas se a propriedade **IO.Ethernet.BackupLocalPortEnable** está configurada para Verdadeiro.

IO.Ethernet.BackupLocalPortEnable[2,3]

Configure em Verdadeiro para forçar o uso de uma porta local específica ao conectar ao endereço IP alternativo ou de *backup* ou configure em Falso para utilizar qualquer porta local disponível.

IO.Ethernet.BackupPort[2,3]

Número da porta do endereço IP alternativo ou de *backup* de um equipamento remoto, usado juntamente com a propriedade **IO.Ethernet.BackupIP**.

IO.Ethernet.IPFilter

Lista de endereços IPv4 ou IPv6 separados por vírgula, que define de quais endereços um Driver aceita ou bloqueia conexões. Pode-se utilizar asteriscos, como por exemplo "192.168.*.*", ou intervalos, como por exemplo "192.168.0.41-50", em qualquer parte dos endereços IP. Para bloquear um endereço IP ou um intervalos de endereços IP, use o caractere til ("~") no início do endereço, conforme os exemplos a seguir:

- **192.168.0.24**: Aceita apenas conexões do endereço IPv4 192.168.0.24
- **192.168.0.41-50**: Aceita conexões dos endereços IPv4 no intervalo entre 192.168.0.41 e 192.168.0.50
- **192.168.0.***: Aceita conexões dos endereços IPv4 no intervalo entre 192.168.0.0 e 192.168.0.255
- **fe80:3bf:877::*:* (expande para fe80:03bf:0877:0000:0000:0000:*:*)**: Aceita conexões de endereços IPv6 no intervalo entre fe80:03bf:0877:0000:0000:0000:0000:0000 e fe80:03bf:0877:0000:0000:0000:ffff:ffff
- **192.168.0.10, 192.168.0.15, 192.168.0.20**: Aceita conexões dos endereços IPv4 192.168.0.10, 192.168.0.15 e 192.168.0.20
- **~192.168.0.95, 192.168.0.***: Aceita conexões dos endereços IPv4 no intervalo entre 192.168.0.0 e 192.168.0.255, exceto o endereço IPv4 192.168.0.95

Quando um Driver recebe uma tentativa de conexão, a lista de filtros é percorrida sequencialmente da esquerda para a direita, procurando por uma autorização ou bloqueio específico para o endereço IP de onde veio a conexão. Se nenhum elemento da lista corresponde ao endereço IP, a autorização ou bloqueio são ditados pelo último elemento da lista:

- Se o último elemento da lista é uma autorização, como por exemplo "192.168.0.24", então todos os endereços IP não encontrados na lista são bloqueados
- Se o último elemento da lista é um bloqueio, como por exemplo "~192.168.0.24", então todos os endereços IP não encontrados na lista são autorizados

Se um endereço IP aparece em mais de um filtro da lista, o filtro mais à esquerda tem precedência. Por exemplo, no caso de "~192.168.0.95, 192.168.0.*", o endereço IP 192.168.0.95 se encaixa nas duas regras, mas a regra que vale é a mais à esquerda, "~192.168.0.95", e portanto o endereço IP é bloqueado.

Quando o **IOKit** bloqueia uma conexão, a mensagem "Blocked incoming socket connection from {IP}!" é logada.

No caso de conexões UDP em modo escuta em *broadcast*, em que um Driver pode receber pacotes de diferentes endereços IP, o bloqueio ou permissão é realizado a cada pacote recebido. Se um pacote é recebido de um endereço IP bloqueado, a mensagem "Blocked incoming packet from {IP} (discarding {N} bytes)!" é logada.

IO.Ethernet.ListenIP

A Endereço IP da interface local de rede por onde um Driver efetua e aceita conexões. Deixe esta propriedade vazia para efetuar e aceitar conexões por qualquer interface local de rede.

IO.Ethernet.ListenPort

9 Número da porta IP utilizada por um Driver para escutar conexões.

IO.Ethernet.MainIP

A Endereço IP de um equipamento remoto. Pode-se utilizar tanto o endereço numérico como o nome de *host* de um equipamento, como por exemplo "192.168.0.7" ou "SERVER2".

IO.Ethernet.MainLocalPort

9 Número da porta local a ser utilizada na conexão ao endereço IP principal de um equipamento remoto. Este valor é usado apenas se a propriedade **IO.Ethernet.MainLocalPortEnable** é igual a Verdadeiro.

IO.Ethernet.MainLocalPortEnable

◩ Configure em Verdadeiro para forçar o uso de uma porta local específica ao conectar ao endereço IP principal ou configure em Falso para utilizar qualquer porta local disponível.

IO.Ethernet.MainPort

9 Número da porta IP em um equipamento remoto, usado em conjunto com a propriedade **IO.Ethernet.MainIP**.

IO.Ethernet.PingEnable

◩ Configure em Verdadeiro para habilitar o envio de um comando **ping** para o endereço IP de um equipamento remoto, antes de tentar conectar-se ao *socket*. O *time-out* de conexão do *socket* não pode ser controlado, por isto o envio de um comando **ping** antes de conectar-se é uma maneira rápida de detectar se a conexão vai falhar. Configure em Falso para desabilitar o comando **ping**.

IO.Ethernet.PingTimeoutMs

9 Tempo de espera por uma resposta de um comando **ping**, em milissegundos.

IO.Ethernet.PingTries

9 Número máximo de tentativas de comandos **ping**. O valor mínimo é 1 (um), incluindo o primeiro comando **ping**.

IO.Ethernet.ShareListenPort

Configure em Verdadeiro para compartilhar a porta de escuta com outros Drivers e processos ou Falso para abrir a porta de escuta em modo exclusivo. Para compartilhar uma porta de escuta com sucesso, todos os Drivers e processos envolvidos devem abrir esta porta em modo compartilhado. Quando uma porta de escuta é compartilhada, cada nova conexão é distribuída para um dos processos que estão escutando. Desta forma, se um Driver Escravo só suporta uma conexão por vez, pode-se utilizar várias instâncias deste Driver escutando na mesma porta, portanto simulando um Driver com suporte a múltiplas conexões.

IO.Ethernet.SupressEcho

Configure em Verdadeiro para eliminar o eco presente em uma comunicação. O eco é a recepção indesejada de uma cópia exata de todos os pacotes de dados que um Driver enviou para um equipamento.

IO.Ethernet.Transport

Define o protocolo de transporte. Os valores possíveis são **T ou TCP**: Utiliza o protocolo TCP/IP ou **U ou UDP**: Utiliza o protocolo UDP/IP.

IO.Ethernet.UseIPv6

Configure em Verdadeiro para utilizar endereços IPv6 em todas as conexões Ethernet ou configure em Falso para utilizar endereços IPv4 (padrão).

Configuração da Interface Modem

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** da Interface **Modem** (TAPI).

Tags de Comunicação

Tags da Interface Modem (N2/B2 = 3)

Os Tags descritos a seguir permitem controlar e diagnosticar a Interface **Modem** (TAPI) em tempo de execução.

IMPORTANTE
Estes Tags estão disponíveis **SOMENTE** enquanto um Driver está em modo **Online**.

IO.TAPI.ConnectionBaudRate

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	5 (cinco)
Configuração por String	IO.TAPI.ConnectionBaudRate

Indica o valor de *baud rate* da conexão atual. Se o modem não está conectado, retorna o valor 0 (zero).

IO.TAPI.Dial

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	1 (um)
Configuração por String	IO.TAPI.Dial

Escreva qualquer valor neste Tag para forçar a Interface **Modem** a iniciar uma chamada. Este comando é assíncrono, apenas iniciando o processo de chamada. Pode-se monitorar o Tag **IO.TAPI.IsModemConnected** para detectar quando uma chamada é estabelecida.

IO.TAPI.HangUp

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	4 (quatro)
Configuração por String	IO.TAPI.HangUp

Qualquer valor escrito neste Tag desliga a chamada atual.

NOTA

Use este comando apenas quando gerenciar a camada física manualmente ou ao explicitamente tentar forçar um Driver a reiniciar a comunicação. Se a camada física está configurada para reconexão automática, um Driver imediatamente tenta restabelecer a conexão.

IO.TAPI.IsModemConnected

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	3 (três)
Configuração por String	IO.TAPI.IsModemConnected

Este Tag indica o estado da conexão do modem. Os valores possíveis são **0**: O modem não está conectado, mas pode estar realizando ou recebendo uma chamada externa ou **1**: O modem está conectado e um Driver completou ou recebeu uma chamada externa com sucesso. Enquanto está neste estado, a camada física consegue enviar ou receber dados.

IO.TAPI.IsModemConnecting

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	6 (seis)
Configuração por String	IO.TAPI.IsModemConnecting

Este Tag indica o estado de conexão do modem, com mais detalhes do que o Tag **IO.TAPI.IsModemConnected**. Os valores possíveis são **0**: O modem não está conectado, **1**: O modem está conectando, ou seja, realizando ou recebendo uma chamada externa, **2**: O modem está conectado. Enquanto está neste estado, a camada física consegue enviar ou receber dados ou **3**: O modem está desconectando a chamada atual.

IO.TAPI.ModemStatus

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Somente Leitura
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	2 (dois)
Configuração por String	IO.TAPI.ModemStatus

Retorna uma **String** com o estado atual do modem. Os valores possíveis são os seguintes:

- **"No status!"**: A Interface **Modem** ainda não foi aberta ou já foi fechada
- **"Modem initialized OK!"**: A Interface **Modem** foi inicializada com sucesso
- **"Modem error at initialization!"**: Um Driver não conseguiu inicializar a linha do modem. Confira o arquivo de log deste Driver para mais detalhes
- **"Modem error at dial!"**: Um Driver não conseguiu começar ou aceitar uma chamada
- **"Connecting..."**: Um Driver iniciou uma chamada com sucesso, e está atualmente processando esta chamada
- **"Ringing..."**: Indica que o modem está recebendo uma chamada externa, mas ainda não a aceitou
- **"Connected!"**: Um Driver conectou-se com sucesso, ou seja, completou ou aceitou uma chamada externa
- **"Disconnecting..."**: Um Driver está desligando a chamada atual
- **"Disconnected OK!"**: Um Driver desligou a chamada atual
- **"Error: no dial tone!"**: Um Driver abortou a chamada porque o sinal de linha disponível não foi detectado
- **"Error: busy!"**: Um Driver abortou a ligação porque a linha estava ocupada
- **"Error: no answer!"**: Um Driver abortou a chamada porque não recebeu resposta do outro modem
- **"Error: unknown!"**: A chamada atual foi abortada por um erro desconhecido

IO.TAPI.PhoneNumber

Tipo de Tag	Tag de Comunicação
Tipo de Acesso	Leitura ou Escrita
Parâmetro N1	-1 (menos um)
Parâmetro N2	0 (zero)
Parâmetro N3	3 (três)
Parâmetro N4	0 (zero)
Configuração por String	IO.TAPI.PhoneNumber

Este Tag é uma **String** que lê ou modifica o número do telefone utilizado pelo Tag **IO.TAPI.Dial**. Ao modificar este Tag, o novo valor é usado apenas no próximo comando **Dial**.

Propriedades

Estas propriedades controlam a configuração da Interface **Modem** (TAPI).

IO.TAPI.AcceptIncoming

Configure em Falso se o modem não pode aceitar chamadas externas, ou seja, se um Driver se comporta como mestre, e configure em Verdadeiro para habilitar a recepção de chamadas, ou seja, se um Driver se comporta como escravo.

IO.TAPI.ModemID

É o número de identificação do modem. Este ID é criado pelo Windows e é usado internamente para identificar o modem dentro de uma lista de equipamentos instalados no computador. Este ID pode não permanecer válido caso o modem seja reinstalado ou a aplicação seja executada em outro computador.

NOTA

Recomenda-se que esta propriedade seja configurada em 0 (zero), indicando que um Driver deve utilizar o primeiro modem disponível.

IO.TAPI.PhoneNumber

O número de telefone utilizado em comandos **Dial**, como por exemplo "0w01234566", em que o caractere "w" força o modem a esperar por um sinal de chamada.

Configuração da Interface RAS

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** da Interface **RAS**.

Tags de Comunicação

Tags da Interface RAS (N2/B2 = 5)

Atualmente, não existem Tags definidos especificamente para gerenciar a Interface **RAS** em tempo de execução.

Propriedades

Estas propriedades controlam a configuração da Interface **RAS**.

NOTA

A Interface **RAS** utiliza a Interface **Ethernet**, que por este motivo também deve ser configurada.

IO.RAS.ATCommand

Comando **AT** a ser enviado através do *socket* para forçar um equipamento RAS a realizar uma ligação usando o canal RAS atual, como por exemplo "ATDT6265545".

IO.RAS.CommandTimeoutSec

Tempo de espera pela mensagem **CONNECT** em resposta a um comando **AT**, em segundos.

Configuração da Interface Serial

Esta seção contém informações sobre a configuração dos **Tags de Comunicação** e das **Propriedades** da Interface **Serial**.

Tags de Comunicação

Tags da Interface Serial (N2/B2 = 2)

Atualmente, não existem Tags definidos especificamente para gerenciar a Interface **Serial** em tempo de execução.

Propriedades

Estas propriedades controlam a configuração da Interface **Serial**.

IO.Serial.Baudrate

Especifica a taxa de *bauds* da porta serial, como por exemplo 9600.

IO.Serial.CTSTimeoutMs

Tempo de espera pelo sinal **CTS**, em milissegundos. Após o sinal **RTS** ser ligado (**ON**), um temporizador é iniciado para esperar pelo sinal **CTS**. Se este temporizador expira, um Driver aborta o envio de bytes através da porta serial. Disponível apenas quando a propriedade **IO.Serial.RTS** está configurada com o valor **Toggle** e a propriedade **IO.Serial.WaitCTS** está configurada em Verdadeiro.

IO.Serial.DataBits

Especifica o número de bits de dados para a configuração da porta serial. Os valores possíveis são **5**: Cinco bits de dados, **6**: Seis bits de dados, **7**: Sete bits de dados ou **8**: Oito bits de dados.

IO.Serial.DelayAfterMs

Número de milissegundos de atraso após o último byte ter sido enviado através da porta serial, mas antes de desligar (**OFF**) o sinal **RTS**. Disponível apenas quando a propriedade **IO.Serial.RTS** está configurada com o valor **Toggle** e a propriedade **IO.Serial.WaitCTS** está configurada em Falso.

IO.Serial.DelayBeforeMs

Número de milissegundos de atraso após o sinal **RTS** ter sido ligado (**ON**), mas antes dos dados serem enviados. Disponível apenas quando a propriedade **IO.Serial.RTS** está configurada com o valor **Toggle** e a propriedade **IO.Serial.WaitCTS** está configurada em Falso.

IO.Serial.DTR

Indica o modo como um Driver lida com o sinal **DTR**. Os valores possíveis são **OFF**: Sinal **DTR** sempre desligado ou **ON**: Sinal **DTR** sempre ligado.

IO.Serial.InterbyteDelayUs

Tempo de espera, em milissegundos (1/1000000 de um segundo), para cada dois bytes enviados pela Interface **Serial**.

IO.Serial.InterframeDelayMs

Tempo de espera, em milissegundos, antes de enviar um pacote após o último pacote enviado ou recebido.

IO.Serial.Parity

Especifica a paridade para a configuração da porta serial. Os valores possíveis são **E ou Even**: Paridade par, **N ou None**: Sem paridade, **O ou Odd**: Paridade ímpar, **M ou Mark**: Paridade de marca ou **S ou Space**: Paridade de espaço.

IO.Serial.Port

Número da porta serial local. Os valores possíveis são **1**: Utiliza a porta COM1, **2**: Utiliza a porta COM2, **3**: Utiliza a porta COM3 ou **n**: Utiliza a porta COM*n*.

IO.Serial.RTS

Indica como um Driver lida com o sinal **RTS**. Os valores possíveis são **OFF**: Sinal **RTS** sempre desligado, **ON**: Sinal **RTS** sempre ligado ou **Toggle**: Liga (**ON**) o sinal **RTS** quando está transmitindo dados e desliga (**OFF**) o sinal **RTS** quando não está transmitindo dados.

IO.Serial.StopBits

Especifica o número de bits de parada para a configuração da porta serial. Os valores possíveis são **1**: Um bit de parada, **2**: Um bit e meio de parada ou **3**: Dois bits de parada.

IO.Serial.SupressEcho

Utilize um valor diferente de 0 (zero) para habilitar a supressão de eco ou 0 (zero) para desabilitá-la.

IO.Serial.WaitCTS

Configure em Verdadeiro para forçar um Driver a esperar pelo sinal **CTS** antes de enviar bytes quando o sinal **RTS** está ligado (**ON**). Disponível apenas quando a propriedade **IO.Serial.RTS** está configurada com o valor **Toggle**.

Histórico de Revisões do Driver

VERSÃO	DATA	AUTOR	COMENTÁRIOS
1.0.10	05/09/2025	M. Ludwig	• Driver atualizado para a biblioteca **IOKit** versão **3.0** e Visual Studio 2022 (*Case 37937*).
1.0.8	09/09/2024	M. Ludwig	• Implementado um mecanismo de *debouncing* com tempo de espera após a primeira solicitação de troca de endereço IP, a fim de evitar filas de requisições de todas as Sessões ativas (*Case 36577*).
1.0.7	18/04/2024	M. Ludwig	• Adicionado o gerenciamento de conexão na camada Session (*Case 35834*).
1.0.6	08/02/2024	M. Ludwig	• Implementada a atualização de Tags para um estado de falha em caso de Sessão *offline* (*Case 35510*).
1.0.5	05/12/2023	M. Ludwig	• Corrigido o envio de dados no primeiro intervalo de *heartbeat* (*Case 35036*). • Corrigido o incremento de SNDMSN e RCVMSN (*Case 35036*). • Implementado o tratamento de colisões em esperas de *Ack* (*Case 35036*).

VERSÃO	DATA	AUTOR	COMENTÁRIOS
			• Corrigida a validação do código CRC em mensagens vitais (*Case 35036*). • Corrigida a leitura de Tags da biblioteca **IOKit** (*Case 35036*). • Corrigida a operação de escrita de bits de Elementos de Bloco (*Case 35036*).
1.0.4	11/09/2023	M. Ludwig	• Corrigida uma função deste Driver que poderia causar instabilidade durante a recepção de dados (*Case 34827*).
1.0.3	29/08/2023	M. Ludwig	• Realizadas melhorias de desempenho e otimizações no código-fonte (*Case 34743*).
1.0.2	24/07/2023	M. Ludwig	• Implementadas configurações para cada Estação de destino (*Case 34303*). • Implementado o envio de dados vitais e não vitais (*Case 34329*). • Implementada a leitura de Tags por *callbacks* (*Case 34449*).
1.0.1	24/04/2023	M. Ludwig	• Versão inicial deste Driver.

Matriz
Rua Mostardeiro, 322/Cj. 902, 1001 e 1002
90430-000 — Porto Alegre — RS
Fone: (+55 51) 3346-4699
Fax: (+55 51) 3222-6226
E-mail: elipse-rs@elipse.com.br

Filial no Paraná
Av. Sete de Setembro, 4698/1708
80240-000 — Curitiba — PR
Fone: (+55 41) 4062-5824
E-mail: elipse-pr@elipse.com.br

Filial no Rio de Janeiro
Av. José Silva de A. Neto, 200/Bl. 4/Sl. 109B
22250-044 — Rio de Janeiro — RJ
Fone: (+55 21) 2430-5912
Suporte Técnico: (+55 21) 2430-5963
E-mail: elipse-rj@elipse.com.br

Filial em São Paulo
Rua dos Pinheiros, 870/Cj. 141 e 142
05422-001 — São Paulo — SP
Fone: (+55 11) 3061-2828
Fax:(+55 11) 3086-2338
E-mail: elipse-sp@elipse.com.br

Filial em Minas Gerais
Rua Antônio de Albuquerque, 156/705
30112-010 — Belo Horizonte — MG
Fone: (+55 31) 4062-5824
E-mail: elipse-mg@elipse.com.br

Filial em Taiwan
9F., No.12, Beiping 2nd St., Sanmin Dist. 807 — Kaohsiung City — Taiwan
Fone: (+886 7) 323-8468
Fax: (+886 7) 323-9656
E-mail: evan@elipse.com.br

Consulte nosso website para informações sobre o representante do seu estado.

www.elipse.com.br
kb.elipse.com.br
forum.elipse.com.br
www.youtube.com/elipsesoftware
elipse@elipse.com.br

Gartner 2014 Cool Vendor

OPC FOUNDATION MEMBER

Gartner, Cool Vendors in Brazil 2014, April 2014.
Gartner does not endorse any vendor, product or service depicted in its research publications, and does not advise technology users to select only those vendors with the highest ratings. Gartner research publications consist of the opinions of Gartner's research organization and should not be construed as statements of fact. Gartner disclaims all warranties, expressed or implied, with respect to this research, including any warranties of merchantability of fitness for a particular purpose.

Microsoft Partner
Gold Independent Software Vendor (ISV)

Copyright © 2022-2025 Elipse Software Ltda. Todos os direitos reservados.