# Driver Novus LogBox

| | |
|---|---|
| **Nome do Arquivo** | NovusLogBox.dll |
| **Fabricante** | Novus Produtos Eletrônicos |
| **Equipamentos** | LogBox3G no serviço NOVUS Cloud |
| **Protocolo** | N/A |
| **Versão** | 1.0.6 |
| **Última Atualização** | 02/04/2026 |
| **Plataforma** | Win32 |
| **Dependências** | IOKit versão 2.0 e bibliotecas LogBox3GCloudLibs.dll, ssleay32.dll, libeay32.dll e libssl32.dll |
| **Leitura com Superblocos** | Não |
| **Nível** | 0 |

## Introdução

Este Driver implementa a comunicação com a biblioteca LogBox3GCloudLibs da Novus Produtos Eletrônicos, que permite aos produtos da **Elipse Software** se comunicarem com equipamentos LogBox3G no serviço NOVUS Cloud.

O Driver Novus LogBox foi desenvolvido em conjunto com a biblioteca **IOKit** da **Elipse Software**.

# Configuração do Driver

Para que o Driver Novus LogBox funcione corretamente, o arquivo de biblioteca dinâmica LogBox3GCloudLibs.dll deve estar localizado na mesma pasta deste Driver, que se conecta diretamente a este arquivo de biblioteca. Os arquivos de bibliotecas dinâmicas ssleay32.dll, libeay32.dll e libssl32.dll também devem estar localizados na mesma pasta, pois são utilizados pela biblioteca da Novus Produtos Eletrônicos.

Todas as configurações são realizadas na caixa de diálogo de configurações deste Driver, especialmente na aba **NovusLogBox**.

Para mais informações sobre a aba **Setup**, consulte o tópico **Documentação das Interfaces de Comunicação**.

A aba **NovusLogBox** contém configurações específicas para este Driver. As opções disponíveis nesta aba estão descritas na tabela a seguir.

**Opções disponíveis na aba NovusLogBox**

| OPÇÃO | DESCRIÇÃO |
|---|---|
| **User email address** | E-mail do usuário cadastrado no serviço NOVUS Cloud |
| **Password** | Senha do usuário cadastrado no serviço NOVUS Cloud |
| **Update time** | Valor, em segundos, para a atualização dos valores dos Canais |

# Referência de Tags

Esta seção contém informações sobre a configuração dos Tags deste Driver. Configura-se estes Tags através de uma **String** no campo **Dispositivo** e por parâmetros **[N/B]**.

A tabela a seguir descreve os valores do parâmetro *N2* ou *B2* que representam cada um dos Canais.

**Opções disponíveis para os parâmetros N2 ou B2**

| N2 OU B2 | DESCRIÇÃO |
|---|---|
| **1** | Entrada analógica 1 (um) |
| **2** | Entrada analógica 2 (dois) |
| **3** | Entrada digital |
| **4** | GPS |
| **5** | Temperatura interna |
| **6** | Percentual de carga da bateria |
| **7** | Tensão da bateria |
| **8** | Tensão da fonte de alimentação |
| **9** | Acumulador ou Totalizador |

# Get Last Value

**Somente Leitura**

| | |
|---|---|
| **Dispositivo** | Número de série de um equipamento LogBox 3G cadastrado no serviço NOVUS Cloud |
| **N1** | 0 (zero) |
| **N2** | Número do Canal a ser lido. Para mais informações, consulte a tabela **Opções disponíveis para os parâmetros N2 ou B2** |
| **N3** | Não usado, deixe em 0 (zero) |
| **N4** | Não usado, deixe em 0 (zero) |

Tag de leitura cuja propriedade **Value** contém o último valor válido no *buffer* referente ao dispositivo e Canal selecionado. Este valor possui a estampa de tempo de registro.

# Get Valid Dates

**Somente Leitura**

| | |
|---|---|
| **Dispositivo** | Número de série de um equipamento LogBox 3G cadastrado no serviço NOVUS Cloud |
| **B1** | 1 (um) |
| **B2** | Número do Canal a ser lido. Para mais informações, consulte a tabela **Opções disponíveis para os parâmetros N2 ou B2** |
| **B3** | Não usado, deixe em 0 (zero) |
| **B4** | Não usado, deixe em 0 (zero) |

Este Tag é utilizado para ler as estampas de tempo inicial e final do primeiro e do último dado válido na nuvem para o dispositivo e Canal selecionado. Os Elementos deste Tag Bloco são os seguintes:

- **Elemento 1**: Data e hora do primeiro valor válido no formato **TIME**
- **Elemento 2**: Data e hora do último valor válido no formato **TIME**
- **Elemento 3**: Número de registros

# Request Collect

### Somente Escrita

| | |
|---|---|
| **Dispositivo** | Número de série de um equipamento LogBox 3G cadastrado no serviço NOVUS Cloud |
| **B1** | 2 (dois) |
| **B2** | Número do Canal a ser lido. Para mais informações, consulte a tabela **Opções disponíveis para os parâmetros N2 ou B2** |
| **B3** | Não usado, deixe em 0 (zero) |
| **B4** | Não usado, deixe em 0 (zero) |

A escrita neste Tag comanda a leitura da coleta de dados compreendidos no intervalo de tempo determinado pelos Elementos deste Tag Bloco. A transferência de dados passa a operar em *background* e o resultado é visualizado no Tag **Collect Data**. Os Elementos deste Tag Bloco são os seguintes:

- **Elemento 1**: Data e hora iniciais da coleta no formato **TIME**
- **Elemento 2**: Data e hora finais da coleta no formato **TIME**

# Status Read

### Somente Leitura

| | |
|---|---|
| **Dispositivo** | Número de série de um equipamento LogBox 3G cadastrado no serviço NOVUS Cloud |
| **N1** | 3 (três) |
| **N2** | Número do Canal a ser lido. Para mais informações, consulte a tabela **Opções disponíveis para os parâmetros N2 ou B2** |
| **N3** | Não usado, deixe em 0 (zero) |
| **N4** | Não usado, deixe em 0 (zero) |

Este Tag exibe o estado da leitura da coleta de dados do respectivo Canal indicado pelo parâmetro *N2*. A propriedade **Value** deste Tag exibe o estado de leitura em *background* com valores numéricos correspondentes ao estado atual. Os valores possíveis são **0**: Ocioso, **1**: Transferência em progresso, **2**: Transferência realizada com sucesso, **3**: Falha na transferência ou **4**: Carregado com dados.

# Collect Data

### Somente Leitura

| | |
|---|---|
| **Dispositivo** | Número de série de um equipamento LogBox 3G cadastrado no serviço NOVUS Cloud |
| **B1** | 4 (quatro) |
| **B2** | Número do Canal a ser lido. Para mais informações, consulte a tabela **Opções disponíveis para os parâmetros N2 ou B2** |
| **B3** | Não usado, deixe em 0 (zero) |
| **B4** | Não usado, deixe em 0 (zero) |

Este Tag é uma lista de *N* blocos, em que *N* é o número de blocos de dados lidos e determinados no espaço de tempo entre a data e hora iniciais e a data e hora finais escritas no Tag **Request Collect**. Caso não existam dados, retorna uma lista vazia. Os Elementos deste Tag Bloco são os seguintes:

- **Elemento 1**: Lista de valores referentes ao dispositivo e Canal selecionado
- **Elemento 2**: Lista de valores referentes à coordenada GPS de latitude
- **Elemento 3**: Lista de valores referentes à coordenada GPS de longitude

Cada bloco retornado nesta lista possui a estampa de tempo do registro informado no serviço NOVUS Cloud.

**NOTAS**

- Esta tabela exige a escrita no Tag **Request Collect**, com intervalo de data e hora inicial e final, para comandar a transferência de dados em *background*.
- Caso o volume de registros requisitados por data ultrapasse o valor de 65535, a leitura deste Tag deve ser repetida até que o Tag **Status Read** saia do estado **4** (quatro, carregado com dados).

# Documentação das Interfaces de Comunicação

Esta seção contém a documentação das Interfaces de Comunicação referentes ao Driver **NovusLogBox**.

# 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.

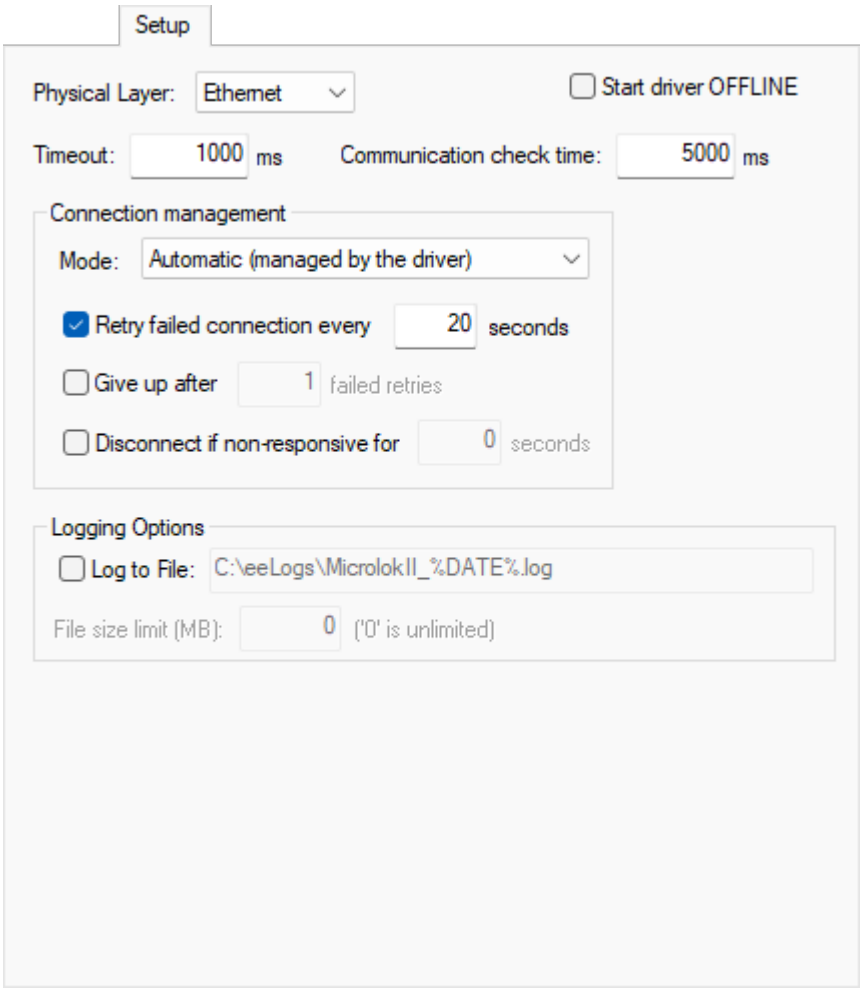
# Caixa de Diálogo de Configuração

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 |

# 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.

# Histórico de Revisões do Driver

| VERSÃO | DATA | AUTOR | COMENTÁRIOS |
|---|---|---|---|
| **1.0.6** | 02/04/2026 | M. Ludwig | • Driver atualizado para a biblioteca **IOKit** versão **3.0** e Visual Studio 2022 (*Cases 38001*). |
| **1.0.5** | 26/05/2021 | M. Ludwig | • Atualizada a biblioteca LogBox3GCloudLibs com a versão mais recente (*Cases 29295, 30936*).<br>• Implementado o Tag **Collect Data** com dados de GPS (*Case 29295*). |
| **1.0.3** | 08/08/2018 | 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 © 2018-2026 Elipse Software Ltda. Todos os direitos reservados.**