

IOKIT
para o Elipse E3 e Elipse SCADA

Versão 1.14

Manual do Usuário

1 Introdução

Este manual é uma referência para os usuários do IOKit Elipse, descrevendo como utilizar e configurar o IOKit tanto no Elipse SCADA quanto no E3, fornecendo informações técnicas do uso prático do IOKit.

O IOKit é um componente compartilhado utilizado pelos drivers de I/O do Elipse, implementando o acesso padrão do nível físico e proporcionando interfaces para:

- Portas seriais
- Modem (através de TAPI)
- Ethernet (utiliza Windows Sockets via TCP/IP ou UDP/IP)
- RAS (Remote Access Server)

Todos os drivers escritos utilizando o IOKit utilizam os seguintes recursos descritos neste manual, como por exemplo:

- Independência de nível físico
- Geração de logs
- Configuração offline
- Gerenciamento de conexões

O IOKit é implementado como um DLL (dynamic link library) linkado com o driver.

IMPORTANTE: Para evitar conflitos entre drivers, o arquivo IOKIT.DLL deve estar presente **APENAS** na pasta **Windows\System32**. Cópias deste DLL em outras pastas podem fazer com que o driver e/ou o IOKit não funcionem corretamente.

2 Configuração

A configuração do I/O kit é feita no dialog de configuração do driver. Para acessar a configuração do dialog no E3 (versão 1.xx), siga os seguintes passos:

- Clique com o botão da direita no driver (IODriver)
- Selecione "Propriedades" no menu popup
- Selecione a página "Driver"
- Pressione o botão "Outros parâmetros"

No E3 versão 2.0 ou posterior, basta clicar no botão  para abrir a configuração do driver.

No Elipse SCADA, siga os seguintes passos:

- Abra o "Organizer"
- Selecione o driver na árvore
- Pressione o botão "Extras..." na página do driver

Atualmente o IOKit permite que apenas uma conexão seja aberta por cada driver. Isto significa que, se você desejar o acesso a duas portas seriais, terá que adicionar dois drivers na sua aplicação e configurar cada um deles para cada porta serial.

2.1 O Dialog do IOKit

O dialog do IOKit permite configurar a conexão de I/O que será utilizada pelo driver e contém cinco páginas:

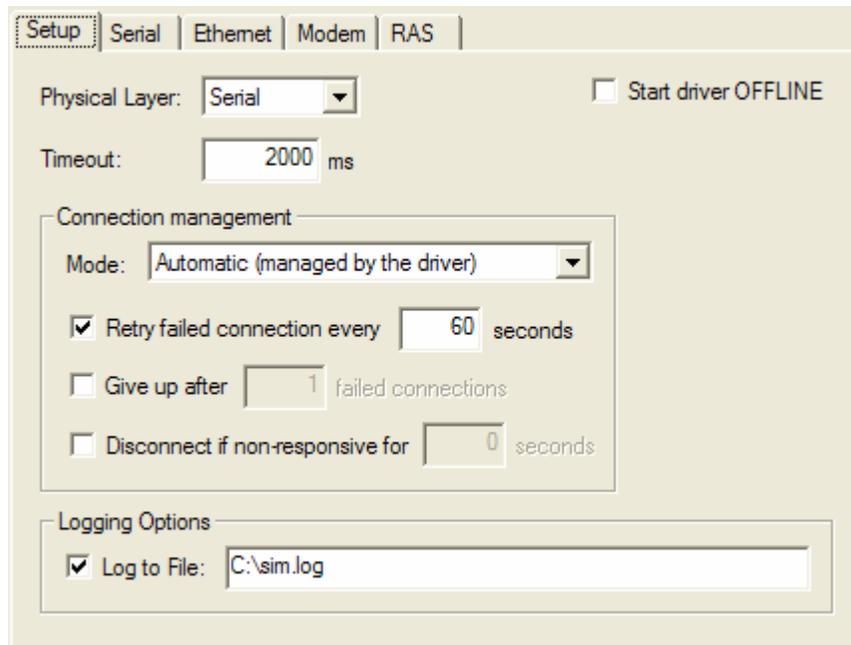
- Setup
- Serial
- Ethernet
- Modem
- RAS

Alguns drivers fornecem páginas adicionais específicas no dialog do IOKit. Consulte os manuais do driver caso deseje maiores informações sobre estas páginas.

2.2 A página do Setup

A página do Setup contém a configuração geral do driver. A página é dividida em três partes distintas:

- 1) Configurações gerais
- 2) Connection management (gerenciamento de conexões)
- 3) logging options (Opções de logs)



Physical Layer

Selecione a interface física a partir da lista. As opções são: **Serial, Ethernet, Modem and RAS**. A interface selecionada deverá ser configurada na sua página específica.

Timeout

Configure o timeout em milissegundos para a camada física. Isto é a medida de tempo que a interface de I/O aguardará para a recepção de UM byte (qualquer byte do buffer de recepção).

Start driver OFFLINE

Assinale esta opção para fazer com que o driver inicie OFFLINE (parado). Isto significa que a interface I/O não será criada até que você configure o driver como ONLINE (utilizando um tag na aplicação). Este modo possibilita a configuração dinâmica da interface I/O em execução. Veja a seção "*Trabalhando em OFFLINE*" para maiores detalhes.

Connection management

Estas configurações configuram como o IOKit irá manter a conexão e qual a política de recuperação contra as falhas.

Mode

Seleciona o modo de gerenciamento de conexão. Selecionando a opção **Automatic** deixará que o driver gerencie a conexão automaticamente como configurado nas opções seguintes. Selecionado a opção **Manual** deixará o gerenciamento inteiramente a cargo da aplicação. Veja a seção "*Estado do Driver*" para maiores detalhes.

Retry failed connection every

Selecione esta opção para habilitar a retentativa de conexão do driver em um determinado intervalo (em segundos). Se a opção 'give up' não estiver configurada o driver continuará retentando até que a conexão seja efetuada ou que a aplicação seja parada.

Give up after

Habilite esta opção para definir um número máximo de tentativas de conexão. Quando o número especificado em **consecutive connection tries** é alcançado o driver irá para o modo OFFLINE, assumindo que um problema de hardware está presente. Se o driver estabelece uma conexão com sucesso, o número de tentativas sem sucesso é zerada. Se esta nova conexão é perdida então o contador de retentativas começa do zero.

Disconnect if non-responsive

Habilite esta opção para fazer com que o driver desconecte se nenhum byte chegou a interface I/O no timeout especificado (em segundos). Este timeout deverá ser maior que o timeout configurado no nível físico anteriormente.

Logging Options

Estas configurações controlam a geração de arquivos de log para o driver.

Log to File

Habilite esta opção e configure o nome do arquivo onde o log será 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 você coloque a macro "%PROCESS%" no nome do arquivo de log, esta será substituída pelo ID do processo atual. Esta opção é particularmente útil se você está utilizando várias instâncias do mesmo driver no E3, fazendo assim que cada instância gere um arquivo separado de log. Por exemplo:

Log to file: c:\e3logs\drivers\sim_%PROCESS%.log

Irá gerar um arquivo de log "c:\e3logs\drivers\sim_00000FDA.log" para o processo 0FDAh

Você também pode utilizar a macro "%DATE%" no nome do arquivo. Neste caso será gerado um arquivo de log por dia (no formato "aaa_mm_dd"). Por exemplo:

Log to file: c:\e3logs\drivers\sim_%DATE%.log

Irá gerar o arquivo "c:\e3logs\drivers\sim_2005_12_31.log" no dia 31 de Dezembro de 2005, e "c:\e3logs\drivers\sim_2006_01_01.log" no dia 1º de Janeiro de 2006.

2.3 A página Serial

Utilize esta página para configurar os parâmetros da interface Serial.

Setup Serial Ethernet Modem RAS

Port: COM1

Baud rate: 9600

Data bits: 8 data bits

Parity: None

Stop bits: 1 stop bit

Enable 'ECHO' suppression

Handshaking

DTR control: OFF

RTS control: Toggle

Wait for CTS before send

CTS timeout: 0 ms

Delay before send: 0 ms

Delay after send: 0 ms

Inter-byte delay (microseconds): 0 µs

Inter-frame delay (milliseconds): 0 ms

Port

Selecione a porta serial a partir da lista (COM1 até COM4) ou digite o nome da porta serial no formato "COMn" (ex: "COM15"). Se você digitar o nome da porta manualmente o dialog aceitará apenas nomes de portas começando com "COM".

Baud rate

Selecione o baud rate a partir da lista (1200, 2400, 4800, 9600, 19200, 38400, 57600 ou 115200) ou digite o baud rate desejado (ex: 600).

Data bits

Selecione 7 ou 8 bits de dados a partir da lista.

Parity

Selecione a paridade a partir da lista (**None**, **Even**, **Odd**, **Mark** ou **List**)

Stop bits

Selecione o número de stop bits a partir da lista (**1**, **1.5** ou **2** stop bits)

Enable 'ECHO' supression

Habilite esta opção para remover o 'eco' recebido após o IOKit enviar algo pela serial. Se o eco não for igual aos bytes recém enviados, o IOKit aborta a comunicação.

Handshaking

Configure o uso dos sinais RTS, CTS e DTR no handshaking (controlando quando o dado pode ser enviado/recebido através da linha serial). Na maioria das vezes configurando o controle DTR para "ON" e o controle RTS para "Toggle" irá funcionar tanto com RS232 quanto com RS485.

DTR control

Selecione "ON" para deixar o sinal DTR sempre ligado enquanto a porta serial é aberta. Selecione "OFF" para desligar o sinal DTR enquanto a porta serial é aberta. Alguns equipamentos exigem que o sinal DTR esteja ligado para permitir a comunicação.

RTS control

Selecione "ON" para deixar o sinal RTS sempre ligado enquanto a porta serial é aberta. Selecione "OFF" para desligar o sinal RTS enquanto a porta serial é aberta. Selecione "Toggle" para ligar o RTS enquanto envia os bytes através da porta serial e desligá-lo quando não está enviando bytes (habilita recepção).

Wait for CTS before send

Disponível apenas quando o controle RTS está configurado para "Toggle". Utilize esta opção para forçar que o driver verifique o sinal CTS antes de enviar os bytes através da serial (após levantar o sinal de RTS). Neste modo o CTS é tratado como um flag de "permissão para envio".

CTS timeout

Determina o tempo máximo (em milisegundos) que o driver irá aguardar pelo sinal de CTS depois de levantar o sinal de RTS. Se o CTS não é levantado dentro deste timeout, o driver acaba por falhar na comunicação atual e retorna erro.

Delay before send

Alguns hardwares de porta serial demoram a habilitar o circuito de envio de dados depois que o sinal RTS é levantado. Configure este parâmetro para aguardar uma determinada quantidade de milisegundos depois de levantar o RTS e antes de enviar o primeiro byte. **IMPORTANTE:** esta espera deve ser utilizada com muito cuidado, pois consome 100% dos recursos da CPU enquanto aguarda. A performance geral do system será degradada conforme este valor aumenta.

Delay after send

É o mesmo que o "Delay before send", mas neste caso a espera é efetuada depois que o último byte é enviado, antes de baixar o sinal de RTS.

Inter-byte delay (microseconds):

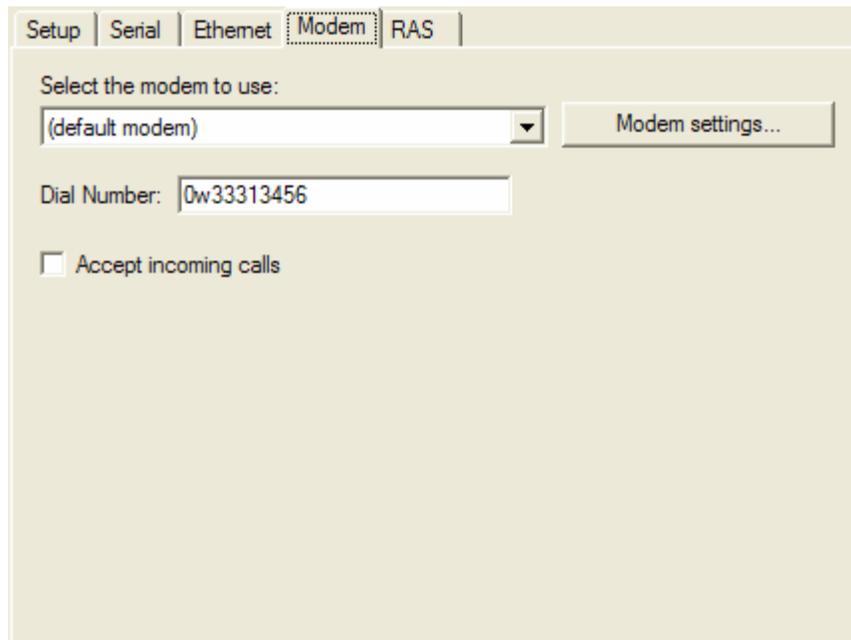
Define uma espera entre cada byte transmitido pelo IOKit, em milionésimos de segundo (1000000 = 1 segundo). Esta opção deve ser utilizada com esperas pequenas (menos que um milissegundo).

Inter-frame delay (milliseconds):

Define uma espera entre pacotes enviados/recebidos pelo IOKit, em milésimos de segundo (1000 = 1 segundo). Esta espera é aplicada caso o IOKit envie dois pacotes consecutivos, ou entre um pacote recebido e o próximo envio.

2.4 A página do Modem

Utilize esta página para configurar os parâmetros da interface Modem. Algumas opções da página Serial afetam a configuração do Modem, portanto é interessante não esquecer de configurar a interface Serial.



A interface modem utiliza os modems TAPI instalados no seu computador.

Select the modem to use

Escolha um modem a partir da lista de modems disponíveis no seu computador. Case você selecione **(default modem)**, então o primeiro modem disponível será utilizado (escolher esta opção é recomendado especialmente quando a aplicação será utilizada em um outro computador).

Modem settings...

Clique neste botão para abrir a janela de configuração do modem selecionado.

Dial Number

Digite o número para discar (este pode ser modificado em execução). Você pode utilizar a

caractere “w” representando uma pausa (espera pelo tom de discagem). Ex: “0w33313456” (discar 0, esperar e então discar 33313456).

Accept incoming calls

Habilitando esta opção faz com que o driver atenda o telefone quando ele receba uma chamada externa. Para utilizar esta opção é necessário que o “Connection management”, na página de Setup, esteja configurada para **Manual**.

2.5 A página Ethernet

Utilize esta página para configurar os parâmetros da interface Ethernet. Estes parâmetros (todos exceto configurações da porta) devem ser também configurados para uso na interface RAS.

The screenshot shows the 'Ethernet' configuration window. At the top, there are tabs for 'Setup', 'Serial', 'Ethernet' (which is selected), 'Modem', and 'RAS'. Below the tabs, the 'Transport' dropdown menu is set to 'TCP/IP'. There is a checkbox labeled 'Listen for connections on port:' with a text box containing the number '4315'. Below this, there are two 'Connect to' sections. The first section has an IP address of '192.168.0.13' and a Port of '4315'. The second section is checked for 'Backup address' and has an IP address of '192.168.0.14' and a Port of '4315'. To the right of these sections, there is a checkbox labeled 'PING before connecting' which is checked. Below it, there is a 'Timeout:' field set to '500 ms' and a 'Retries:' field set to '0'.

Transport

Selecione **TCP/IP** para um socket TCP (stream). Selecione **UDP/IP** para utilizar um socket UDP (datagram, connectionless).

Listen for connections on port:

Utilize esta opção para aguardar por novas conexões em uma porta IP específica (comum em drivers **Escravos**). Caso você deixe esta opção desmarcada então o driver conectará ao endereço e porta especificado em “Connect to”.

Connect to

Estes parâmetros configuram o endereço IP e a Porta do dispositivo remoto.

IP

Digite o endereço IP do dispositivo remoto. Você pode usar tanto o IP separado por pontos quanto uma URL (no caso de uma URL, o driver usa o serviço de DNS disponível para mapear a URL para um endereço IP). Exemplos: “192.168.0.13” ou “server1”

Port

Digite a porta IP do dispositivo remoto (0 até 65535).

Backup address

Habilite o endereço de backup se o dispositivo dispõe de um endereço de IP alternativo (no caso do primeiro endereço falhar).

IP

Digite o endereço IP do dispositivo remoto. Você pode usar tanto o IP separado por pontos quanto uma URL (no caso de uma URL, o driver usa o serviço de DNS disponível para mapear a URL para um endereço IP). Exemplos: "192.168.0.13" ou "server1"

Port

Digite a porta IP do dispositivo remoto (0 até 65535).

PING before connecting

Habilite esta opção para pingar (verificar se o dispositivo pode ser encontrado na rede) o dispositivo antes de tentar uma conexão com o socket. Esta é uma maneira rápida de determinar a chance de uma conexão bem sucedida antes de tentar abrir um socket com o dispositivo (o timeout de uma conexão com um socket pode ser bem alto).

Timeout

Especifique o número de milissegundos de espera por uma resposta do ping. Você deve usar o comando de linha de comando "PING" para verificar o tempo normal de resposta, configurando este parâmetro para um valor acima desta média. Normalmente você pode configurar um valor entre 1000 e 4000 milissegundos (1 até 4 segundos).

Retries

Número de tentativas de ping (não conta a tentativa inicial). Se todas as tentativas falharem então a conexão com o socket será abortada.

2.6 A página RAS

Use esta página para configurar os parâmetros da interface RAS. Você necessitará também configurar a página Ethernet.

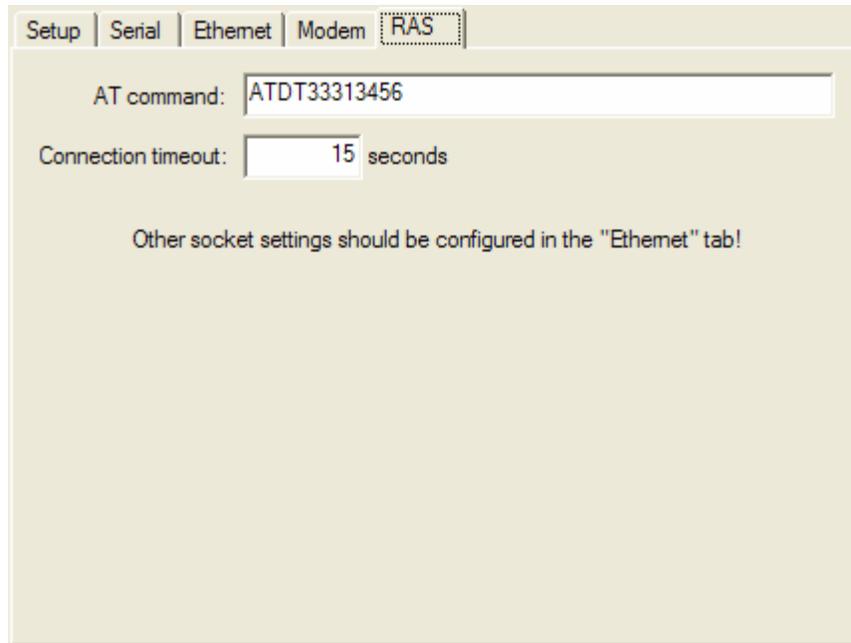
A interface RAS abre uma conexão socket com um dispositivo RAS. O dispositivo RAS é um servidor de modems acessível através de TCP/IP, aguardando por conexões socket em uma porta IP. Para cada conexão aceita nesta porta ele dá acesso a um modem.

Quando conectando a um dispositivo RAS, primeiramente o IOKit conecta ao socket no IP e porta configurado na página Ethernet. Depois que o socket é aberto, os seguintes passos de inicialização/conexão são efetuados:

- 1) Limpeza do socket (remove qualquer mensagem de saudação TELNET recebida do dispositivo RAS)
- 2) Envia o comando de discagem AT (em ASCII) no socket
- 3) Aguarda pela recepção de uma resposta "CONNECT"
- 4) Caso o timeout expire, a conexão é abortada
- 5) Se o "CONNECT" é recebido dentro do timeout, o socket é disponibilizado para comunicação com o dispositivo (a conexão está estabelecida)

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

Depois que a conexão é estabelecida, a interface RAS monitora os dados recebidos pelo driver. Caso um string de "NO CARRIER" é encontrado, 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 página de Setup).



AT command

String com o comando AT completo usado para discar ao dispositivo destino. Exemplo: "ATDT33313456" (discagem usando tom para o número 33313456).

Connection timeout

Número de segundos a aguardar por uma resposta "CONNECT" do modem, após o envio do comando AT.

3 Referência dos Tags do IOKIT (N1/B1 = -1)

O IOKit define tags internos utilizados para várias tarefas. Todos os tags do IOKit tem **N1/B1 = -1** (ou 65535 no E3) e **N2/B2 = 0**. Os parâmetros **N3/B3** tem o seguinte significado:

- **N3/B3=0**: Tags de status e configurações gerais do IOKit
- **N3/B3=1**: Tags da interface serial
- **N3/B3=2**: Tags da interface ethernet
- **N3/B3=3**: Tags da interface modem
- **N3/B3=4**: Tags da interface RAS

3.1 Tags gerais do IOKit (N2/B2 = 0)

Os tags seguintes são fornecidos pelo IOKit para todas as interfaces de I/O suportadas.

BLOCO [tamanho=4] Read Driver Events (-1, 0, 0, 1) - "IO.IOKitEvent"	apenas leitura
<p>Este bloco retorna eventos do driver gerados por várias fontes do IOKit (veja <i>Apêndice I – Eventos do IOKit</i> para a lista de todos os eventos gerados pelo IOKit). A propriedade timestamp do bloco informa o tempo que o evento ocorreu. Os elementos do bloco são:</p> <ul style="list-style-type: none"> - Elemento 0: Tipo do evento <ul style="list-style-type: none"> 0 = Informação 1 = Aviso 2 = Erro - Elemento 1: Fonte do evento <ul style="list-style-type: none"> 0 = Driver (específico do driver) -1 = IOKit (Eventos genéricos de I/O do Kit) -2 = Interface Serial -3 = Interface Modem -4 = Interface Ethernet -5 = Interface RAS - Elemento 2: Número do Erro (específico de cada fonte do evento) - Elemento 3: Mensagem do Evento (string, específico de cada evento) <p>* O driver mantém um máximo de 100 eventos internamente. Caso um novo evento seja reportado o evento mais antigo é descartado.</p>	

TAG Physical Layer Status (-1, 0, 0, 2) - "IO.PhysicalLayerStatus"	apenas leitura
<p>Este tag indica o estado do nível físico. O valor deste tag pode ser:</p> <p>0 – Nível físico parado (o driver está em offline, o nível físico falhou ao iniciar ou excedeu o número máximo de tentativas de reconexão)</p> <p>1 – Nível físico iniciado mas não conectado (o driver está online, mas o nível físico não está conectado. Se você habilitou o gerenciamento automático de conexão, o nível físico ou está conectando, desconectado ou aguardando por uma tentativa de reconexão. Caso você tenha selecionado o gerenciamento manual de conexão, o nível físico permanecerá neste estado até que você force-o a conectar)</p>	

TAG Physical Layer Status (-1, 0, 0, 2) – “IO.PhysicalLayerStatus”	apenas leitura
---	----------------

2 – Nível físico conectado (o nível físico está pronto para ser utilizado). Isto NÃO significa que o dispositivo está conectado, mas apenas que o meio de acesso está funcionando.

TAG/BLOCO Set Configuration Parameters (-1, 0, 0, 3) – “IO.SetConfigurationParameters”	somente escrita
---	-----------------

Use este tag para modificar em execução qualquer propriedade do dialog (a lista completa de propriedades pode ser encontrada no *Apêndice II – Propriedades do IOKit*).

Este tag funciona apenas enquanto o driver estiver OFFLINE. Para iniciar o driver em offline, verifique a opção “Start driver OFFLINE”, presente na página Setup do driver. Você pode também escrever em um TAG simples ou escrever em um BLOCO contendo os parâmetro que você deseja modificar (escritas individuais de elemento de bloco não são suportadas, o bloco inteiro deve ser escrito de uma vez só!!!!)

No Eclipse SCADA, você precisará usar um tag BLOCO. Cada parâmetro que você queira modificar usa dois elementos de bloco. Por exemplo, se você deseja configurar 3 parâmetros, então o tamanho do bloco deverá ser 6 (3 * 2). O primeiro elemento é o nome da propriedade (como um string), enquanto o segundo elemento é o valor da propriedade. Observe o seguinte código de script do Elipse SCADA:

```
// 'Block' deverá ser um tag Bloco com leitura automática,
// leitura por scan e escrita automática DESABILITADOS

// 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()
```

Se você estiver usando o E3, a habilidade de criar arrays em execução permite que você use tanto o I/O tag quando o tag Bloco. Você pode também usar o método Driver.Write para enviar os parâmetros diretamente ao driver sem a necessidade de criar um tag. Veja os exemplos 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

' você tem dois métodos de enviar os parâmetros
' Método 1: usando um tag I/O
tag.WriteEx arr

' Método 2: sem uso de tag
Driver.Write -1, 0, 0, 3, arr
```

Uma variação do exemplo acima usa um array bidimensional:

```
Dim arr(10)
```

TAG/BLOCO Set Configuration Parameters (-1, 0, 0, 3)
– "IO.SetConfigurationParameters"

somente escrita

```
' configura os elementos do array, note que o array foi dimensionado
' para 10 elementos. Elementos de array 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
```

O driver não valida os nomes dos parâmetros nem os valores passados, portanto seja cuidadoso para escrever os nomes das propriedades corretamente. O comando de escrita falhará se o array configurado estiver incorretamente montado. Você pode verificar o log do driver ou utilizar o parâmetro *writeStatus* da função *WriteEx* para a causa exata do 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 configurando os parâmetros do driver: " + strError
End If
```

TAG Work ONLINE (-1, 0, 0, 4) – "IO.WorkOnline"

leitura/escrita

Este tag informa o status atual do driver e permite que você inicie ou pare o nível físico.

0 – Driver OFFLINE: o nível físico está fechado (parado). Este modo permite a configuração dinâmica dos parâmetros do driver usando o tag *Set Configuration Parameters*.

1 – Driver ONLINE: o nível físico está aberto (rodando). Enquanto ONLINE, o nível físico pode estar conectado ou desconectado (você pode verificar o status atual com o tag *Physical Layer Status*)

O seguinte exemplo em E3 coloca o driver em offline, muda a porta COM e coloca o driver em online novamente:

```
'Para o driver colocando-o em OFFLINE
Driver.Write -1, 0, 0, 4, 0

'Muda a porta para COM2
Driver.Write -1, 0, 0, 3, Array("IO.Serial.Port", 2)

'Coloca o driver em ONLINE
Driver.Write -1, 0, 0, 4, 1
```

O comando de Write pode falhar quando configurando o driver para ONLINE (escrevendo 1). Neste caso o driver continua em OFFLINE. A causa dessa falha pode ser:

- O tipo do nível físico configurado incorretamente (provavelmente um valor inválido foi configurado para a propriedade "IO.Type")
- O driver pode ter ficado sem memória suficiente
- O nível físico falhou criando seu próprio thread de tarefa (verifique o arquivo de log, procurando pela mensagem "Failed to create physical layer thread!")
- O nível físico falhou ao inicializar. A causa da falha depende do tipo do nível físico. Pode ser causado por uma porta serial inválida, falha na inicialização dos Windows sockets, falha inicializando o TAPI (modem), etc. A causa será gerada no arquivo de log.

IMPORTANTE: caso você obtenha sucesso colocando o driver em ONLINE, isto não significa

TAG Work ONLINE (-1, 0, 0, 4) – "IO.WorkOnline"	leitura/escrita
---	-----------------

que o nível físico está pronto para uso (pronto para efetuar I/O com o dispositivo externo). Você deve verificar o tag *Physical Layer Status* para ter certeza que o nível físico está conectado e pronto para efetuar operações de troca de dados.

3.2 Tags da Interface Serial (N2/B2 = 2)

Atualmente não existem tags definidos especialmente para o gerenciamento da Interface Serial em execução.

3.3 Tags da Interface Modem (N2/B2 = 3)

Os seguintes tags permitem o controle e diagnóstico da Interface Modem em execução. Estes tags estão disponíveis APENAS quando o driver estiver ONLINE.

TAG Get/Set Phone Number (-1, 0, 3, 0) – "IO.TAPI.PhoneNumber"	leitura/escrita
--	-----------------

(String) Este tag lê/modifica o número do telefone usado no comando *Dial*. Se você modificar este tag, o novo valor será usado somente no próximo comando *Dial*.

TAG Dial (-1, 0, 3, 1) – "IO.TAPI.Dial"	somente escrita
---	-----------------

Escreva qualquer valor neste tag para forçar a interface modem a iniciar a discagem. Este comando é assíncrono, portanto apenas inicia o processo de discagem. Você pode monitorar o tag *Is Modem Connected* para detectar quando ligação está estabelecida.

TAG Get Modem Status (-1, 0, 3, 2) – "IO.TAPI.ModemStatus"	somente leitura
--	-----------------

Retorna um string com o status atual do modem. Os valores possíveis são:

- **"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!"** – o driver falhou ao inicializar a linha do modem. Verifique o arquivo de log para maiores detalhes do erro.
- **"Modem error at dial!"** – o driver falhou ao iniciar ou aceitar uma ligação.
- **"Connecting..."** – o driver iniciou com sucesso uma chamada e agora a está processando.
- **"Ringing..."** – indica que o modem está recebendo uma chamada (mas ainda não foi aceita).
- **"Connected!"** – o driver está conectado com sucesso (tanto por completar um ligação quanto por aceitar uma chamada externa).
- **"Disconnecting..."** – o driver está desligando a ligação atual.
- **"Disconnected OK!"** – o driver desligou a ligação atual.
- **"Error: no dial tone!"** – o driver abortou a ligação pois o tom de discagem não foi detectado.
- **"Error: busy!"** – o driver abortou a ligação porque a linha estava ocupada.

TAG Get Modem Status (-1, 0, 3, 2) – "IO.TAPI.ModemStatus" somente leitura

- **"Error: no answer!"** – o driver abortou a ligação porque não obteve resposta do outro modem.

- **"Error: unknown!"** – o driver abortou a ligação porque ocorreu um erro desconhecido.

TAG Is Modem Connected (-1, 0, 3, 3) – "IO.TAPI.IsModemConnected" somente leitura

Este tag indica o estado da conexão do modem. Os valores possíveis são:

0 – o modem não está conectado (pode estar conectando ou aceitando uma ligação)

1 – o modem está conectado, o driver tanto completou uma ligação quanto aceitou uma ligação externa com sucesso. O nível físico está apto a enviar/receber dados enquanto estiver neste estado.

TAG Hang-up Call (-1, 0, 3, 4) – "IO.TAPI.HangUp" apenas escrita

Escreva qualquer valor neste tag para terminar a ligação atual.

NOTA: Você deve usar este comando apenas se você está gerenciando o nível físico manualmente ou se você deseja forçar explicitamente o driver a reiniciar a conexão. Caso o nível físico tenha sido configurado para reconexão automática, o driver irá automaticamente tentar restabelecer a conexão.

TAG Connection Baud-rate (-1, 0, 3, 5)
– "IO.TAPI.ConnectionBaudRate" somente leitura

Indica o baud rate da conexão atual. Caso o modem não esteja conectado o valor deste tag será 0.

TAG Is Modem Connecting (-1, 0, 3, 6)
– "IO.TAPI.IsModemConnecting" somente leitura

Este tag indica o estado da conexão do modem. Os valores possíveis são:

0 – o modem não está conectado.

1 – o modem está conectando (discando ou aceitando uma ligação externa).

2 – o modem está conectado, o driver tanto completou uma ligação quanto aceitou uma ligação externa com sucesso. O nível físico está apto a enviar/receber dados enquanto estiver neste estado.

3 – o modem está desconectando (a chamada atual está sendo finalizada).

3.4 Tags da Interface Ethernet Interface (N2/B2 = 4)

Os seguintes tags permitem o controle e diagnóstico da Interface Ethernet em execução (também válidos quando a Interface RAS estiver selecionada). Estes tags estão disponíveis APENAS quando o driver estiver ONLINE.

TAG	IPSelect (-1, 0, 4, 0) – "IO.Ethernet.IPSelect"	leitura/escrita
Indica qual dos IPs especificados na configuração da interface Ethernet está ativo: <ul style="list-style-type: none">0: o IP principal está selecionado (ativo)1: o IP de reserva está selecionado (ativo)		
Se a interface Ethernet estiver conectada, este tag indica em qual dos dois IPs que a conexão foi estabelecida. Se estiver desconectada, este tag indica em qual IP o IOKit tentará conectar primeiro na próxima tentativa de conexão.		
Durante a conexão, se o IP ativo não estiver disponível, o IOKit tenta a conexão com o outro IP. Se esta conexão tiver sucesso, então este passa para o estado ativo.		
Para alterar o IP ativo, basta escrever 0 ou 1 neste tag. Se o driver estiver conectado, esta escrita forçará uma reconexão com o IP especificado. Se o driver estiver desconectado, esta escrita simplesmente configura qual o primeiro IP que será utilizado na próxima tentativa de conexão.		

TAG	IPSwitch (-1, 0, 4, 1) – "IO.Ethernet.IPSwitch"	somente escrita
Qualquer valor escrito neste tag força uma troca do IP ativo (do principal para o backup, ou do backup para o principal). Se o driver estiver conectado, esta escrita forçará uma reconexão com o novo IP ativo. Se o driver estiver desconectado, esta escrita simplesmente configura qual o primeiro IP que será utilizado na próxima tentativa de conexão.		

3.5 RAS Interface Tags (N2/B2 = 5)

Atualmente não existem tags definidos especialmente para o gerenciamento da Interface RAS em execução.

4 Tópicos Avançados

Este capítulo contém informações detalhadas do funcionamento interno do IOKit.

4.1 Estados do Driver

O driver pode estar em 4 distintos estados:

- 1) Parado
- 2) OffLine (iniciado)
- 3) Desconectado (iniciado, online)
- 4) Conectado (iniciado, online)

O estado **parado** é quando o driver não está sendo executado ou não está carregado (tipicamente quando você está configurando a aplicação).

Tão logo você executa a aplicação (**inicia**), o driver vai para o estado **offline**. Neste estado ele detecta as condições mínimas para rodar (basicamente o DLL do driver foi carregado) e está aguardando por mais parâmetros de configuração ou por um comando que peça que ele vá para o estado **online**.

O estado **online** tem duas divisões principais: **conectado** ou **desconectado**. O driver está **desconectado** quando o nível físico não é capaz de transmitir/receber dados. O driver está **conectado** quando o nível físico está pronto para utilização.

NOTA: os estados definidos aqui aplicam-se apenas ao **nível físico**. Tendo o nível físico conectado faz com que o driver acesse o dispositivo de I/O (plc), porém o dispositivo pode ainda não estar respondendo. Estando o nível físico conectado é um bom começo, porém você deve sempre verificar os tags específicos do driver que informam o estado do driver.

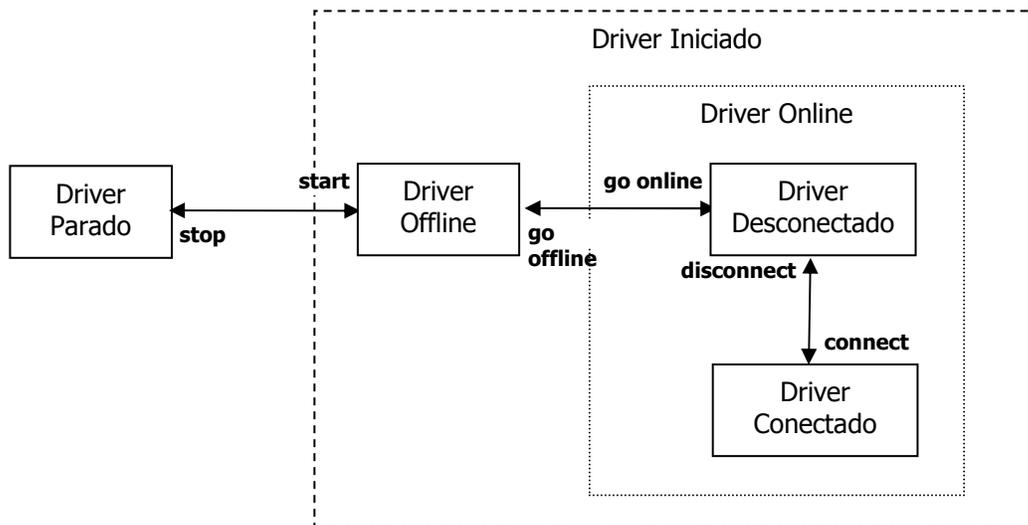


Figura 1 – Estados do Driver IOKit

4.2 Trabalhando OFFLINE

O modo OFFLINE foi planejado para sistemas onde o driver necessita ser configurado em execução. Este é o caso quando você simplesmente não sabe o tipo de conexão ou parâmetros até que a aplicação esteja sendo executada.

Em OFFLINE, todos os tags de I/O do driver irão falhar (todas as escritas e leituras). Os únicos tags permitidos são os *Tags gerais do IOKit* (N1/B1=-1 e N2/B2=0).

Embora você possa chavear o driver entre os modos ONLINE e OFFLINE em execução, geralmente você irá tomar os seguintes passos:

- 1) Configurar a opção "Start driver OFFLINE" no dialog do driver "Extras...". Esta opção fará com que o driver inicie no estado OFFLINE.
- 2) Configurar os parâmetros do driver em um script da aplicação. O script pode ser executado automaticamente ou de acordo com uma solicitação do usuário.
- 3) Configurar o driver para ONLINE

4.3 Gerenciamento da Conexão

Quando o driver está ONLINE, ele entra em um estado de tentativa de conectar o nível físico. Esta conexão pode ser efetuada de três maneiras:

- **conexão automática:** o driver mantém a conexão como configurado no dialog do driver.
- **conexão manual:** o driver se mantém desconectado e aguarda por comandos de aplicação (ex: comando dial) para conectar o nível físico.
- **modo de espera:** o driver se mantém desconectado e age como um escravo, aceitando conexões solicitadas por outros dispositivos na linha física.

Esta seção também descreve a **deteção de inatividade**, que automaticamente desconecta o nível físico caso não receba nenhum byte em um determinado período de tempo.

4.3.1 Conexão Automática

Selecione a conexão automática de duas formas:

- no dialog do driver, selecione "Automatic (managed by the driver)" na lista do modo de conexão.
- configure a propriedade "*IO.ConnectionMode*" para 0 (automático).

O algoritmo de conexão automática inicia quando o driver é configurado para o modo ONLINE. Nesta hora o driver estabelece a conexão inicial. A partir deste ponto, o driver irá comportar-se de acordo com as opções de gerenciamento da conexão.

Caso você habilite o resgate da conexão ("*Retry failed connection every nn seconds*") então o driver irá automaticamente tentar reconectar o nível físico se a conexão está perdida. Se o resgate da conexão é desabilitado e a conexão é perdida então o driver irá retornar para o modo offline.

Você também pode definir o número máximo de tentativas de conexão ("*Give up after nn failed connections*"), limitando o número de reconexões com o nível físico. Caso todas as tentativas forem feitas, então o driver é abortado e colocado novamente no estado offline. O *failed connections counter* é zerado (setado para 0) quando a conexão for bem sucedida.

4.3.2 Conexão Manual

Quando configurado para trabalhar em modo manual, o driver mantém o estado **desconectado** após ser colocado em **online**. Então você pode forçar manualmente a conexão do nível físico utilizando os tags de gerenciamento da conexão.

Por exemplo, se você estiver usando o nível físico do modem, você poderá ter uma tela onde edita o número do telefone (usando um setpoint) e um botão para iniciar a discagem. Caso a

discagem funcione então o driver irá automaticamente ir para o estado **conectado**. Você pode desconectar o driver novamente, escrevendo no tag *hang-up*.

4.3.3 Modo de espera

O modo de espera é quase idêntico ao modo manual, mas a diferença está que o nível físico será programado para aceitar conexões externas. Para o nível de ethernet, você teria que marcar a opção "Listen for connections on port". Para o nível de modem, você teria que marcar a opção "Accept incoming calls".

Para o funcionamento do modo de espera, é necessário configurar o driver para funcionar no modo manual. O driver irá continuar no estado **desconectado** até que o nível físico sinalize que o uma conexão está disponível. Você pode verificar quando a conexão é estabelecida monitorando o tag *Physical Layer Status*.

4.3.4 Detecção de Inatividade

O driver pode ser programado para desconexão automática do nível físico se ele se mantém inativo por um período de tempo. O driver considera que o nível físico está inativo se ele continuamente solicita dados (tenta receber caracteres) sem receber NENHUM dado. Caso algum caractere seja recebido (mesmo que não seja um caractere válido do protocolo), o temporizador de inatividade é reinicializado.

O temporizador de inatividade é iniciado sempre que ocorre um timeout na recepção de caracteres no nível físico. O temporizador é reinicializado (parado) quando qualquer byte é recebido pelo nível físico.

O tempo máximo de inatividade deve ser maior que o timeout do nível físico, caso contrário o nível físico pode ser considerado inativo durante o funcionamento normal do driver. É recomendado um mínimo de 10 segundos para o tempo máximo de inatividade.

4.4 Threads do Driver

Tanto o E3 quanto o Elipse SCADA utilizam os benefícios de multithreading nas suas arquiteturas de captura de dados. O Multithreading permite aos drivers operarem troca de dados simultaneamente, sem bloqueios entre os mesmos. Nesta seção nós vamos estudar os threads de I/O tanto do Elipse SCADA quanto do E3.

4.4.1 Threads no Elipse SCADA

Quando é utilizado o Elipse SCADA (ELIPSE32.EXE), você tipicamente tem os seguintes threads (apenas threads de I/O estão listados aqui):

- O thread PRINCIPAL: este thread roda a interface do usuário e todos os scripts da aplicação. Todas as solicitações **síncronas** do driver (leituras/escritas chamadas por scripts ou pela interface do usuário, incluindo escritas assíncronas) são geradas neste thread. O thread PRINCIPAL é bloqueado enquanto estas solicitações síncronas estão sendo processadas pelo driver.
- O thread do DRIVER: este thread é onde o driver é executado. Existe um thread de DRIVER para cada driver declarado na aplicação. Este manuseia as leituras em background (polling) dos tags e também as solicitações síncronas geradas pelo thread principal. Se o driver está configurado como "manter o comportamento 16-bit", então este thread não existe e todas as solicitações (síncronas e leituras em background) são manipulados pelo thread PRINCIPAL.
- O thread do NÍVEL FÍSICO: este thread é iniciado quando o driver entra em ONLINE. Ele manipula as solicitações de I/O enviados pelo thread do DRIVER e também é responsável por gerenciar a conexão (conectar, desconectar, retentar conexão, etc). As solicitações

do thread do DRIVER podem ser apenas manipuladas pelo thread do NÍVEL FÍSICO quando o driver está **conectado**.

4.4.2 Threads no E3

Quando utilizando o E3, cada driver tem seu próprio processo (IOSERVER.EXE). Todos os drivers são gerenciados pelo módulo runtime do E3 (E3RUN.EXE). Nós vamos apenas comentar aqui os threads do IOSERVER:

- O thread PRINCIPAL: este thread recebe solicitações do E3Run. Estas solicitações incluem: iniciar/parar o driver, comandos de escrita, iniciar/parar a leitura por varredura de tags.
- O thread de CALLBACK: este thread é responsável por enviar os valores de volta ao E3RUN.
- O thread do DRIVER: este thread (como o thread do DRIVER do Elipse SCADA) é onde o driver realmente é executado.
- O thread do NÍVEL FÍSICO: este thread (como o thread do NÍVEL FÍSICO do Elipse SCADA) manuseia solicitações de I/O enviadas pelo thread do DRIVER, gerenciando também a conexão física. É iniciado apenas quando o driver está ONLINE.

5 APÊNDICE I – Eventos do IOKit

Este apêndice lista os eventos atualmente gerados pelo IOKit. Para acessar estes eventos você deve declarar o tag *Read Driver Events* block (B1=-1, B2=0, B3=0, B4=-1, Tamanho=4)

Tipo	Fonte	Código	Mensagem
Erro (2)	Serial (-2)	0	Error opening serial port: %s <ul style="list-style-type: none"> - tried to open port COM%u twice - failed to open port %s (windows error %u) - failed to configure port %s (windows error %u)
Erro (2)	Modem (-3)	0	Error opening modem line: %s <ul style="list-style-type: none"> - tried to open line twice - failed to initialize line (tapi error %s) - failed to negotiate API version (tapi error %s) - failed to open line (tapi error %s) - failed to configure status messages (tapi error %s) - failed to get line address status (tapi error %s) - out of memory getting line address status - failed to get device capabilities (tapi error %s) - device ID=%u not found - out of memory getting device capabilities
Info (0)	Modem (-3)	2	Begin of available modem list
Info (0)	Modem (-3)	3	%u:%s <ul style="list-style-type: none"> - %u é o ID do modem (em decimal) - %s é a descrição do modem
Info (0)	Modem (-3)	4	End of available modem list
Erro (2)	Ethernet (-4)	0	Error connecting to %s on port %u: %s Error connecting to %s on port %u (backup IP): %s <ul style="list-style-type: none"> - tried to connect socket twice - null address - socket error %s(%d) calling %s()
Erro (2)	Ras (-5)	1	Timeout waiting for CONNECT
Info (0)	Ras (-5)	3	RAS response: '%s'
Erro (2)	Ras (-5)	4	RAS connection error: NO CARRIER
Info (0)	IOKit (-1)	1	Initializing physical layer...
Erro (2)	IOKit (-1)	2	Physical layer initialization failed!
Info (0)	IOKit (-1)	3	Physical layer initialized!
Info (0)	IOKit (-1)	4	Connecting physical layer...
Erro (2)	IOKit (-1)	5	Failed to connect physical layer!
Info (0)	IOKit (-1)	6	Physical layer connected!
Erro (2)	IOKit (-1)	7	Physical layer connection lost!
Info (0)	IOKit (-1)	8	Reconnecting physical layer...
Erro (2)	IOKit (-1)	9	Failed to reconnect physical layer!

Tipo	Fonte	Código	Mensagem
Info (0)	IOKit (-1)	10	Physical layer reconnected!
Info (0)	IOKit (-1)	11	Physical layer connected automatically!
Info (0)	IOKit (-1)	12	Reconnecting physical layer (retry #%u)...
Erro (2)	IOKit (-1)	13	Failed to reconnect physical layer (retry #%u)!
Info (0)	IOKit (-1)	14	Physical layer reconnected (on retry #%u)!
Info (0)	IOKit (-1)	15	Terminating physical layer...
Info (0)	IOKit (-1)	16	Physical layer terminated! (%u bytes sent, %u bytes received)
Info (0)	IOKit (-1)	17	Disconnecting physical layer...
Erro (2)	IOKit (-1)	18	Physical layer aborted (exhausted all retries)!
Erro (2)	IOKit (-1)	19	Physical layer non-responsive for %u seconds, disconnecting...

6 APÊNDICE II – Propriedades do IOKit

Este apêndice lista os parâmetros suportados pelo IOKit. Os parâmetros podem ser modificados em execução pela escrita no tag *Set Configuration Parameters*.

6.1 Parâmetros Gerais:

Estes são os parâmetros gerais do nível físico:

IO.Type	String
<p>Define o tipo da interface de nível físico usado pelo driver. Os valores possíveis são:</p> <ul style="list-style-type: none"> - "N" ou "None": não usa nenhuma interface de comunicação. - "S" ou "Serial": use uma porta serial local (COMn). - "M" ou "Modem": use um modem local (interno ou externo) acessado via TAPI (Telephony Application Programming Interface). - "E" ou "Ethernet": use um socket TCP/IP ou UDP/IP. - "R" ou "RAS": use uma interface RAS (Remote Access Server). O driver conecta com o dispositivo RAS usando uma interface Ethernet e então utiliza um comando AT (discagem). 	
IO.TimeoutMs	Inteiro
<p>Define o timeout do nível físico (em milissegundos; 1 segundo = 1000).</p>	
IO.StartOffline	Boleano
<p>TRUE para iniciar o driver offline, FALSE para iniciar o driver online.</p> <p>NOTA: não faz sentido em modificar esta propriedade em runtime, pois ele só pode ser modificado se o driver já estiver offline. Para colocar o driver online em execução, escreva 1 no tag <i>Work ONLINE</i>.</p>	
IO.ConnectionMode	Inteiro
<p>Controla o modo de gerenciamento da conexão:</p> <ul style="list-style-type: none"> - 0: modo automático (o driver gerencia a conexão) - 1: modo manual (a aplicação gerencia a conexão) 	
IO.RecoverEnable	Boleano
<p>TRUE para habilitar o driver a resgatar conexões perdidas, FALSE para fazer com que o driver fique OFFLINE quando a conexão é perdida.</p>	

IO.RecoverPeriodSec Inteiro

Número de segundos a aguardar entre duas tentativas de reconexão.

NOTA: a primeira reconexão é feita imediatamente após a conexão ser perdida.

IO.GiveUpEnable Boleano

TRUE para definir o número máximo de tentativas de reconexão. Se todas as reconexões falharem, o driver ficará OFFLINE. Caso seja configurado para **FALSE** o driver irá tentar indefinidamente até que obtenha sucesso na reconexão.

IO.GiveUpTries Inteiro

Número de tentativas de reconexão antes de abortar a reconexão. Por exemplo, caso seja utilizado o valor **1**, então o driver irá tentar apenas uma reconexão quando a conexão é perdida. Caso falhe o driver irá para OFFLINE.

IO.InactivityEnable Boleano

TRUE para habilitar e **FALSE** para desabilitar a detecção de inatividade. O nível físico será desconectado caso fique inativo por um período de tempo. Será considerado inativo apenas se o nível físico é capaz de enviar dados, mas não recebe nenhum dado de retorno.

IO.InactivityPeriodSec Inteiro

Número de segundos de checagem da inatividade. Caso o nível físico está inativo por este período de tempo ele será desconectado.

6.2 Parâmetros do Log:

Estes parâmetros controlam a geração do arquivo de log.

IO.Log.Enable Boleano

Coloque em **TRUE** para habilitar e em **FALSE** para desabilitar a criação do arquivo de log.

IO.Log.FileName String

String definindo o nome do arquivo de log.

6.3 Parâmetros da Interface Serial

Estes parâmetros controlam a configuração da Interface Serial:

IO.Serial.Port

Inteiro

Número da porta local serial:

- **1:** use COM1
- **2:** use COM2
- **3:** use COM3
- ...

IO.Serial.Baudrate

Inteiro

Define o baud rate da porta serial. Exemplo: 9600

IO.Serial.DataBits

Inteiro

Define o número de bits de dados configurados para a porta serial. Os valores possíveis são:

- **5:** 5 data bits
- **6:** 6 data bits
- **7:** 7 data bits
- **8:** 8 data bits

IO.Serial.StopBits

Inteiro

Define o número de bits de parada configurados para a porta serial. Os valores possíveis são:

- **1:** um stop bit
- **2:** um stop bit e meio
- **3:** dois stop bits

IO.Serial.Parity

String

Define a paridade configurada para a 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 mark
- **"S"** ou **"Space"**: paridade space

IO.Serial.RTS

String

IO.Serial.RTS

String

Indica como o driver irá manipular o sinal de RTS:

- **"OFF"**: RTS sempre desligado
- **"ON"**: RTS sempre ligado
- **"Toggle"**: somente liga o RTS quando está transmitindo dados.

IO.Serial.DTR

String

Indica como o driver irá manipular o sinal de DTR:

- **"OFF"**: DTR sempre desligado
- **"ON"**: DTR sempre ligado

IO.Serial.DelayBeforeMs

Inteiro

Número de milissegundos de espera antes de ligar o RTS, antes de enviar os dados. Esta opção é apenas disponível quando o RTS está em **"Toggle"** e o WaitCTS é **FALSE**.

IO.Serial.DelayAfterMs

Inteiro

Número de milissegundos de espera depois que o último byte é enviado pela serial antes de desligar o sinal de RTS. Disponível apenas quando o RTS está **"Toggle"** e o WaitCTS é **FALSE**.

IO.Serial.WaitCTS

Boleano

TRUE para fazer com que o driver aguarde pelo sinal de CTS antes de enviar os bytes quando o RTS é ligado. Disponível apenas quando a opção do RTS esta configurada para **"Toggle"**.

IO.Serial.CTSTimeoutMs

Inteiro

Número de milissegundos de espera pelo sinal de CTS. Depois que o RTS é ligado, um temporizador é iniciado para aguardar pelo sinal de CTS. Caso este temporizador expire, o driver aborta o envio dos bytes através da porta serial. Disponível apenas quando o RTS está em **"Toggle"** e o WaitCTS é **TRUE**.

IO.Serial.InterbyteDelayUs

Inteiro

Número de microsegundos (1/1000000 de segundo) de espera entre dois bytes enviados pela interface serial.

IO.Serial.InterframeDelayMs

Inteiro

Número de milissegundos a esperar entre o envio de dois pacotes consecutivos, ou entre a recepção de um pacote e o envio de outro pacote.

IO.Serial.SupressEcho

Inteiro

Use um valor diferente de zero para habilitar a recepção de 'eco', ou zero para desabilitar.

6.4 Parâmetros da Interface TAPI

Estes parâmetros controlam a configuração da interface Modem.

IO.TAPI.AcceptIncoming

Inteiro

FALSE se o modem não deve aceitar chamadas (o driver age como mestre), **TRUE** para habilitar o recebimento de chamadas (o driver age como escravo).

IO.TAPI.PhoneNumber

String

Número do telefone a ser usado nos comandos de discagem. Por exemplo: "0w01234566" (o 'w' força o modem que espere que tom de discagem).

IO.TAPI.ModemID

Inteiro

O ID (identificador) do modem. Este ID é criado pelo Windows e é utilizado internamente para identificar o modem dentro da lista de dispositivos instalador no computador. Você não deve supor que o ID permanecerá o mesmo caso o modem seja reinstalado ou se você rodar a aplicação em outro computador.

É fortemente recomendado que este parâmetro **seja configurado para 0**, indicando que o driver deve utilizar o primeiro modem disponível.

6.5 Parâmetros da Interface Ethernet

Estes parâmetros controlam a configuração da Interface Ethernet (observe que a Interface Ethernet é também utilizada pela Interface RAS).

IO.Ethernet.MainIP

String

Endereço IP do dispositivo destino. Você pode usar tanto o IP por pontos quando o nome do HOST do dispositivo. Exemplos: "192.168.0.6" ou "SERVER1"

IO.Ethernet.MainPort

Inteiro

IO.Ethernet.MainPort	Inteiro
Número da porta IP do dispositivo destino (usado em conjunto com o MainIP).	
IO.Ethernet.BackupEnable	Boleano
TRUE para habilitar o endereço IP backup. Caso ocorra uma falha ao conectar ao MainIP o driver tentará usar o endereço backup. FALSE para desabilitar o uso do endereço IP backup.	
IO.Ethernet.BackupIP	String
Endereço IP do dispositivo destino alternativo. Você pode usar tanto o IP por pontos quando o nome do HOST do dispositivo. Exemplos: "192.168.0.6" ou "SERVER1"	
IO.Ethernet.BackupPort	Inteiro
Número da porta IP do dispositivo destino alternativo (usado em conjunto com o BackupIP).	
IO.Ethernet.PingEnable	Boleano
TRUE para habilitar o ping do endereço IP do dispositivo destino antes da tentativa de conexão ao socket. O timeout de conexão ao socket não pode ser controlado, portanto pingar o endereço antes de conectar é uma maneira rápida de detectar se a conexão irá falhar. FALSE desabilita o ping.	
IO.Ethernet.PingTimeoutMs	Inteiro
Número de milissegundos de espera por uma resposta de ping.	
IO.Ethernet.PingTries	Inteiro
Número máximo de tentativas de ping (mínimo é 1, incluindo o primeiro ping).	
IO.Ethernet.Transport	Inteiro
Define o protocolo transporte: <ul style="list-style-type: none">- "T" ou "TCP": use TCP/IP- "U" ou "UDP": use UDP/IP	
IO.Ethernet.AcceptConnection	Boleano

IO.Ethernet.AcceptConnection

Boleano

FALSE se o driver não deverá aceitar conexões externas (o driver age como um mestre), **TRUE** para habilitar o driver a receber conexões (o driver age como um escravo).

IO.Ethernet.ListenPort

Inteiro

Número da porta IP, utilizada pelo driver para aguardar conexões.

6.6 Parâmetros da Interface RAS

Estes parâmetros controlam a configuração da Interface RAS. Observe que a interface RAS usa a interface Ethernet, portanto você deverá ter o cuidado de configurar a interface Ethernet.

IO.RAS.ATCommand

String

Comando AT enviado através do socket para forçar o dispositivo RAS a discar usando o canal RAS atual. Exemplo: "ATDT6265545"

IO.RAS.CommandTimeoutSec

Inteiro

Número de segundos de espera pela mensagem "CONNECT" em resposta ao comando AT.

7 Histórico de Revisões

Versão	Data	Autor	Comentários
v1.14	2007-01-15	F. Englert	<p>Corrigido: o IOKit pode travar enquanto um driver serial está sendo terminado. (Case 5313)</p> <p>Melhorado o tempo de resposta de drivers escravos pela porta serial. (Case 7280)</p> <p>O IOKit agora enumera os modems disponíveis (com os seus respectivos IDs) quando o driver é ativado com um modem ID inválido/desconhecido. (Case 7474)</p> <p>Corrigido: as primeiras operações de I/O (leituras/escritas) podem falhar enquanto o IOKit está tentando fazer a primeira conexão (Case 7614)</p> <p>Corrigido: o IOKit pode gerar um GPF quando um número de ponto flutuante (float ou double) muito grande é lido/escrito (ex: -4585887366944162000000000000.0) (Case 7806)</p>
v1.13	2006-08-10	F. Englert	<p>Corrigido: A estampa de tempo de valores lidos ou escritos pode ficar incorreta por 1 milissegundo. (Case 7261)</p>
v1.12	2006-07-03	F. Englert	<p>Corrigido: Vazamento de memória quando o driver retorna um HVALUE dimensionado como um array de HVALUES. (Case 7092)</p> <p>Corrigido: Driver com um thread separado de comunicação pode travar ou falhar quando o IOKit é colocado OFFLINE. (Case 7099)</p>
v1.11	2006-05-11	F. Englert	<p>Agora é possível verificar a versão do IOKitLib que foi compilada com um determinado driver. Uma função com nome do tipo "IOKitLib_v1.11()" é exportada pela DLL do driver, permitindo verificar facilmente qual versão do IOKitLib foi utilizada (Case 6968)</p> <p>Corrigido: o tempo de timeout não é respeitado no envio de dados da interface Ethernet. Este bug poderia pendurar o driver caso o PLC mantivesse um socket aberto sem ler os dados recebidos neste socket. (Case 6935)</p> <p>Corrigido: a opção 'disconnect if inactive' não funciona se o driver foi implementado em 'modo de escuta' (Case 6933)</p> <p>Corrigido: não são feitas tentativas de reconexão da interface Serial caso falhe a abertura da porta (Case 6865)</p>
v1.10	2005-12-20	F. Englert	<p>O IOKit agora pode gerar um arquivo de log por dia se a macro "%DATE%" for colocada no nome do arquivo (Case 2047)</p> <p>Corrigido: o log do IOKit pode misturar as linhas com TX: ou RX: geradas por threads de comunicação diferentes (Case 6530)</p>

Versão	Data	Autor	Comentários
v1.09	2005-12-07	F. Englert	<p>Corrigido: a interface de Modem não lista ou não abre um modem se existirem modems incorretamente instalados no sistema (Case 3613)</p> <p>Agora uma mensagem é mostrada se o usuário habilita a opção "Start driver OFFLINE", perguntando se o usuário tem certeza que quer iniciar OFFLINE (Case 4584)</p> <p>Os parâmetros de configuração do IOKIT agora são listados no arquivo de log quando o driver é iniciado (Case 4584)</p> <p>Corrigido: os delays na configuração do chaveamento da interface serial não funcionam em Windows NT (Case 5525)</p> <p>Adicionada a opção "Inter-frame delay (milliseconds)" na configuração da interface Serial (também disponível na interface Modem) (Case 5525)</p> <p>A mensagem escrita no arquivo de log quando o PLC fecha o socket ("recv3() returned error (unknown) (0)") foi alterada para "socket gracefully closed by the remote partner" (Case 5599)</p> <p>A interface Ethernet agora utiliza o WinSock v2.2 se estiver disponível (as versões anteriores do IOKit utilizavam apenas o WinSock v1.1) – alguns PLCs exigem esta versão para comunicar corretamente (Case 5617)</p> <p>Adicionada a opção "Enable 'ECHO' supression" (habilita a supressão de 'ECO' na comunicação) na interface Serial (também disponível na interface Modem) (Case 5647)</p> <p>Adicionado o botão "Modem settings..." na página de configuração da interface Modem, permitindo que o usuário configure o modem selecionado (Case 5656)</p> <p>Adicionado suporte a auto-declaração dos tags internos do IOKit (utilizado pela ferramenta "TagBrowser" do E3) (Case 6016)</p> <p>Adicionadas três novas opções de 'baudrate' na página de configuração da interface Serial: 38700, 57600 e 115200 baud (Case 6076)</p> <p>O IOKit agora pode receber e identificar a fonte (IP) de pacotes UDP enviados em 'broadcast'. Esta ferramenta permite que alguns drivers utilizem o broadcast UDP para identificar os equipamentos presentes na rede (Case 6085)</p> <p>O IOKit agora não tenta reconectar imediatamente se uma conexão recém feita for perdida (Case 6294)</p> <p>Adicionada a opção "None" à lista de interfaces disponíveis no IOKit, permitindo que alguns drivers implementem o seu próprio meio de acesso (Case 6316)</p>
v1.08		F. Englert	Adicionado o status "Ringing..." e o tag "Is Modem

Versão	Data	Autor	Comentários
			<p>Connecting” na interface Modem (Case 4368)</p> <p>Implementado o IP Reserva, Ping e UDP na interface Ethernet (Cases 3014, 3015 e 3017)</p> <p>Colocada proteção para evitar <i>buffer overrun</i> na geração de logs (Case 4365)</p> <p>Implementado o delay entre bytes durante a transmissão na interface Serial e Modem (Case 4343)</p> <p>Portado para Linux e Windows CE (Case 4280)</p> <p>Implementados os serviços de ativação e desativação de log em runtime pela janela do Gerenciador de Drivers do E3Run (Case 4513)</p> <p>Adicionados logs extras para ajudar na depuração de drivers: indicação de interface desconectada no sputs e sgets (Case 4851)</p> <p>A versão do IOKit agora é mostrada no título da página de propriedades dos drivers (“Driver nonono (IOKit v1.08”) (Case 4778)</p> <p>O estado anterior da porta serial agora é restaurado quando a porta é fechada. Isto evita que outros programas que usam a porta serial na configuração padrão do windows deixem de funcionar (Case 4484)</p>
v1.07	2004-05-04	A. Corrêa	Manual traduzido para o Português
v1.07	2004-01-26	F. Englert	A interface serial às vezes demora 1 segundo antes de manipular os bytes que estão chegando (afetando apenas os drivers escravos) (Case 3279)
v1.06	2003-12-18	F. Englert	O IOKit agora verifica se a versão do driver para ter certeza que nela foi utilizada uma versão compatível do IOKit (Case 3018)
v1.05	2003-11-27	F. Englert	Implementado a opção “Listen for connections on port:” na interface Ethernet. Drivers escravos via Ethernet agora são suportados. (Case 3018)
v1.04	2003-10-27	F. Englert	A interface RAS agora corretamente desconecta um socket se ele recebe uma seqüência de string “NO CARRIER” depois que a conexão é estabelecida. (Case 2643)
v1.03	2003-10-09	F. Englert	<p>A página Ethernet não deixa o parâmetro da porta IP ser editada quando a interface RAS está selecionada. (Case 2675)</p> <p>Permite que o string ‘%PROCESS%’ seja inserido no nome do arquivo de log, a fim de inserir o nome do ID do processo no nome do arquivo. Exemplo: “c:\driver_%PROCESS%.log” irá criar um arquivo de log “c:\driver_0000.log”, onde 0000 é</p>

I/O Kit – Manual do Usuário

Versão	Data	Autor	Comentários
			o ID do processo. (Case 2676) As propriedade "IO.Log.Enable" e "IO.Log.FileName" não funcionam e elas deveriam ser aplicadas imediatamente depois de modificadas utilizando o tag Set Parameters. (Case 2678)
v1.02	2003-10-07	F. Englert	A interface RAS agora conecta a uma porta IP simples (a porta especificada na página Ethernet). O hardware RAS irá verificar se existe um modem disponível, designar um modem ao socket ou negar a conexão ao socket. (Case 2656)
v1.01	2003-10-06	F. Englert	A interface RAS agora desconecta o socket caso ele receba uma seqüência de string "NO CARRIER" depois que a conexão é estabelecida. (Case 2643) <i>(esta característica não está funcionando corretamente, por favor utilize a versão 1.04)</i>
v1.00	2003-08-11	F. Englert	Primeira versão. As seguintes opções Ethernet não estão disponíveis ainda: ping, endereço de backup, UDP, usar novo socket and aceitar conexão.