



UFAM

EtherNet/IP™



Protocolo de comunicação Ethernet/IP

Gledyson Cidade

Thiago Rodrigo Monteiro Salgado

Universidade Federal do Amazonas

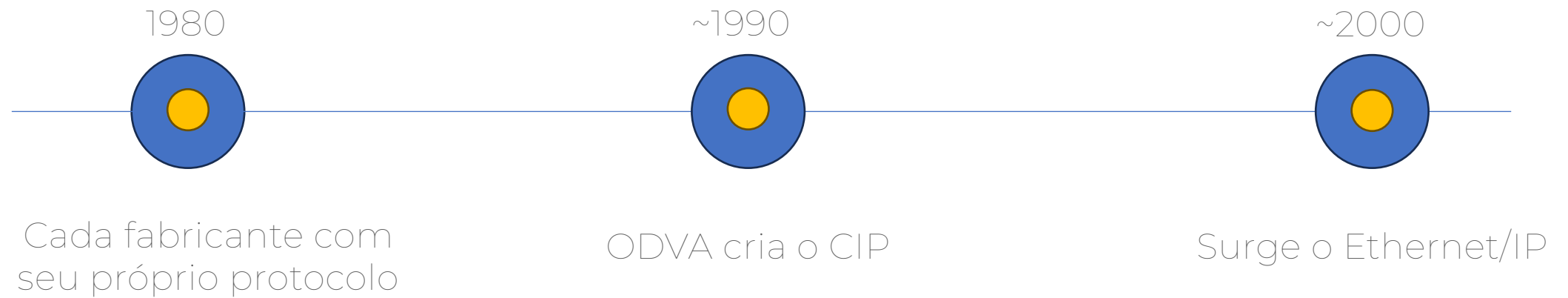
Faculdade de tecnologia – Programa de pós-graduação em Engenharia Elétrica

Setembro de 2025

Conteúdo

1. Definição
2. Comparativos
3. Implementação
4. Referências

Linha cronológica do Ethernet/IP

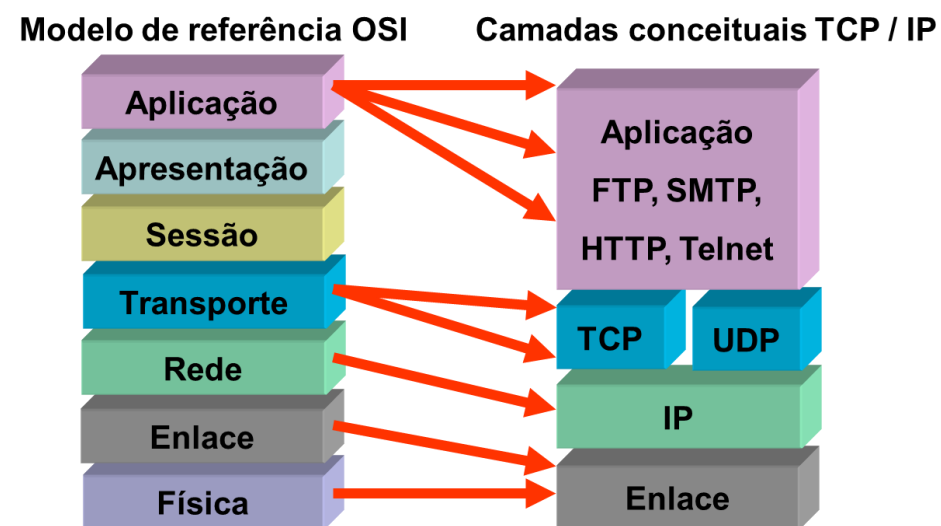


**

ODVA (Open DeviceNet Vendors Association)
CIP (Common Industrial Protocol)

Evolução até o Ethernet/IP

- O CIP é somente para a camada de aplicação, independente da rede física;
- Surgem os protocolos DeviceNet sobre o CAN e ControlNet por fibra, ambos utilizando CIP na aplicação;
- Evolução do Ethernet com o padrão IEEE 802.3 e chegada dos switches full-duplex (evita colisões e envia e recebe ao mesmo tempo).

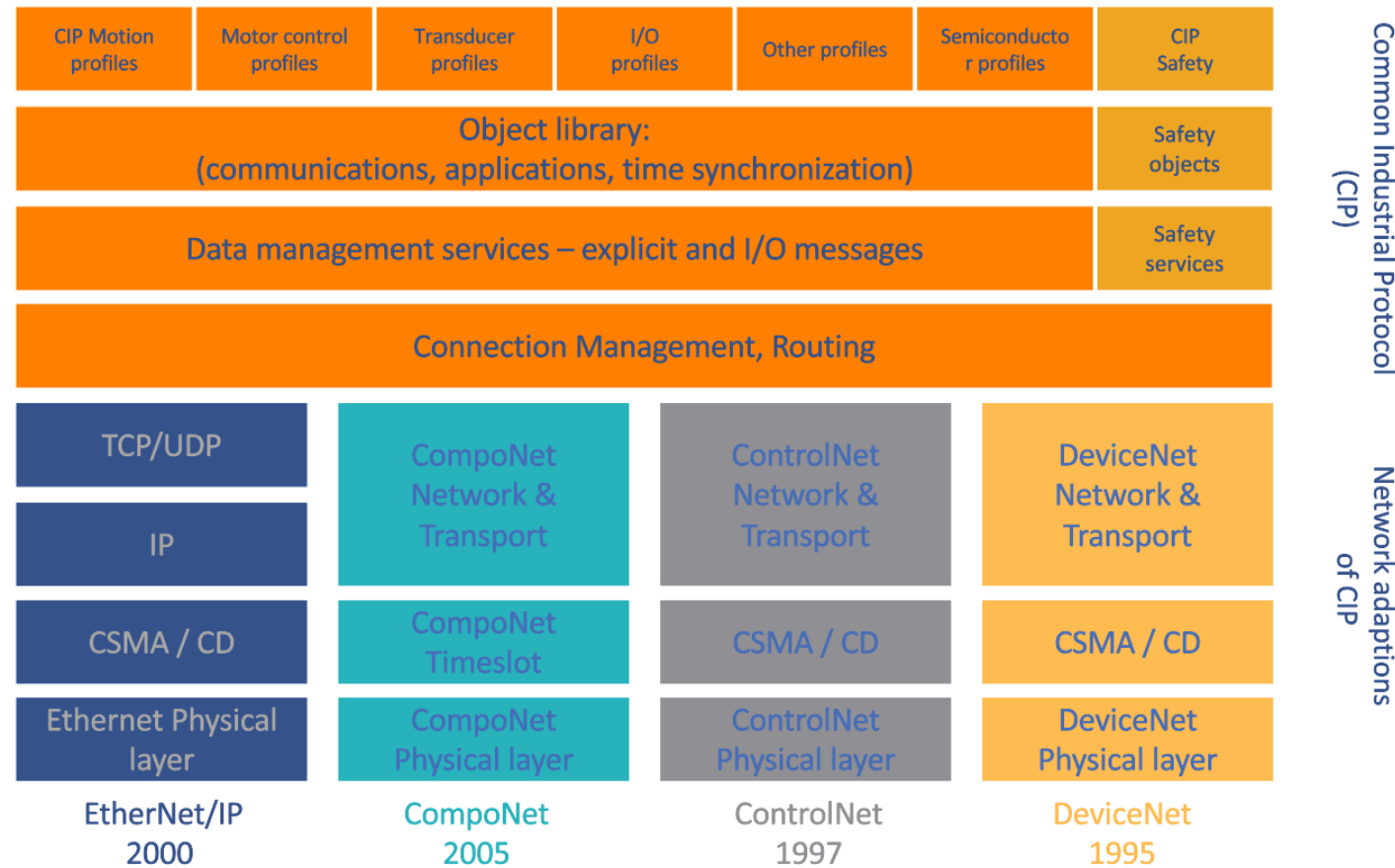


Fonte: [estrategia](http://estrategia.com.br)

Características e vantagens

- Baseado em Ethernet IEEE 802.3;
- Utiliza tanto o TCP / IP quanto o UDP. Um para configurações e outro para transmissão de dados;
- Utiliza o modelo produtor-consumidor;
- Interoperabilidade e escalabilidade (milhares de dispositivos e QoS (Quality of Service));
- Segurança via CIP Security (autenticação, criptografia, certificados);
- Vai do chão de fábrica até o nível de gestão (MES/ERP).

Características e vantagens



Fonte: [netlion](http://netlion.com)

Características da arquitetura

- Camada de aplicação:
 - CIP;
 - Produtor-Consumidor;
- Camada de transporte:
 - TCP/IP e UDP.
- Camada de Enlace:
 - CSMA/CD (detecção de colisão) e Full-Duplex;
- Camadas físicas:
 - Ethernet (IEE 802.3) ou Wi-Fi (802.11).

Tipos de dados

- BOOL - valores lógicos (0 ou 1);
- SINT / INT / DINT / LINT - inteiros (8, 16, 32, 64 bits);
- USINT / UINT / UDINT / ULINT - inteiros sem sinal (8 a 64 bits);
- REAL - ponto flutuante de 32 bits (float);
- LREAL - ponto flutuante de 64 bits (double);
- STRING - textos;
- ARRAYS - vetores de qualquer tipo (ex.: BOOL[8], INT[10]).
- STRUCT - estruturas que agrupam vários tipos (ex.: posição = {x, y, ângulo}).

TAGs

- Uma TAG é o nome lógico atribuído a uma variável ou recurso em um CLP/dispositivo. Dessa forma representando um endereço na memória com significado. Exemplo:
 - acceleration (REAL), motor_status (BOOL),
- Utilizando as TAGs podemos ler ou escrever os valores pelo Ethernet/IP;

Objetos CIP (a parte de atributos, serviços e instâncias)

O CIP é um protocolo 'orientado a objeto'. Então assim como POO ele possui um objeto, instâncias, atributos e serviços.

- Objeto: é como a declaração de uma variável ou um recurso:
 - Atributos: são os dados daquele objeto, como por exemplo velocidade de um motor, fabricante, entre outros;
 - Serviços: são ações, como funções que podemos executar. Exemplo: ligar, desligar, resetar;
 - Instâncias: são como cópias daquele objeto. Podemos ter várias instâncias.

Objetos CIP (a parte de atributos, serviços e instâncias)

Exemplo:

- Objeto: 'Motor'

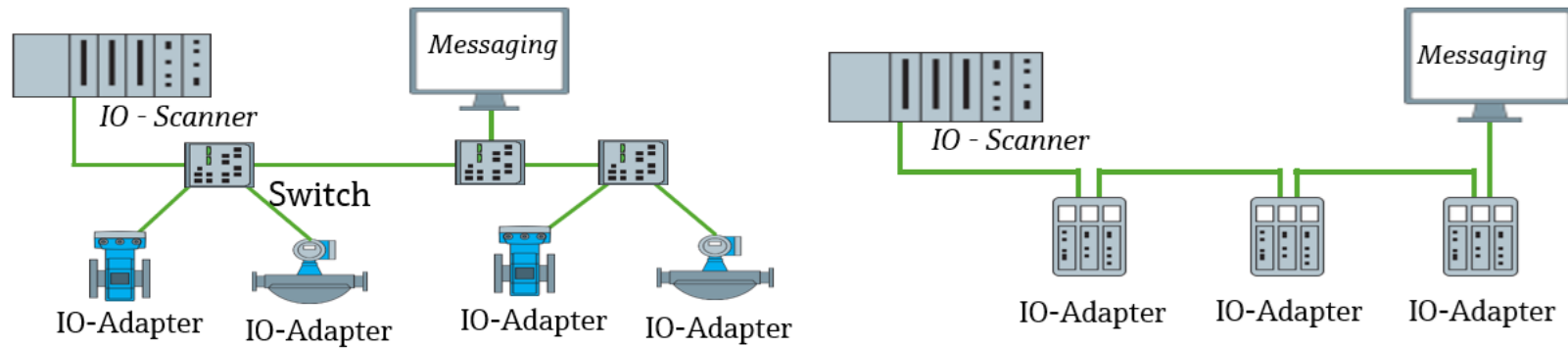
- Instância 1 = Motor A

- Atributo: velocidade = 1500 rpm
 - Atributo: status = ON
 - Serviço: Reset

- Instância 2 = Motor B

- Atributo: velocidade = 1200 rpm
 - Atributo: status = OFF
 - Serviço: Reset

Topologias



Fonte: [netlion](http://netlion.com)

Permite em sua topologia Estrela (esquerda), linear (direita) e até mesmo Anel.

Conteúdo

1. Definição
2. Comparativos
3. Implementação
4. Referências

Objetos CIP (a parte de atributos, serviços e instâncias)

O CIP é um protocolo 'orientado a objeto'. Então assim como POO ele possui um objeto, instâncias, atributos e serviços.

- Objeto: é como a declaração de uma variável ou um recurso:
 - Atributos: são os dados daquele objeto, como por exemplo velocidade de um motor, fabricante, entre outros;
 - Serviços: são ações, como funções que podemos executar. Exemplo: ligar, desligar, resetar;
 - Instâncias: são como cópias daquele objeto. Podemos ter várias instâncias.

Comparativos

Característica	Modbus	PROFINET	OPC UA	EtherCAT	EtherNet/IP
Ano / Origem	1979 / Modicon	2003 / PI (Siemens)	2008 / OPC Foundation	2003 / Beckhoff	2000 / ODVA
Transporte	RS-485 / TCP/IP	Ethernet / TCP/IP / RT	TCP/IP / Pub-Sub	Ethernet direto (on-the-fly)	Ethernet TCP/UDP/IP
Modelo	Mestre-escravo	Produtor-consumidor	Cliente-servidor / Pub-Sub	Mestre-escravo determinístico	Produtor-consumidor
Determinismo	Baixo	Alto (com RT/IRT)	Médio (boa para TI, não hard real-time)	Muito alto (μ s)	Alto (UDP + QoS + TSN)
Velocidade típica	até 115 kbps (RTU) / 100 Mbps (TCP)	100 Mbps / 1 Gbps	+100 Mbps!	100 Mbps / 1 Gbps	100 Mbps / 1 Gbps
Escalabilidade	Alta, mas lenta	Alta	Muito alta (íntegra TI/OT)	Alta (milhares)	Alta (milhares)
Interoperabilidade	Alta (muito difundido)	Alta, mas forte presença Siemens	Muito alta (aberto, multi-plataforma)	Boa, mas concentrado em automação de movimento	Alta (ODVA, multi-vendor, CIP security)
Aplicações típicas	CLPs antigos, sensores, energia	Automação de processos, máquinas Siemens	Integração corporativa, IIoT	Robótica, CNC, motion control	Automação discreta, robôs, CLPs, chão de fábrica
Segurança	Quase inexistente	Segurança via TI/TSN	Forte (criptografia, autenticação)	Baixa nativa	CIP Security (TLS/DTLS)
Vantagens	Simples, barato, universal	Determinismo, suporte Siemens	Interoperabilidade, integração TI/OT	Altíssima velocidade e sincronismo	Aberto, baseado em Ethernet padrão
Limitações	Baixa velocidade, sem segurança	Mais dependente do ecossistema Siemens	Não é determinístico rígido	Mais caro, nichado	UDP depende de QoS/TSN para determinismo

Comparativos

Característica	Modbus	PROFINET	OPC UA	EtherCAT	EtherNet/IP
Ano / Origem	1979 / Modicon	2003 / PI (Siemens)	2008 / OPC Foundation	2003 / Beckhoff	2000 / ODVA
Transporte	RS-485 / TCP/IP	Ethernet / TCP/IP / RT	TCP/IP / Pub-Sub	Ethernet direto (on-the-fly)	Ethernet TCP/IP e UDP
Modelo	Mestre-escravo	Produtor-consumidor	Cliente-servidor / Pub-Sub	Mestre-escravo determinístico	Produtor-consumidor
Determinismo	Baixo	Alto (com RT/IRT)	Médio (boa para TI, não hard real-time)	Muito alto (μ s)	Alto (UDP + QoS + TSN)
Escalabilidade	Alta, mas lenta	Alta	Muito alta (íntegra TI/OT)	Alta (milhares)	Alta (milhares)
Interoperabilidade	Alta (muito difundido)	Alta, mas forte presença Siemens	Muito alta (aberto, multi-plataforma)	Boa, mas concentrado em automação de movimento	Alta (ODVA, multi-vendor, CIP security)
Aplicações típicas	CLPs antigos, sensores, energia	Automação de processos, máquinas Siemens	Integração corporativa, IIoT	Robótica, CNC, motion control	Automação discreta, robôs, CLPs, chão de fábrica
Vantagens	Simples, barato, universal	Determinismo, suporte Siemens	Interoperabilidade, integração TI/OT	Altíssima velocidade e sincronismo	Aberto, baseado em Ethernet padrão
Limitações	Baixa velocidade, sem segurança	Mais dependente do ecossistema Siemens	Não é determinístico rígido	Mais caro, nichado	UDP depende de QoS/TSN para determinismo

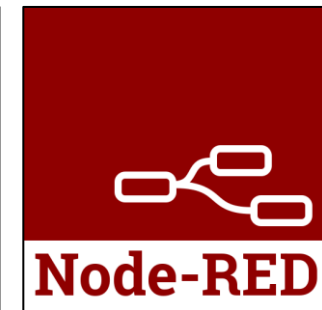
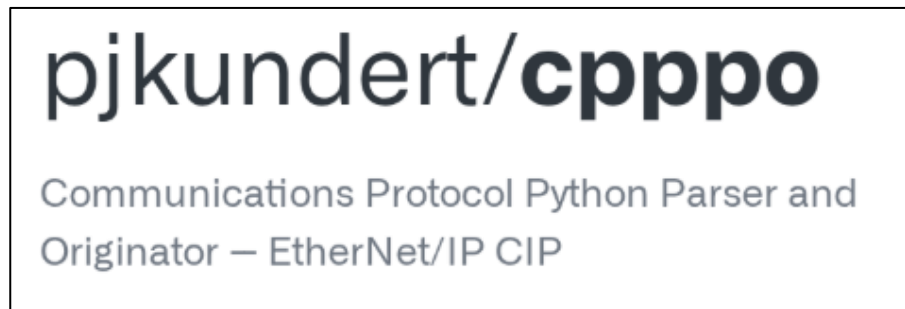
Conteúdo

1. Definição
2. Comparativos
3. Implementação
4. Referências

Arquitetura da aplicação


Para realização desta aplicação, foi construído um simulador - servidor (produtor) Ethernet/IP e, assim como o *client* (consumidor), foi desenvolvido em Python utilizando a biblioteca CPPPO, justamente pela sua capacidade de simular.

Para demonstrar a aplicação foi criada uma simples interface utilizando node-red.




Servidor

enip_server é o comando responsável por inicializar o servidor, seguindo de `-v` para passar as variáveis que você deseja criar dentro do simulador.



```
1 enip_server -v Register1=INT Temperature=REAL Flags=BOOL[8]  
2
```



```
1 tags = {  
2     "acceleration": "REAL",  
3     "speed_factor": "REAL",  
4     "inputs": "BOOL[8]",  
5     "outputs": "BOOL[8]",  
6 }
```

Servidor

enip_server é o comando responsável por inicializar o servidor, seguindo de `-v` para passar as variáveis que você deseja criar dentro do simulador.

```
/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/mnstsalg/Documents/2 - Pessoal/Estudo Pessoal/mestrado/protocolos de comunicacao/codes/communication-protocols
-and-industrial-applications/sim_ethernet_ip_server.py"
- Iniciando: C:\Users\mnstsalg\AppData\Local\Programs\Python\Python39\Scripts\enip_server.EXE -v acceleration=REAL speed_factor=REAL inputs=BOOL[8] outputs=BOOL[8]
09-21 20:21:08.645 MainThread enip.srv NORMAL main Loaded config files: []
09-21 20:21:08.647 MainThread enip.srv NORMAL main New Tag: acceleration REAL[ 1]
09-21 20:21:08.647 MainThread enip.srv NORMAL main New Tag: speed_factor REAL[ 1]
09-21 20:21:08.647 MainThread enip.srv NORMAL main New Tag: inputs BOOL[ 8]
09-21 20:21:08.647 MainThread enip.srv NORMAL main New Tag: outputs BOOL[ 8]
09-21 20:21:08.647 MainThread enip.srv NORMAL main EtherNet/IP Simulator: ('', 44818)
09-21 20:21:08.648 MainThread network NORMAL server_mai enip_srv server PID [30860] starting on ('', 44818)
09-21 20:21:08.648 MainThread network NORMAL server_mai enip_srv server PID [30860] responding to external done/disable signal via <class 'cpppo.dotdict.apidict'> {'done': False, 'disable': False, 'latency': 0.1}
09-21 20:21:08.651 Thread-1 enip.srv NORMAL enip_srv EtherNet/IP Server enip_UDP begins serving peer None
```


Cliente

O cliente utiliza a mesma biblioteca: CPPPO, que permite a conexão utilizando CIP com o servidor.

```
1 class EthernetIPProxy:
2
3     def __init__(self, host: str = "127.0.0.1", port: int = 44818, timeout: float = 5.0):
4         self.host = host
5         self.port = port
6         self.timeout = timeout
7         self.connection: Optional[EthernetIPProxy._ClientProtocol] = None
8
9     def __enter__(self):
10         self.connection = proxy_simple(self.host, port=self.port, timeout=self.timeout)
11         self.connection.__enter__()
12         return self
13
```

Cliente

O cliente utiliza a mesma biblioteca: CPPPO, que permite a conexão utilizando CIP com o servidor. Realizando a leitura.



```
1 @app.route("/acceleration", methods=["GET"])
2 def get_acceleration():
3     with EthernetIPProxy() as plc:
4         value = plc.read('acceleration')
5         return jsonify(value)
```

Interface

No node-red, é possível interagir utilizando o client e por meio das consultas ao produtor é possível ver a interface atualizando.

ETHERNET / IP

INPUT OUTPUTS			SPEED FACTOR ACC	WRITE BOOLEANS		SPEED FACTOR AND ACCELERATION	
OUTPUT 0	<input type="checkbox"/>	INPUTS 0	<input type="checkbox"/>	Current speed factor 0.0	<input type="checkbox"/>	OUTPUT 0	speed factor <input type="text"/>
OUTPUT 1	<input type="checkbox"/>	INPUTS 1	<input type="checkbox"/>	Current acceleration 0.0	<input type="checkbox"/>	OUTPUT 1	Acceleration <input type="text"/>
OUTPUT 2	<input type="checkbox"/>	INPUTS 2	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 2	<input type="button" value="MUDAR ACCELERATION"/> <input type="button" value="SPEED FACTOR"/>
OUTPUT 3	<input type="checkbox"/>	INPUTS 3	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 3	
OUTPUT 4	<input type="checkbox"/>	INPUTS 4	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 4	
OUTPUT 5	<input type="checkbox"/>	INPUTS 5	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 5	
OUTPUT 6	<input type="checkbox"/>	INPUTS 6	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 6	
OUTPUT 7	<input type="checkbox"/>	INPUTS 7	<input type="checkbox"/>		<input type="checkbox"/>	OUTPUT 7	

Interface

No node-red, é possível interagir utilizando o client e por meio das consultas ao produtor é possível ver a interface atualizando.

ETHERNET / IP

INPUT OUTPUTS

OUTPUT 0	<div></div>	INPUTS 0	<div></div>
OUTPUT 1	<div></div>	INPUTS 1	<div></div>
OUTPUT 2	<div></div>	INPUTS 2	<div></div>
OUTPUT 3	<div></div>	INPUTS 3	<div></div>
OUTPUT 4	<div></div>	INPUTS 4	<div></div>
OUTPUT 5	<div></div>	INPUTS 5	<div></div>
OUTPUT 6	<div></div>	INPUTS 6	<div></div>
OUTPUT 7	<div></div>	INPUTS 7	<div></div>

SPEED FACTOR ACC

Current speed factor
13.0

Current acceleration
101.0

WRITE BOOLEANS

INPUT 0	OUTPUT 0
INPUT 1	OUTPUT 1
INPUT 2	OUTPUT 2
INPUT 3	OUTPUT 3
INPUT 4	OUTPUT 4
INPUT 5	OUTPUT 5
INPUT 6	OUTPUT 6
INPUT 7	OUTPUT 7

SPEED FACTOR AND ACCELERATION

speed factor

13

Acceleration

101

MUDAR ACCELERATION

SPEED FACTOR

Conteúdo

1. Definição
2. Comparativos
3. Implementação
4. Referências

Referências

1. Kundert, P.J. cpppo. Disponível em: <https://github.com/pjkundert/cpppo>.
2. Netilion. Untitled 4. Disponível em: <https://netilion.endress.com/blog/pt/untitled-4/>.
3. Academia de Redes. Protocolos de Roteamento: TCP-IP v4. Disponível em: <https://academiaderedes.com/conteudos/protocolos-de-rotaemento/tcp-ip-v4/>.
4. Tanenbaum, Andrew S. Redes de Computadores. 4. ed.
5. [vídeo]. Disponível em: <https://www.youtube.com/watch?v=mm-NHrLtRWI>.
6. [vídeo]. Disponível em: <https://www.youtube.com/watch?v=IfeW2MWijG4>.
7. ODVA. Ethernet/IP – Key Technologies. Disponível em: <https://www.odva.org/technology-standards/key-technologies/ethernet-ip/>.
8. [vídeo]. Disponível em: <https://www.youtube.com/watch?v=vNdaVqslDHA>.
9. Envisia. Conceitos básicos do protocolo Ethernet/IP. Disponível em: <https://www.envisia.com.br/2022/08/22/conceitos-basicos-do-protocolo-ethernet-ip/>.
10. Rockwell Automation. Ethernet/IP: White Paper. Documento técnico ENET-WP001.

Anexos

[Link](#) para o repositório da aplicação:

