STRATAON

miniPLC

STRATA-L800

MANUAL API

STRATA-L800 - MANUAL API – v1.2 - fev/2026

STRATA-L800 – miniPLC

Versão 1.2 – fev/2026

Versão de firmware 1.1.0 e superior

Sumário

1. Introdução

Este documento descreve a API de comunicação do **STRATA-L800 miniPLC**, destinada a desenvolvedores que desejam integrar sistemas externos (supervisórios, aplicações web, serviços em nuvem, gateways ou softwares embarcados) com o dispositivo.

A API expõe uma interface HTTP/REST leve, executando diretamente no firmware do **STRATA-L800 miniPLC**, e permite:

- Monitorar o estado do dispositivo, incluindo saúde do sistema, identificação, tempo de operação e uso de recursos;
- Ler o estado das entradas, saídas e variáveis internas, de forma compacta e eficiente;
- Comandar saídas digitais e variáveis lógicas/numéricas remotamente;
- Consultar o mapeamento lógico dos pontos de IO, facilitando a integração com HMIs e sistemas supervisórios;
- Ler e ajustar a data e hora do equipamento, garantindo sincronização com sistemas externos.

O foco desta API é oferecer uma interface estável, previsível e de baixo overhead, adequada para ambientes industriais e de automação, onde simplicidade, robustez e clareza de contrato são mais importantes do que frameworks complexos.

1.1. Escopo

Este documento cobre exclusivamente os seguintes endpoints expostos pelo firmware:

- Endpoints REST (/api/v1/*) para leitura e comando de IO e estado do sistema;

Outros serviços internos do **STRATA-L800 miniPLC** (configurações avançadas, lógica de controle, comunicação de barramento, MQTT, etc.) não fazem parte deste documento.

1.2. Público-alvo

Este material é destinado a:

- Desenvolvedores de software supervisório (SCADA/HMI);
- Desenvolvedores de aplicações web ou backend que consumam dados do **STRATA-L800 miniPLC**;
- Integradores de sistemas industriais;
- Desenvolvedores de firmware ou gateways que necessitem controlar ou monitorar o **STRATA-L800 miniPLC** via rede IP.

Pressupõe-se conhecimento básico de:

- HTTP e REST;
- JSON;
- Conceitos de automação industrial (entradas, saídas, variáveis lógicas).

1.3. Filosofia da API

A API do miniPLC segue alguns princípios fundamentais:

- **Simplicidade**: chamadas diretas, payloads pequenos e formatos previsíveis;
- **Determinismo**: leitura e escrita explícitas, sem efeitos colaterais ocultos;
- **Compatibilidade**: interface estável, pensada para uso contínuo em campo;
- **Baixo consumo de recursos**: adequada a dispositivos embarcados.

Este documento deve ser utilizado como referência oficial de integração para qualquer software que precise se comunicar com o **STRATA-L800 miniPLC** via rede IP.

2. Convenções gerais

Base URL

http://<IP_DO_EQUIPAMENTO>/api/v1

Formato:

- HTTP / JSON
- Stateless
- Versionada

A API permite:

- Monitoramento do equipamento (health/info)
- Leitura de entradas e variáveis (snapshot e item)
- Escrita direta de saídas e variáveis (item via POST)
- Leitura de mapa/labels dos IOs
- Leitura e ajuste de data/hora

3. Autenticação por token

Toda requisição DEVE conter um token.

Header suportado :

- Authorization: Bearer <TOKEN>

Exemplo (WRITE):

Authorization: Bearer fd6ebc246a03b019c7c28a5bead5078989add8385583d98091d6890260c23f56

Tipos de token:

- READ – permite apenas GET
- WRITE – permite GET e POST

Regras:

- GET → READ ou WRITE
- POST → somente WRITE
- Token ausente/inválido → HTTP 401
- Token READ tentando POST → HTTP 403

PADRÃO DE RESPOSTA

Sucesso:

```
{
  "ok": true,
  ...
}
```

Erro:

```
{
  "ok": false,
  "error": "<codigo>",
  "message": "<descricao>"
}
```

4. Map

GET /api/v1/map

Token: READ

Uso:

- Supervisão
- Ler nome das variáveis disponíveis

Descrição

Mapa/nomenclatura padrão (32 posições por grupo).

Exemplo (curl):

```
curl -i \
  http://192.168.17.60/api/v1/map \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Resposta (200)

```
{
  "ok": true,
  "di": ["DI1","DI2","DI3", "..."],
  "do": ["DO1","DO2","DO3", "..."],
  "vb": ["VB1","VB2","VB3", "..."],
  "vl": ["VL1","VL2","VL3", "..."]
}
```

5. Health

GET /api/v1/health

Token: READ

Uso:

- Supervisão / Watchdog
- Monitoramento de disponibilidade

Exemplo (curl):

```
curl -i \  http://192.168.17.60/api/v1/health \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Resposta:

```
{
  "ok": true,
  "status": "ok",
  "uptime_s": 45231,
  "epoch": 1773143400
}
```

6. Info

GET /api/v1/info

Token: READ

Exemplo (curl):

```
curl -i \
  http://192.168.17.60/api/v1/info \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Resposta (exemplo real):


```
{
  "ok": true,
  "board_id": "miniplc002001",
  "serial_id": "002001",
  "version": "1.0.6",
  "language": "pt_BR",
  "media": "wired",
  "mac": "3e:e9:0e:8b:94:40",
  "ip": "192.168.17.60",
  "dns1": "",
  "dns2": "",
  "gateway": "192.168.17.1",
  "mask": "255.255.255.0",
  "runtime": { "uptime_s": 1492, "epoch": 1770742148 }
}
```


Descrição

Identificação do dispositivo e informações básicas de rede.

7. Data e hora

7.1. Ler data/hora

GET /api/v1/get_datetime
Token: READ

Exemplo (curl):
```
curl -i \
  http://192.168.17.60/api/v1/get_datetime \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Resposta:
```
{
  "ok": true,
  "datetime": "11/02/2026 12:00:13",
  "epoch": 1770811213
}
```

7.2. Ajustar data/hora

POST /api/v1/set_datetime
Token: WRITE

Body JSON:
```
{
  "year": 2026,
  "mon": 2,
  "day": 11,
  "hour": 12,
  "min": 0,
  "sec": 0
}
```

Exemplo (curl):
```
curl -i -X POST \
  http://192.168.17.60/api/v1/set_datetime \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -H "Content-Type: application/json" \
  -d '{"year":2026,"mon":2,"day":11,"hour":12,"min":0,"sec":0}'
```

Validações:

- Ano: 1970–9999
- Mês: 1–12
- Dia compatível com mês/ano bissexto
- Hora: 0–23
- Min/Sec: 0–59

8. IO – Snapshot (Todos os canais)

Tipos suportados:

- di → Entrada digital (read-only)
- do → Saída digital
- vb → Booleano virtual
- vl → Inteiro virtual

GET /api/v1/io
Token: READ

Retorna um snapshot completo (arrays):

```
{
  "ok": true,
  "di": [0,1,0,...],
  "do": [1,0,1,...],
  "vb": [false,true,...],
  "vl": [10,250,...]
}
```

Exemplo (curl):

```
curl -i \
  http://192.168.17.60/api/v1/io \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Observação:
O snapshot (/api/v1/io) não garante sincronização com mudanças subsequentes.
Para leitura determinística de um único ponto, utilize /api/v1/io/<kind>/<idx>.

9. IO – Item (Leitura)

GET /api/v1/io/<kind>/<idx>
Token: READ

Onde:

- <kind> = di | do | vb | vl
- <idx> = índice do canal (ex.: 1..32)

Exemplos (curl):

Ler entrada DI1:

```
curl -i \
  http://192.168.17.60/api/v1/io/di/1 \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Ler saída DO1:

```
curl -i \
  http://192.168.17.60/api/v1/io/do/1 \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Ler booleano virtual VB5:

```
curl -i \
  http://192.168.17.60/api/v1/io/vb/5 \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Ler inteiro virtual VL2:

```
curl -i \
  http://192.168.17.60/api/v1/io/vl/2 \
  -H "Authorization: Bearer <READ_TOKEN>"
```

Resposta (exemplo):

```
{
  "ok": true,
  "kind": "do",
  "idx": 1,
  "value": 0
}
```

10. IO – Item (Escrita) + Pulso (Timer)

POST /api/v1/io/<kind>/<idx>

Token: WRITE

Onde:
- <kind> = do | vb | vl
- <idx> = índice do canal (ex.: 1..32)

Body (JSON):

```
{
  "value": <valor>,
  "pulse_ms": <tempo_ms_opcional>
}
```

Tipos e regras de value

- do: value = 0/1 ou false/true
- vb: value = 0/1 ou false/true
- vl: value = inteiro 32 bits (faixa conforme firmware)

Campo opcional pulse_ms (somente para do)

- pulse_ms em milissegundos
- Se ausente ou 0 → escrita direta (sem temporização)
- Se pulse_ms > 0:
  - a saída vai para o estado indicado em value imediatamente
  - permanece assim por pulse_ms
  - depois retorna ao estado anterior (estado antes do comando)
- Um novo comando na mesma saída substitui o pulso anterior (se existir)

Observação: pulse_ms deve ser usado apenas com kind=do. Para vb/vl, ignore/não envie pulse_ms.

Exemplos (curl) – Escrita direta (sem pulso)

10.1. Ligar DO1:

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/do/1 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":true}'
```

10.2. Desligar DO1:

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/do/1 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":false}'
```

10.3. Setar VB5 = true:

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/vb/5 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":true}'
```

10.4. Setar VL2 = 123:

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/vl/2 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":123}'
```

Exemplos (curl) – Pulso temporizado (somente DO)

10.5. Pulso para "ligar relé" por 800 ms (liga e volta):

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/do/1 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":true,"pulse_ms":800}'
```

10.6. Pulso inverso (desliga saída por 5000 ms e volta):

Útil se a saída estiver ligada e você quer "cortar" por um tempo.

```
curl -i -X POST \
  http://192.168.17.60/api/v1/io/do/1 \
  -H "Authorization: Bearer <WRITE_TOKEN>" \
  -d '{"value":false,"pulse_ms":5000}'
```

Resposta (exemplo)

```
{
  "ok": true,
  "kind": "do",
  "idx": 1,
  "value": 1
}
```