STRATAON

miniPLC

STRATA-L800

MANUAL API

STRATA-L800 - MANUAL API – v1.3 - jul/2026

STRATA-L800 – miniPLC

Versão 1.3 – jul/2026

Versão de firmware 1.2.9 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 miniPLC;
- Integradores de sistemas industriais;
- Desenvolvedores de firmware ou gateways que necessitem controlar ou monitorar o 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. Base URL

http://<ip-do-miniplc>/api/v1

Nos exemplos apresentados neste manual usamos o endereço IP 192.168.17.60. Para testes use o endereço IP real de miniPLC.

3. Autenticação

A autenticação utiliza Bearer Token no cabeçalho HTTP Authorization. Existem dois níveis de permissão:

- READ: permite operações GET.
- WRITE: permite operações GET e POST.

Quando a API estiver desabilitada, o equipamento retorna HTTP 403.

Fluxo de autenticação

- Cliente envia Authorization: Bearer <TOKEN>.
- Firmware verifica se a API está habilitada.
- Token é comparado ao token READ e WRITE configurados.
- Permissão é validada conforme o método solicitado.
- Em caso de sucesso, o endpoint é executado.
- Em caso de falha retorna HTTP 401 ou 403.

4. CORS

O miniPLC responde automaticamente às requisições OPTIONS (preflight). Os principais cabeçalhos enviados são:

- Access-Control-Allow-Origin: *
- Access-Control-Allow-Methods: GET, POST, OPTIONS
- Access-Control-Allow-Headers: Authorization, Content-Type

5. Endpoints implementados

Método	URI	Permissão	Descrição
GET	/health	READ	Estado geral
GET	/info	READ	Informações
POST	/reboot	WRITE	Reiniciar equipamento
GET	/map	READ	Mapa das variáveis
GET	/io	READ	Estado das IOs
POST	/io/{tipo}/{id}	WRITE	Escrita
GET	/get_datetime	READ	Ler relógio
POST	/set_datetime	WRITE	Ajustar relógio

6. Estrutura padrão das respostas

Resposta de sucesso:

```
{
  "ok": true
}
```

Resposta de erro:

```
{
  "ok": false,
  "error": "invalid_token"
}
```

7. Sistema

7.1. Health

GET /api/v1/health

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/health
Objetivo	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
}
```

7.2. Info

GET /api/v1/info

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/info
Objetivo	Obter identificação do dispositivo e informações básicas de rede.

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 }

}

7.3. Reboot

POST /api/v1/reboot

O endpoint de reboot permite reiniciar o equipamento de forma controlada. É recomendado após alterações críticas de configuração.

Item	Descrição
Método	POST
Permissão	WRITE
URI	/api/v1/reboot
Objetivo	Reiniciar o equipamento
Resposta	JSON com confirmação

Resposta típica:

```
{
  "ok": true,
  "message":"Reboot agendado.",
  "delay_ms":500
}
```

8. IO

8.1. Map

GET /api/v1/map

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/map
Objetivo	Ler nome das variáveis disponíveis

O endpoint /api/v1/map fornece o mapa lógico das variáveis exportadas pela API.

Campo	Descrição	Observação
di	Entradas digitais	1..32
do	Saídas digitais	1..32
bool	Variáveis booleanas	1..32
long	Variáveis inteiras	1..32

Exemplo (curl):

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

Resposta (exemplo real):

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

8.2. IO – Snapshot

GET /api/v1/io

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/io
Objetivo	Obter estado atual de todas as variáveis do miniplc

Tipos de variáveis:

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

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

Resposta (Retorna um snapshot completo (arrays)):

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

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

8.3. IO – Item (Leitura)

GET /api/v1/io/<tipo>/<id>

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/io/<tipo>/<id>
Objetivo	Obter estado atual de uma variável do miniplc

Tipos de variáveis:

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

Id de variáveis:

- <id> = índice da variáveis (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 DO3:
curl -i http://192.168.17.60/api/v1/io/do/3 -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>"

Resposta (exemplo):

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

8.4. IO – Item (Escrita)

POST /api/v1/io/<tipo>/<id>

Item	Descrição
Método	POST
Permissão	WRITE
URI	/api/v1/io/<tipo>/<id>
Objetivo	Alterar estado/valor de uma variável do miniplc

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)

Id de variáveis:

- <id> = índice da variáveis (ex.: 1..32)

Formato do Body (JSON):

- Geral:

```
{
  "value": <valor>
}
```

- Opcional apenas para DO

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

O campo pulse_ms é opcional e usado apenas 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 durante o tempo determinado por pulse_ms
    - depois retorna ao estado anterior (estado antes do comando)
- Um novo comando na mesma saída substitui o pulso anterior (se existir)

Exemplos (curl) – Escrita direta (sem pulso)

Ligar DO1:

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

Desligar DO1:

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

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}'
```

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) – Escrita direta (com pulso)

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}'
```

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}'
```

9. Datetime

9.1. Ler data/hora

GET /api/v1/get_datetime

Item	Descrição
Método	GET
Permissão	READ
URI	/api/v1/get_datetime
Objetivo	Obter data e hora do miniplc

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
}
```

9.2. Ajustar data/hora

POST /api/v1/set_datetime

Item	Descrição
Método	POST
Permissão	WRITE
URI	/api/v1/set_datetime
Objetivo	Atualizar data e hora do miniplc

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

Resposta (exemplo)

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