STRATAON

smartPDU

STRATA-P810

MANUAL API

STRATA-P810 – smartPDU

Versão 1.1 – jul/2026

Versão de firmware 2.3.2 e superior

Sumário

1. Introdução

Este documento descreve a API de comunicação do **STRATA-P810 smartPDU**, 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-P810 smartPDU**, e permite:

- Monitorar o estado do dispositivo, incluindo saúde do sistema, identificação, tempo de operação e uso de recursos;
- Ler o estado das tomadas de forma compacta e eficiente;
- Comandar as tomadas remotamente;
- Ler e ajustar a data e hora do equipamento, garantindo sincronização com sistemas externos;
- Ler e ajustar os recursos de agendas e watchdogs do dispositivo;
- Comandar Reboot remotamente

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, comandos, configuração e estado do sistema;

Outros serviços internos do **STRATA-P810 smartPDU** (configurações de rede, 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 equipamento;
- Integradores de sistemas industriais;
- Desenvolvedores de firmware ou gateways que necessitem controlar ou monitorar o equipamento 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 **STRATA-P810 smartPDU** 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 equipamento via rede IP.

2. Visão geral

Todos os endpoints da API REST v1 usam o prefixo /api/v1. Exemplo:

```
http://192.168.0.5/api/v1/outputs
```

O endereço IP **192.168.0.5** é usado apenas como exemplo neste manual. Altere para o endereço em uso.

Os principais recursos disponíveis são: status do dispositivo, controle de saídas, data/hora, watchdogs, agendas e reboot remoto.

3. Autenticação Bearer

Quando a API estiver habilitada, os endpoints GET exigem token READ ou WRITE. Os endpoints POST exigem token WRITE. Os tokens de READ e WRITE são gerados pelo próprio equipamento através da Web UI de configuração.

```
Authorization: Bearer SEU_TOKEN
```

Token	Permite
READ	Leitura via GET
WRITE	Leitura via GET e alteração via POST

HTTP	Significado
401	Token ausente ou inválido
403	Token sem permissão de escrita ou API desabilitada

4. CORS e Swagger

Para uso via navegador ou Swagger UI, o firmware deve responder OPTIONS nos endpoints da API. O método OPTIONS não deve exigir Bearer, pois é usado pelo navegador como preflight CORS.

5. Formato de erros

Erros gerais da API podem seguir este formato:

```
{
  "ok": false,
  "code": "bad_request",
  "message": "Requisição inválida."
}
```

A camada de autenticação pode retornar:

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

Erro	Significado
missing_bearer_token	Header Authorization ausente
invalid_token	Token inválido
write_token_required	Token READ usado em operação WRITE
api_disabled	API desabilitada na configuração
bad_request	JSON inválido, campo ausente ou valor inválido

6. Endpoints

Método	Endpoint	Permissão	Descrição
GET	/api/v1/health	READ	Estado geral do dispositivo
GET	/api/v1/info	READ	Informações do dispositivo
POST	/api/v1/reboot	WRITE	Reinicia o dispositivo
GET	/api/v1/outputs	READ	Lista todas as saídas
GET	/api/v1/output/{id}	READ	Lê uma saída específica
POST	/api/v1/output/{id}	WRITE	Ajusta uma saída específica
GET	/api/v1/datetime	READ	Lê data/hora atual
POST	/api/v1/datetime	WRITE	Ajusta data/hora
GET	/api/v1/watchdogs	READ	Lista watchdogs
GET	/api/v1/watchdog/{id}	READ	Lê watchdog específico
POST	/api/v1/watchdog/{id}	WRITE	Atualiza watchdog específico
GET	/api/v1/schedules	READ	Lista agendas
GET	/api/v1/schedule/{id}	READ	Lê agenda específica
POST	/api/v1/schedule/{id}	WRITE	Atualiza agenda específica

Os IDs usados em output, watchdog e schedule vão de 1 a 8.

7. System

7.1. GET /api/v1/health

Retorna o estado geral do dispositivo.

```
curl -X GET "http://192.168.0.5/api/v1/health" -H "Authorization: Bearer SEU_TOKEN"
```

```
{
  "ok": true,
  "status": "ok",
  "uptime_s": 12345
}
```

7.2. GET /api/v1/info

```
curl -X GET "http://192.168.0.5/api/v1/info" -H "Authorization: Bearer SEU_TOKEN"
```

```
{
  "ok": true,
  "model": "PB0810",
  "firmware": "1.0.0"
}
```

7.3. POST /api/v1/reboot

Agenda o reboot do dispositivo após enviar a resposta HTTP. Requer token WRITE.

```
curl -X POST "http://192.168.0.5/api/v1/reboot" -H "Authorization: Bearer SEU_TOKEN_WRITE"
```

```
{
  "ok": true,
  "message": "Reboot scheduled"
}
```

8. Outputs

8.1. GET /api/v1/outputs

Lista todas as saídas.

```
curl -X GET "http://192.168.0.5/api/v1/outputs" -H "Authorization: Bearer SEU_TOKEN"
```

```
{
  "ok":                                                                                                    true,
  "outputs":                                                                                                  [
    {"id":1,"tag":"plug01","name":"Tomada                                                    1","active":1,"state":1}
  ]
}
```

8.2. GET /api/v1/output/{id}

```
curl -X GET "http://192.168.0.5/api/v1/output/1" -H "Authorization: Bearer SEU_TOKEN"
```

8.3. POST /api/v1/output/{id}

Ajusta o estado de uma saída. Requer token WRITE.

```
{
  "state":                                                                                                   1
}
```

```
curl -X POST "http://192.168.0.5/api/v1/output/1" -H "Content-Type: application/json" -H "Authorization: Bearer SEU_TOKEN_WRITE" -d '{"state":1}'
```

9. DateTime

9.1. GET /api/v1/datetime

```
curl -X GET "http://192.168.0.5/api/v1/datetime" -H "Authorization: Bearer SEU_TOKEN"

{
  "ok": true,
  "datetime": "04/07/2026 15:30:00",
  "epoch": 1783193400
}
```

9.2. POST /api/v1/datetime

Ajusta a data/hora. Requer token WRITE.

```
{
  "year": 2026,
  "mon": 7,
  "day": 4,
  "hour": 15,
  "min": 30,
  "sec": 0
}

curl -X POST "http://192.168.0.5/api/v1/datetime" -H "Content-Type: application/json" -H "Authorization: Bearer SEU_TOKEN_WRITE" -d '{"year":2026,"mon":7,"day":4,"hour":15,"min":30,"sec":0}'
```

10. Watchdogs

O **STRATA-P810 smartPDU** possui 8 watchdogs, identificados por ID 1..8. Internamente os tags vão de watchdog01 a watchdog08.

10.1. GET /api/v1/watchdogs

```
curl -X GET "http://192.168.0.5/api/v1/watchdogs" -H "Authorization: Bearer SEU_TOKEN"
```

```
{
  "ok": true,
  "count": 8,
  "watchdogs": [
    {"id":1,"tag":"watchdog01"},
    {"id":2,"tag":"watchdog02"}
  ]
}
```

10.2. GET /api/v1/watchdog/{id}

```
curl -X GET "http://192.168.0.5/api/v1/watchdog/1" -H "Authorization: Bearer SEU_TOKEN"
```

10.3. POST /api/v1/watchdog/{id}

Atualiza a configuração de um watchdog. Requer token WRITE.

```
{
  "host": "192.168.1.20",
  "plug": "plug01",
  "active": 1,
  "ping_interval": 5,
  "ping_timeout": 50,
  "ping_fail_limit": 3,
  "time_plug_off": 10,
  "time_monitor_off": 10,
  "retry_limit": 0,
  "retry_long_time": 60
}
```

Campo	Unidade	Descrição
host	-	IP ou host monitorado
plug	-	Tomada associada, exemplo plug01

active	0/1	Habilita ou desabilita o watchdog
ping_interval	segundos	Intervalo entre pings
ping_timeout	milissegundos	Timeout do ping
ping_fail_limit	tentativas	Falhas consecutivas para acionar
time_plug_off	segundos	Tempo mantendo a tomada desligada
time_monitor_off	segundos	Espera antes de voltar a monitorar
retry_limit	ciclos	0 significa ilimitado
retry_long_time	minutos	Intervalo entre ciclos de rearme

11. Schedules

O **STRATA_P810 smartPDU** possui 8 agendas, identificadas por ID 1..8. Internamente os tags vão de schedule01 a schedule08.

11.1. GET /api/v1/schedules

```
curl -X GET "http://192.168.0.5/api/v1/schedules" -H "Authorization: Bearer SEU_TOKEN"
```

```
{
  "ok": true,
  "count": 8,
  "schedules": [
    {"id":1,"tag":"schedule01"},
    {"id":2,"tag":"schedule02"}
  ]
}
```

11.2. GET /api/v1/schedule/{id}

```
curl -X GET "http://192.168.0.5/api/v1/schedule/1" -H "Authorization: Bearer SEU_TOKEN"
```

11.3. POST /api/v1/schedule/{id}

Atualiza a configuração de uma agenda. Requer token WRITE.

```
{
  "plug": "plug01",
  "active": 1,
  "sun_on": "00:00",
  "sun_off": "00:00",
  "mon_on": "08:00",
  "mon_off": "18:00",
  "tue_on": "08:00",
  "tue_off": "18:00",
  "wed_on": "08:00",
  "wed_off": "18:00",
  "thu_on": "08:00",
  "thu_off": "18:00",
  "fri_on": "08:00",
  "fri_off": "18:00",
  "sat_on": "00:00",
  "sat_off": "00:00"
}
```

Campos de horário usam o formato HH:MM, de 00:00 a 23:59.

Prefixo	**Dia**
sun	Domingo
mon	Segunda-feira
tue	Terça-feira
wed	Quarta-feira
thu	Quinta-feira
fri	Sexta-feira
sat	Sábado

12. Recomendações de integração

- Use sempre Content-Type: application/json nos métodos POST.
- Use token WRITE somente nos sistemas que realmente precisam alterar estado/configuração.
- Para integrações somente de monitoramento, use token READ.
- Evite chamar POST /api/v1/reboot automaticamente sem confirmação do usuário.
- Trate HTTP 401 e 403 separadamente.
- Para Swagger UI em navegador, garanta que os endpoints OPTIONS estejam registrados no firmware.

13. Checklist rápido para testes

```
# Health
curl -X GET "http://192.168.0.5/api/v1/health" -H "Authorization: Bearer SEU_TOKEN"

# Outputs
curl -X GET "http://192.168.0.5/api/v1/outputs" -H "Authorization: Bearer SEU_TOKEN"

# Set output 1 ON
curl -X POST "http://192.168.0.5/api/v1/output/1" -H "Content-Type: application/json" -H "Authorization: Bearer
SEU_TOKEN_WRITE" -d '{"state":1}'

# DateTime
curl -X GET "http://192.168.0.5/api/v1/datetime" -H "Authorization: Bearer SEU_TOKEN"

# Watchdogs
curl -X GET "http://192.168.0.5/api/v1/watchdogs" -H "Authorization: Bearer SEU_TOKEN"

# Schedules
curl -X GET "http://192.168.0.5/api/v1/schedules" -H "Authorization: Bearer SEU_TOKEN"
```