

Protocolos

Manual do usuário

Revisão 232 (313241b) · 05/11/2026

docs.smarthomecontrol.com.br/protocolos/

Sumário

Native Device: introdução 3
Native Device: comunicação UDP e TCP 4
Native Device: formato da mensagem 5
Ações 10
Ações do Native Device 10
/ac/rs: saídas de relé 12
/ac/bl: persianas (firmware 3.x) 13
/ac/pers: persianas 15
LOUD: controle das zonas 16
/ac/xcN: intensidade por cena 20
/ac/xClick: clique remoto 21
/ac/xHold: segurar remoto 22
/aa/tcp: enviar mensagem TCP 23
/aa/udp: enviar mensagem UDP 25
/aa/wol: Wake-on-LAN 27
/aa/delay: esperar entre ações da cena 29
/if/device: informações do equipamento 30
/ac/ping: equipamento respondendo 31
/ac/status: estado da rede e do equipamento 32

Native Device: introdução

O **Native Device** é o protocolo de comunicação dos equipamentos. Cada equipamento segue o mesmo formato de mensagem e informa quais ações oferece.

Características

Característica	Descrição
Protocolo aberto	O formato das mensagens está documentado nesta referência.
Texto ASCII	As mensagens são texto legível.
Delimitadores	Os caracteres `$`, `>`, `@`, `?`, `<` e `#` marcam o início e o fim de cada campo.
Número de sequência	Cada mensagem leva um número que permite descartar cópias da mesma mensagem (veja número de sequência).
Vários canais	A mesma mensagem pode trafegar por UDP multicast (`239.5.0.2` , porta `5202`), TCP (porta `5203`) e XNET (veja canais).
Ações descritas pelo equipamento	Cada equipamento informa as próprias ações e parâmetros (veja descrição).

Veja: Protocolos (canais de comunicação), Query Strings (formato da mensagem) e Ações.

Native Device: comunicação UDP e TCP

Os equipamentos **Native Device** aceitam mensagens por **UDP multicast**, por **TCP** e, quando cabeados, pela **XNET**.

Canal	Endereço e porta	Uso
UDP multicast	`239.5.0.2`, porta `5202`	Uma mensagem chega a todos os equipamentos da rede. Use para descobrir equipamentos e para comandos a vários equipamentos.
TCP	IP do equipamento, porta `5203`	Conexão direta com um equipamento, com confirmação de entrega.
XNET	rede cabeada	Equipamentos ligados por cabo. As mensagens em bytes da XNET estão em Control Xcene: interface XNET.

Enviar por UDP multicast

Envie a mensagem para `239.5.0.2:5202`. Todos os equipamentos da rede recebem a mesma mensagem, sem que você precise saber o IP de cada um. Só o equipamento com o ID da mensagem executa; com o ID `0000`, todos executam (veja o formato da mensagem).

Enviar por TCP

Abra uma conexão com o IP do equipamento na porta `5203` e envie a mensagem. O equipamento responde nessa mesma conexão.

Para onde vai a resposta

Para receber as respostas, escute a conexão TCP que você abriu e o UDP multicast.

Com o parâmetro `_to`, o equipamento envia a resposta para esse IP e também pelo UDP multicast. Para saber a que pedido cada resposta corresponde e descartar a repetida, veja o número de sequência.

Native Device: formato da mensagem

Cada mensagem Native Device é uma linha de texto com sete partes, marcadas pelos separadores `$`, `>`, `@`, `?`, `<` e `#`.

Formato

```
$6A6F>/ac/rs@0?r0=off&r1=off&_to=192.168.0.10<99999#
```

Parte	Descrição
`$6A6F`	ID do equipamento que recebe a mensagem: os 4 últimos caracteres do endereço MAC, em letras maiúsculas (está na etiqueta). O ID `0000` endereça todos os equipamentos da rede ao mesmo tempo, e cada um responde com o próprio ID.
`>/ac/rs`	Ação (caminho) a executar ou consultar. Veja a lista de ações.
`@0`	Número do protocolo: o que fazer com a ação. Veja a tabela de protocolos.
`?r0=off&r1=off`	Parâmetros, no formato `nome=valor`, separados por `&`. Sem parâmetros, a consulta devolve todos os valores da ação.
`&_to=192.168.0.10`	Opcional: IP de quem envia a mensagem, para a resposta. Coloque-o por último: `?r0=off&_to=192.168.0.10`, ou `?_to=192.168.0.10` quando não há outros. Veja os parâmetros de controle.
`<99999`	Número de sequência. Veja número de sequência.
`#`	Fim da mensagem.

Os separadores são obrigatórios, mesmo com o campo vazio: uma mensagem sem parâmetros ainda leva o `?`.

Parâmetros de controle

Parâmetros que começam com `_` não são dados da ação: eles controlam a mensagem. Vão na parte dos valores, junto dos outros parâmetros.

Parâmetro	Para que serve
`_to`	IP de quem envia a mensagem. O equipamento manda a resposta para esse IP. Veja para onde vai a resposta.
`_n`	Índice do item, nas ações que guardam vários itens: a zona em `/lb/Zona` do LOUD, o botão em `/lb/buttons`, o item de cena em `/cs/btN`. Nas cenas de botão, as ações de clique usam `_n` = `0`, `1`, `2` …, e as de segurar usam `-1`, `-2`, `-3` …
`_ts`	Data e hora da última alteração, em segundos desde 1º de janeiro de 1970 (UTC). Vai junto quando você grava uma configuração. O equipamento guarda o valor e o devolve quando você consulta a configuração.
`_su`	Nas ações `/aa/*`: identificador livre, de até 8 caracteres. O equipamento guarda e devolve o valor. Ele não muda o envio.

A descrição também lista parâmetros com `_`. Eles seguem as mesmas regras, e os aplicativos não os mostram como saídas.

Número de sequência

O número de sequência vem entre `<` e `#`. Use um número diferente em cada mensagem. O equipamento ignora uma mensagem cuja sequência ele já recebeu há pouco: assim, a mesma mensagem que chega por UDP e por TCP só é executada uma vez.

O valor `0` desliga essa verificação: o equipamento executa todas as cópias. Não use `0` em comandos.

A resposta traz um número de sequência próprio, diferente do pedido. Para saber a que pedido uma resposta corresponde, compare o ID, a ação, o protocolo e o `_n`.

Protocolos

Protocolo	Nome	Descrição
`@0`	Set Data	Consulta valores (sem parâmetros) ou grava valores (com parâmetros).
`@1`	Delete Data	Apaga itens de uma ação, como os itens de cena de um botão. Veja Delete Data.
`@6`	Get All	Lê todos os itens de uma ação de uma vez. Veja Get All.
`@8`	Send All	Envia a mesma mensagem a vários equipamentos ao mesmo tempo. Veja Send All.
`@9`	Description	Pede ao equipamento a descrição das ações e dos parâmetros que ele oferece. Veja Descrição.

@1 Delete Data

Para apagar itens de cena de um botão, envie `@1` com o `_n` do primeiro item a apagar. Exemplo: apagar do terceiro item de clique em diante, no botão 0.

```
$72B0>/cs/bt0@1?_n=2&_ts=1760900000<48213#
```

- Com `_n` igual a `0` ou maior, o equipamento apaga as ações de clique desse índice em diante.
- Com `_n` negativo, apaga as ações de segurar desse índice em diante (`-2` apaga `-2`, `-3` …).

A resposta traz `status=OK` quando deu certo:

```
$72B0>/cs/bt0@1?code=0000&status=OK&_n=2<90311#
```

@6 Get All

Para ler todos os itens de uma ação que guarda vários itens, envie `@6` sem parâmetros. Exemplo: ler a configuração de todos os botões.

```
$72B0>/lb/buttons@6?<48214#
```

O equipamento responde com uma mensagem por item. Cada uma vem com o protocolo `@0`, o `code` e o `_n` do item. O campo `len` da descrição informa quantos itens a ação tem.

@8 Send All

O Studio SHC usa `@8` para enviar a mesma rede Wi-Fi a vários equipamentos (opção **Enviar para todos (rede cabeada)**):

```
$72B0>/lb/wifi@8?ssid=MinhaRede&pass=12345678<48215#
```

Resposta

A resposta tem o mesmo formato da mensagem. Ela vem com o ID do equipamento que respondeu, a ação, o protocolo e os valores atuais.

O primeiro valor é o `code`, que diz se o pedido deu certo:

Protocolo	Deu certo quando
`@0`, `@6`, `@8`	`code` começa com `1` (ex.: `code=1001`, ou `code=1` no LOUD). `code=0001` indica erro.
`@1`	a resposta traz `status=OK`.
`@9`	a resposta chega (a descrição não traz `code`).

```
$72B0>/ac/rs@0?code=1001&r0=on&r1=off<77120#
```

Descrição (@9)

A descrição é a lista das ações que um equipamento oferece, com os parâmetros de cada uma: nome, tipo e valores aceitos. Cada equipamento guarda a própria descrição. Com ela, você sabe o que pode enviar a um equipamento sem consultar outra documentação.

O Studio SHC usa a descrição para montar as telas do equipamento: saídas, entradas e ações. Para buscar a descrição de novo, use o botão **Recarregar**.

Pedir a descrição

Para pedir a descrição de todas as ações, envie a ação `/*` com o protocolo `@9`:

```
$72B0>/*@9?<48216#
```

Para pedir a descrição de uma ação só, envie o caminho dela com o protocolo `@9`. Exemplo: a descrição de `/ac/rs`.

```
$72B0>/ac/rs@9?<48217#
```

A resposta

O equipamento responde com **uma mensagem por ação**: com `/*`, uma para cada ação; com o caminho de uma ação, uma mensagem só. Cada mensagem traz o caminho da ação, o protocolo `@9` e, no lugar dos valores, um texto JSON:

```
$72B0>/ac/rs@9?{"name":"rs","path":"/ac/rs","params":[{"name":"r0","type":"Boolean","params":["togg
```

- No pedido com `/*`, o número de mensagens é o valor `nAC` da ação `/if/device`. Espere até receber esse número de mensagens. No Studio SHC, o campo **Ações** mostra quantas chegaram, como **6/18**.

- Decodifique o texto como URL antes de ler (`%7B` vira `{`). Num texto sem codificação, a decodificação não muda nada.
- O JSON termina na chave que fecha o primeiro `{` . O que vier depois, como `&_ts=1760900000` , são parâmetros de controle.

O mesmo JSON, organizado:

```
{
  "name": "rs",
  "path": "/ac/rs",
  "params": [
    { "name": "r0", "type": "Boolean", "params": ["toggle", "on", "off"] },
    { "name": "r1", "type": "Boolean", "params": ["toggle", "on", "off"] }
  ],
  "paramsResponse": []
}
```

Esse equipamento tem duas saídas de relé, `r0` e `r1` . Cada uma aceita `toggle` , `on` ou `off` . Veja como usar em `/ac/rs` .

Campos da ação

Campo	O que traz
`name`	Nome curto da ação (ex.: `rs`).
`path`	Caminho da ação, para usar nas mensagens (ex.: `/ac/rs`).
`params`	Parâmetros que a ação aceita. A consulta (`@0` sem parâmetros) devolve os valores atuais deles.
`paramsResponse`	Parâmetros que só vêm na resposta: você não os envia. Ex.: em `/if/device` , `model` , `version` e `nAC` .
`len`	Quantos itens a ação guarda, de `_n` = `0` até `len` − 1. Nas ações sem `_n` , vem `1` ou o campo não vem. Firmware anterior ao 3.1.0 usa o nome `lens` (ex.: `"lens":1` em `/lb/device`).

Campos de cada parâmetro

Campo	O que traz
`name`	Nome do parâmetro na mensagem (ex.: `r0` , `vl`). Começa com `_` quando é parâmetro de controle.
`type`	Tipo do valor. Veja a tabela de tipos abaixo.
`params`	Lista com os valores aceitos (ex.: `on` , `off`) e as restrições, que começam com `$` (ex.: `$max=99`).
`friendlyName`	Opcional. Nome para mostrar ao usuário (ex.: `volume`).
`description`	Opcional. Explicação do parâmetro.

Tipo	Valor
`Boolean`	Um dos valores listados em `params`, como `toggle`, `on` e `off`.
`Chosen`	Um dos valores listados em `params` (ex.: as entradas `AUX1`, `BT`, `OPT` do LOUD).
`Integer`	Número inteiro, entre `$min` e `$max`.
`String`	Texto, com tamanho máximo `$max`.
`Buffer`	Bytes em hexadecimal, 2 caracteres por byte (ex.: `0D` = Enter). `$max` limita os caracteres e `$maxBytes`, os bytes.
`Array[N]:Tipo`	Firmware legado: N parâmetros do mesmo tipo, numerados de `0` a N − 1. Ex.: `r` com `Array[4]:Boolean` equivale a `r0` a `r3`.

Restrições

As restrições vêm dentro de `params`, no formato `$nome=valor`:

Restrição	Significado	Exemplo
`$min`	Menor valor aceito.	`$min=-15`
`$max`	`Integer`: maior valor aceito. `String` e `Buffer`: número máximo de caracteres.	`$max=99`
`$maxBytes`	`Buffer`: número máximo de bytes.	`$maxBytes=128`
`$inc`, `$dec`	Aparecem nos parâmetros `Integer`, no formato `$inc=[a,b]`.	`$inc=[1,20]`

Exemplo de parâmetro `Integer`: o volume de uma zona do LOUD aceita de `0` a `99`.

```
{ "name": "vl", "friendlyName": "volume", "type": "Integer", "params": ["$inc=[1,20]", "$dec=[1,20]
```

Parâmetros agrupados

```
$6A6F>/ac/rs@0?r[0,1]=off&_to=192.168.0.1<99999#
```

Parâmetros com o mesmo valor podem ser agrupados: `r[0,1]=off` equivale a `r0=off&r1=off`. Essa forma vale para firmwares 2.x ou mais novos; firmwares 1.x não a aceitam.

Ações

Ações do Native Device

Cada equipamento informa as ações que oferece na descrição (protocolo `@9`). As ações valem por UDP, TCP ou XNET. Nem todo equipamento tem todas as ações: consulte a descrição do equipamento.

Lista de ações

Caminho	Nome	O que faz
`/ac/rs`	Rele State	Consulta e aciona as saídas de relé.
`/ac/bl`	Persianas	Consulta e aciona as persianas do MRQ 10 (firmware 3.4.x).
`/ac/pers`	Persiana State	Consulta e aciona as persianas (firmware 2.x).
`/ac/zonaN`	LOUD	Consulta e controla as zonas do amplificador LOUD.
`/ac/xcN`	Cena do Control	Intensidade das cargas em cada cena (0 a 100; 0 a 240 em firmware anterior à 2.4.17).
`/ac/xClick`	xClick	Faz um botão disparar um clique remoto em outro equipamento.
`/ac/xHold`	xHold	Faz um botão disparar um “segurar” remoto em outro equipamento.
`/ac/bt`	Botão	Aviso que o equipamento envia quando um botão físico é pressionado (`b=` índice do botão). Não é um comando. Enviado pelos equipamentos com firmware 3.x ou superior.
`/ac/serial`	Serial	Envia um evento serial pelos botões. Existe em alguns modelos.
`/cl/actions`	Executar botão	Executa as ações gravadas num botão: `action=` índice do botão e `hold=0` (ações de clique) ou `hold=1` (ações de segurar).
`/aa/tcp`	Ação TCP	Envia uma mensagem por TCP para o IP e a porta configurados.
`/aa/udp`	Ação UDP	Envia uma mensagem por UDP para o IP e a porta configurados.
`/aa/wol`	Wake-on-LAN	Liga um computador pela rede, a partir do endereço MAC.
`/aa/delay`	Espera	Põe uma espera entre duas ações da cena de um botão.
`/if/device`	Informações	Devolve modelo, nome, versão e outros dados do equipamento.
`/ac/ping`	Ping	Confere se o equipamento responde; devolve modelo, tempo ligado e memória livre.
`/ac/status`	Estado	Devolve IP, sinal do Wi-Fi e memória livre do equipamento.

Exemplo: executar as ações de clique do botão de índice 1 no equipamento `72B0`.

```
  $72B0>/cl/actions@0?action=1&hold=0<12345#
```

/ac/rs: saídas de relé

A ação `/ac/rs` consulta e aciona as saídas de relé do equipamento. Cada saída é um parâmetro: `r0` é a saída 1, `r1` a saída 2, e assim por diante, até o número de saídas do equipamento.

Consultar

```
$6A6F>/ac/rs@0?_to=192.168.0.10<99999#
```

Resposta

```
$6A6F>/ac/rs@0?r0=off&r1=off&_to=192.168.0.10<52817#
```

Cada parâmetro vem com `on` (ligada) ou `off` (desligada).

Saídas ligadas (`_en`)

No MRQ 10 com firmware 3.4.x, a resposta traz também o campo `_en` : as saídas que aceitam comando por `/ac/rs` , uma por bit. O bit 0 é a saída 1 (`r0`), o bit 1 é a saída 2 (`r1`), e assim por diante.

```
$72B0>/ac/rs@0?code=1001&r0=off&r1=off&r2=off&r3=off&r4=off&r5=off&r6=off&r7=off&r8=off&r9=off&_en=
```

No exemplo, `_en=1008` é `1111110000` em binário: as saídas 5 a 10 (`r4` a `r9`) aceitam comando. As saídas 1 a 4 (`r0` a `r3`) estão sendo usadas por 2 persianas, e o equipamento ignora comando para elas.

- A consulta traz o estado de todas as saídas, inclusive as das persianas.
- Sem o campo `_en` na resposta, todas as saídas aceitam comando.
- Para acionar as persianas, use `/ac/bl` .

Acionar

Envie o parâmetro da saída com o valor desejado:

Valor	Efeito
`on`	Liga a saída.
`off`	Desliga a saída.
`toggle`	Inverte o estado atual da saída.

```
$6A6F>/ac/rs@0?r0=on<12345#
```

Para acionar várias saídas de uma vez, envie um parâmetro para cada uma: `r8=on&r9=on` . A resposta traz o estado de todas as saídas.

/ac/bl: persianas (firmware 3.x)

A ação `/ac/bl` consulta e aciona as persianas do equipamento. Cada persiana usa um par de relés: o primeiro sobe e o segundo desce. A persiana 1 usa as saídas 1 e 2 (`r0` e `r1`); a persiana 2, as saídas 3 e 4 (`r2` e `r3`); e assim por diante.

Disponibilidade

`/ac/bl` existe no MRQ 10 com firmware 3.4.x, com até 5 persianas. No firmware 2.x, as persianas usam `/ac/pers` .

Parâmetros

Cada persiana tem três parâmetros, numerados a partir de 0. O número é o índice da persiana: `u0` , `d0` e `t0` são da persiana 1.

Parâmetro	Nome	`on`	`off`
`uN`	Subir	Sobe a persiana. Se ela estiver descendo, troca o sentido.	Para a persiana.
`dN`	Descer	Desce a persiana. Se ela estiver subindo, troca o sentido.	Para a persiana.
`tN`	Parar/ Alternar	Em movimento: para. Parada: anda no sentido contrário ao último movimento.	Para a persiana.

Na resposta, `uN` vem `on` enquanto a persiana sobe, `dN` vem `on` enquanto ela desce, e `tN` vem `on` enquanto ela anda, em qualquer sentido.

Quais persianas estão ligadas

O número de persianas vem da configuração do equipamento: o parâmetro `n` de `/lb/blinds` (de 0 a 5). O campo `_en` da resposta mostra as persianas ligadas, uma por bit: o bit 0 é a persiana 1, o bit 1 é a persiana 2, e assim por diante.

`_en`	Em binário	Persianas ligadas
`1`	`00001`	Persiana 1
`3`	`00011`	Persianas 1 e 2
`31`	`11111`	As 5 persianas

- A consulta só traz os parâmetros das persianas ligadas.
- Comando para uma persiana desligada não aciona nada. A resposta vem com `code=1011` .
- Os relés de uma persiana ligada saem de `/ac/rs` : use `/ac/bl` para acioná-los.

Consultar

```
$72B0>/ac/bl@0?<83554#
```

Resposta

Exemplo com 2 persianas ligadas e paradas:

```
$72B0>/ac/bl@0?code=1001&u0=off&d0=off&t0=off&u1=off&d1=off&t1=off&_en=3<3363824#
```

Acionar

Envie o parâmetro da persiana com `on`. Exemplo: subir a persiana 1.

```
$72B0>/ac/bl@0?u0=on<63613#
```

A resposta traz o estado de todas as persianas ligadas:

```
$72B0>/ac/bl@0?code=1111&u0=on&d0=off&t0=on&u1=off&d1=off&t1=off&_en=3<3368851#
```

Para parar a persiana 1, envie `t0=on` (ou `u0=off`).

/ac/pers: persianas

legacy

Firmware antigo

`/ac/pers` funciona só nos equipamentos com firmware antigo (2.x, linha PERSIANA). No MRQ 10 com firmware 3.4.x, use `/ac/bl`.

A ação `/ac/pers` consulta o estado das persianas do equipamento. Cada persiana tem três parâmetros, numerados a partir de 0: `on0`, `up0` e `down0` são da persiana 1; `on1`, `up1` e `down1`, da persiana 2; e assim por diante, até o número de persianas do equipamento.

Consultar

```
$6A6F>/ac/pers@0?_to=192.168.0.10<99999#
```

Resposta

```
$6A6F>/ac/pers@0?on0=off&on1=on&...&up0=off&up1=on&...&down0=on&down1=off&...&_to=192.168.0.10<5281
```

Cada parâmetro vem com `on` ou `off`.

Acionar

Envie o parâmetro `onN` da persiana com o valor desejado:

Valor	Efeito
`on`	Sobe a persiana.
`off`	Desce a persiana.
`toggle`	Troca o sentido do movimento.

Se a persiana está subindo e recebe `off`, ela para. Um novo `off` faz a persiana descer.

```
$6A6F>/ac/pers@0?on0=on<12345#
```

LOUD: controle das zonas

Esta página mostra como descobrir e controlar as zonas de um amplificador LOUD pela rede local.

> **Zonas e entradas variam por aparelho**
>
> O protocolo é o mesmo para toda a linha LOUD, mas o número de zonas e de entradas depende do aparelho. Os exemplos abaixo mostram até 6 zonas (`zona0` a `zona5`) e as entradas `AUX1` a `AUX5` , `BT` e `OPT` . O APL 420S, por exemplo, tem 2 zonas. Para saber o que um aparelho tem, peça a descrição (seção 3): ele responde com as zonas e os parâmetros que expõe.

1. Formato da mensagem

A mensagem segue o formato Native Device: ID, ação, protocolo, parâmetros, `_to` , sequência e fim. Os protocolos usados aqui são `@0` (ler ou definir valores) e `@9` (descrição).

```
$3A7F>/ac/zona0@0?vl=50&_to=192.168.0.10<123456#
```

Neste exemplo, o equipamento `3A7F` recebe o volume 50 na zona 0. O número `123456` é a sequência: troque-o a cada envio.

2. Equipamento usado nos exemplos

Os exemplos desta página usam um LOUD de ID `3A7F` com 6 zonas.

Propriedade	Valor
ID	`3A7F`
Modelo	Multiroom
Zonas de controle	`/ac/zona0` , `/ac/zona1` , `/ac/zona2` , `/ac/zona3` , `/ac/zona4` , `/ac/zona5`
Config avançada de zonas	`/lb/Zona` (indexada por `_n=0..5`)

Outras ações (Bluetooth, Inputs, IR, WiFi, modo Dualzone/Singlezone/4.1) também estão disponíveis via path `/lb/*` , mas não são cobertas aqui.

3. Descoberta — Protocolo 9 (DESCRIPTION)

A descrição informa quais zonas e parâmetros o aparelho tem. Como pedir e como ler a resposta está em Descrição (@9).

Pedido de todas as ações do aparelho:

```
$3A7F>/*@9?<12345#
```

Pedido só da zona 0:

```
$3A7F>/ac/zona0@9?<12346#
```

Cada zona vem numa mensagem própria. Modelo de resposta da zona 0:

```
$3A7F>/ac/zona0@9?{"name":"zona0","path":"/ac/zona0","params":[
 {"name":"pw","friendlyName":"power","params":["toggle","on","off"],"type":"Boolean"},
 {"name":"mt","friendlyName":"mute","params":["toggle","on","off"],"type":"Boolean"},
 {"name":"in","friendlyName":"input","params":["AUX1","AUX2","AUX3","AUX4","AUX5","BT","OPT"],"type
 {"name":"vl","friendlyName":"volume","params":["$inc=[1,20]","$dec=[1,20]","$max=99","$min=0"],"ty
 {"name":"bs","friendlyName":"bass","params":["$inc=[1,20]","$dec=[1,20]","$max=0","$min=-15"],"typ
 {"name":"tr","friendlyName":"treble","params":["$inc=[1,20]","$dec=[1,20]","$max=0","$min=-15"],"t
 {"name":"md","friendlyName":"middle","params":["$inc=[1,20]","$dec=[1,20]","$max=0","$min=-15"],"t
]}<58210#
```

O que cada parâmetro da zona faz está na seção 4.

4. Parâmetros de uma zona (`/ac/zonaN`)

Param	Nome amigável	Tipo	Valores / Faixa	Exemplo
`pw`	power	Boolean	`on` , `off` , `toggle`	`pw=on`
`mt`	mute	Boolean	`on` , `off` , `toggle`	`mt=off`
`in`	input	Chosen	`AUX1` , `AUX2` , `AUX3` , `AUX4` , `AUX5` , `BT` , `OPT`	`in=BT`
`vl`	volume	Integer	`0` a `99`	`vl=50`
`bs`	bass	Integer	`-15` a `0` (dB)	`bs=-5`
`tr`	treble	Integer	`-15` a `0` (dB)	`tr=-3`
`md`	middle	Integer	`-15` a `0` (dB)	`md=-2`

5. Comandos práticos

Todos os exemplos usam a zona 0 (`/ac/zona0`) do equipamento `3A7F` . Para outras zonas, troque o path (`/ac/zona1` , `/ac/zona2` , …). A sequência (`<123#`) deve ser trocada a cada envio.

5.1 Consultar estado completo

Envio:

```
$3A7F>/ac/zona0@0?<111#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&pw=on&mt=off&in=AUX1&vl=50&bs=0&tr=0&md=0<7341#
```

`code=1` = sucesso.

5.2 Ligar / desligar / toggle

```
$3A7F>/ac/zona0@0?pw=on<112#
$3A7F>/ac/zona0@0?pw=off<113#
$3A7F>/ac/zona0@0?pw=toggle<114#
```

Resposta (eco de confirmação):

```
$3A7F>/ac/zona0@0?code=1&pw=on<7342#
```

5.3 Mutar / desmutar

```
$3A7F>/ac/zona0@0?mt=on<115#
$3A7F>/ac/zona0@0?mt=off<116#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&mt=on<7343#
```

5.4 Trocar fonte de áudio

```
$3A7F>/ac/zona0@0?in=AUX1<117#
$3A7F>/ac/zona0@0?in=BT<118#
$3A7F>/ac/zona0@0?in=OPT<119#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&in=BT<7344#
```

5.5 Definir volume

```
$3A7F>/ac/zona0@0?vl=75<120#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&vl=75<7345#
```

5.6 Ajustar equalização (graves, médios, agudos)

```
$3A7F>/ac/zona0@0?bs=-5<121#
$3A7F>/ac/zona0@0?md=-3<122#
$3A7F>/ac/zona0@0?tr=-2<123#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&bs=-5<7346#
```

5.7 Comando combinado (múltiplos parâmetros)

```
$3A7F>/ac/zona0@0?pw=on&in=BT&vl=60&mt=off<124#
```

Resposta:

```
$3A7F>/ac/zona0@0?code=1&pw=on&in=BT&vl=60&mt=off<7347#
```

6. Configuração avançada da zona (`/lb/Zona`)

`/lb/Zona` guarda a configuração persistente de cada zona (nome, limites, valores iniciais). O parâmetro `_n` indica qual zona está sendo lida ou gravada, de 0 até o número de zonas menos 1 (0 a 5 no equipamento de exemplo).

Parâmetros

Param	Tipo	Faixa / Valores	Descrição
`nmc`	String	até 15 caracteres	Nome amigável da zona
`vli`	Integer	`0` a `30`	Volume inicial (ao ligar)
`vll`	Integer	`40` a `99`	Volume máximo permitido
`ii`	Chosen	`AUX1..AUX5`, `BT`, `OPT`	Fonte padrão ao ligar
`mdda`	Chosen	`Normal`, `Power Automático`	Modo de ligar automático
`md`	Integer	`-15` a `0`	Médio default
`tb`	Integer	`-15` a `0`	Agudo default
`bs`	Integer	`-15` a `0`	Grave default

6.1 Ler a configuração da zona 0

Envio:

```
$3A7F>/lb/Zona@0?_n=0<130#
```

Resposta:

```
$3A7F>/lb/Zona@0?code=1&nmc=Sala&vli=15&vll=99&ii=AUX1&mdda=Normal&md=-5&tb=-3&bs=-2&_n=0<7348#
```

6.2 Renomear a zona 0 para “Sala”

```
$3A7F>/lb/Zona@0?nmc=Sala&_n=0<131#
```

O valor em `nmc` pode ser URI-encoded se tiver espaços/acentos (ex.: `nmc=Sala%20de%20Estar`).

6.3 Definir volume inicial e máximo da zona 1

```
$3A7F>/lb/Zona@0?vli=20&vll=80&_n=1<132#
```

7. Notas rápidas

- **Sequência única por envio**: gere um número novo a cada mensagem. Veja número de sequência.
- **`_to=<IP>`**: opcional. Veja os parâmetros de controle.
- **Separadores**: obrigatórios mesmo com o campo vazio. Ex.: leitura sem parâmetros é `$3A7F>/ac/zona0@0?<SEED#`. Veja o formato da mensagem.
- **Envios sequenciais**: envie um pacote por vez, com intervalo de cerca de 50 ms entre envios consecutivos.
- **`code`** na resposta: `code=1` indica sucesso. Veja a resposta.
- **Broadcast `0000`**: substitua `$3A7F` por `$0000` para enviar a todos os equipamentos da rede. Use para descoberta (`/if/device`), não para controle.

/ac/xcN: intensidade por cena

A ação `/ac/xcN` informa a intensidade de cada saída na cena `N` (no exemplo, `/ac/xc0`). Cada saída é um parâmetro: `l0` é a saída 1, `l1` a saída 2, e assim por diante.

> ⚠ **Faixa de valores**
>
> Firmware 2.4.17 ou mais novo: de `0` (apagado) a `100` (intensidade máxima). Firmware anterior: de `0` (apagado) a `240` (intensidade máxima). Para converter em porcentagem, calcule `(valor / 240) × 100`. Exemplo: `185` corresponde a 77%.

Consultar

```
$6A6F>/ac/xc0@0?_to=192.168.0.10<99999#
```

Resposta

```
$6A6F>/ac/xc0@0?l0=0&l1=80&l2=0...&_to=192.168.0.10<52817#
```

/ac/xClick: clique remoto

A ação `/ac/xClick` faz um botão disparar um clique remoto no botão `bt` do Control `id`. A consulta devolve a configuração atual.

Disponibilidade

O MRQ 10 com firmware 3.x e o Keypad Xcene com firmware 3.1.0 não têm esta ação. Confira na descrição do equipamento.

Consultar

```
$6A6F>/ac/xClick@0?_to=192.168.0.10<99999#
```

Resposta

```
$6A6F>/ac/xClick@0?id=3&bt=5&_to=192.168.0.10<52817#
```

- `id` : ID do Control que recebe o comando.
- `bt` : número do botão nesse Control.

/ac/xHold: segurar remoto

A ação `/ac/xHold` faz um botão disparar um “segurar” remoto no botão `bt` do Control `id`. A consulta devolve a configuração atual.

> **Disponibilidade**
>
> O MRQ 10 com firmware 3.x e o Keypad Xcene com firmware 3.1.0 não têm esta ação. Confira na descrição do equipamento.

Consultar

```
$6A6F>/ac/xHold@0?_to=192.168.0.10<99999#
```

Resposta

```
$6A6F>/ac/xHold@0?id=3&bt=5&_to=192.168.0.10<52817#
```

- `id`: ID do Control que recebe o comando.
- `bt`: número do botão nesse Control.

/aa/tcp: enviar mensagem TCP

A ação `/aa/tcp` faz o equipamento abrir uma conexão TCP com outro aparelho da rede e enviar uma mensagem. Use-a para acionar dispositivos externos, como projetores e controladores.

Disponibilidade

No equipamento, as ações `/aa/*` existem nos firmwares 2.4.x e a partir do 3.1.0. Nos aplicativos de configuração (Studio SHC e Xcene Config), elas só aparecem com a opção **Ações avançadas** ligada.

Parâmetros

Parâmetro	Conteúdo
`ip`	IP de destino.
`port`	Porta de destino, de 1 a 65535.
`payload`	Conteúdo da mensagem, em hexadecimal: 2 caracteres por byte (ex.: `0D` = Enter), até 256 caracteres (128 bytes). Os separadores `:`, `-` e `.` são ignorados, mas contam no limite: `41:42-43` envia `ABC`.
`persistent`	`on` mantém a conexão aberta depois do envio, e o próximo envio ao mesmo destino usa a mesma conexão. `off` (padrão) fecha a conexão depois de cada envio.
`_su`	Identificador livre, de até 8 caracteres. Veja os parâmetros de controle.

Enviar

Com `@0` e os parâmetros, o equipamento envia a mensagem na hora. Exemplo: enviar `ligar` para 192.168.0.50, na porta 5000.

```
$72B0>/aa/tcp@0?ip=192.168.0.50&port=5000&payload=6C69676172<48230#
```

No exemplo, `6C69676172` é a palavra “ligar” em hexadecimal.

Parâmetro que não vai na mensagem fica com o último valor usado. Envie sempre `ip`, `port` e `payload`.

A resposta chega logo depois do pedido, antes da entrega:

```
$72B0>/aa/tcp@0?code=1111&ip=192.168.0.50&port=5000&payload=&persistent=off&_su=&bytes=5&err=0<3399
```

Parâmetro da resposta	Conteúdo
`bytes`	Quantos bytes o equipamento colocou na fila de envio.
`err`	`0` = mensagem aceita para envio. `1` = `payload` inválido (caractere que não é hexadecimal, número ímpar de caracteres ou mais de 128 bytes). `2` = sem `ip`. `3` = fila de envio cheia. `4` = sem memória. Com erro, nada é enviado.

O `payload` vem sempre vazio na resposta. `err=0` confirma que o equipamento aceitou a mensagem, mas não que o destino a recebeu.

Usar numa cena

Para o envio acontecer quando um botão é pressionado, grave a ação como item da cena do botão (`/cs/btN`). No item, `path` é `/aa/tcp` e `query` traz os parâmetros separados por `;`. Exemplo: o item 1 da cena do botão 0 envia `ligar` para 192.168.1.50, na porta 8008.

```
$72B0>/cs/bt0@6?code=1001&_n=1&id=72B0&path=/aa/tcp&query=ip%3D192.168.1.50;payload%3D6C69676172;po
```

Consultar

```
$72B0>/aa/tcp@0?_to=192.168.0.10<99999#
```

Resposta

A consulta não envia nada. Ela devolve `code=1001` e os parâmetros da ação:

```
$72B0>/aa/tcp@0?code=1001&ip=192.168.0.50&port=5000&payload=&persistent=off&_su=&bytes=0&err=0&_to=
```

/aa/udp: enviar mensagem UDP

A ação `/aa/udp` faz o equipamento enviar uma mensagem por UDP para outro aparelho da rede. Use-a para acionar dispositivos externos.

> **Disponibilidade**
>
> No equipamento, as ações `/aa/*` existem nos firmwares 2.4.x e a partir do 3.1.0. Nos aplicativos de configuração (Studio SHC e Xcene Config), elas só aparecem com a opção **Ações avançadas** ligada.

Parâmetros

Parâmetro	Conteúdo
`ip`	IP de destino.
`port`	Porta de destino, de 1 a 65535.
`payload`	Conteúdo da mensagem, em hexadecimal: 2 caracteres por byte (ex.: `0D` = Enter), até 256 caracteres (128 bytes). Os separadores `:`, `-` e `.` são ignorados, mas contam no limite: `41:42-43` envia `ABC`.
`_su`	Identificador livre, de até 8 caracteres. Veja os parâmetros de controle.

Enviar

Com `@0` e os parâmetros, o equipamento envia a mensagem na hora. Exemplo: enviar `ligar` para 192.168.0.50, na porta 5000.

```
$72B0>/aa/udp@0?ip=192.168.0.50&port=5000&payload=6C69676172<48230#
```

No exemplo, `6C69676172` é a palavra “ligar” em hexadecimal.

Parâmetro que não vai na mensagem fica com o último valor usado. Envie sempre `ip`, `port` e `payload`.

A resposta chega logo depois do pedido, antes da entrega:

```
$72B0>/aa/udp@0?code=1111&ip=192.168.0.50&port=5000&payload=&_su=&bytes=5&err=0<3395745#
```

Parâmetro da resposta	Conteúdo
`bytes`	Quantos bytes o equipamento colocou na fila de envio.
`err`	`0` = mensagem aceita para envio. `1` = `payload` inválido (caractere que não é hexadecimal, número ímpar de caracteres ou mais de 128 bytes). `2` = sem `ip`. `3` = fila de envio cheia. `4` = sem memória. Com erro, nada é enviado.

O `payload` vem sempre vazio na resposta. `err=0` confirma que o equipamento aceitou a mensagem, mas não que o destino a recebeu.

Usar numa cena

Para o envio acontecer quando um botão é pressionado, grave a ação como item da cena do botão (`/cs/btN`). No item, `path` é `/aa/udp` e `query` traz os parâmetros separados por `;`. Exemplo: o item 3 da cena do botão 0 envia `ligar` para 192.168.1.50, na porta 8008.

```
$72B0>/cs/bt0@6?code=1001&_n=3&id=72B0&path=/aa/udp&query=ip%3D192.168.1.50;payload%3D6C69676172;po
```

Consultar

```
$72B0>/aa/udp@0?_to=192.168.0.10<99999#
```

Resposta

A consulta não envia nada. Ela devolve `code=1001` e os parâmetros da ação:

```
$72B0>/aa/udp@0?code=1001&ip=192.168.0.50&port=5000&payload=&_su=&bytes=0&err=0&_to=192.168.0.10<33
```

/aa/wol: Wake-on-LAN

A ação `/aa/wol` liga um computador pela rede local, a partir do endereço MAC dele. O equipamento envia o pacote Wake-on-LAN por UDP: 6 bytes `FF` seguidos do MAC repetido 16 vezes.

> **Disponibilidade**
>
> No equipamento, as ações `/aa/*` existem nos firmwares 2.4.x e a partir do 3.1.0. Nos aplicativos de configuração (Studio SHC e Xcene Config), elas só aparecem com a opção **Ações avançadas** ligada.

Parâmetros

Firmware	Parâmetros
3.1.0 ou mais novo	`mac`: endereço MAC em hexadecimal, 6 bytes (`001B44112233` ou `00:1B:44:11:22:33`). `ip`: endereço de broadcast; vazio envia para `255.255.255.255`. Se o roteador bloquear esse endereço, use o broadcast da sua rede (ex.: `192.168.0.255`). `port`: porta UDP; `0` envia para a porta 9. `_su`: identificador livre, de até 8 caracteres (veja os parâmetros de controle).
2.4.x	`mac0` a `mac5`: os 6 bytes do MAC, em decimal.

Enviar

Com `@0` e os parâmetros, o equipamento envia o pacote na hora. Exemplo: ligar o computador de MAC `00:1B:44:11:22:33`.

```
$72B0>/aa/wol@0?mac=001B44112233&ip=192.168.0.255&port=9<48231#
```

```
$72B0>/aa/wol@0?code=1111&mac=001B44112233&ip=192.168.0.255&port=9&_su=&err=0<3400980#
```

Parâmetro da resposta	Conteúdo
`err`	`0` = pacote aceito para envio. `1` = `mac` inválido (menos de 6 bytes ou caractere que não é hexadecimal). Com erro, nada é enviado.

Parâmetro que não vai na mensagem fica com o último valor usado. Envie sempre `mac`, `ip` e `port`.

Usar numa cena

Para o envio acontecer quando um botão é pressionado, grave a ação como item da cena do botão (`/cs/btN`). No item, `path` é `/aa/wol` e `query` traz os parâmetros separados por `;`, como em `/aa/tcp`.

Consultar

```
$72B0>/aa/wol@0?_to=192.168.0.10<99999#
```

Resposta

A consulta não envia nada. Ela devolve `code=1001` e os parâmetros da ação:

```
$72B0>/aa/wol@0?code=1001&mac=001B44112233&ip=192.168.0.255&port=9&_su=&err=0&_to=192.168.0.10<3382
```

/aa/delay: esperar entre ações da cena

A ação `/aa/delay` põe uma espera entre duas ações da cena de um botão. Exemplo: ligar o amplificador, esperar 2 segundos e depois mudar a entrada de áudio.

> **Disponibilidade**
>
> No equipamento, `/aa/delay` existe a partir do firmware 3.1.0. Nos aplicativos de configuração (Studio SHC e Xcene Config), as ações `/aa/*` só aparecem com a opção **Ações avançadas** ligada.

Parâmetros

Parâmetro	Conteúdo
`ms`	Tempo de espera, em milissegundos, de `0` a `60000` (60 segundos).
`_su`	Identificador livre, de até 8 caracteres. Veja os parâmetros de controle.

Usar numa cena

Grave a espera como item da cena do botão (`/cs/btN`), entre as duas ações. No item, `path` é `/aa/delay` e `query` traz `ms`. Exemplo: o item 2 da cena do botão 0, lido com `@6`, espera 2 segundos.

```
$72B0>/cs/bt0@6?code=1001&_n=2&id=72B0&path=/aa/delay&query=ms%3D2000&invertToOff=off&offOnRelease=
```

Durante a espera, o equipamento continua respondendo a outras mensagens.

Mensagem avulsa

Enviada fora de uma cena, a mensagem só é confirmada: o equipamento responde na hora com `code=1111` e não segura nenhuma outra mensagem.

```
$72B0>/aa/delay@0?ms=3000<32254#
```

```
$72B0>/aa/delay@0?code=1111&ms=3000&_su=<3404753#
```

Consultar

```
$72B0>/aa/delay@0?_to=192.168.0.10<99999#
```

```
$72B0>/aa/delay@0?code=1001&ms=0&_su=&_to=192.168.0.10<3383694#
```

/if/device: informações do equipamento

A ação `/if/device` devolve os dados do equipamento. Com o ID `0000`, todos os equipamentos da rede respondem, cada um com os seus dados: é assim que os aplicativos descobrem os equipamentos.

Consultar

```
$0000>/if/device@0?_to=192.168.0.10<99999#
```

Resposta

```
$72B0>/if/device@0?code=1001&model=MRQ10&deviceName=MRQ10-72B0&version=3.3.0...&_to=192.168.0.10<52
```

Parâmetro	Conteúdo
`code`	Resultado do pedido. Veja a resposta.
`model`	Modelo do equipamento (ex.: `MRQ10`, `KEYPADWIFI`).
`deviceName`	Nome do equipamento na rede. Firmware 3.x: `<MODELO>-<ID>` (ex.: `MRQ10-72B0`). Firmware 2.4.x: `Xcene_<MODELO>-<ID>`. LOUD: `Loud_420SV3-<ID>`.
`version`	Versão do firmware.
`serialNumber`	Número de série.
`type`	Tipo do firmware (ex.: `RELE`, `ND`).
`id`	ID do equipamento: os 4 últimos caracteres do MAC, em maiúsculas.
`date`	Data de fabricação.
`nAC`	Número de ações que o equipamento oferece: é o número de mensagens que ele manda quando você pede a descrição.

/ac/ping: equipamento respondendo

A ação `/ac/ping` confere se o equipamento está respondendo. A resposta traz o modelo, há quanto tempo o equipamento está ligado e a memória livre. Ela só consulta: não muda nada no equipamento.

Consultar

```
$72B0>/ac/ping@0?<10428#
```

Resposta

```
$72B0>/ac/ping@0?code=1001&model=MRQ10&uptime=3244&heap=15856<3245066#
```

Parâmetro	Conteúdo
`code`	Resultado do pedido. Veja a resposta.
`model`	Modelo do equipamento (ex.: `MRQ10`).
`uptime`	Tempo desde que o equipamento ligou, em segundos. Volta a contar do zero quando o equipamento reinicia.
`heap`	Memória livre do equipamento.

/ac/status: estado da rede e do equipamento

A ação `/ac/status` devolve um resumo do equipamento: modelo, tipo do firmware, IP, sinal do Wi-Fi e memória livre. Ela só consulta: não muda nada no equipamento.

Consultar

```
$72B0>/ac/status@0?<17896#
```

Resposta

```
$72B0>/ac/status@0?code=1001&model=MRQ10&app=NDV2&ip=192.168.1.16&rssi=-50&wifi_state=3&heap=15856&
```

Parâmetro	Conteúdo
`code`	Resultado do pedido. Veja a resposta.
`model`	Modelo do equipamento (ex.: `MRQ10`).
`app`	Tipo do firmware, o mesmo `type` de `/if/device` (ex.: `NDV2`).
`ip`	IP do equipamento na rede Wi-Fi.
`rssi`	Sinal do Wi-Fi, em dBm, de `-128` a `0`. Quanto mais perto de 0, melhor.
`wifi_state`	Estado da conexão Wi-Fi, de `0` a `4`.
`heap`	Memória livre do equipamento.
`sdk`	Versão do sistema base do firmware, cortada em 15 caracteres.