Configuration procedure for new devices

If your device is not a built-in module listed in the module configuration menu, a user contributed device template (explanation) may be available. Otherwise, follow this procedure for configuring the ESP chip pins used by your device.

Some smart devices have additional functionality which may be handled by codes sent to a separate MCU in the device. Devices with functions offloaded to a separate MCU require additional coding in the software or via rules. Certain supported Tasmota BASE devices have built-in code to handle MCU controlled devices. Using a template with an appropriate BASE device may have the programming logic required to manage the MCU commands for your device. If an existing BASE device with the logic for your device is ***not*** available, a modified device driver will be required. This case is outside of the scope of this article.

If your device is similar to the existing built-in module (e.g., a particular MCU or power monitoring algorithm) it is best to use that as a starting point. When you are not sure which module is suitable for your device, use `Generic module (18)`.

ESP8266

Finding Relays and Lights

Step 1.

Begin this procedure by disabling power state saves. Some improper GPIO assignments can cause device reboots. Disabling this setting avoids repeated flash v Tasmota to return to a fail safe state in case of a bad configuration. Ensure that boot loop control is not disabled.

```
Backlog SetOption0 0; SetOption36 1
```

Step 2.

Assign every available GPIO to successive `Relay<x>` components. For the initial GPIO probe, exclude "dedicated" GPIO such as GPIO0/GPIO2 and Tx/Rx, etc. You these assignments:

```
{"NAME":"ID Relays","GPIO":[0,0,0,0,224,225,0,0,226,227,228,229,230,0],"FLAG":0,"BASE":18}
```

Save the configuration. Once the device reboots, use the virtual buttons on the web UI to find which of the assigned GPIO actually control the physical relays anc which GPIO act on which device peripheral.

Step 2a.

If you are unable to control some of the relays or LEDs on the device, they may be attached to the "dedicated" GPIO skipped in the initial probe. Now assign those

```
{"NAME":"ID Relays 2","GPIO":[224,225,226,227,0,0,0,0,0,0,0,0,0],"FLAG":0,"BASE":18}
```

If your device is based on the ESP8285 and you are still unable to control some of the relays or LEDs on the device, they may be attached to GPIO9 or GPIO10. No repeat step *2a*:

```
{"NAME":"ID Relays 3","GPIO":[0,0,0,0,0,0,224,225,0,0,0,0,0,0],"FLAG":0,"BASE":18}
```

Step 2b.

Once you have found which GPIOs control the relays and LEDs, set these "active" GPIO to associate them with the corresponding `Relay<x>`, `LED<x>`, or `LEDLin` use of inverted (i.e., `Relay<x>i` / `LED<x>i` / `LEDLinki`) component). Bulbs have mainly `PWM`.

For proper operation, in the final device configuration, assignment of like components must begin from `1` and be assigned sequentially! Regular and inverte (e.g., `Relay1`, then `Relay2`; `Led1`, then `Led2i` and so on).

Buttons, Switches, nonPWM Lights or Power Monitoring

Step 1.

Now assign every remaining GPIO (excluding, once again, remaining "dedicated" GPIO like GPIO0/GPIO2 and Tx/Rx, etc.) to successive `Switch1` .. `Switch8` comp
remaining GPIOs assign them as `Counter`)

```
{"NAME":"ID Other","GPIO":[160,352,353,354,161,163,0,0,163,164,165,166,167,0],"FLAG":0,"BASE":18}
```

Step 2.

Save the configuration. Once the device reboots, use the web UI Console to run the Status 8 (sensors) command. This will display the current state of each GPIO
in state `ON` will probably be SM16716 CLK or SM16716 DAT component.

Step 3.

Run `SetOption114 1` to show switch activations in console.

If you have a power monitoring device, when under load the power monitoring chip should trigger switches or counters. Then its just a matter of finding the righ
their combination.

Step 4.

Once you have found which GPIO are connected to each input, change the GPIO setting in the configuration to your input component or use case (f.e. `Button<x>`
may dictate the use of regular or inverted (i.e., `Switch<x>i` / `Button<x>i`) settings. For buttons, you may need to determine whether the internal pull-up is usec
where ***n*** indicates no pull-up.

ESP32

Step 1.

Begin this procedure by disabling power state saves. Some improper GPIO assignments can cause device reboots. Disabling this setting avoids repeated flash v
Tasmota to return to a fail safe state in case of a bad configuration. Ensure that boot loop control is not disabled.

```
Backlog SetOption0 0; SetOption36 1
```

Step 2.

Assign every available GPIO to successive `Relay<x>` components. For the initial GPIO probe, exclude "dedicated" GPIO such as Tx/Rx, etc. You can use a Templat
assignments:

```
{"NAME":"Find Relays","GPIO":
[224,0,225,0,226,227,1,1,228,229,230,231,232,233,234,235,236,237,238,239,240,241,242,243,0,0,0,0,244,245,246,247,248,249,250,251],"F
```

Save the configuration. Once the device reboots, use the virtual buttons on the web UI to find which of the assigned GPIO actually control the physical relays anc
which GPIO act on which device peripheral.

Buttons, Switches, nonPWM Lights or Power Monitoring

Step 1.

Now assign every remaining available GPIO to successive `Switch<x>` components. For the initial GPIO probe, exclude "dedicated" GPIO such as Tx/Rx, etc. You c
these assignments:

```
{"NAME":"Find Switches","GPIO":
[160,0,161,0,162,163,1,1,164,165,166,167,168,169,170,171,172,173,174,175,176,177,178,179,0,0,0,0,180,181,182,183,184,185,186,187],"F
1 | SO65 0;"}
```

If you have a power monitoring device, when under load the power monitoring chip should trigger switches which will display in the console. Then its just a matt
monitoring components and their combination.

Step 4.

Once you have found which GPIO are connected to each input, change the GPIO setting in the configuration to your input component or use case (f.e. `Button<x>`
may dictate the use of regular, inverted or no pulldown settings. For buttons, you may need to determine whether the internal pull-up or pull-down is used or not

Finishing Configuration

Step 1

Once you have determined which GPIO your device uses, set any remaining GPIOs to `None (0)`.

Save the configuration.

Once the device reboots, set the options change back to defaults with:

```
Backlog SetOption0 1; SetOption114 0
```

Step 2.

Submit the new configuration: Since you have now configured a device not previously known to the Tasmota user base, you can export the template and submit it to the Tasmota Supported Devices Repository.

Devices with TuyaMCU

In case your device is a Tuya device with an MCU which controls everything see TuyaMCU for instructions on how to configure it.

Video for a tutorial on this procedure.