ESPHome is a system to create specialized firmware for ESP8266/ESP32 in a declarative way by crafting simple yet powerful configuration files. The resulting firmware can integrate with and be controlled remotely through various smart home automation systems.

This tutorial describes how to build and flash firmware for the Sowillo Box board and connect it to the *Home Assistant* dashboard.

In essence, firmware creation for specific hardware is writing config files in YAML directives with a list of necessary features, also describing the Board Software Package layer. We provide the required files below to serve as a starting point.

Linux distro, please, install before the build:

- Python-3.8.

Windows, please, before the build:

- Install esphome...

First, build steps:

1. install ESPHome utility:

   ```
   $ pip3 install esphome --user
   ```

2. create project.yaml configuration with ESPHome command line wizard:

   ```
   $ esphome project.yaml wizard
   ```

3. a few answers will be needed:

   a. unique device *area* name,

   b. *ESP32* platform,

   c. *esp32dev* board name,

   d. WiFi network SSID name and password, board API password.

4. Now config YAML file is ready for the next step - compilation by command:

   ```
   $ esphome project.yaml compile
   ```

5. No errors should be at console output after command execution - then we could continue to flash device:

   ```
   $ esphome project.yaml run
   ```

6. USB port number should be entered in the proper dialog during ESPHome utility burning the board.

7. The skeleton of basic config values generation is done. Now BSP definitions of the Sowillo Box board should be added at the end of the `project.yaml` file:

```yaml
dallas:
  - pin: GPIO32
    update_interval: 10s
    id: dallas_32

sensor:
  - platform: dallas
    name: temp1
    dallas_id: dallas_32
    index: 0
    resolution: 12
    unit_of_measurement: "°C"
    filters:
      - offset: 0
      - median:
          window_size: 4
          send_every: 1
          send_first_at: 1

  - platform: dallas
    name: temp2
    dallas_id: dallas_32
    index: 1
    resolution: 12
    unit_of_measurement: "°C"
    filters:
      - offset: 0
      - median:
          window_size: 4
          send_every: 1
          send_first_at: 1

  - platform: adc
    pin: GPIO33
    name: "Current"
    attenuation: 11db
    update_interval: 5s
    unit_of_measurement: "A"
    filters:
      - offset: -1.9
      - multiply: 5
      - median:
          window_size: 20
          send_every: 1
          send_first_at: 1
```

```
  # Sensors with general information.
  # Uptime sensor.
  - platform: uptime
    name: Node Uptime

  # WiFi Signal sensor.
  - platform: wifi_signal
    name: WiFi RSSI
    update_interval: 10s

binary_sensor:
  - platform: gpio
    name: "Button"
    pin:
      number: 14
      inverted: True
      mode: INPUT_PULLUP

switch:
  - platform: gpio
    name: "Relay1"
    pin: 13

  - platform: gpio
    name: "Relay2"
    pin: 27

  - platform: template
    name: RF Button 1
    turn_on_action:
      - remote_transmitter.transmit_rc_switch_raw:
          code: '000010110111001010101000011'
          protocol:
            pulse_length: 396
            sync: [1,31]
            zero: [1,3]
            one: [3,1]
            inverted: false
          repeat: 10

#EV1527 codes
remote_receiver:
  pin: GPIO16
  dump:
    - rc_switch
  tolerance: 50%
```

```
  filter: 4us
  idle: 4ms

remote_transmitter:
  pin: GPIO17
  carrier_duty_percent: 100%
```

8. The saved project.yaml file could be recompiled to translate changes to the latest firmware binary file and flashed by one command:

   ```
   $ esphome project.yaml run
   ```

9. Setup Home Assistant server:
   <...>

10. Connect provisioned SowilloBox board with Your tuned firmware to Home Assistance dashboard in simple steps described here:
    https://esphome.io/guides/getting_started_hassio.html

Troubleshooting

This section is currently empty. If you encounter an issue, please *contact us* or open an issue in our bug tracker.