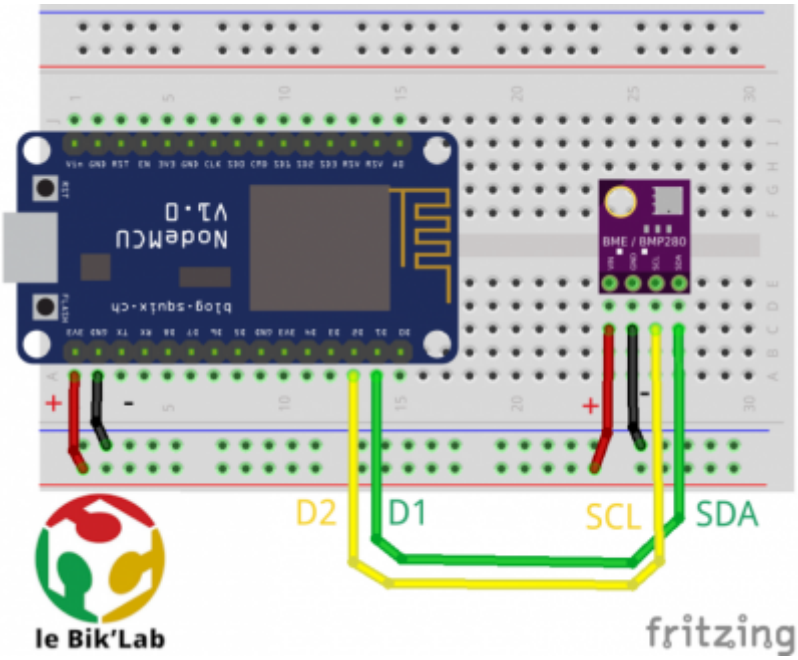
ESPhome / tsl2561 (luminosité)

Montage



Attention à la polarité ! (**+/-**) Schéma placeholder! page en cours d'écriture, seul le code est bon!



Suivre le schéma de montage en respectant les conventions de couleur pour les fils.

Documentation technique

La documentation technique complète est disponible :

- en ligne : Kit DS1820
- en téléchargement (PDF) :

fiche_kit_ds18b20.pdf

Toute cette documentation est diffusée sous licence Creative Commons CC BY 4.0 Deed pour en faciliter la réutilisation.

Code

Code basique

ds18b20simple.yaml

```
esphome:
  name: "station5" # le nom de l'objet connecté

esp32:
  board: nodemcu-32s # ajuster selon la platforme, ok pour nos kits
  framework:
    type: arduino

logger:

# Enable Home Assistant API
api:
  encryption:
    key: "JhwNLgVDiZLAtKsukQRb2//wYz/olZdI/mBx22uX9WA="
    # voir
https://esphome.io/components/api.html#configuration-variables
    # il y a sur la page un générateur de clé aléatoire
    # changez et mettez une autre valeur pour votre noeud

ota:
  password: "secret_ota_password" # changer pour une valeur de votre
choix
  password: "secret_ota_password" # changer pour une valeur de votre
choix

wifi:
  ssid: "wifi_ssid" # nom du réseau wifi
  password: "wifi_password" # mot de passe du réseau wifi

# définition du bus dallas 1 wire :
dallas:
  - pin: GPIO23  # à changer si nécéssaire

sensor:
  - platform: dallas # ajout d'une sonde ds18b20 sur le bus dallas
défini plus haut
  # documentation esphome :
https://esphome.io/components/sensor/dallas.html
    index: 0
    name: "Temperature DS18B20 (station5)"
```

Code avancé

ds18b20adv.yaml

```
substitutions:
  devicename: station5 # nom de l'objet connecté, changer ici
uniquement

esphome:
  name: ${devicename} # la valeur définie plus haut sera placée ici

esp32:
  board: nodemcu-32s # ajuster selon la platforme, ok pour nos kits
  framework:
    type: arduino

logger:

# Enable Home Assistant API
api:
  encryption:
    key: "JhwNLgVDiZLAtKsukQRb2//wYz/olZdI/mBx22uX9WA="
    # voir
https://esphome.io/components/api.html#configuration-variables
    # il y a sur la page un générateur de clé aléatoire
    # changez et mettez une autre valeur pour votre noeud

ota:
  # le mot de passe OTA sera récupéré dans sectets.yaml
  password: !secret ota_password
  # décommenter la ligne ci après pour définir le mot de passe ici
#  password: "wifi_password" # mot de passe du réseau wifi
wifi:
  # le ssid et le password wifi seront récupérés dans sectets.yaml
  ssid: !secret wifi_ssid
  password: !secret wifi_password
  #décommenter pour définir le wifi dans ce fichier, supprimer au
dessus
#  ssid: "wifi_ssid" # nom du réseau wifi
#  password: "wifi_password" # mot de passe du réseau wifi

# définition du bus dallas 1 wire :
dallas:
  - pin: GPIO23  # à changer si nécéssaire

sensor:
  - platform: dallas # ajout d'une sonde ds18b20 sur le bus dallas
défini plus haut
  # documentation esphome :
https://esphome.io/components/sensor/dallas.html
```

```
    index: 0
    name: "Temperature DS18B20 (${devicename})"

  # mesure la force du signal wifi reçu en dB
  # documentation : https://esphome.io/components/sensor/wifi_signal
  - platform: wifi_signal
    name: "signal WiFi (${devicename})"
    update_interval: 10s

  # donne l'utpime (depuis combien de temps l'objet connecté est il
allumé)
  # documentation : https://esphome.io/components/sensor/uptime
  # ici on ajoute des calculs pour avoir un format lisible (j:h:m:s)
  - platform: uptime
    name: "${devicename} Uptime Sensor"
    id: ${devicename}_uptime_sensor
    update_interval: 60s
    on_raw_value:
      then:
        - text_sensor.template.publish:
            id: ${devicename}_uptime_human
            state: !lambda |-
              int seconds =
round(id(${devicename}_uptime_sensor).raw_state);
              int days = seconds / (24 * 3600);
              seconds = seconds % (24 * 3600);
              int hours = seconds / 3600;
              seconds = seconds % 3600;
              int minutes = seconds /  60;
              seconds = seconds % 60;
              return (
                (days ? String(days) + "j " : "") +
                (hours ? String(hours) + "h " : "") +
                (minutes ? String(minutes) + "m " : "") +
                (String(seconds) + "s")
              ).c_str();

text_sensor:
  # on présente l'adresse IP du noeud
  # documentation
https://esphome.io/components/text_sensor/wifi_info.html
  - platform: wifi_info
    ip_address:
      name: "adresse IP (${devicename})"
      id: ${devicename}_ip_address

  - platform: template
  # on présente l'uptime sous forme texte, voir plus haut
    name: "Uptime (${devicename})"
    id: ${devicename}_uptime_human
```

```
    icon: mdi:clock-start

  # version de espHome utilisée
  # https://esphome.io/components/text_sensor/version
  - platform: version
    name: "Version d'ESPHome installée"
    id: ${devicename}_ESPHome_Version
```

From:
https://wiki.lebiklab.fr/ - **Wiki Le BIK'LAB**

Permanent link:
https://wiki.lebiklab.fr/doku.php?id=projets:home-assistant:esphome:noeud-basique-mesures-environnementales:ds18b20&rev=1698374891

Last update: **04/04/2024 15:35**