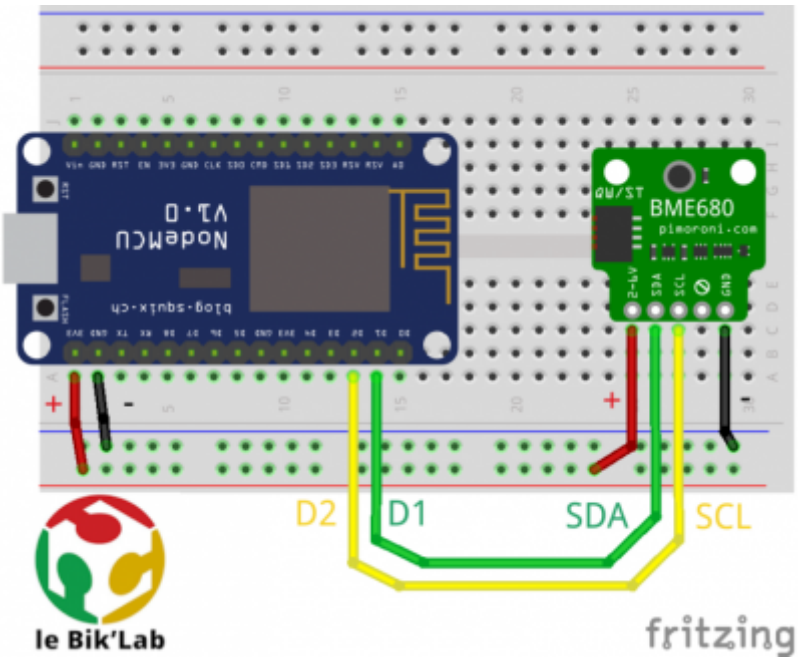
ESPhome / BME 680

Montage

Attention à la polarité ! (**+/-**)



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 BME680
- en téléchargement (PDF) :

fiche_kit_bme680.pdf

Code

Code basique

bme680simple.yaml

```yaml
esphome:
  name: "station2" # 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

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

## on définit les GPIO du bus I2C
i2c:
  sda: D1 # à changer si carte différente (GPIO21 pour esp32)
  scl: D2 # à changer si carte différente (GPIO22 pour esp32)
  scan: True
  id: bus_a

bme680_bsec:
  # documentation esphome : https://esphome.io/components/sensor/bme680_bsec.html
  address: 0x77
  iaq_mode: static
  sample_rate: ulp

sensor:
  - platform: bme680_bsec # multi capteurs atmosphériques (qualité de l'air)
  # documentation esphome : https://esphome.io/components/sensor/bme680_bsec.html
    temperature: # température de l'air
      name: "Temperature (station2)"
```

```
      id: station2_bme680_temperature
      sample_rate: lp
      filters:
        - offset: -2.0
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    pressure: # pression atmosphérique
      name: "Pression atmosphérique (station2)"
      id: station2_bme680_pressure
      sample_rate: lp
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    humidity: # humidité dans l'air
      name: "Hygrométrie (station2)"
      id: station2_bme680_humidity
      sample_rate: lp
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    iaq: # indice de qualité de l'air (calculé)
      name: "indice de qualité de l'air (station2)"
      id: station2_bme680_iaq
      filters:
        - sliding_window_moving_average:
            window_size: 10
            send_every: 10
    co2_equivalent: # taux de CO2 calculé (estimation aproximative)
      name: "taux CO2 équivalent (station2)"
      id: station2_bme680_eco2
      filters:
        - sliding_window_moving_average:
            window_size: 10
            send_every: 10
    breath_voc_equivalent: # cov calculés dans le souffle d'une
personne
      name: "COV équivalents dans le souffle"
      id: station2_bme680_bvoc
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
```

Code avancé

bme680adv.yaml

```yaml
substitutions:
  devicename: station2 # 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


## on définit les GPIO du bus I2C
i2c:
  sda: D1 # à changer si carte différente (GPIO21 pour esp32)
  scl: D2 # à changer si carte différente (GPIO22 pour esp32)
  scan: True
  id: bus_a

sensor:
  - platform: bme680_bsec # multi capteurs atmosphériques (qualité de l'air)
```

```
  # documentation esphome :
https://esphome.io/components/sensor/bme680_bsec.html
    temperature: # température de l'air
      name: "Temperature (${devicename})"
      id: station2_bme680_temperature
      sample_rate: lp
      filters:
        - offset: -2.0
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    pressure: # pression atmosphérique
      name: "Pression atmosphérique (${devicename})"
      id: station2_bme680_pressure
      sample_rate: lp
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    humidity: # humidité dans l'air
      name: "Hygrométrie (station2)"
      id: station2_bme680_humidity
      sample_rate: lp
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20
    iaq: # indice de qualité de l'air (calculé)
      name: "indice de qualité de l'air (${devicename})"
      id: station2_bme680_iaq
      filters:
        - sliding_window_moving_average:
            window_size: 10
            send_every: 10
    co2_equivalent: # taux de CO2 calculé (estimation aproximative)
      name: "taux CO2 équivalent (${devicename})"
      id: station2_bme680_eco2
      filters:
        - sliding_window_moving_average:
            window_size: 10
            send_every: 10
    breath_voc_equivalent: # cov calculés dans le souffle d'une
personne
      name: "COV équivalents dans le souffle (${devicename})"
      id: station2_bme680_bvoc
      filters:
        - sliding_window_moving_average:
            window_size: 20
            send_every: 20

  # 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
```