ESP32 components library with BME280

Links to the component library:

- Component Library -> https://github.com/UncleRus/esp-idf-lib
- Documentation -> https://esp-idf-lib.readthedocs.io/en/latest/
- Examples -> https://github.com/UncleRus/esp-idf-lib/tree/master/examples

Example how to implement Components Library with BME280 sensor:

Create new ESP-IDF project in Eclipse

- Create a new Espressif IDF Project: menu *File -> New -> Espressif IDF Project*.

- Enter a project name, click *Next*

- On the next window leave *the Create a project using one of the templates* unchecked and click on finish. New project with simple application code example will be created.

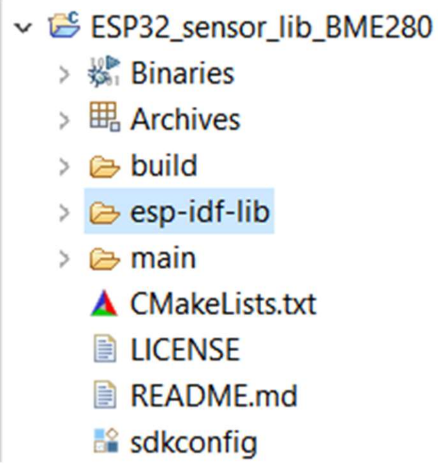
- Build the project.

- Adjust the project configuration:

    - Click on sdkconfig file, SDK Configuration window will open.
        - Serial flasher config -> Flash size: 4 MB (or size in your ESP)
        - Partition Table -> Partition Table: ??? (According the project need)

Add Component Library to the project

- Clone ESP IDF components/sensors library and to the project
    - Copy to the project folder.
    - Refresh project to view new folder.



```
ESP32_sensor_lib_BME280
  Binaries
  Archives
  build
  esp-idf-lib
  main
  CMakeLists.txt
  LICENSE
  README.md
  sdkconfig
```

- Adjust the top level CMakeLists.txt file to include the sensor library component. (*Check Examples folder*)

```
# The following lines of boilerplate have to be in your project's
# CMakeLists in this exact order for cmake to work correctly
cmake_minimum_required(VERSION 3.5)

set(EXTRA_COMPONENT_DIRS esp-idf-lib/components)
include($ENV{IDF_PATH}/tools/cmake/project.cmake)
project(app-template)
```

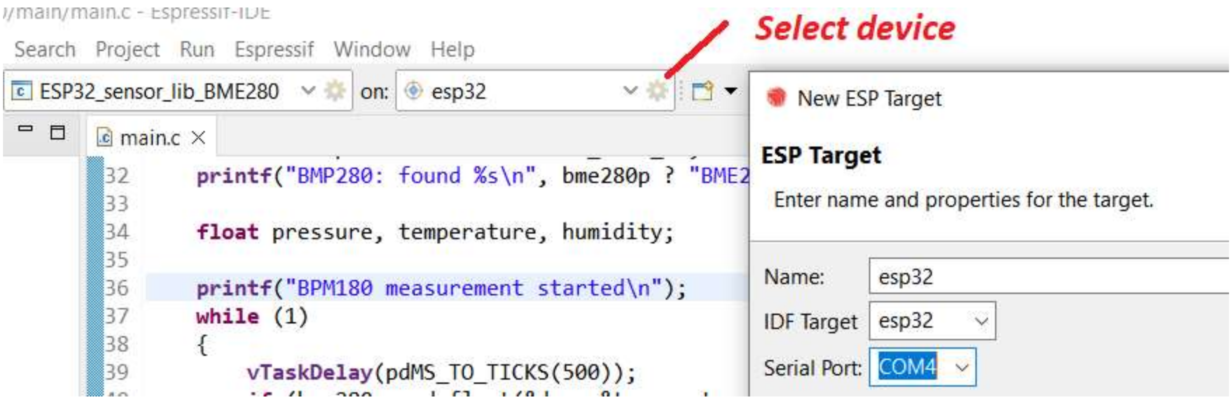
- Adjust the "*Kconfig.projbuild*" file or add one from example to the *'main'* folder.

```
menu "Example configuration"
    config EXAMPLE_I2C_MASTER_SCL
        int "SCL GPIO Number"
        default 5 if IDF_TARGET_ESP8266
        default 6 if IDF_TARGET_ESP32C3
        default 22 if IDF_TARGET_ESP32 || IDF_TARGET_ESP32S2 || IDF_TARGET_ESP32S3
        help
            GPIO number for I2C Master clock line.

    config EXAMPLE_I2C_MASTER_SDA
        int "SDA GPIO Number"
        default 4 if IDF_TARGET_ESP8266
        default 5 if IDF_TARGET_ESP32C3
        default 21 if IDF_TARGET_ESP32 || IDF_TARGET_ESP32S2 || IDF_TARGET_ESP32S3
        help
            GPIO number for I2C Master data line.
endmenu
```

- Adjust application files (*main.c, …*).

```
#include <stdio.h>
#include <stdbool.h>
#include <unistd.h>

#include <inttypes.h>
#include <freertos/FreeRTOS.h>
#include <freertos/task.h>
#include <esp_system.h>
#include <bmp280.h>
#include <string.h>


#ifndef APP_CPU_NUM
      #define APP_CPU_NUM PRO_CPU_NUM
#endif


//BMP280 application start
void bmp280_test(void *pvParameters)
{
    bmp280_params_t params;
    bmp280_init_default_params(&params);
    bmp280_t dev;
    memset(&dev, 0, sizeof(bmp280_t));

    printf("BME280 initialization started\n");
    ESP_ERROR_CHECK(bmp280_init_desc(&dev, BMP280_I2C_ADDRESS_0, 0,
CONFIG_EXAMPLE_I2C_MASTER_SDA, CONFIG_EXAMPLE_I2C_MASTER_SCL));
    ESP_ERROR_CHECK(bmp280_init(&dev, &params));

    bool bme280p = dev.id == BME280_CHIP_ID;
    printf("BMP280: found %s\n", bme280p ? "BME280" : "BMP280");

    float pressure, temperature, humidity;

    printf("BME280 measurement started\n");
    while (1)
    {
        vTaskDelay(pdMS_TO_TICKS(500));
        if (bmp280_read_float(&dev, &temperature, &pressure, &humidity) != ESP_OK)
        {
            printf("Temperature/pressure reading failed\n");
            continue;
        }

        /* float is used in printf(). you need non-default configuration in
         * sdkconfig for ESP8266, which is enabled by default for this
         * example. see sdkconfig.defaults.esp8266
         */
        printf("Pressure: %.2f Pa, Temperature: %.2f C", pressure, temperature);
        if (bme280p)
            printf(", Humidity: %.2f\n", humidity);
        else
            printf("\n");
    }
}


//Main application start
void app_main(void)
{
      printf("BME280 test application\n");

      ESP_ERROR_CHECK(i2cdev_init());
      xTaskCreatePinnedToCore(bmp280_test, "bmp280_test", configMINIMAL_STACK_SIZE *
15, NULL, 5, NULL, APP_CPU_NUM);
}
```

- Build the project, select the right device and upload application to the ESP32



```
    printf("BMP280: found %s\n", bme280p ? "BME2

    float pressure, temperature, humidity;

    printf("BPM180 measurement started\n");
    while (1)
    {
        vTaskDelay(pdMS_TO_TICKS(500));
```

ESP Target

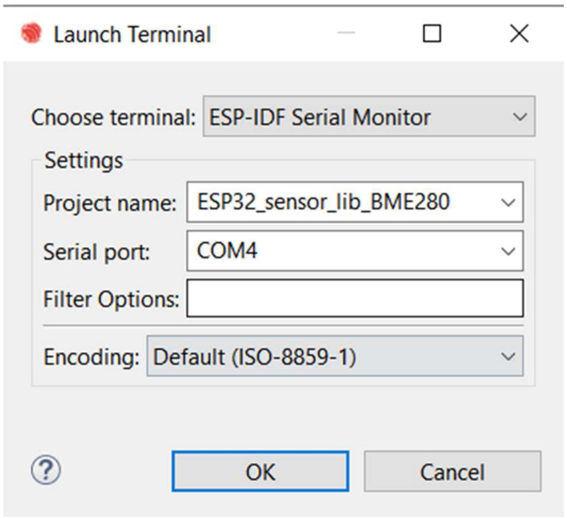
Enter name and properties for the target.

Name: esp32

IDF Target: esp32

Serial Port: COM4

- Open terminal window *Window -> Show View -> Terminal* . In the opened window, select "*Open a Terminal*" icon. Select *Terminal – ESP-IDF Serial Monitor*, *Project name*, *Serial Port*. Click OK. Results in the terminal window should appear.



Choose terminal: ESP-IDF Serial Monitor

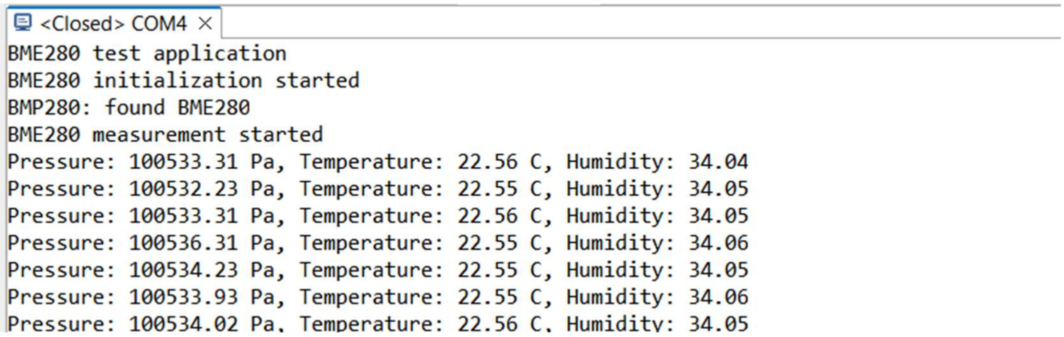
Settings

Project name: ESP32_sensor_lib_BME280

Serial port: COM4

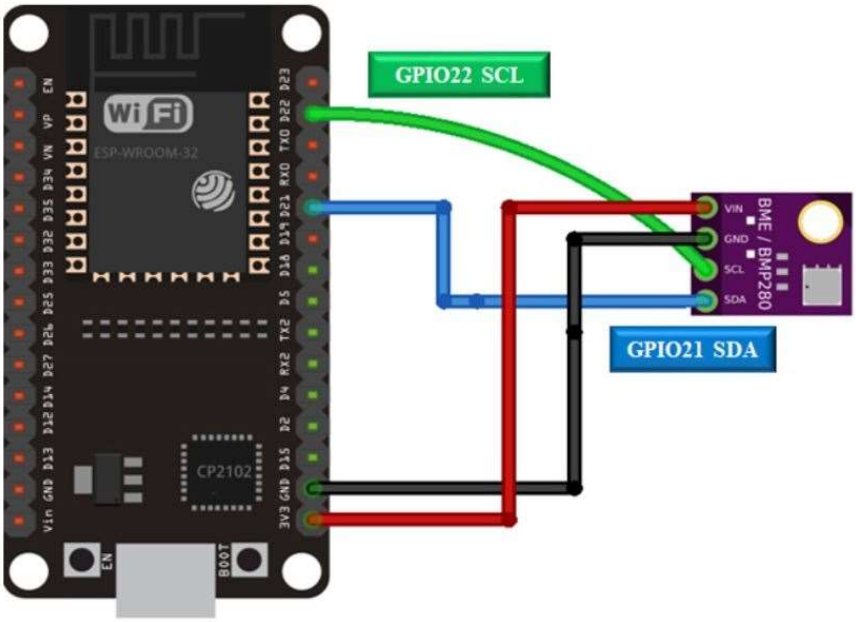
Filter Options:

Encoding: Default (ISO-8859-1)



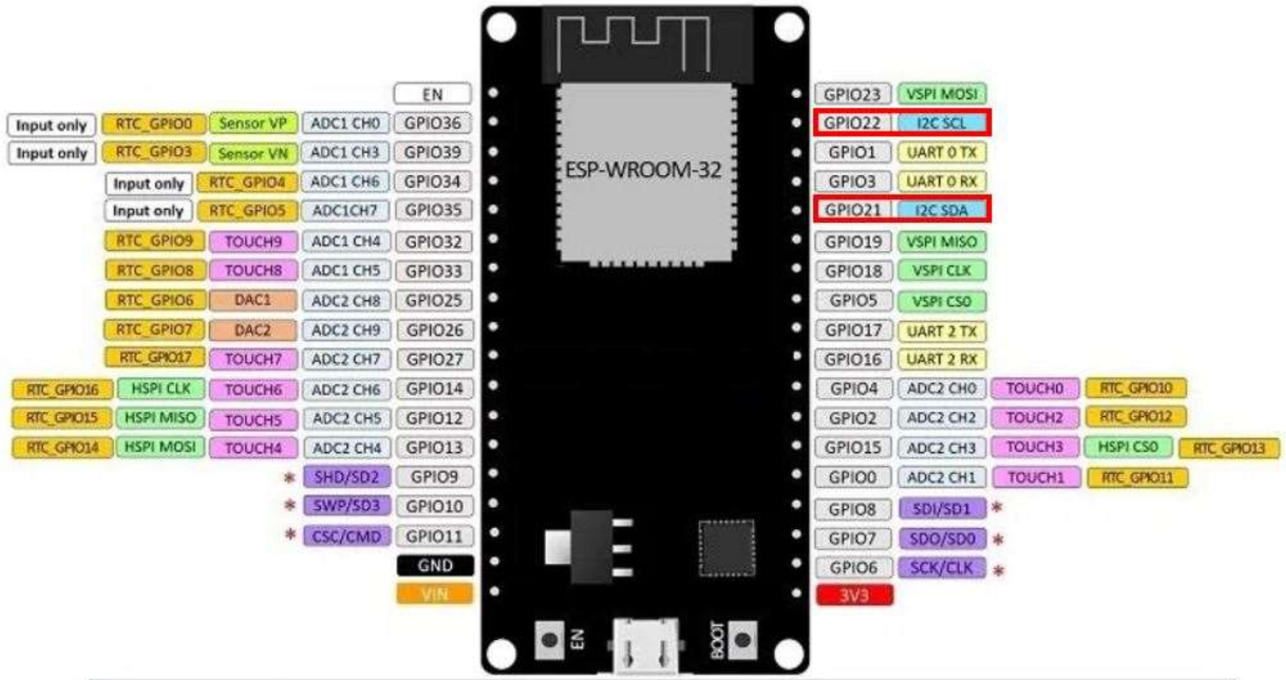
```
BME280 test application
BME280 initialization started
BMP280: found BME280
BME280 measurement started
Pressure: 100533.31 Pa, Temperature: 22.56 C, Humidity: 34.04
Pressure: 100532.23 Pa, Temperature: 22.55 C, Humidity: 34.05
Pressure: 100533.31 Pa, Temperature: 22.56 C, Humidity: 34.05
Pressure: 100536.31 Pa, Temperature: 22.55 C, Humidity: 34.06
Pressure: 100534.23 Pa, Temperature: 22.55 C, Humidity: 34.05
Pressure: 100533.93 Pa, Temperature: 22.55 C, Humidity: 34.06
Pressure: 100534.02 Pa, Temperature: 22.56 C, Humidity: 34.05
```

The BME/BMP280 sensor connection to the ESP32 shown in the following schematic diagram.



ESP32 DEVKIT V1 – DOIT

version with 36 GPIOs


ESP-WROOM-32
EN
Input only RTC_GPIO0 Sensor VP ADC1 CH0 GPIO36
Input only RTC_GPIO3 Sensor VN ADC1 CH3 GPIO39
Input only RTC_GPIO4 ADC1 CH6 GPIO34
Input only RTC_GPIO5 ADC1CH7 GPIO35
RTC_GPIO9 TOUCH9 ADC1 CH4 GPIO32
RTC_GPIO8 TOUCH8 ADC1 CH5 GPIO33
RTC_GPIO6 DAC1 ADC2 CH8 GPIO25
RTC_GPIO7 DAC2 ADC2 CH9 GPIO26
RTC_GPIO17 TOUCH7 ADC2 CH7 GPIO27
RTC_GPIO16 HSPI CLK TOUCH6 ADC2 CH6 GPIO14
RTC_GPIO15 HSPI MISO TOUCH5 ADC2 CH5 GPIO12
RTC_GPIO14 HSPI MOSI TOUCH4 ADC2 CH4 GPIO13
* SHD/SD2 GPIO9
* SWP/SD3 GPIO10
* CSC/CMD GPIO11
GND
VIN
GPIO23 VSPI MOSI
GPIO22 I2C SCL
GPIO1 UART 0 TX
GPIO3 UART 0 RX
GPIO21 I2C SDA
GPIO19 VSPI MISO
GPIO18 VSPI CLK
GPIO5 VSPI CS0
GPIO17 UART 2 TX
GPIO16 UART 2 RX
GPIO4 ADC2 CH0 TOUCH0 RTC_GPIO10
GPIO2 ADC2 CH2 TOUCH2 RTC_GPIO12
GPIO15 ADC2 CH3 TOUCH3 HSPI CS0 RTC_GPIO13
GPIO0 ADC2 CH1 TOUCH1 RTC_GPIO11
GPIO8 SDI/SD1 *
GPIO7 SDO/SD0 *
GPIO6 SCK/CLK *
3V3
EN
BOOT

* Pins SCK/CLK, SDO/SD0, SDI/SD1, SHD/SD2, SWP/SD3 and SCS/CMD, namely, GPIO6 to GPIO11 are connected to the integrated SPI flash integrated on ESP-WROOM-32 and are not recommended for other uses.