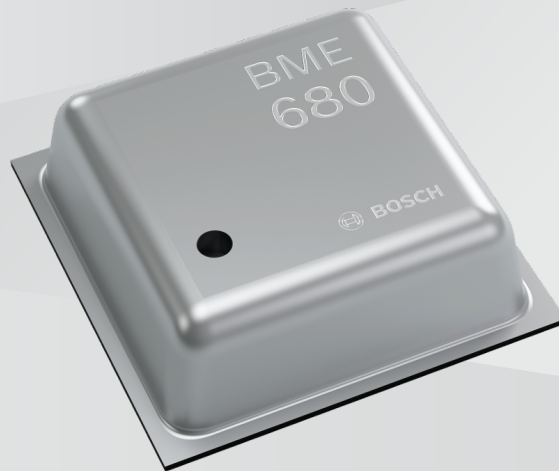


Arduino Integration Guide

Bosch Software Environmental Cluster (BSEC)



Contents

1	Instructions for using the BSEC Library in Arduino 1.8.19	3
----------	--	----------

1 Instructions for using the BSEC Library in Arduino 1.8.19

Until the Arduino IDE natively supports pre-compiled libraries, the following steps/hacks will need to be followed to integrate the BSEC library into your project.

Installation and getting started

1. Install the latest Arduino IDE

As of this publication, the latest Arduino IDE 1.8.19 can be downloaded from this [link](#)

2. Install the BSEC and Bme680_Data library

Either download these libraries as zip and import BSEC.zip and Bme680_Data.zip into the Arduino IDE. Refer to [this](#) guide on how to import libraries.

3. Modify the platform.txt file

If you have already used the previous example code and hack guide, remove the linker flag `-libalgobsec` in the `platform.txt` file found in `hardware\esp8266\3.0.2\platform.txt` and reference to the `compiler.c.elf`. ↔ `extra_flags`.

The standard arduino-builder now passes the linker flags under `compiler.libraries.ldflags`. Most `platform.txt` files do not already include this new optional variable. You will hence need to declare this variable's default and add it to the end of the combine recipe. It is recommended to declare it in the following section like below,

```
# These can be overridden in platform.local.txt
compiler.c.extra_flags=
compiler.c.elf.extra_flags=
compiler.S.extra_flags=
compiler.cpp.extra_flags=
compiler.ar.extra_flags=
compiler.objcopy.eep.extra_flags=
compiler.elf2hex.extra_flags=
compiler.libraries.ldflags=-lalgobsec
```

and add it in the combine recipe like the below example

ESP8266 community forum's ESP8266 core

Original line [122](#),

```
## Combine gc-sections, archives, and objects
recipe.c.combine.pattern="{compiler.path}{compiler.c.elf.cmd}" {build.exception_flags} -Wl,-Map
"-Wl,{build.path}/{build.project_name}.map" {compiler.c.elf.flags} {compiler.c.elf.extra_flags} -o
"{build.path}/{build.project_name}.elf" -Wl,--start-group {object_files} "{archive_file_path}" {compiler.c.elf.libs}
-Wl,--end-group "-L{build.path}"
```

should become

```
## Combine gc-sections, archives, and objects
recipe.c.combine.pattern="{compiler.path}{compiler.c.elf.cmd}" {build.exception_flags} -Wl,-Map
"-Wl,{build.path}/{build.project_name}.map" {compiler.c.elf.flags} {compiler.c.elf.extra_flags} -o
"{build.path}/{build.project_name}.elf" -Wl,--start-group {object_files} "{archive_file_path}" {compiler.c.elf.libs}
{compiler.libraries.ldflags} -Wl,--end-group "-L{build.path}"
```

4. Only for the ESP8266 - modify the linker script

This step needs to be done for BSEC example code as it has an interface with library and it need not be done for bme680 data logging example code. Due to the current size of the BSEC library, upon compilation, you will receive an error: section '.text' will not fit in region 'iram1_0_seg'. In order to solve this, you will need to modify the linker script and specifically define where the library should be placed in memory.

You will need to modify the file eagle.app.v6.common.ld typically found in {YourESP8266PPackage\Directory}\tools\sdk\ld.

With reference to the linker script [here](#),

After line 208 `*libwps.a:(.literal.* .text.*)`, add `*libalgobsec.a:(.literal.* .text.*)`, which should look like,

```
*libupgrade.a:(.literal.* .text.*)
*libwpa.a:(.literal.* .text.*)
*libwpa2.a:(.literal.* .text.*)
*libwps.a:(.literal.* .text.*)
*libalgobsec.a:(.literal.* .text.*)
```

5. Copy the binaries

If you have already used the previous example code remove the libalgobsec.a file from the core directory or any other location you might have copied it to and instead, copy the file algo\normal_↵ version\bin\esp\esp8266\libalgobsec.a from the BSEC package into where the Arduino library is installed on your system typically hardware\esp8266\3.0.2\tools\sdk\lib folder.

6. Verify and upload the example code

Start or restart the Arduino IDE.

Open the example code found under File>Examples>Bsec software library>basic for BSEC or open the example code found under File>Examples>BME680 datalogging for bme680 data logging.

Select your board and COM port. Upload the example. Open the Serial monitor. You should see an output on the terminal.

7. Tested board list

The current list of tested micro-controllers include,

Core MCU	Tested boards
atmega2560	Arduino MEGA 2560
cortex-m0plus	Arduino Zero
cortex-m3	Arduino Due
cortex-m4f	Adafruit BlueFruit NRF52 Feather, STM32 Nucleo F411RE
esp32	Sparkfun ESP32 Thing
esp8266	Adafruit Feather HUZZAH