Lab 3 - External Sensors

Welcome to lab 3! In this lab you will learn how to use Zephyr to talk to external hardware components.

Requirements and Setup

For this lab, you will need a working Zephyr installation. The setup is the same as for lab 2, so there are no additional components that need to be installed.

On the hardware side you will need the BME680 board and the Qwiic-Grove connector cable from your lab kit. In the standard configuration, the first I2C bus is routed to Grove connector 3 on the Maker Pi Pico board.

Resources

You may find the following resources helpful during this lab:

- The BME680 datasheet (in the `/docs` folder inside the template)
- Zephyr I2C documentation
- Zephyr Sensor API documentation
- Zephyr Sensor examples
- If you have never worked with I2C before you can have a look at this tutorial

Template

You can download the template from Studium.

Building the application

To build an application you use the **west** command line tool. A link to its documentation is given in the section above. In general, to build an application you have to navigate to the folder that holds the application's **CMakeLists.txt** file and issue

```
west build -b rpi_pico2/rp2350a/m33
```

This will automatically create a `/build` folder in the current directory and start compiling your application. West has many optional flags (as exemplified with `-b rpi_pico2/rp2350a/m33`) so it is worth looking into the documentation which ones might be interesting for you!

If west fails to build your project with an error about a missing “build” command you probably have not set all required environment variables. This can usually be solved by sourcing the *zephyr-env.sh* (Linux, macOS) or *zephyr-env.cmd* (Windows) file in the `/zephyrproject/zephyr/` folder.

You can program your board as you did in lab 1 by copying the resulting uf2 file (located at `/build/zephyr/zephyr.uf2`) onto the USB device that shows up when you start your board in boot mode.

You can also use **west flash** if you have the debugging setup connected.

Part 1

In the first part of the lab you shall read out the temperature from the sensor every 3 seconds by directly issuing the required I2C commands, and print it out to the console. From the sensor’s datasheet you can find how to configure the sensor for the desired measurement, where to locate the results and how to process them.

Hints:

- In the template, there is header file called `bme680_reg.h`, which already holds the register addresses of the most important registers in the sensor
- The template includes code that initializes a device pointer to the I2C0 peripheral and reads the Chip ID register
- You only need to read the temperature, not any other value from the sensor (you can of course experiment with the other sensing channels if you have the time)
- The data you get from the sensor is more or less the raw reading from the sensor’s internal ADC, so you will have to convert that into a temperature value. The formula for doing that is in the datasheet

Part 2

In this part of the lab you will again read out the temperature from the BME680 every 3 seconds, but shall make use of the Zephyr Sensor API. The choice is yours whether to use the Fetch and Get API or the more recent Read and Decode API.

Hints:

- You will want to update the device tree overlay `rpi_pico2_rp2350_m33.overlay` to include the BME680 sensor. You can look at the Zephyr examples to see how it is done for other boards.