I²C basics.

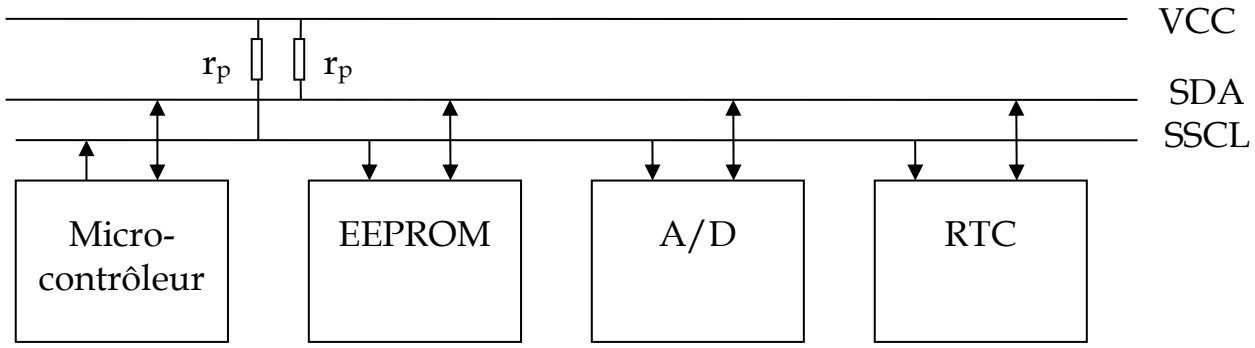
I²C is the acronym for Inter-Integrated Circuit (bus).

This bus was developed in the 80s by Philips Semiconductors to allow interfacing the control circuits of a television in a standard way with a microcontroller. This bus is sometimes called TWI (Two Wire Interface) or simply Wire. Currently, this bus is used with a large number of circuits (RTC, flash memory, sensors, displays…)

The bus uses two connections called SDA (Serial Data Line) and SCL (Serial Clock).

For these lines the **0 level is dominant**, which corresponds to open-drain transistors controlling the output pins. A pull-up resistor is needed on each line to fix the "idle" state (level 1).

The simplest configuration has a master circuit (the microcontroller) and some slave circuits.



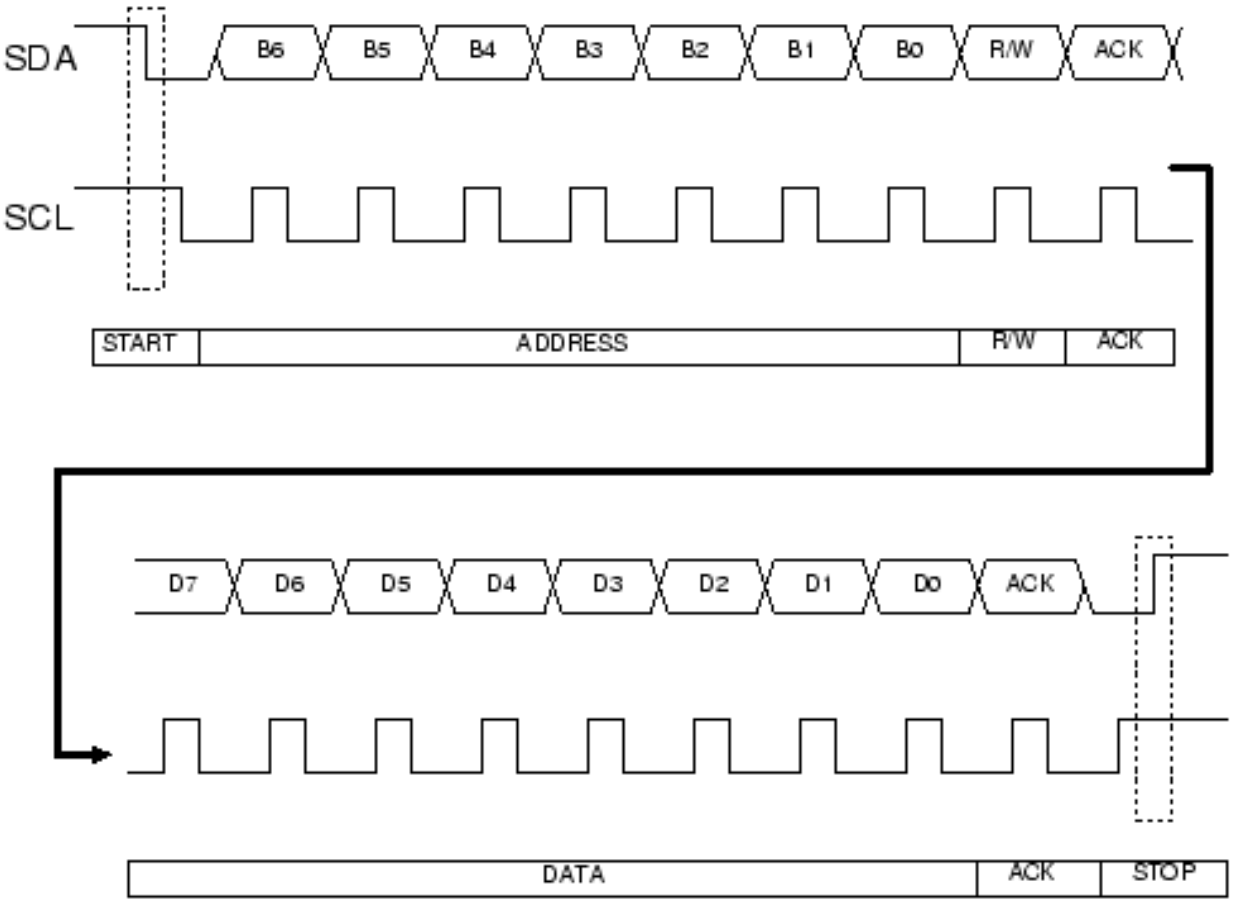
A message generated by the processor consists of a START condition, followed by the physical address of the slave concerned by the communication and its direction.

The slave address is coded on 7 bits followed by a 0 if the following transaction is a write or by a 1 if we need a read operation.

Next comes an acknowledgment bit generated by the slave (0 if all is OK), then the first data item sent by the processor or by the slave depending on the direction of transmission followed by an acknowledgment generated by the receiver, then the next byte, and so on.

When all the bytes are sent a STOP condition is issued.

The clock (SCL) frequency is typically 400 kHz, and data bits can only change when clock is low (otherwise it will be interpreted as a SART or STOP condition).

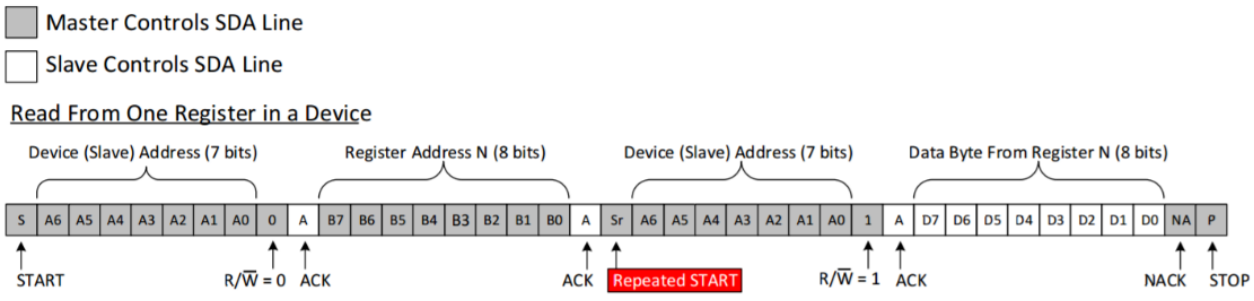


More advanced modes of communication are possible with some circuits, for instance a 10 bits address mode.

In addition to the physical address of the circuit which allows the selection of the component, it is also necessary to carry out an addressing of the internal register of the circuit to which we want to access.

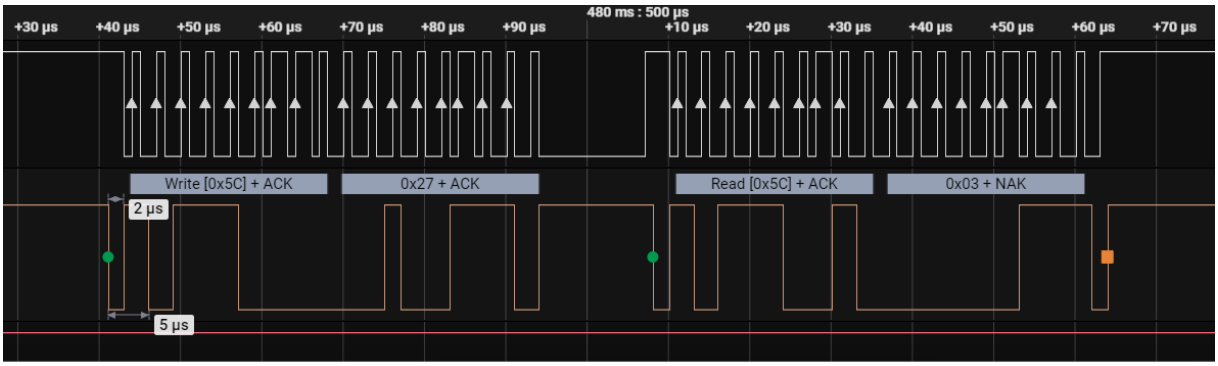
This happens in two steps: writing of the address of the internal register in the slave and then issuing a read or write operation.

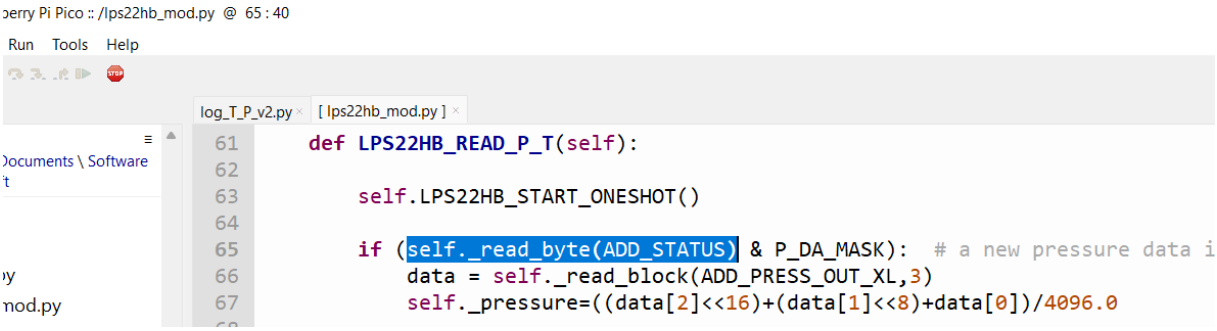
It can be done with two separate transactions or with a single transaction with a repeated START (see the example below).



In the following example, two transactions are used to read the internal register at address 0x27 of a temperature and pressure sensor LPS22HB (physical address 0x5C).

The negative acknowledge (NAK) is used by the master to tell the slave to stop sending bytes.





The highlighted part of this MicroPython program corresponds to the diagram.

See here for the detailed code:
https://github.com/pcamus/embedded-sensors/tree/main/Pico_IMU_10DOF/my_IMU_code