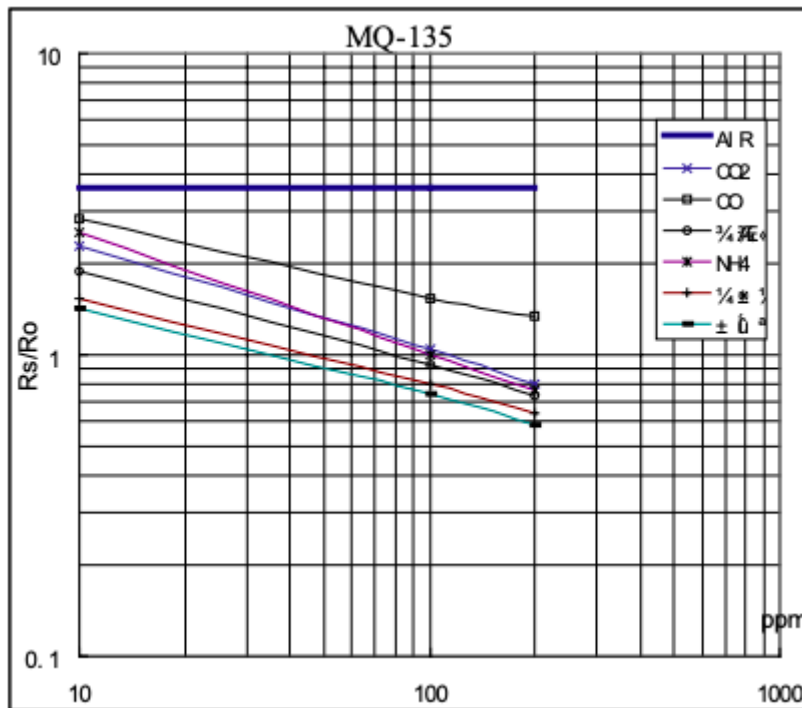


# M\_Q135.h

\*Rzero is the value of the resistor in the MQ-135 sensor in the presence of clean air. Where the CO<sub>2</sub> value is around 397.13 ppm as given below. The Rzero value varies for each individual sensor and can be found using another Arduino [code](#)

- RLOAD meanwhile is the external resistor value that is connected to the MQ-135 sensor, which in this case is 10kΩ.
- Important parameters are for calculating CO<sub>2</sub> of 116.6 and 2.79 are taken from a calibration graph as seen from the [datasheet](#). This shows the sensitivity of the sensor varies for each gas.



## M\_Q135.h

```
<code>
/*****
****/
/*!
@file    MQ135.h
@author  G.Krocker (Mad Frog Labs)
@license GNU GPLv3
@section HISTORY

v1.0 - First release
*/
/*****
****/
#ifndef MQ135_H
#define MQ135_H
#if ARDUINO >= 100
```

```
#include "Arduino.h"
#else
#include "WProgram.h"
#endif

/// The load resistance on the board
#define RLOAD 10.0
/// Calibration resistance at atmospheric CO2 level
#define RZERO 65 //Dependent on the sensor
/// Parameters for calculating ppm of CO2 from sensor resistance
#define PARA 116.6020682
#define PARB 2.769034857

/// Parameters to model temperature and humidity dependence
#define CORA 0.00035
#define CORB 0.02718
#define CORC 1.39538
#define CORD 0.0018

/// Atmospheric CO2 level for calibration purposes
#define ATMOCO2 400

class MQ135 {
private:
uint8_t _pin;

public:
MQ135(uint8_t pin);
float getCorrectionFactor(float t, float h);
float getResistance();
float getCorrectedResistance(float t, float h);
float getPPM();
float getCorrectedPPM(float t, float h);
float getRZero();
float getCorrectedRZero(float t, float h);
};
#endif
</code>
```

[Back to report](#)

From:

<https://student-wiki.eolab.de/> - **HSRW EOLab Students Wiki**

Permanent link:

<https://student-wiki.eolab.de/doku.php?id=amc2021:group!extras:mq-135:start&rev=1630810004>

Last update: **2023/01/05 14:38**

