|  |  |
| --- | --- |
| Mark |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Team name: | *A2* | | |
| Homework number: | *07* | | |
| Due date: | 03/11/2024 | | |
|  |  |  |  |
| Contribution | NO | Partial | Full |
| La Barbera Marco |  |  | *x* |
| Lotto Giulio |  |  | *x* |
| Majocchi Tommaso |  |  | *x* |
| Maffezzini Andrea |  |  | *x* |
| Pompilio Matteo |  |  | *x* |
| Notes: none | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Project name | I2C thermometer | | |
| Not done | Partially done  (major problems) | Partially done  (minor problems) | Completed |
|  |  |  | *x* |
| We successfully completed the homework.  Next, we will explain all the steps for accomplishing our goals:  **Part 1b:**  First of all, we configure the board pinout for the I2C transmission with the temperature sensor (pins PB8 and PB9 as verified in the schematic file) and for the UART transmission with the pc, pin PA2:    Then, from “connectivity*”* we enable the I2C communication in this way:    And the UART for the remote transmission of the temperature value to the matlab console, in DMA mode as always:    Finally, we set the timer with the usual values (timeout parametrized by the constant TEMPO):    Then in the “main.c” file we initialize the TIM2 base generation and the I2C transmission:    For this purpose, we declare two constants (timeout and size) and two global variables: LM75\_WR\_ADDRESS and LM75\_REGISTER, respectively the address of the sensor and the address of the register where the converted temperature is stored. LM75\_RD\_ADDRESS will be used in the HAL\_I2C\_Master\_Receive function. The lsb bit of these two similar addresses distinguishes the write from the read request.       * *TEMPO* will dictate the frequency of the timer; * *data*; * *data\_old*; * ;   The *snprintf()* function, given *string* and itssize *string\_lenght,* parses the specified values in the specified formats in the buffer. Finally, we transmit the buffer with Direct Memory Access through our *uart2* interface, with its characters’ length.  Switching to MATLAB, we can now run the script “UART\_read\_data.m” to read the voltage values at a baud rate of 115200 bps (as set on our board). We successfully receive the data on our console: | | | |
| Professor comments: | | | |