|  |  |
| --- | --- |
| **Mark** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Team name: | *B01* | | |
| Homework number: | *HOMEWORK 6* | | |
| Due date: | 14/11/2023 | | |
|  |  |  |  |
| Contribution | NO | Partial | Full |
| Paolo Salvatore Galfano |  |  | *x* |
| William Stucchi |  |  | *x* |
| Giada Silvestrini |  |  | *x* |
| Francesco Scroccarello |  |  | *x* |
| Francesco Maria Tranquillo |  |  | *x* |
| Notes: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Project name | Temperature sensor and Accelerometer | | |
| Not done | Partially done  (major problems) | Partially done  (minor problems) | Completed |
|  |  |  | *x* |
| Explanation:  We successfully completed the homework.  **Part a:**  In the GUI: 1. Set pins PB8 and PB9 to I2C1\_SCL and I2C1\_SDA, respectively 2. Enabled I2C1 and the event interrupt in the NVIC tab 3. Enabled Timer 1 by setting Internal Clock as source, and by setting the prescaler to 9999 and the counter period to 8399, to make it trigger every one second.  Software: In the main function, we used “HAL\_I2C\_Master\_Transmit” to select the register 0x00 (temperature register). However, as we have seen by removing it, this is not needed. Then, we started the timer 1 with interrupts, by calling “HAL\_TIM\_Base\_Start\_IT”.  We have defined the timer callback in which we start the I2C receive every one second by calling “HAL\_I2C\_Master\_Receive\_IT”. In the I2C interrupt callback, we read the data written though I2C and we split it into MSB and LSB. We summed the MSB (the integer part) to the MSB of the LSB (the decimal part) multiplied by 0.5. We then printed the converted temperature with “HAL\_UART\_Transmit”.  **Part b**:  In the GUI: 1. Set pins PB8 and PB9 to I2C1\_SCL and I2C1\_SDA, respectively 2. Enabled I2C1 3. We enabled DMA for USART2 with request USART2\_TX, and also the USART global interrupt in the NVIC tab. 4. Enabled Timer 1 by setting Internal Clock as source, and by setting the prescaler to 9999 and the counter period to 8399, to make it trigger every one second.  Software: First, we wrote the code to check which address is assigned to the accelerometer like we have seen in the lab lecture. We also configure three registers of the accelerometer at addresses: 1. 0x20, value 00010111  2. 0x21, value 00000000 3. 0x23, value 00000000 using the documentation as reference.  Then, we started the timer 1 with interrupts, by calling “HAL\_TIM\_Base\_Start\_IT”.  We defined the callback “HAL\_TIM\_PeriodElapsedCallback” in which we set a flag to 1.  In the main while loop, if the flag is 1, we reset the flag and read the x, y and z accelerations in the following way: 1. Transmit the subaddress of the register (0x29, 0x2B and 0x2D for X, Y and Z)  2. Receive the value of the acceleration and store it into a local, signed 8-bit integer variable 3. Divide the value read by 64 storing it in a float variable 4. Transmit the string with the acceleration values with DMA by calling “HAL\_UART\_Transmit\_DMA”  **Part c**: Starting from part b, in the GUI: 1. We enabled DMA for I2C1\_RX requests and the event interrupt in the NVIC tab  Software: In the main function, we determined the accelerator address and setup the three registers like in part b. In the “HAL\_TIM\_PeriodElapsedCallback”: 1. We transmit 0x29 with the MSB set to 1 (i.e. 0xA9) to enable the multiple read protocol 2. We call “HAL\_I2C\_Master\_Receive\_DMA”, passing as destination the address of a 5-bytes buffer.  In the “HAL\_I2C\_MasterRxCpltCallback”, we read the three acceleration values at index 0, 2 and 4 and convert it to g values and sending them to the terminal like in part b. | | | |
| Professor comments: | | | |