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

|  |  |  |  |
| --- | --- | --- | --- |
| Team name: | *A2* | | |
| Homework number: | *HOMEWORK 06* | | |
| Due date: | *12/11/23* | | |
|  |  |  |  |
| Contribution | NO | Partial | Full |
| Hui Jiang |  |  | *x* |
| Mattia Sironi |  |  | *x* |
| Gabriele Landi |  |  | *x* |
| Arturo Caliandro |  |  | *x* |
| Luigi Lizzini |  |  | *x* |
| Notes:  none | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Project name | I2C temperature sensor with interrupt | | |
| Not done | Partially done   (major problems) | Partially done   (minor problems) | Completed |
|  |  |  | *x* |

|  |
| --- |
| We have done the whole homework, here is our explanation:  Firstly, we have configured the board as shown below:  We have set the Pins PB8 and PB9 to I2C1\_SCL, I2C1\_SDA respectively.(by default, the USART2 is already configured).  Then we enabled the I2C1:  After doing that, we enabled the timer2 as shown below:  Where Prescaler and Counter Period are set to 8399 and 9999 respectively as (8399+1)\*(9999+1)/83e6 = 1  Later, we went to the NVIC table in order to enable the TIM2 global interrupt:  Now we have finished all the configurations in the GUI, then we went to main.c,firstly, we defined two global variables in order to write the two HAL I2C Master functions easily:  Then in the main, we started the I2C and Timer2 in a safe way:  Then, we created the callback function as shown below:  In the callback function, we first define a series of variables that we will use them later(and we will explain some of them later). Then we call the HAL\_I2C\_Master\_Receive. But you may notice in the function the number of bytes we are going to receive is 6 instead of 2. The reason we do this is to resolve the bug that is proponed in the slides:  In order to explain this bug and how we solve it, we need to clarify one thing:  When the LM75B is accessed the conversion in process is not interrupted (that is, the I2C-bus section is totally independent of the Sigma-Delta converter section) and accessing the LM75B continuously without waiting at least one conversion time between communications will not prevent the device from updating the Temp register with a new conversion result. The new conversion result will be available immediately after the Temp register is updated.  The above paragraph we copied from the LM75B.pdf, the important thing is when the sensor is updating the result in the register, the I2C can still access the Temp register. Therefore, if we access the Temp register when the sensor is updating the result, we may encounter something weird like in the slides, the microcontroller read 26,26.875,25.875: where we noticed that the second one has a integer part equal to the integer part of the first one but the decimal part of the second part is equal to the second part of the third one, thus when the microcontroller was reading the second one via the I2C, the sensor is doing the conversion, and just have changed the decimal part but not the integer part and when the microcontroller was reading the third one, the conversion has done. That is the bug we need to solve.  In order to solve the bug, we read 3 values instead of 1. We now back to the code: if the first value read and the second value read is the same, means we don’t meet the bug, so we just transfer the read value in the right form: first, the variable temperature\_final is int16\_t so the compiler can recognize the two’s complement. Then we just move the first element of the datas\_temperature(the integer part of the fisrt temperature read) 8 positions to right(igual to datas\_temperature[0] 00000000) and put the second element of the datas\_temperature(the decimal part of the fisrt temperature read) to the least 8 significant bits of the variable temperature\_final using the logic operation “or”. Otherwise, the fisrt and second read temperature are different which means we encounter the bug. So we just put the third read temperature in the temperature\_final. Which according to the conversion time description:  The LM75B performs the temperature-to-data conversions with a much higher speed than the LM75A. While the LM75A takes almost the whole of conversion period (Tconv) time of about 100 ms to complete a conversion, the LM75B takes only about 1⁄ 10 of the period, or about 10 ms. Therefore, the conversion period (Tconv) is the same, but the temperature conversion time (tconv(T)) is different between the two parts. A shorter conversion time is applied to significantly reduce the device’s average power dissipation. During each conversion period, when the conversion is completed, the LM75B becomes idled and the power is reduced, resulting in a lesser average power consumption.  The third read temperatura is for sure the correct new result. Then we divided by 256.0 in order to change it to degree.  Finally, if something wrong with the HAL\_I2C\_Master\_Receive, we send a message related to this using the UART.  And the code works as we expected:  Where we have put our finger on the sensor for a while. |
| Professor comments: |