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

|  |  |  |  |
| --- | --- | --- | --- |
| Team name: | *B01* | | |
| Homework number: | *HOMEWORK 9* | | |
| Due date: | 5/12/2023 | | |
|  |  |  |  |
| Contribution | NO | Partial | Full |
| Francesco Scroccarello |  |  | *x* |
| Paolo Salvatore Galfano |  |  | *x* |
| William Stucchi |  |  | *x* |
| Giada Silvestrini |  |  | *x* |
| Francesco Maria Tranquillo |  |  | *x* |
| Notes: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Project name | Keyboard | | |
| Not done | Partially done  (major problems) | Partially done  (minor problems) | Completed |
|  |  |  | *x* |
| Explanation:  We successfully completed the homework.  9b) From the GUI, we activated the pins PC2, PC3, PC12, PC13 as GPIO\_Input, and PC8, PC9, PC10, PC11 as GPIO\_Output; moreover, we enabled TIM1 with clock\_source Internal Clock, prescaler at 8400-1 and counter period to 10-1 (to have a final frequency of 1000 Hz).  In the main code, we defined a structure “keyboard\_t” containing:   * Value: the character to be printed * State: the GPIO\_PinState * Pressed: binary variable (1 if the corresponding button has been pressed, 0 otherwise) * Printed: binary variable (1 if the corresponding button has been printed, 0 otherwise)   Consequently, we defined a matrix of “keyboard\_t” of dimension 4x4 that we used to map each keyboard button, setting the value to be printed to the corresponding character, and initializing the other 3 variables.  After that, we redefined the timer callback by setting a flag “activate” to 1.  In the while loop, firstly we set the pin of the selected column to be read; then, when “activate” flag is 1, we read the states of the row pins (in particular, we read them in the following order: 12, 13, 2, 3; because we observed that the sequence correctly maps our keyboard). We now use the pressed and printed variables to keep track of which button is pressed at runtime, if it has already been printed (because we want to print the character just once per click, without printing it multiple times while we keep the button pressed). Note that, to allow multiple buttons to be pressed simultaneously and their values printed, we included the pressed and printed variables in the struct Keyboard\_t, such that each button can be independently controlled. | | | |
| Professor comments: | | | |