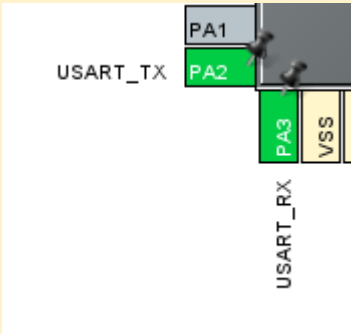


Mark	/11
------	-----

Team name:	A5		
Homework number:	HOMEWORK 04		
Due date:	13/10/2024		
Contribution	NO	Partial	Full
Alessio Spineto			x
Riccardo Lamarca			x
Sofia Cecchetto			x
Annamaria De Togni			x
Emma Crespi			x
Notes: none			

Project name	USART DMA + LCD		
Not done	Partially done (major problems)	Partially done (minor problems)	Completed
<p>Part 1a: Complete the UART project with Direct Memory Access functions.</p> <p>The two USART pins are enabled by default.</p>  <p>In the connectivity section of the GUI, we selected USART2, opening its 'Mode and Configuration' page. There, in the DMA settings, we added a new DMA request, selecting the transfer mode USART2_TX.</p>			

USART2 Mode and Configuration

Mode

Mode: Asynchronous

Hardware Flow Control (RS232): Disable

Configuration

Reset Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

DMA Request	Stream	Direction	Priority
USART2_TX	DMA1 Stream 6	Memory To Peripheral	High

Add Delete

DMA Request Settings

Peripheral		Memory
Mode: Normal	Increment Address: <input type="checkbox"/>	<input checked="" type="checkbox"/>
Use Fifo: <input type="checkbox"/>	Threshold: <input type="text"/>	Data Width: Byte
	Burst Size: <input type="text"/>	Byte

In the same window, under Parameter Settings, we selected a 115200 baud rate and 8 bit word length.

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Configure the below parameters :

Search (Ctrl+F)

Basic Parameters

Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

Advanced Parameters

Data Direction	Receive and Transmit
Over Sampling	16 Samples

Inside the *main()* function, we first defined the string that we wanted to transmit, then we computed its length with *strlen()* function. In the *while(1)* loop we transmitted our string with *HAL_UART_Transmit_DMA*. We then implemented the requested delay of one second between each transmission using the *HAL_Delay* function.

```

90
91 /* Initialize all configured peripherals */
92 MX_GPIO_Init();
93 MX_DMA_Init();
94 MX_USART2_UART_Init();
95 /* USER CODE BEGIN 2 */
96 char* frase= "Name Surname 2001\r\n";
97 int length = strlen(frase);
98 /* USER CODE END 2 */
99
100 /* Infinite loop */
101 /* USER CODE BEGIN WHILE */
102 while (1)
103 {
104     HAL_UART_Transmit_DMA(&huart2, frase, length);
105     HAL_Delay(1000);
106     /* USER CODE END WHILE */
107
108     /* USER CODE BEGIN 3 */

```

After connecting our pc to the Nucleo board, we can use a terminal emulator of our choice to see the output, by setting the same baud rate as the one set to transmit data. Then, we also selected the right COM port connected to the Nucleo board.

```
11:25:06.968 -> Name Surname 2001
11:25:07.947 -> Name Surname 2001
11:25:08.926 -> Name Surname 2001
11:25:09.950 -> Name Surname 2001
11:25:10.931 -> Name Surname 2001
11:25:11.913 -> Name Surname 2001
11:25:12.894 -> Name Surname 2001
```

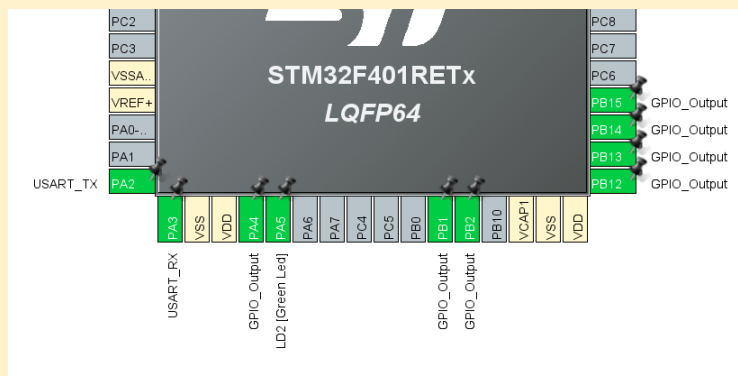
< [Progress Bar]

☒ Scorrimento automatico ☒ Visualizza orario Entrambi (NL & CR) 115200 baud

```
120 RCC_ClkInitStruct RCC_ClkInitStruct = {0}.
```

Part 1b: Write on the LCD the name of each member of your group, one per line, in alphabetical order.

We configured the PA4, PB1, PB2, PB12-15 pins as *GPIO_Output* and imported and included the "PMDB16_LCD" library.



```
17 */
18 /* USER CODE END Header */
19 /* Includes -----
20 #include "main.h"
21
22 /* Private includes -----
23 /* USER CODE BEGIN Includes */
24 #include "PMDB16_LCD.h"
25 #include <string.h>
26 /* USER CODE END Includes */
27
```

We defined an array containing the names of the group members and an *insertion_sort* function to sort the names in alphabetical order.

```

59 /* Private user code -----*/
60 /* USER CODE BEGIN 0 */
61
62 int members_index;
63
64 char *members[] = {"Riccardo", "Alessio", "Sofia", "Anna", "Emma"};
65
66 int length = sizeof(members)/sizeof(char*);
67
68 void insertion_sort(char* arr[], int n)
69 {
70     for (int i = 1; i < n; ++i) {
71         char* key = arr[i];
72         int j = i - 1;
73
74         /* Move elements of arr[0..i-1], that are
75          greater than key, to one position ahead
76          of their current position */
77         while (j >= 0 && strcmp(arr[j], key)>0) {
78             arr[j + 1] = arr[j];
79             j = j - 1;
80         }
81         arr[j + 1] = key;
82     }
83 }
84
85

```

```

91
92 int main(void)
93 {
94
95     /* USER CODE BEGIN 1 */
96     insertion_sort(members, length);
97     /* USER CODE END 1 */
98

```

We initialized the LCD controller and its backlight and defined an array containing the names in the main.c before the while(1) loop.

```

113 /* USER CODE END SysInit */
114
115 /* Initialize all configured peripherals */
116 MX_GPIO_Init();
117 MX_USART2_UART_Init();
118 /* USER CODE BEGIN 2 */
119
120 lcd_initialize();
121 lcd_backlight_ON();
122
123 /* USER CODE END 2 */
124

```

First, we manually wrote the first name in the second row. After that, at each iteration the names scroll in alphabetical order appearing first in row '1' and then in row '0'. The elements of the *members* array are selected using the remainder of the integer division by 5, which does not require the counter to be reset.

```
127  /* Infinite loop */
128  /* USER CODE BEGIN WHILE */
129
130  int counter = 1;
131  lcd_println(members[0], 1);
132  HAL_Delay(1000);
133  while (1)
134  {
135
136      lcd_clear();
137      lcd_println(members[(counter + 1) % length], 1);
138      lcd_println(members[(counter ) % length], 0);
139      counter = (counter + 1)%length;
140      HAL_Delay(1000);
141
142      /* USER CODE END WHILE */
143
144      /* USER CODE BEGIN 3 */
145  }
```

Professor comments: