

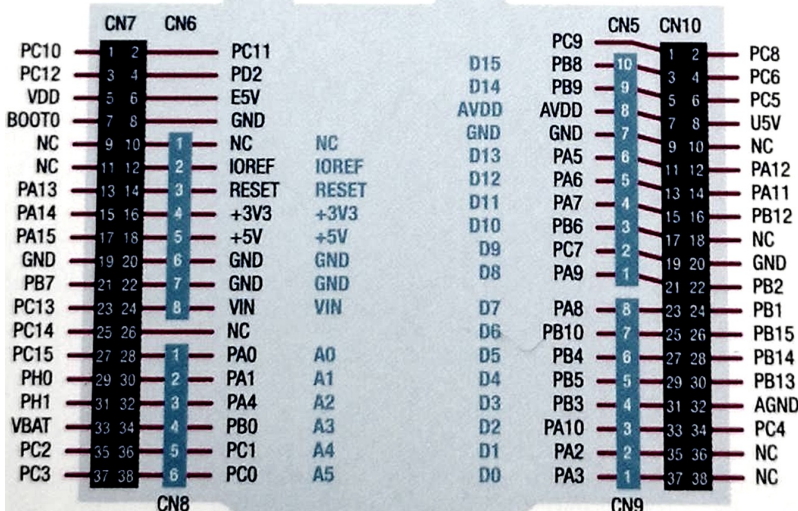


```
1 // ----- Lab2 - 01 & 02 -----
2 // blink (on/off) with a period of 0.2 sec. for an on board LED
3 // blink (on/off) with a period of 0.2 sec. for an external LED
4
5 /* Infinite loop */
6 /* USER CODE BEGIN WHILE */
7 while (1)
8 {
9     HAL_Delay(100);
10    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
11    /* USER CODE END WHILE */
12    /* USER CODE BEGIN 3 */
13 }
14 /* USER CODE END 3 */
15
16 // ----- Lab2 - 03 -----
17 // toggle an LED (on/off) with pushing USER push-button. (Debouncing is required)
18
19 while (1)
20 {
21     // blue button(PC13) is pressed(GPIO_PIN_RESET) state)
22     if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
23     {
24         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
25         HAL_Delay(100);
26         // debouncing the button
27         while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
28             ;
29     }
30
31     // ----- Lab2 - 04 -----
32     // communicate UART, serial terminal
33     // screen /dev/tty.usbmodem11203 115200, onclr
34     // quit: ctrl + A then ctrl + \
35
36     uint8_t ch;
37     while (1)
38     {
39         if (HAL_UART_Receive(&huart2, &ch, 1, 100) == HAL_OK)
40         {
41             HAL_UART_Transmit(&huart2, &ch, 1, 100);
42         }
43
44         // ----- Lab2 - 05 -----
45         // toggle an LED status (on/off) with commands via serial console.
46         // (Type "on" or "off" then press Enter to on or off the LED
47
48         /* Infinite loop */
49         /* USER CODE BEGIN WHILE */
50         int idx = 0;
51         uint8_t ch;
52         uint8_t buf[256];
53         while (1)
54         {
55             if (HAL_UART_Receive(&huart2, buf + idx, 1, 1000) == HAL_OK)
56             {
57                 HAL_UART_Transmit(&huart2, buf + idx, 1, 1000);
58                 if (buf[0] == 'o' && buf[1] == 'n' && buf[2] == '\r' && ch != 1)
59                 {
60                     HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
61                     HAL_Delay(100);
62                     ch = 1;
63                 }
64                 else if (buf[0] == 'o' && buf[1] == 'f' && buf[2] == 'f' && buf[3] == '\r' && ch != 0)
65                 {
66                     HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
67                     HAL_Delay(100);
68                     ch = 0;
69                 }
70                 if (buf[idx] == '\r')
71                 {
72                     idx = 0;
73                     buf[2] = '0';
74                     buf[3] = '0';
75                 }
76                 else
77                 {
78                     idx++;
79                 }
80             }
81         }
82     }
```



```
1 // Skill Test Sec 2 2562
2 // 1. Make all lights blink every 150 ms (on 50 ms, off 100 ms)
3 // 2. push button TOGGLE (on 100 ms, off 50 ms)
4 // 3. hold to all light on and button dont response anymore
5 // to push button anymore.
6
7 /* USER CODE BEGIN WHILE */
8
9 uint8_t state = 1;
10 uint8_t buttonState = GPIO_PIN_SET;
11 while (1)
12 {
13     if (state == 1)
14     {
15         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_SET); // LED on
16         HAL_Delay(50);
17         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_RESET); // LED off
18         HAL_Delay(100);
19
20         // if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET) {
21         //     state++;
22         //     // debouncing the button
23         //     while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET);
24         // }
25         // Debouncing
26         if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET && buttonState == GPIO_PIN_SET)
27         {
28             HAL_Delay(50); // Debounce delay
29             if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
30                 state++; // Increment state
31         }
32         buttonState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13);
33     }
34     else if (state == 2)
35     {
36         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_SET); // LED on
37         HAL_Delay(100);
38         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_RESET); // LED off
39         HAL_Delay(50);
40
41         // if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET) {
42         //     state++;
43         //     // debouncing the button
44         //     while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET);
45         // }
46         // Debouncing
47         if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET && buttonState == GPIO_PIN_SET)
48         {
49             HAL_Delay(50); // Debounce delay
50             if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
51                 state++; // Increment state
52         }
53         buttonState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13);
54     }
55     else
56     {
57         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_SET); // All LEDs on
58     }
59 }
60
61 // Debounce function
62 uint8_t buttonState = GPIO_PIN_SET;
63 if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET && buttonState == GPIO_PIN_SET)
64 {
65     HAL_Delay(50); // Debounce delay
66     if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
67     {
68         state++; // Increment state
69         // stage controller
70         if (state > 2)
71             state = 0; // Reset state after the third press
72     }
73 }
74 buttonState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13);
```

NUCLEO-F411RE



Arduino Uno

ST morpho



```
1 // ----- Lab3 - 01 -----
2 // blink (on/off) green LED with changeable period by pushing USER push-button
3 // from periods of 0.2 sec. → 1 sec. → 5 sec. and back to 0.2 sec.
4 // (using External interrupt/event controller (EXTI) is required)
5
6 /* Private user code -----*/
7 /* USER CODE BEGIN 0 */
8 uint16_t blinkTime = 200;
9
10 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
11 {
12     HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_15);
13     HAL_Delay(100);
14
15     if (blinkTime == 200)
16         blinkTime = 1000;
17     else if (blinkTime == 1000)
18         blinkTime = 5000;
19     else
20         blinkTime = 200;
21
22     __HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
23 }
24 /* USER CODE END 0 */
25
26 /* Infinite loop */
27 /* USER CODE BEGIN WHILE */
28 while (1)
29 {
30     HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
31     HAL_Delay(blinkTime);
32     /* USER CODE END WHILE */
33
34     /* USER CODE BEGIN 3 */
35 }
36 /* USER CODE END 3 */
```



```
1 // ----- Skill Test Sec 1 2562 -----
2 // 1. Make red and green lights blink every ms.
3 // 2. Use user push button to to make red and green lights blink every ms.
4 // 3. Hold push button at least
5 // push button anymore.
6
7 /* USER CODE BEGIN WHILE */
8 uint8_t state = 1;
9
10 while (1)
11 {
12     if (state == 1)
13     {
14         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6);
15         HAL_Delay(75); // Delay for 75ms
16         if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
17         {
18             state++;
19             // debouncing the button
20             while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
21                 ;
22         }
23     }
24     else if (state == 2)
25     {
26         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6);
27         HAL_Delay(15); // Delay for 75ms
28         if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
29         {
30             state++;
31             // debouncing the button
32             while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
33                 ;
34         }
35     }
36     else
37     {
38         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_RESET); // All LEDs off
39     }
40 }
```



```
1 // ----- skilltest2561 -----
2 // Implement the following:
3 // 1 The indicator LEDs blinks every one second (500ms on, 500ms off) after reset
4 // 2 Pushing the USER button the first time will change the state such that only one LED is blinking. ( choose
5 // any)
6 // 3 Pushing the USER button the second time will change the state such that only two LEDs are blinking.
7 // 4 Pushing the USER button the third time will go back to the first state after reset
8
9 // Setup: HSE clock, GPIO mode Rising edge, NVIC 2 bit 2
10
11 /* Private user code -----*/
12 /* USER CODE BEGIN 0 */
13 uint16_t state = 0;
14
15 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
16 {
17     HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_15);
18     HAL_Delay(100);
19
20     if (state == 0)
21         state = 1;
22     else if (state == 1)
23         state = 2;
24     else
25         state = 0;
26
27     __HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
28 }
29
30 while (1)
31 {
32     if (state == 0)
33     {
34         // HAL_GPIO_WritePin: Set Status
35         // off: GPIO_PIN_RESET, on: GPIO_PIN_SET
36         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6, GPIO_PIN_RESET); // All LEDs off
37         // debouncing the button
38         while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
39             ;
40     }
41     else if (state == 1)
42     {
43         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET); // Turn off LED
44         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
45         HAL_Delay(500); // Delay for 500ms
46         // debouncing the button
47         while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
48             ;
49     }
50     else if (state == 2)
51     {
52         HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5 | GPIO_PIN_6);
53         HAL_Delay(500);
54         // debouncing the button
55         while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
56             ;
57     }
58 }
```