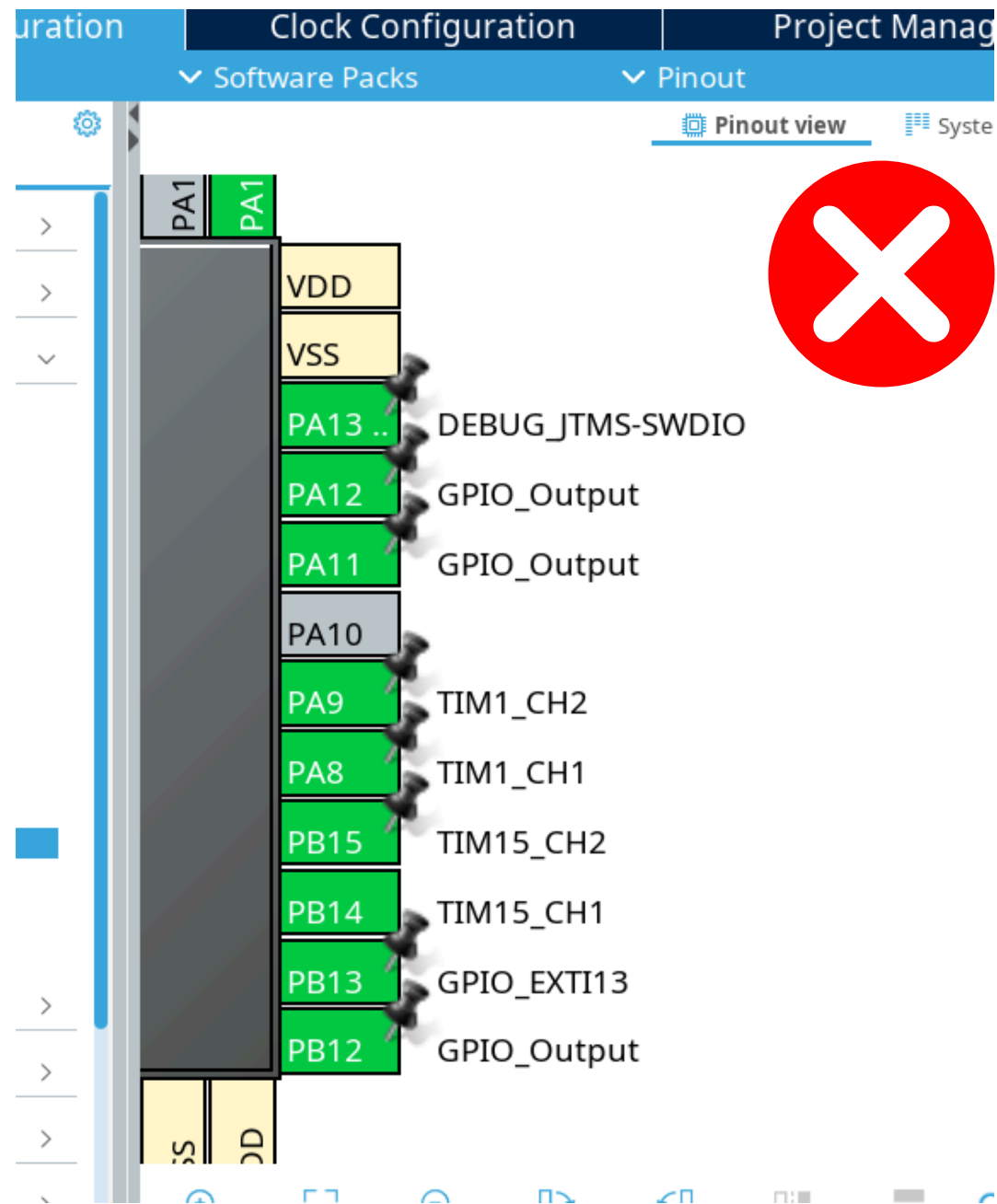
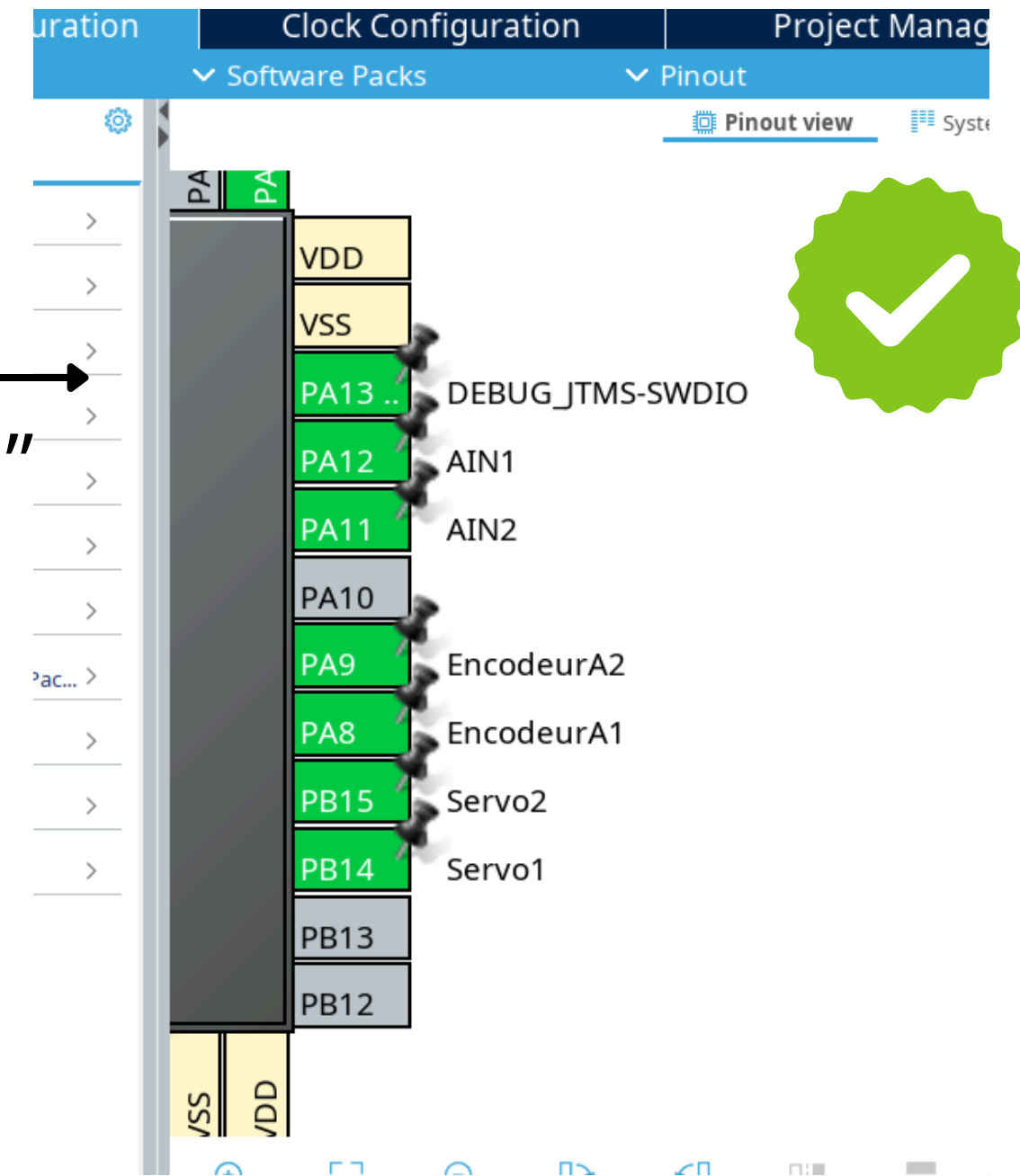


**Comment avoir un projet
STM propre**

1) On ne laisse pas le nom de pins qu'on doit manipuler (dans les fonctions) par défaut



Clique droit + "enter user label"



Comment nommer les pins?

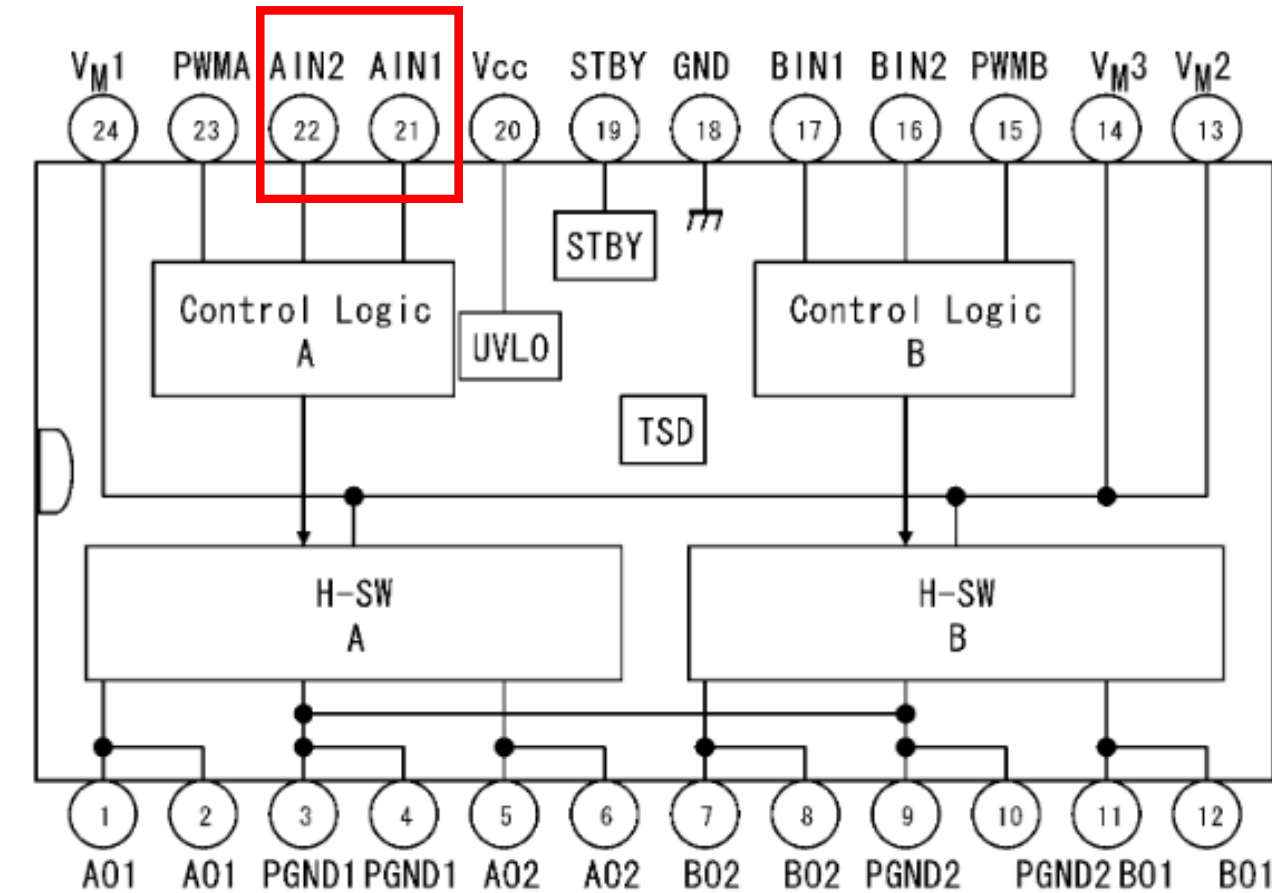
TOSHIBA

TB6612FNG

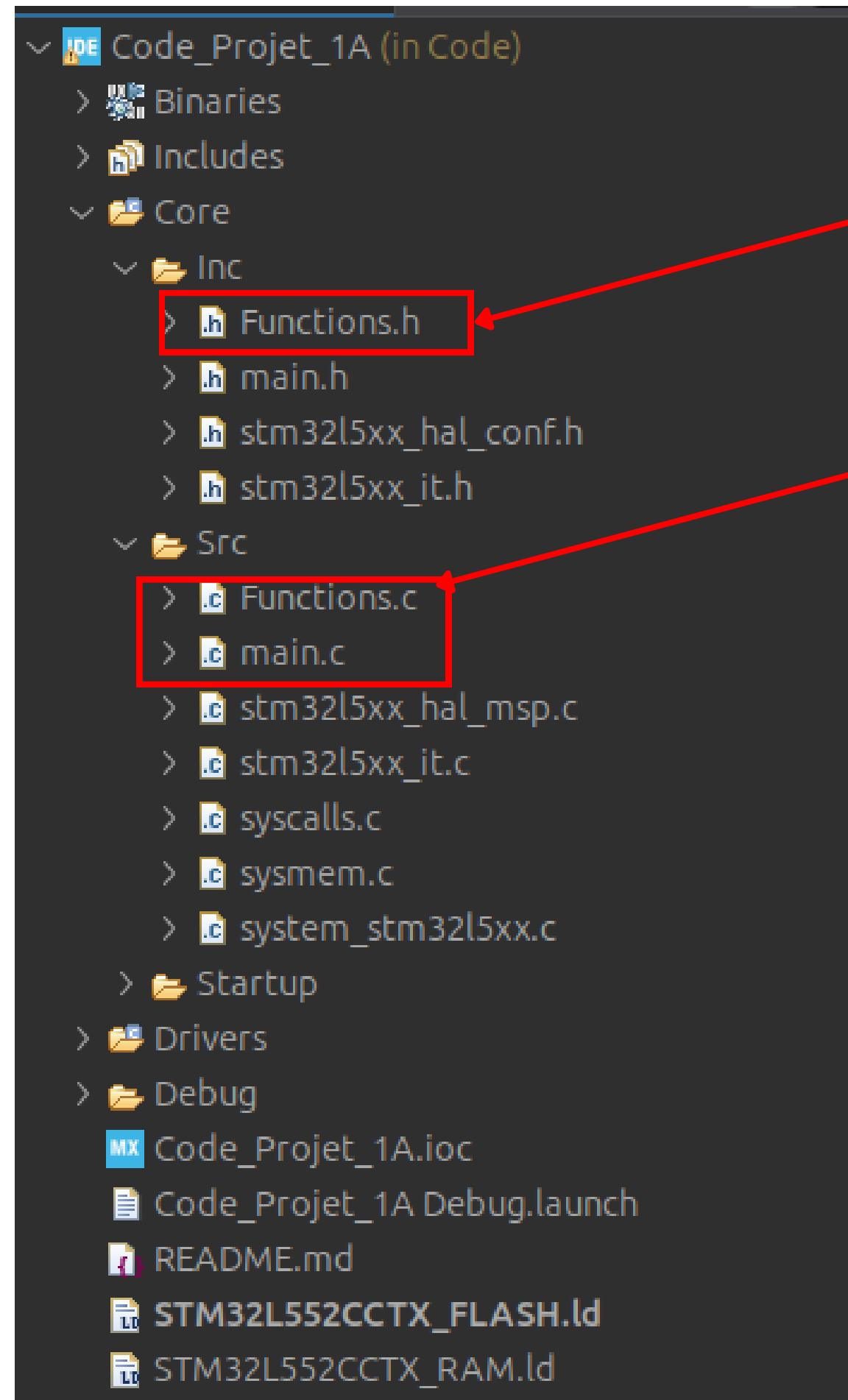
Préférer :

- Celui des composants/datasheets
- Appellations courtes et évidentes

Block Diagram



2) On organise le projet



Headers, avec les déclarations des fonctions, éventuellement des macros et des types

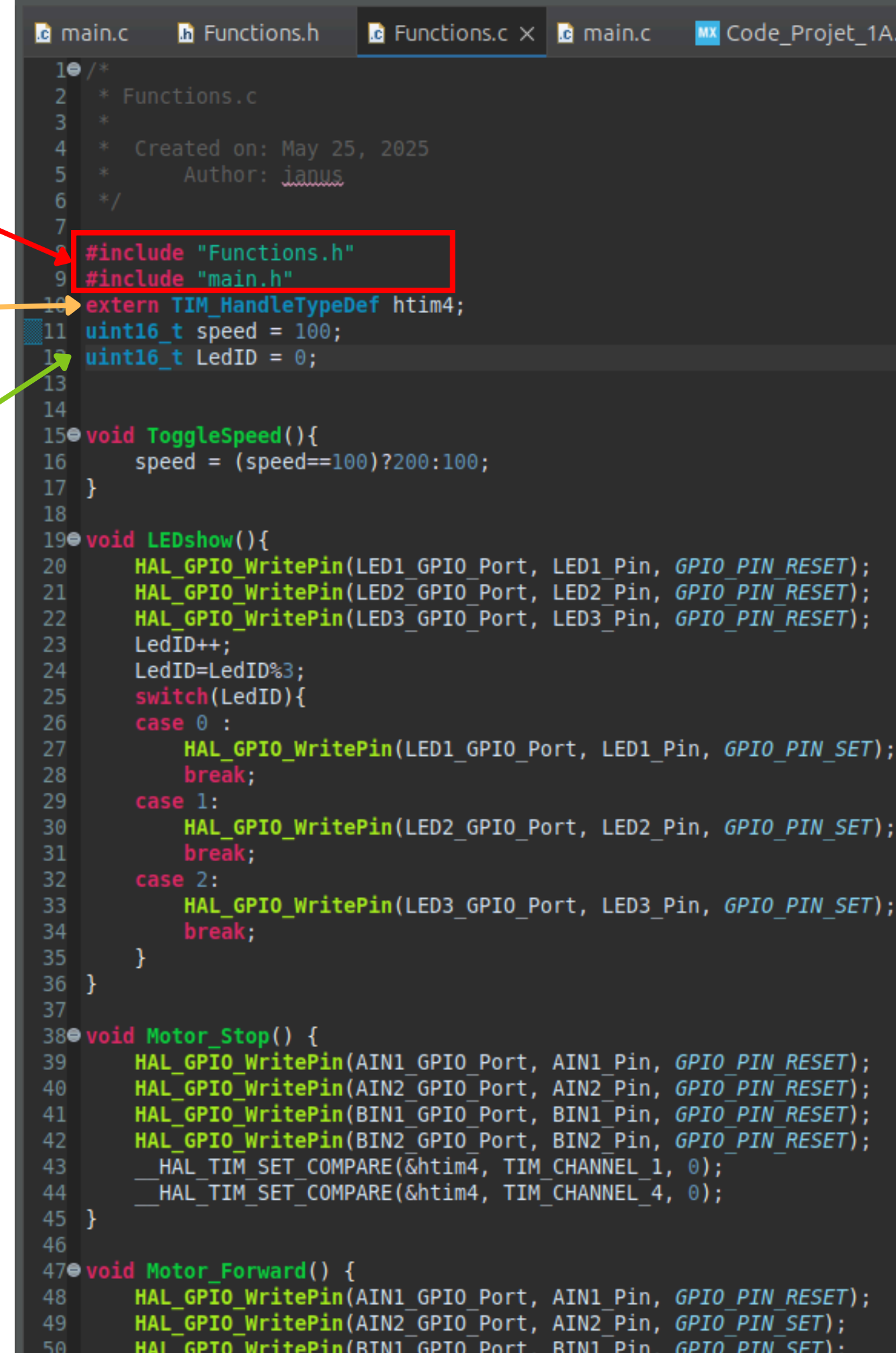
Sources, avec les définitions des fonctions

Faut pas oublier les imports nécessaires

Déclarer les variables externes nécessaires

Nommez les variables de manière courte et intuitive

Rajoutez des commentaires si la fonction n'est pas courte/intuitive !



```
1 /*
2  * Functions.c
3  *
4  * Created on: May 25, 2025
5  * Author: ianux
6  */
7
8 #include "Functions.h"
9 #include "main.h"
10
11 extern TIM_HandleTypeDef htim4;
12 uint16_t speed = 100;
13 uint16_t LedID = 0;
14
15 void ToggleSpeed(){
16     speed = (speed==100)?200:100;
17 }
18
19 void LEDshow(){
20     HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_RESET);
21     HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_RESET);
22     HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin, GPIO_PIN_RESET);
23     LedID++;
24     LedID=LedID%3;
25     switch(LedID){
26     case 0 :
27         HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin, GPIO_PIN_SET);
28         break;
29     case 1:
30         HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, GPIO_PIN_SET);
31         break;
32     case 2:
33         HAL_GPIO_WritePin(LED3_GPIO_Port, LED3_Pin, GPIO_PIN_SET);
34         break;
35     }
36 }
37
38 void Motor_Stop() {
39     HAL_GPIO_WritePin(AIN1_GPIO_Port, AIN1_Pin, GPIO_PIN_RESET);
40     HAL_GPIO_WritePin(AIN2_GPIO_Port, AIN2_Pin, GPIO_PIN_RESET);
41     HAL_GPIO_WritePin(BIN1_GPIO_Port, BIN1_Pin, GPIO_PIN_RESET);
42     HAL_GPIO_WritePin(BIN2_GPIO_Port, BIN2_Pin, GPIO_PIN_RESET);
43     __HAL_TIM_SET_COMPARE(&htim4, TIM_CHANNEL_1, 0);
44     __HAL_TIM_SET_COMPARE(&htim4, TIM_CHANNEL_4, 0);
45 }
46
47 void Motor_Forward() {
48     HAL_GPIO_WritePin(AIN1_GPIO_Port, AIN1_Pin, GPIO_PIN_RESET);
49     HAL_GPIO_WritePin(AIN2_GPIO_Port, AIN2_Pin, GPIO_PIN_SET);
50     HAL_GPIO_WritePin(BIN1_GPIO_Port, BIN1_Pin, GPIO_PIN_SET);
51 }
```

Résultat :

un main léger, facile à
lire/relire/débugger

```
main.c x Functions.h Functions.c main.c Code_Projet_1A.ioc
80 /* USER CODE BEGIN PFP */
81
82 /* USER CODE END PFP */
83
84 /* Private user code -----*/
85 /* USER CODE BEGIN 0 */
86 int _write(int file, char *ptr, int len)
87 {
88     HAL_UART_Transmit(&huart2, (uint8_t*)ptr, len, HAL_MAX_DELAY);
89     return len;
90 }
91
92 void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
93     if (huart->Instance == UART4) {
94         LEDshow();
95         printf("Commande reçue : %c\r\n", received_data); // Affichage de la réception
96
97         switch (received_data) {
98             case 'z': Motor_Forward(); printf("Avance\n"); break;
99             case 's': Motor_Backward(); printf("Recule\n"); break;
100             case 'q': Motor_Left(); printf("Gauche\n"); break;
101             case 'd': Motor_Right(); printf("Droite\n"); break;
102             case ' ': Motor_Stop(); printf("Stop\n"); break;
103             case 't': ToggleSpeed(); printf("ToggledSpeed\n"); break;
104             default: printf("Commande inconnue\n"); break;
105         }
106
107         HAL_UART_Receive_IT(&huart4, &received_data, 1);
108     }
109 }
110
111
112 /* USER CODE END 0 */
113
114 /**
115  * @brief The application entry point.
116  * @retval int
117  */
118
119 int main(void)
120 {
121
122     /* USER CODE BEGIN 1 */
123
124     /* USER CODE END 1 */
125
126     /* MCU Configuration-----*/
127
128     /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
129     HAL_Init();
```