Lab Assignment 3

Mohit Maurya (22110145)

Parag Sarvoday Sahu (22110179)

Q1: Implementation of an N-bit Johnson counter

This code defines buttons as external interrupt so *increment_button*, *select_button* are the variable corresponding to the two buttons. They act as interrupt when pressed, so depending on the state of machine the appropriate action is performed.

Defining the required variables and arrays.

```
int increment_button;
int select_button;

int bit_value[3] = {2,3,4} ;

int32_t temp_two_expo = 1;
uint32_t i=0;
uint32_t counter=0;
uint32_t current_time;
uint32_t previous_time;
int debounce_time = 200;
```

Defining *flash_led* function to flash LEDs to indicate the selection of N (number of bits).

```
void flash_led(void){

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 1);
        HAL_Delay(800);

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 0);
        HAL_Delay(200);

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 1);
        HAL_Delay(800);

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
```

```
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 0);
        HAL_Delay(200);

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 1);
        HAL_Delay(800);

        HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 0);
        HAL_Delay(200);

}
```

display_number function to convert decimal number to their binary format and display them using onboard LEDs.

```
void display_number(int num){
        //lights us LEDs corresponding to the binary value of the given
input
        //LSB to MSB -> LD1, LD2, LD3 and B12

        int shift_num = num;
        HAL_GPIO_WritePin(GPIOB,LD1_Pin, shift_num & 1 );
        shift_num = shift_num >> 1;
        HAL_GPIO_WritePin(GPIOB, LD2_Pin, shift_num & 1 );
        shift_num = shift_num >> 1;
        HAL_GPIO_WritePin(GPIOB, LD3_Pin, shift_num & 1 );
        shift_num = shift_num >> 1;
        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, shift_num & 1 );
}


//PB15 for cycle_button

//PB10 for set_button
```

The main logic block.

```
while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
        //N selection phase
          while (select_button == 0){
                display_number(bit_value[i]);
                        if (increment_button){
```

```
                                i=(i+1)%3;
                                increment_button=0;
                        }
                }
                flash_led();

        //Counting phase
                while(1){
                        display_number(counter);
                        if (counter == ((temp_two_expo << bit_value[i]))){
                                counter=0;
                        }
                }
  }
```

Code block to process the press of the button.

```
/* USER CODE BEGIN 4 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  UNUSED(GPIO_Pin);

  // PIN 15 FOR increment BUTTON
  // PIN 10 FOR select_BUTTON
  current_time = HAL_GetTick();
  if (select_button==1){ /// counting phase
          if (GPIO_Pin == GPIO_PIN_15 && (current_time - previous_time >
debounce_time) ){
                          counter=counter+1;
          }
          if (GPIO_Pin == GPIO_PIN_10 && (current_time - previous_time >
debounce_time) ){
                  counter=0;
          }
          previous_time = current_time;
  }
  else {
          if (GPIO_Pin == GPIO_PIN_15 && (current_time - previous_time >
debounce_time) ){
                  increment_button=1;
                }

          if (GPIO_Pin == GPIO_PIN_10 && (current_time - previous_time >
debounce_time) ){
                  select_button=1;
          }
          previous_time = current_time;
  }
}
/* USER CODE END 4 */
```

Q2: Implementation of a countdown timer using the 7-segment LED display

This codes defines button as input so *user_button variable* is set (1) as long as button is pressed and it is reset (0) when its left unpressed. The 7-segment display provided was common anode, meaning the LED glows when it is provided with a 'low signal'. The 7-segment driver is written following this arrangement.

C Codes:

Declaring all the variables used in the program

```
/* USER CODE BEGIN PV */
int counter = 0;
int selected_value;
int long_press_flag = 0;
int short_press_flag=0;
int user_button;
int seven_map[] = {63, 6, 91, 79, 102, 109, 125, 7, 127, 111};

/* USER CODE END PV */
```

Driver for 7-segment LED (notice the negation given as the third argument):

```
void display_seven_segment(int num){
	//complete this function
	int shift_num = seven_map[num];

	HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, !(shift_num & 1) ); //A
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, !(shift_num & 1) );  //B
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOC, GPIO_PIN_0, !(shift_num & 1) );  //C
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOD, GPIO_PIN_3, !(shift_num & 1) ); //D
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOG, GPIO_PIN_2, !(shift_num & 1)); //E
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOG, GPIO_PIN_3, !(shift_num & 1) ); //F
	shift_num = shift_num >> 1;
	HAL_GPIO_WritePin(GPIOE, GPIO_PIN_2, !(shift_num & 1)); //G
}
```

A similar *flash_led* function as before:

```
void flash_leds(void){
	//make the LEDs flash
	HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
		HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 1);
		HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
		HAL_Delay(800);
```

```
                HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
                HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 0);
                HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
                HAL_Delay(200);

                HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
                HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 1);
                HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
                HAL_Delay(800);

                HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
                HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 0);
                HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
                HAL_Delay(200);

                HAL_GPIO_WritePin(GPIOB, LD1_Pin, 1);
                HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 1);
                HAL_GPIO_WritePin(GPIOB, LD3_Pin, 1);
                HAL_Delay(800);

                HAL_GPIO_WritePin(GPIOB, LD1_Pin, 0);
                HAL_GPIO_WritePin(GPIOB, GPIO_PIN_7, 0);
                HAL_GPIO_WritePin(GPIOB, LD3_Pin, 0);
                HAL_Delay(200);
}
```

A function block to process the button press returns 1 if a long press is detected or else a 0 for a short press. No use of interrupt this time because that makes it difficult to detect a long press. As interrupt does not detect the time for which the button was kept pressed.

```
int check_long_press(void){
                HAL_Delay(50);
                user_button = HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13);
                if (user_button){
                        short_press_flag=1;
                }
                for (int i=0;i<15;i++){
                        user_button = HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13);
                        if (user_button==0){
                                return 0;
                        }
                        HAL_Delay(200);

                }
                short_press_flag=0;
                return 1;
}
```

The main logic block:

while (1)

```
{
  /* USER CODE END WHILE */
        display_seven_segment(counter);
        user_button = HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13);
        if (user_button==1){
                long_press_flag=check_long_press();
        }
        if (short_press_flag){
                counter=(counter+1)%10;
                display_seven_segment(counter);
                short_press_flag=0;
        }
        if (long_press_flag){
                flash_leds();
                selected_value=counter;
               while(1){
                       display_seven_segment(counter);
                       counter--;
                       if (counter==-1){
                               counter=selected_value;
                       }
                       HAL_Delay(500);
               }
        }
```