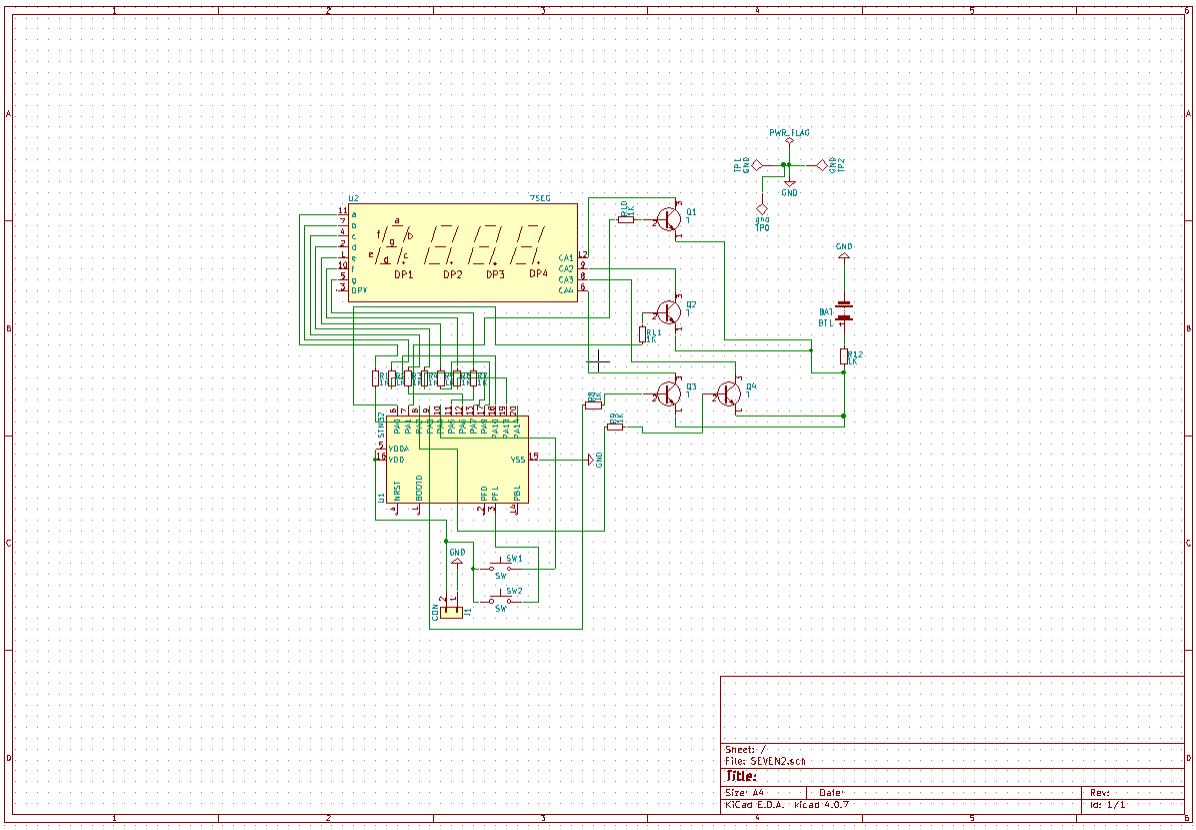
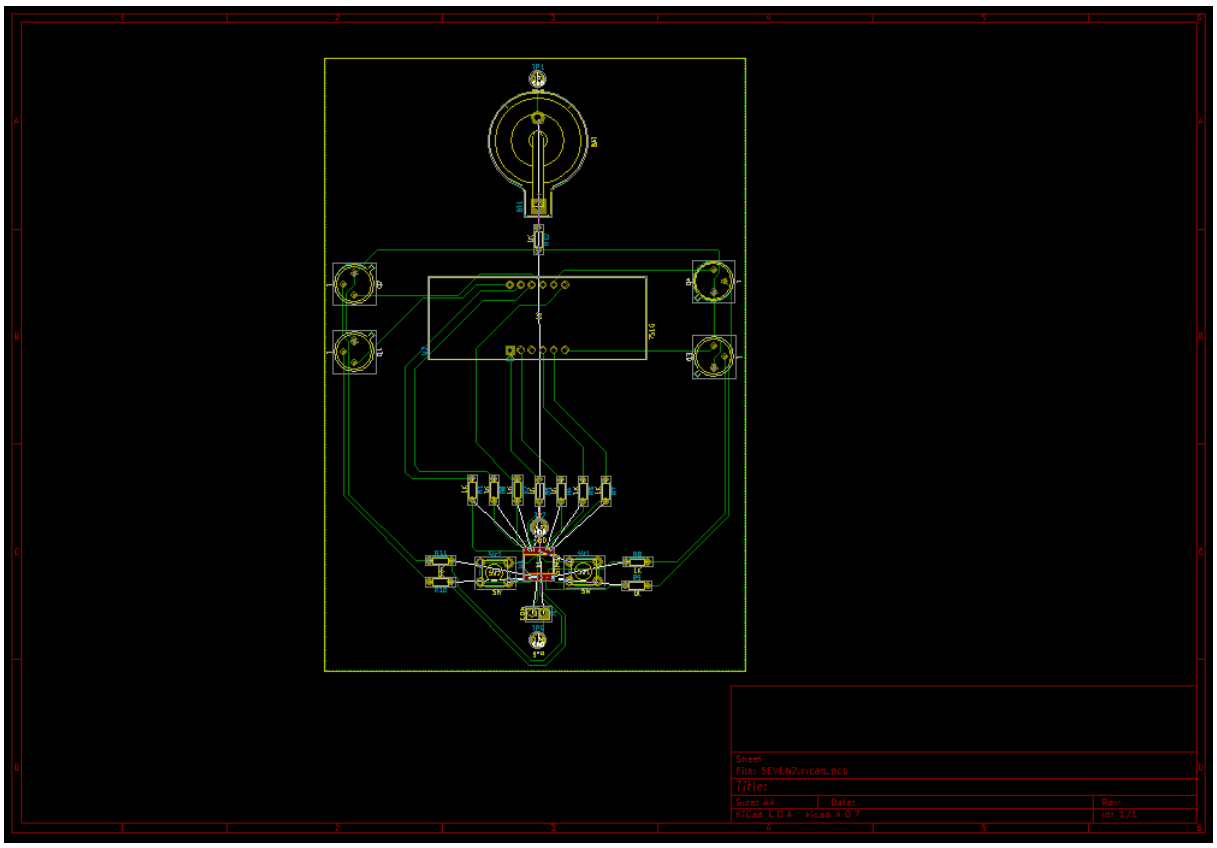
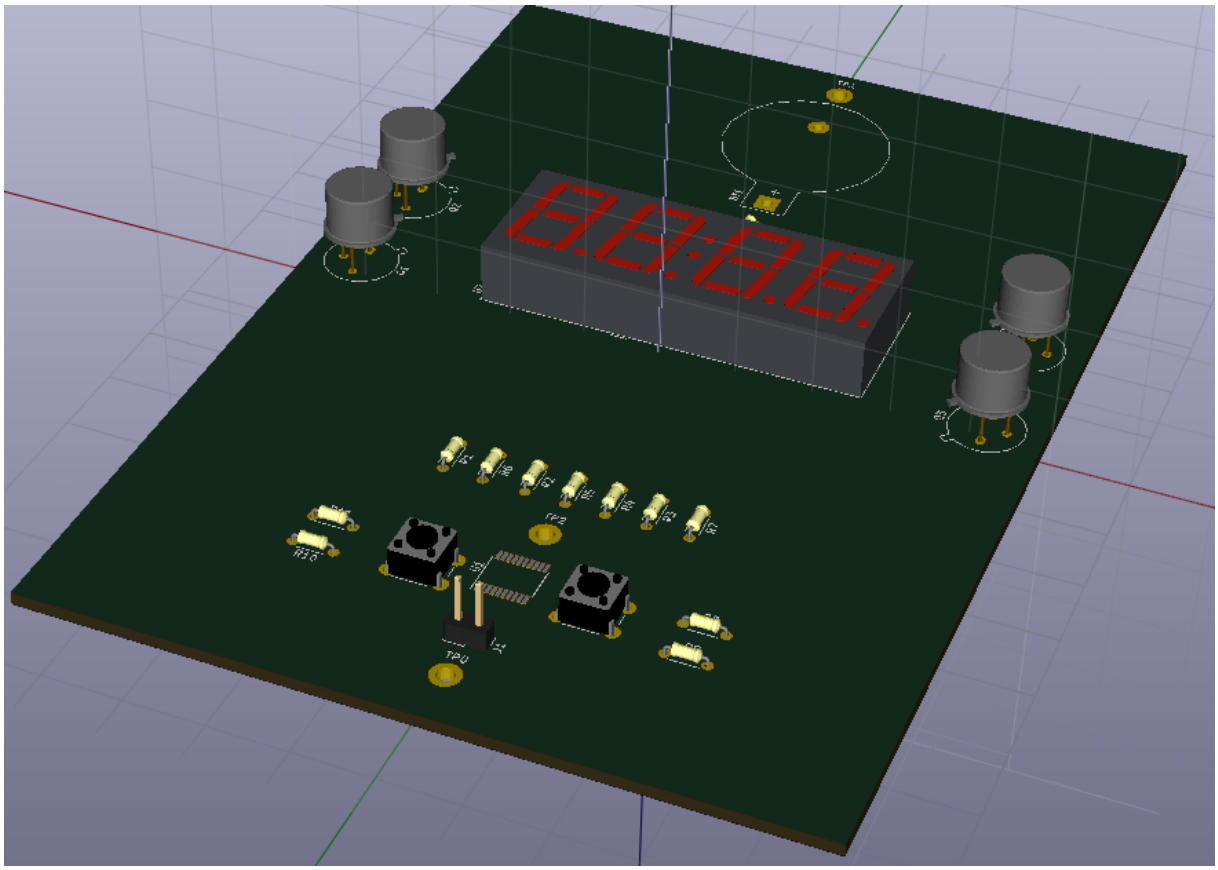
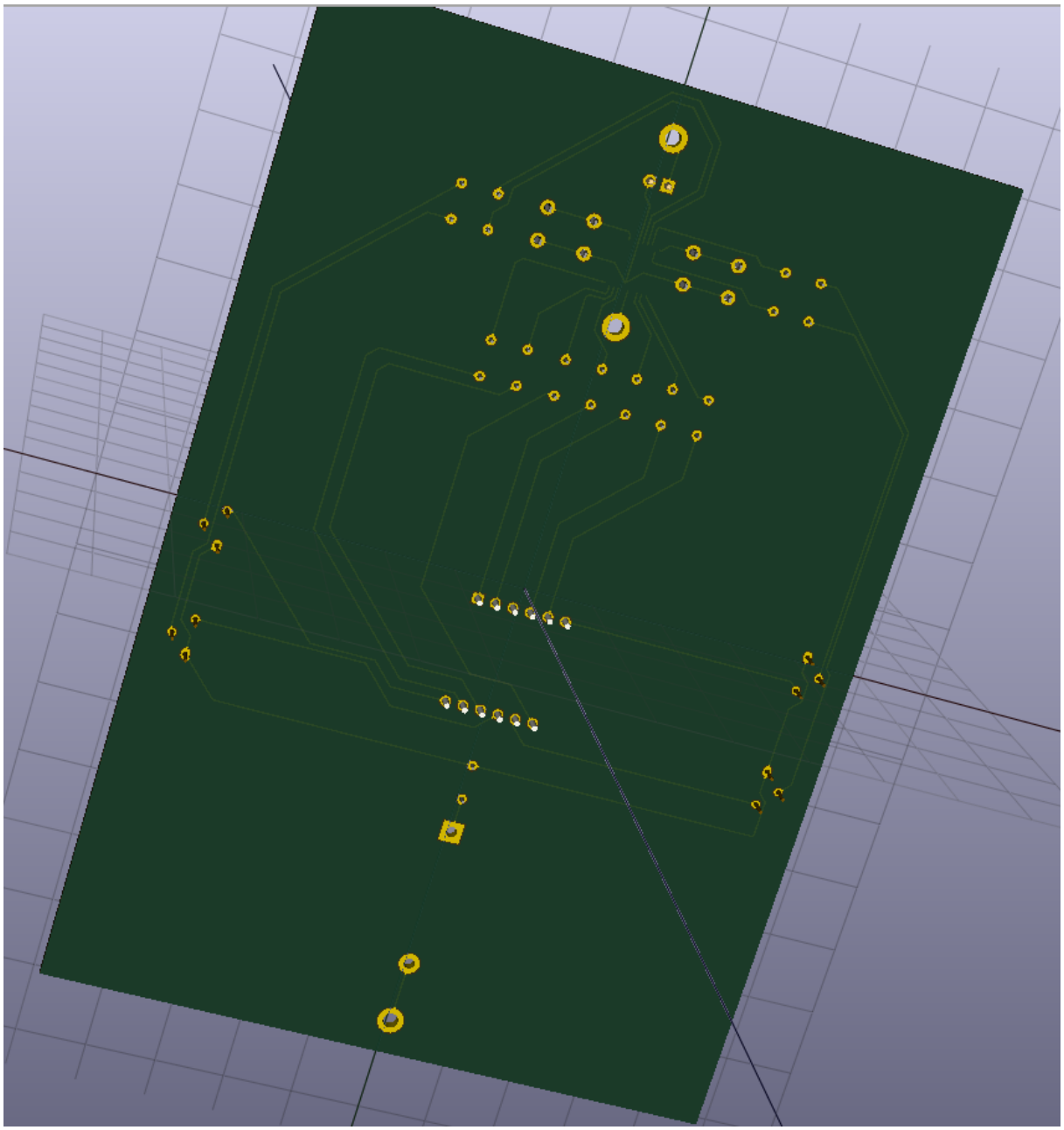

PWR_FLAG
GND
U2
7SEG
DP1
DP2
DP3
DP4
DPY
Q1
Q2
Q3
Q4
BAT
SW1
SW2
Sheet: /
File: SEVEN2.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1


Sheet
File: SEVEN2.kicad_pcb
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1





```c
#include "main.h"
#include "stm32f0xx_hal.h"
#include "tim.h"
#include "gpio.h"

uint16_t mode;
uint16_t digit1;
uint16_t digit2;
uint16_t digit3;
uint16_t digit4;

void SystemClock_Config(void);
/**
  * @brief System Clock Configuration
  * @retval None
```

```
  */
void SystemClock_Config(void)
{

  RCC_OscInitTypeDef RCC_OscInitStruct;
  RCC_ClkInitTypeDef RCC_ClkInitStruct;

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = 16;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL6;
  RCC_OscInitStruct.PLL.PREDIV = RCC_PREDIV_DIV1;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

    /**Initializes the CPU, AHB and APB busses clocks
    */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    _Error_Handler(__FILE__, __LINE__);
  }

  HAL_RCC_MCOConfig(RCC_MCO, RCC_MCO1SOURCE_SYSCLK, RCC_MCODIV_1);

    /**Configure the Systick interrupt time
    */
  HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

    /**Configure the Systick
    */
  HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

  /* SysTick_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}

/**
  * @brief  This function is executed in case of error occurrence.
  * @param  file: The file name as string.
  * @param  line: The line in file as a number.
  * @retval None
  */
```

```c
void _Error_Handler(char *file, int line)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return
state */
  while(1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t* file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line
number,
     tex: printf("Wrong parameters value: file %s on line %d\r\n", file,
line) */
  /* USER CODE END 6 */
}
#endif

void EXTI0_1_IRQHandler(void)
{
      for (int i = 0; i < 10000; i++);
      if(mode < 4)
            mode = mode + 1;
      else
            mode = 0;
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_1);
}

/**
* @brief This function handles EXTI line 4 to 15 interrupts.
*/
void EXTI4_15_IRQHandler(void)
{
      for (int i = 0; i < 10000; i++);
      if (mode == 1){
            if (digit1 >= 9)
                  digit1 = 0;
            else
                  digit1++;
      }
      else if (mode == 2){
            if (digit2 >= 9)
```

```
                digit2 = 0;
            else
                digit2++;
        }
        else if (mode == 3){
            if (digit3 >= 9)
                digit3 = 0;
            else
                digit2++;
        }
        else if (mode == 4){
            if (digit4 >= 9)
                digit4 = 0;
            else
                digit2++;
        }
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_4);
}



void zero(){
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_RESET);
}
void one(){
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_RESET);
}
void two(){
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void three(){
      HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
      HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
```

```
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void four(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void five(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void six(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void seven(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_RESET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_RESET);
}
void eight(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void nine(){
	HAL_GPIO_WritePin(a_GPIO_Port, a_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(b_GPIO_Port, b_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(c_GPIO_Port, c_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(d_GPIO_Port, d_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(e_GPIO_Port, e_Pin, GPIO_PIN_RESET);
```

```
	HAL_GPIO_WritePin(f_GPIO_Port, f_Pin, GPIO_PIN_SET);
	HAL_GPIO_WritePin(g_GPIO_Port, g_Pin, GPIO_PIN_SET);
}
void set1(){
	switch(digit1){
	case 0:
		zero();
		break;
	case 1:
		one();
		break;
	case 2:
		two();
		break;
	case 3:
		three();
		break;
	case 4:
		four();
		break;
	case 5:
		five();
		break;
	case 6:
		six();
		break;
	case 7:
		seven();
		break;
	case 8:
		eight();
		break;
	case 9:
		nine();
		break;
	}
}
void set2(){
	switch(digit2){
		case 0:
			zero();
			break;
		case 1:
			one();
			break;
		case 2:
			two();
			break;
		case 3:
			three();
			break;
		case 4:
			four();
			break;
```

```
            case 5:
                  five();
                  break;
            case 6:
                  six();
                  break;
            case 7:
                  seven();
                  break;
            case 8:
                  eight();
                  break;
            case 9:
                  nine();
                  break;
      }
}
void set3(){
      switch(digit3){
            case 0:
                  zero();
                  break;
            case 1:
                  one();
                  break;
            case 2:
                  two();
                  break;
            case 3:
                  three();
                  break;
            case 4:
                  four();
                  break;
            case 5:
                  five();
                  break;
            case 6:
                  six();
                  break;
            case 7:
                  seven();
                  break;
            case 8:
                  eight();
                  break;
            case 9:
                  nine();
                  break;
      }
}
void set4(){
      switch(digit4){
            case 0:
```

```
                zero();
                break;
            case 1:
                one();
                break;
            case 2:
                two();
                break;
            case 3:
                three();
                break;
            case 4:
                four();
                break;
            case 5:
                five();
                break;
            case 6:
                six();
                break;
            case 7:
                seven();
                break;
            case 8:
                eight();
                break;
            case 9:
                nine();
                break;
        }
}

int main(void)
{
  /* USER CODE BEGIN 1 */
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en4_GPIO_Port, en4_Pin, GPIO_PIN_RESET);
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_RESET);

  /* MCU Configuration--------------------------------------------------------
-----*/

  /* Reset of all peripherals, Initializes the Flash interface and the
Systick. */
  HAL_Init();

  mode = 0;
  digit1 = 0;
  digit2 = 0;
  digit3 = 0;
  digit4 = 0;

  /* Configure the system clock */
```

```
  SystemClock_Config();

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM14_Init();

  while (1)
  {
      HAL_GPIO_WritePin(en4_GPIO_Port, en4_Pin, GPIO_PIN_RESET);
      set1();
      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_SET);

      HAL_Delay(1);

      HAL_GPIO_WritePin(en1_GPIO_Port, en1_Pin, GPIO_PIN_RESET);
      set2();
      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_SET);

      HAL_Delay(1);

      HAL_GPIO_WritePin(en2_GPIO_Port, en2_Pin, GPIO_PIN_RESET);
      set3();
      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_SET);

      HAL_Delay(1);

      HAL_GPIO_WritePin(en3_GPIO_Port, en3_Pin, GPIO_PIN_RESET);
      set4();
      HAL_GPIO_WritePin(en4_GPIO_Port, en4_Pin, GPIO_PIN_SET);

      HAL_Delay(1);
  }
}
```