Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh

**Khoa Cơ khí Chế tạo Máy**

**Bộ môn Cơ Điện tử**

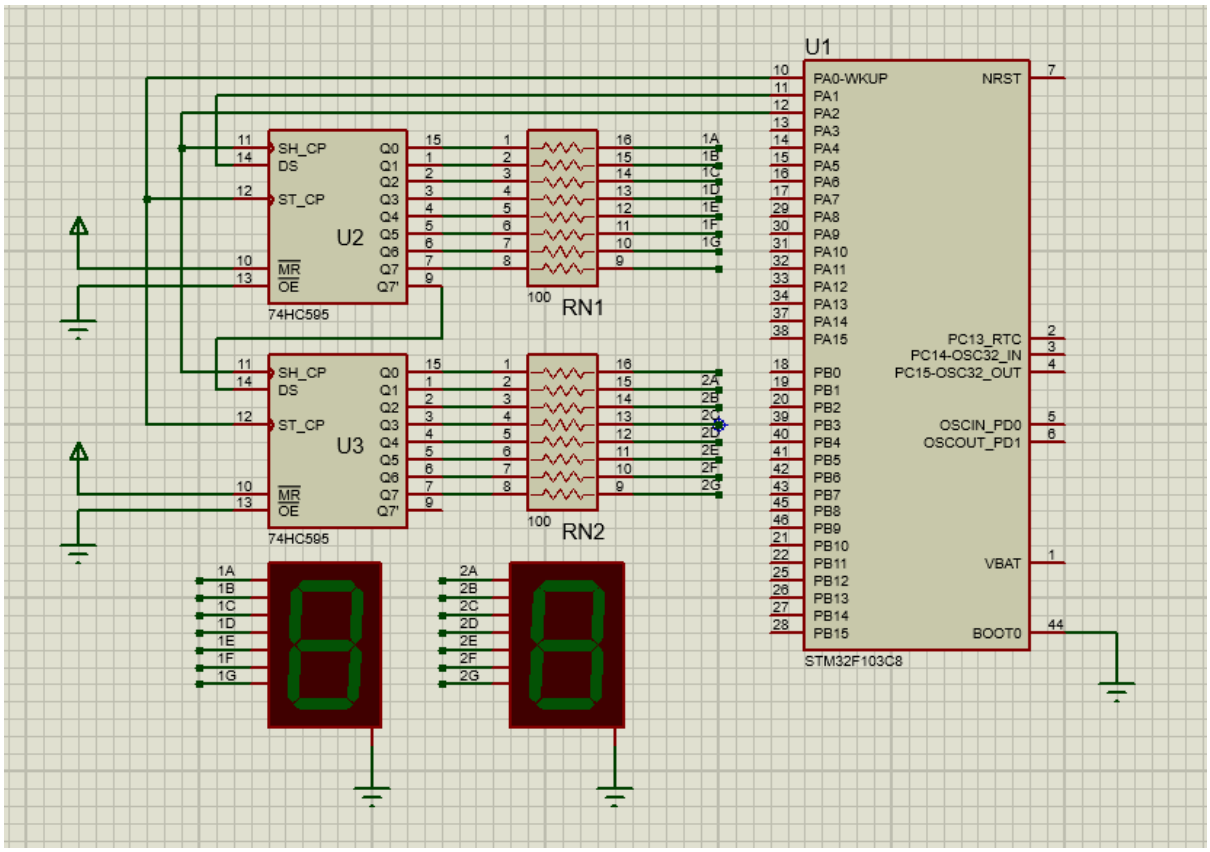
# BÁO CÁO DỰ ÁN

## 1. Thông tin người thực hiện:

| STT | Người thực hiện | MSSV | Ngày |
|---|---|---|---|
| 01 | Nguyễn Hữu Chí | 20146479 | 4/1/2023 |

## 2. Yêu cầu dự án

- Lập trình 2 Led 7 đoạn sáng số từ 00 đến 99 và lặp lại. Trì hoãn giữa 2 lần đếm 0.2s

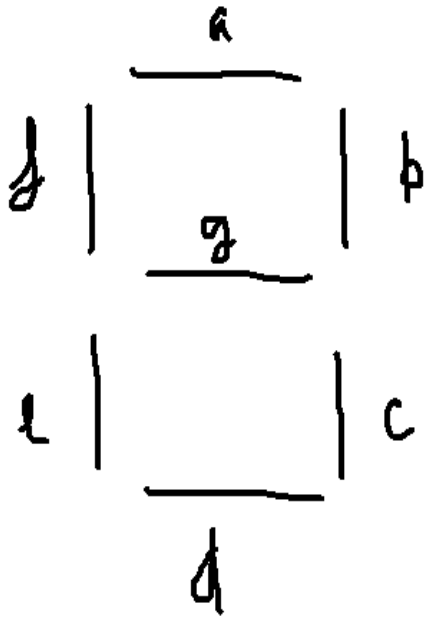



## 3. Nội dung:

### 3.1. Phân tích dự án:

- Để Led sáng được thì các chân của led phải được cấp mức 1. Do đây là Led 7 đoạn có cực âm nối chung (cathode chung) và nối xuống đất

- Với 1 số hiển thị trên led thì ta sẽ có 1 bộ mã nhị phân (hoặc mã hex) tương ứng




- Ví dụ: số 1 sẽ có mã là 0b01100000 (bit 0 cuối cùng không quan trọng do ta không xét tới bit này) tương ứng với các chân abcdefg

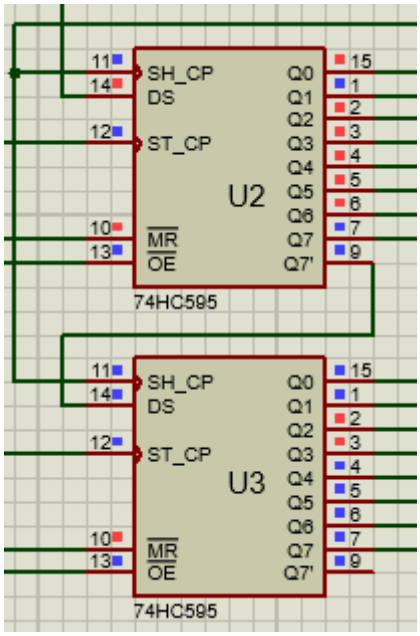
- Do đó ta cần khai báo mã đầu vào cho các số từ 0->9 bằng dòng code dưới đây (mã hex)

const uint8_t LED[] = {0xFC,0x60,0xDA,0xF2,0x66,0xB6,0xBE,0xE0,0xFE,0xF6,0x00};

0x00 là trạng thái tắt 7 LED

- 74HC595 hoạt động với 3 chân input: DS là chân data (0 hoặc 1), SH_CP là chân clock dùng để đẩy data ra các chân Qx, ST_CP là chân lat dùng để xuất mức logic trên các chân Qx

Ví dụ: chân DS đang ở mức 1, thì có 1 xung clock vào, khi chân lat được kích Q0 sẽ xuất mức 1 (các Qx còn lại xuất mức 0). Tiếp tục chuyển DS thành mức 0, cho 2 xung clock vào, sau đó kích chân lat, ta sẽ có kết quả Q0 và Q1 xuất mức 0, Q2 xuất mức 1, các Qx còn lại mức 0 (tín hiệu bị đẩy đi, bit nào vào trước sẽ bị đẩy lên Qx cao hơn)




Ta nối chân Q7’ của U2 vào DS của U3 nhằm mục đích tạo ra dãy data nối tiếp từ U2 sang U3. Ví dụ bit 1 đang ở Q7’ của U2 có 1 xung clock vào (bị đẩy đi) thì bit 1 này sẽ được đẩy sang Q0 của U3. Tương tự nếu bit 1 đang ở Q7’ của U3 có 1 xung clock vào (bị đẩy đi) thì bit 1 này sẽ được đẩy ra ngoài

Ở bài này chúng ta sẽ tạo 3 hàm con để giải quyết bài toán:

```
void LED_CLK(uint32_t n){
	while(n>0){
		HAL_GPIO_WritePin(CLK_GPIO_Port,CLK_Pin,1);
		HAL_GPIO_WritePin(CLK_GPIO_Port,CLK_Pin,0);
		n = n-1;
	}
}
```

- Hàm này dùng để tạo xung clock. Ví dụ LED_CLK(5); tương ứng với 5 xung clock

---

```
void LED_push(uint8_t n){
	for(uint8_t i = 0; i<8; i++){
		HAL_GPIO_WritePin(DAT_GPIO_Port,DAT_Pin,LED[n]&(1<<i));
		LED_CLK(1);
	}
}
```

- Hàm này dùng để xử lý bộ mã ban đầu sau đó gán giá trị cho các chân của Led. Ví dụ LED_push(3) (set số 3); thì hàm này sẽ truy cập vào LED[3] tương ứng với 0xF2 = 0b11110010.

`LED[n]>>i&1u`. Toán tử này dùng để lấy bit thứ i trên bộ mã nhị phân của số 3


Sau khi lấy được bit thứ 0 (bit cuối) sau đó set giá trị bit này cho chân data thì data sẽ được đẩy đi bằng 1 xung clock. Tiếp tục bit thứ 1,2,3,4,5,6,7 (lúc này Led 7 đoạn chưa hiển thị số 3 cần thêm 1 xung kích từ chân lat thì mới có thể hiển hị được)

---

```
void LED_print(uint32_t n){
	LED_push(n%10);
	HAL_GPIO_WritePin(DAT_GPIO_Port,DAT_Pin,0);
	LED_CLK(1);
	LED_push((n/10)%10);
}
```

- Hàm này dùng để set số hiển thị lên trên 2 led 7 đoạn. Ví dụ LED_print(35) thì LED_push(35%10); trả về số 5 sau đó set số 5 lên Led 7 đoạn (hàng chục)

Lưu ý: yêu cầu bài toán thì chân Q0 của U3 không được kết nối với Led 7 đoạn nên ta sẽ đẩy thêm 1 bit bất kì nữa để dữ liệu không bị sai địa chỉ (hiển thị sai)
HAL_GPIO_WritePin(DAT_GPIO_Port,DAT_Pin,0);

LED_CLK(1); 2 dòng này để giải quyết lưu ý trên

LED_push((35/10)%10); trả về số 3 sau đó set số 3 lên Led 7 đoạn (hàng chục) và số 5 được đẩy sang hàng đơn vị

---

- Trong vòng while chính:

```
for (uint32_t i = 0; i <100; i++) {
	LED_print(i);
	HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,1);
	HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,0);
	HAL_Delay(200);
```

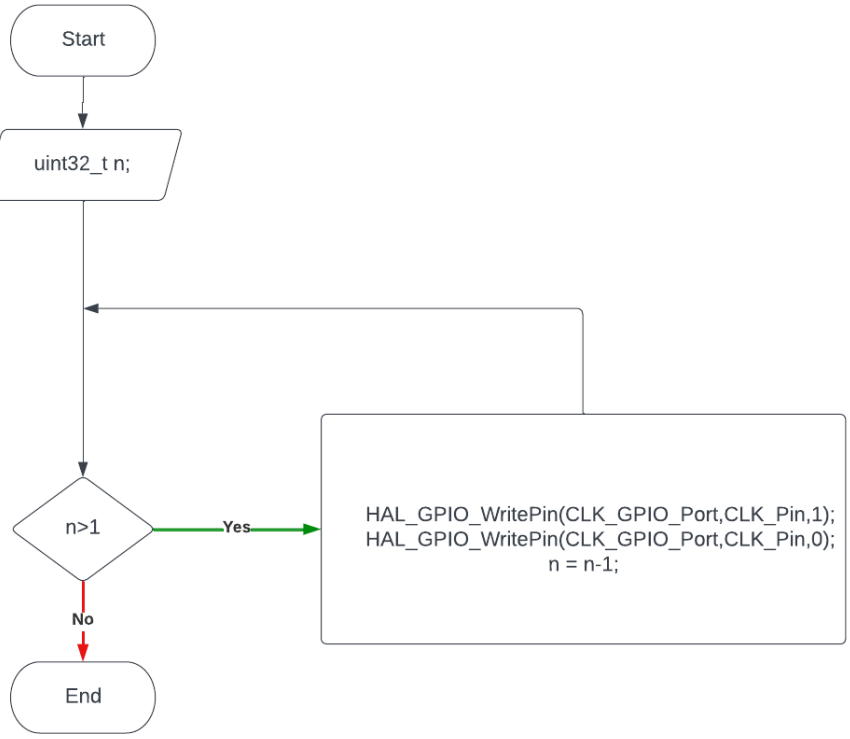
- Tạo 1 vòng for chạy từ 0 -> 99 để Led hiển thị số tương ứng

```
HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,1);
HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,0);
HAL_Delay(200);
```

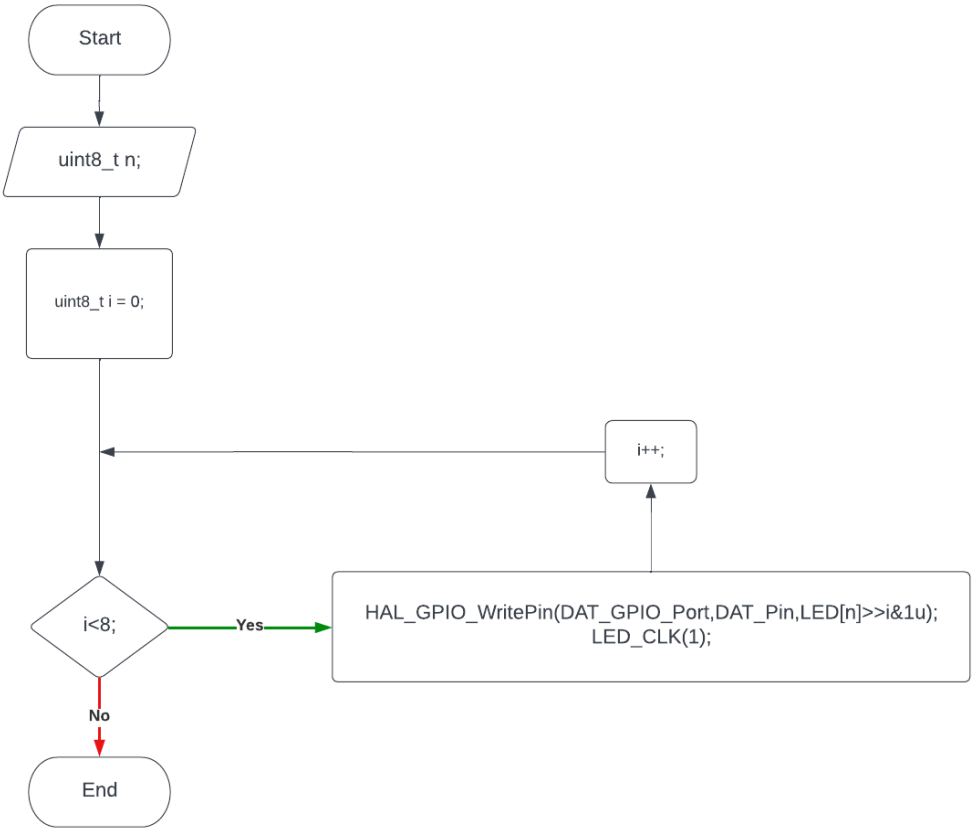
- 3 dòng này dùng để hiển thị số lên Led như đã nói sau đó delay 2s sau mỗi lần hiển thị

## 3.2. Lưu đồ lập trình:

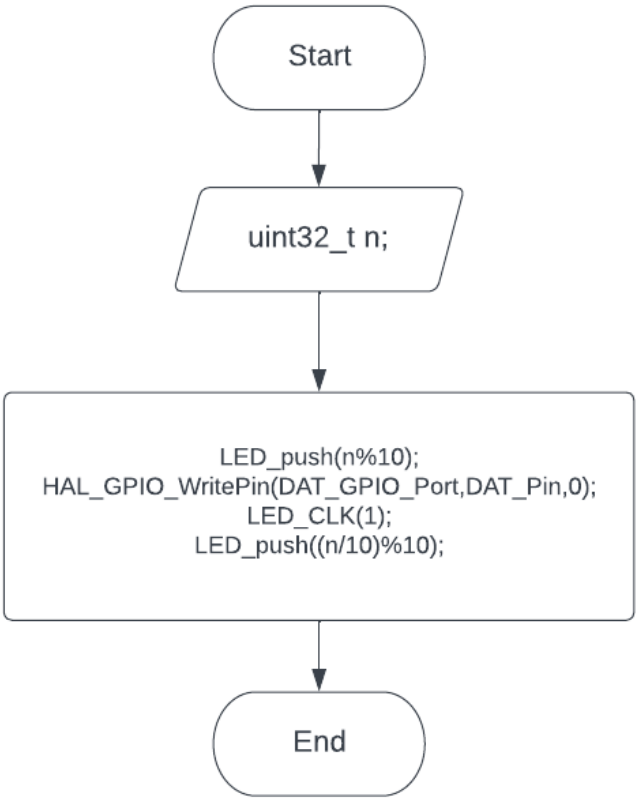
LED_CLK() function



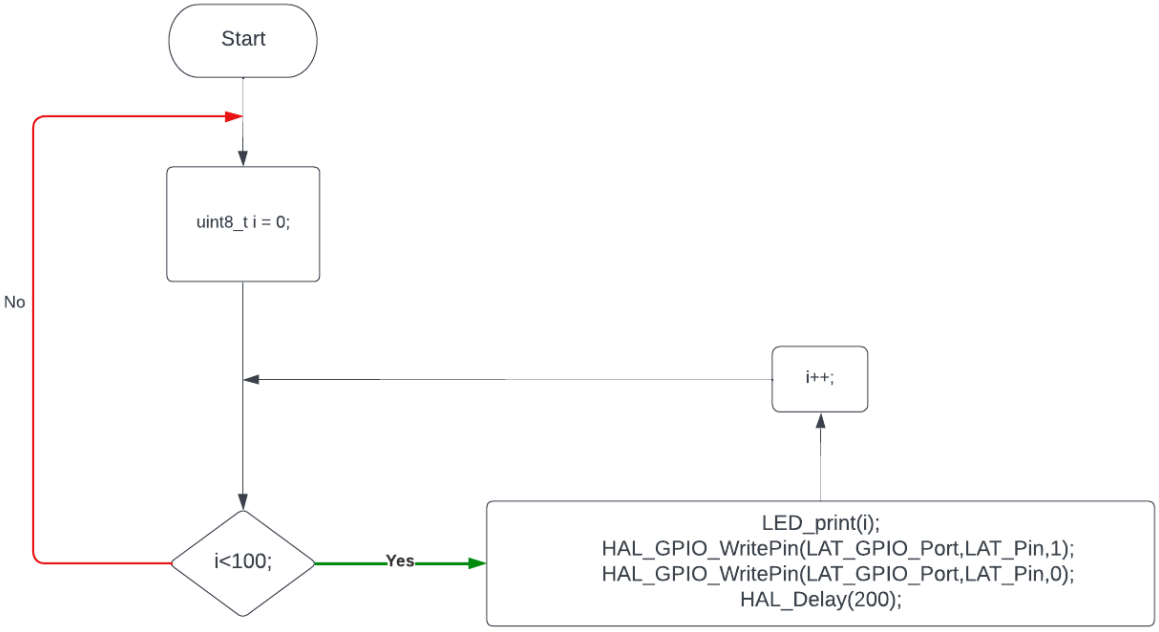

LED_push() function



LED_print() function




Vòng lặp chính



### 3.3. Mã nguồn chương trình:

```
/* USER CODE BEGIN Header */
/**

******************************************************************************
  * @file           : main.c
  * @brief          : Main program body

******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *

******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
```

```
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/

/* USER CODE BEGIN PV */
const uint8_t LED[] = {0xFC,0x60,0xDA,0xF2,0x66,0xB6,0xBE,0xE0,0xFE,0xF6,0x00};
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
// xung clock
void LED_CLK(uint32_t n){
	while(n>0){
		HAL_GPIO_WritePin(CLK_GPIO_Port,CLK_Pin,1);
		HAL_GPIO_WritePin(CLK_GPIO_Port,CLK_Pin,0);
		n = n-1;
```

```
        }
}

// day tung so len led, day 8 lan
void LED_push(uint8_t n){
        for(uint8_t i = 0; i<8; i++){
                HAL_GPIO_WritePin(DAT_GPIO_Port,DAT_Pin,LED[n]>>i&1u);
                LED_CLK(1);
        }
}
// truy cap gia tri tai tung vi tr? cua so n, bat dau tu hang don vi. Sau do dung ham LED_push day so len led lan luot. Co kiem tra so 0 dung dau
void LED_print(uint32_t n){
        LED_push(n%10);
        HAL_GPIO_WritePin(DAT_GPIO_Port,DAT_Pin,0);
        LED_CLK(1);
        LED_push((n/10)%10);
}
/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

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

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

```
  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
                    for (uint32_t i = 0; i <100; i++) {
                            LED_print(i);
                            HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,1);
                            HAL_GPIO_WritePin(LAT_GPIO_Port,LAT_Pin,0);
                            HAL_Delay(200);
                    }
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

```
/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSI;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
```

```
}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, LAT_Pin|DAT_Pin|CLK_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pins : LAT_Pin DAT_Pin CLK_Pin */
  GPIO_InitStruct.Pin = LAT_Pin|DAT_Pin|CLK_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
```

```
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  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,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

## 4. Ghi chú khác (nếu có)