Embedded Systems Lab 01 Report

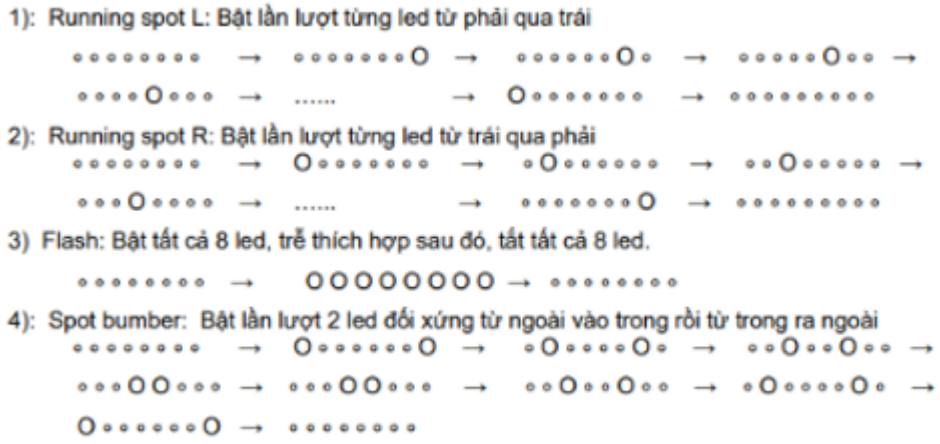
Họ và tên: Lê Ngọc Quang Hưng
MSSV: 20225975

Cam kết: Nội dung và mã nguồn trong báo cáo thực hành này là do tôi và bạn Nguyễn Thị Thu Huyền cùng tự làm. Bất cứ nội dung nào tham khảo từ bên ngoài thì sẽ được nêu rõ nguồn gốc và tác giả.

Table of Contents

Bài 3.1

Đây là bài tập để điều khiển dãy đèn LED theo các mode khác nhau như hình minh hoạ:



Mã nguồn gốc:

```
unsigned char LED_Value;
void DisplayLEDs(int mode)
{
  if (mode == 1)
  {
    LED_Value = (LED_Value >> 1) | (LED_Value << 7);
  }

  if (LED_Value & 0x80)
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET);

  if (LED_Value & 0x40)
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_SET);
```

```
  else
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET);

  if (LED_Value & 0x20)
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET);

  if (LED_Value & 0x10)
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_RESET);

  if (LED_Value & 0x8)
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_RESET);

  if (LED_Value & 0x4)
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);

  if (LED_Value & 0x2)
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);

  if (LED_Value & 0x1)
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
  else
    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
}
```

Mã nguồn sau khi sửa:

```
unsigned char LED_Value;
unsigned char LED_Values_spot_bumper[] = {0x81, 0x42, 0x24, 0x18, 0x24, 0x42};
//added line
int current_index = 0; //added line
void DisplayLEDs(int mode)
{
  if (mode == 1)
  {
    LED_Value = (LED_Value >> 1) | (LED_Value << 7);
  }

  // added code
  else if (mode == 2)
  {
    LED_Value = (LED_Value << 1) | (LED_Value >> 7);
  }
```

```
    else if (mode == 3)
    {
      LED_Value = ~LED_Value;
    }
    else if (mode == 4)
    {
      LED_Value = LED_Values_spot_bumper[current_index];
      current_index++;
      if (current_index >= 6)
      {
        current_index = 0;
      }
    }
    //end added code

    if (LED_Value & 0x80)
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET);

    if (LED_Value & 0x40)
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET);

    if (LED_Value & 0x20)
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET);

    if (LED_Value & 0x10)
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOB, GPIO_PIN_15, GPIO_PIN_RESET);

    if (LED_Value & 0x8)
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, GPIO_PIN_RESET);

    if (LED_Value & 0x4)
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, GPIO_PIN_RESET);

    if (LED_Value & 0x2)
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);

    if (LED_Value & 0x1)
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
    else
      HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
```

```
}
...

int main(void)
{
  ...

  while (1)
  {
    ...
    //added code
    LED_Value = 1;
    for (int i = 0; i <= 15; i++)
        {
          DisplayLEDs(1);
          HAL_Delay(100);
        }
    LED_Value = 0;
    DisplayLEDs(1);


    LED_Value = 1;
    for (int i = 0; i <= 14; i++)
        {
          DisplayLEDs(2);
          HAL_Delay(100);
        }
    LED_Value = 0;
    DisplayLEDs(1);


    LED_Value = 0xff;
    for (int i = 0; i <= 4; i++)
        {
          DisplayLEDs(3);
          HAL_Delay(100);
        }
    LED_Value = 0;
    DisplayLEDs(1);

    LED_Value = 0x81;
    for (int i = 0; i <= 12; i++)
        {
          DisplayLEDs(3);
          HAL_Delay(100);
        }
    LED_Value = 0;
    DisplayLEDs(1);
    //end added code


    ...
  }
}
```

Cơ chế điều khiển LED của hàm `DisplayLEDs` là sử dụng 8 bit giá trị của biến LED_Value để điều khiển LED tương ứng, 1 với bật và 0 với tắt. Phần mã nguồn gốc đã có sẵn ví dụ cho chế độ chạy đèn từ phải qua trái sử dụng cơ chế này. Để có thể chạy các chế độ khác nhau, ta cần thêm vào hàm `DisplayLEDs` các chế độ mới và cập nhật giá trị của biến LED_Value tương ứng với chế độ đó. Trong bài này, ta đã thêm 3 chế độ mới:

- Chế độ 2: chạy đèn từ trái qua phải.

Mã nguồn xử lý chế độ 2:

```
else if (mode == 2)
{
  LED_Value = (LED_Value << 1) | (LED_Value >> 7);
}
```

Tương tự với mode 1 nhưng ngược lại, ta dịch bit sang trái và dịch bit cuối cùng sang đầu để mô phỏng việc bật tắt đèn LED từ trái qua phải.

- Chế độ 3: bật tất cả 8 led và tắt sau một khoảng trễ nhất định.

Mã nguồn xử lý chế độ 3:

```
else if (mode == 3)
{
  LED_Value = ~LED_Value;
}
```

Sử dụng toán tử NOT để đảo ngược giá trị của biến LED_Value, từ tắt cả 8 led thành bật cả 8 led và ngược lại.

- Chế độ 4: bật lần lượt 2 led đối xứng từ ngoài vào trong và từ trong ra ngoài.

Mã nguồn xử lý chế độ 4:

```
unsigned char LED_Values_spot_bumper[] = {0x81, 0x42, 0x24, 0x18, 0x24, 0x42};
int current_index = 0;
else if (mode == 4)
{
  LED_Value = LED_Values_spot_bumper[current_index];
  current_index++;
  if (current_index >= 6)
  {
    current_index = 0;
  }
}
```

Sử dụng mảng LED_Values_spot_bumper để lưu giá trị của biến LED_Value có bit 1 tương ứng với các vị trị của từng 2 led đối xứng. Mỗi lần chạy chế độ 4, ta sẽ chạy qua lần lượt các giá trị trong mảng này để bật tắt 2 led đối xứng. Khi chạy hết mảng, ta sẽ quay lại vị trí đầu tiên để tiếp tục chạy.

Để chạy được các chế độ LED, ta sẽ thêm vào hàm main các đoạn mã điều khiển LED tương ứng với chế độ đó.

Ví dụ:

```
LED_Value = 1;
for (int i = 0; i <= 15; i++)
    {
      DisplayLEDs(1);
      HAL_Delay(100);
    }
LED_Value = 0;
DisplayLEDs(1);
```

Ban đầu ta khởi tạo giá trị của biến LED_Value là giá trị bắt đầu của chế độ đó. Ví dụ ở mode 1 thì giá trị LED_Value khởi đầu sẽ là 1 (0x1) và cứ mỗi lần gọi hàm `DisplayLEDs(1)` thì giá trị của LED_Value sẽ dịch bit sang phải một bit và gán bit cuối cùng vào bit đầu tiên để tạo nên hiệu ứng đèn LED sáng từ phải qua trái.

Đoạn mã trên sẽ chạy LED theo chế độ 1, bật tắt từng led một theo thứ tự từ phải qua trái và được lặp lại hai lần. Do mỗi lần hàm `DisplayLEDs` được gọi, vị trí của LED sáng chỉ dịch sang phải một bit nên ta cần lặp lại 16 lần để chạy hết 8 led lặp lại hai lần.

Cuối cùng, ta gọi hàm `DisplayLEDs(1)` với giá trị 0 để tắt hết các LED trước khi chuyển sang chế độ khác.

Tương tự với các chế độ khác, ta sẽ thay đổi giá trị của LED_Value và số lần gọi hàm `DisplayLEDs` tương ứng để tạo nên hiệu ứng lặp lại hai lần.

Ví dụ với chế độ 3 thì ta cần giá trị khởi đầu của LED_Value là 0xff và 3 lần gọi hàm `DisplayLEDs(3)` để đạt kết quả mong muốn.

Bài 3.2

Bài 3.3

Bài 3.4