1. **PWM控制小灯逐步亮起来**

这个程序能够实现用PWM模块控制小灯逐步亮起来的效果，大家调试一下，理解该程序

样例程序：

#include "stm32l4xx.h"

// 延时函数

void delay\_ms(uint32\_t ms) {

for (uint32\_t i = 0; i < ms \* 1000; i++) {

\_\_asm("nop");

}

}

int main(void) {

// 使能GPIOB时钟

RCC->AHB2ENR |= RCC\_AHB2ENR\_GPIOBEN;

// 使能TIM16时钟

RCC->APB2ENR |= RCC\_APB2ENR\_TIM16EN;

// 配置PB8为复用功能模式

GPIOB->MODER &= ~(GPIO\_MODER\_MODE8\_Msk);

GPIOB->MODER |= (2 << GPIO\_MODER\_MODE8\_Pos);

// 配置PB8复用功能为TIM16\_CH1（AF14）

GPIOB->AFR[1] &= ~(GPIO\_AFRH\_AFSEL8\_Msk);

GPIOB->AFR[1] |= (14 << GPIO\_AFRH\_AFSEL8\_Pos);

// 配置TIM16

// 预分频器

TIM16->PSC = 0;

// 自动重载值

TIM16->ARR = 1000;

// 比较值

TIM16->CCR1 = 0;

// 配置PWM模式1

TIM16->CCMR1 &= ~(TIM\_CCMR1\_OC1M\_Msk);

TIM16->CCMR1 |= (TIM\_CCMR1\_OC1M\_1 | TIM\_CCMR1\_OC1M\_2);

// 使能通道1预装载

TIM16->CCMR1 |= TIM\_CCMR1\_OC1PE;

// 使能主输出

TIM16->BDTR |= TIM\_BDTR\_MOE;

// 使能通道1输出

TIM16->CCER |= TIM\_CCER\_CC1E;

// 使能TIM16

TIM16->CR1 |= TIM\_CR1\_CEN;

uint16\_t dutyCycle = 0;

uint16\_t step = 1;

while (1) {

// 逐渐增加占空比

TIM16->CCR1 = dutyCycle;

delay\_ms(10);

dutyCycle += step;

if (dutyCycle >= TIM16->ARR) {

dutyCycle = 0;

}

}

}

1. **下面的程序也可以实现，但需要大家理解并调试出来**

#include "stm32l4xx.h"

volatile uint16\_t pwm\_value = 0;

volatile int8\_t dir = 1;

void TIM16\_PWM\_Init(void) {

// 时钟使能

RCC->APB2ENR |= RCC\_APB2ENR\_TIM16EN;

RCC->AHB2ENR |= RCC\_AHB2ENR\_GPIOBEN;

// 配置PB8为TIM16\_CH1

GPIOB->MODER &= ~GPIO\_MODER\_MODE8;

GPIOB->MODER |= GPIO\_MODER\_MODE8\_1;

GPIOB->AFR[1] |= (14 << 0); // AF14 = TIM16\_CH1

// 配置TIM16

TIM16->PSC = 31; // 分频后1MHz

TIM16->ARR = 999; // PWM周期1ms

TIM16->CCR1 = 0; // 初始占空比0%

TIM16->CCMR1 |= (6 << 4); // OC1M = PWM模式1（0b110）

TIM16->CCER |= TIM\_CCER\_CC1E;

TIM16->CR1 |= TIM\_CR1\_CEN; // 启动定时器

}

int main(void) {

TIM16\_PWM\_Init();

// 配置SysTick

SysTick->LOAD = 31999; // 1ms中断

SysTick->CTRL = SysTick\_CTRL\_CLKSOURCE\_Msk |

SysTick\_CTRL\_TICKINT\_Msk |

SysTick\_CTRL\_ENABLE\_Msk;

while(1) {

// 主循环无需操作，由中断处理

}

}

// SysTick中断服务函数

void SysTick\_Handler(void) {

static uint16\_t counter = 0;

if (++counter >= 10) { // 每10ms更新一次

counter = 0;

pwm\_value += dir;

if (pwm\_value >= 999) dir = -1;

else if (pwm\_value <= 0) dir = 1;

TIM16->CCR1 = pwm\_value;

}

}