

舵机阻尼模式

API使用说明

设置阻尼模式并设置功率

```
// 舵机阻尼模式
FSUS_STATUS FSUS_DampingMode(Usart_DataTypeDef *usart, uint8_t servoId, uint16_t power);
```

- `usart` 舵机控制对应的串口数据对象 `Usart_DataTypeDef`
- `servoId` 舵机的ID
- `power` 舵机的功率，单位mW

使用示例

```
// 连接在转接板上的串口舵机ID号
uint8_t servoId = 0;
// 阻尼模式下的功率，功率越大阻力越大
uint16_t power = 500;
// 设置舵机为阻尼模式
FSUS_DampingMode(servoUsart, servoId, power);
```

设置阻尼模式的功率

功能简介

设置不同的功率，体验舵机阻尼力的变换。

源代码

```
/*
 * 串口舵机阻尼模式
 */
#include "stm32f10x.h"
#include "usart.h"
#include "sys_tick.h"
#include "fashion_star_uart_servo.h"

// 使用串口1作为舵机控制的端口
// <接线说明>
// STM32F103 PA9(Tx) <----> 串口舵机转接板 Rx
```

```

// STM32F103 PA10(Rx) <----> 串口舵机转接板 Tx
// STM32F103 GND <----> 串口舵机转接板 GND
// STM32F103 V5 <----> 串口舵机转接板 5V
// <注意事项>
// 使用前确保已设置usart.h里面的USART1_ENABLE为1
// 设置完成之后，将下行取消注释
Usart_DataTypeDef* servoUsart = &usart1;

// 连接在转接板上的串口舵机ID号
uint8_t servoId = 0;
// 阻尼模式下的功率，功率越大阻力越大
uint16_t power = 500;
int main (void)
{
    // 滴嗒定时器初始化
    SysTick_Init();
    // 串口初始化
    Usart_Init();
    // 设置舵机为阻尼模式
    FSUS_DampingMode(servoUsart, servoId, power);
    while (1)
    {
        //主循环什么也不做
        // 等待1000ms
        SysTick_DelayMs(1000);
    }
}

```

阻尼模式与角度回读

功能简介

设置舵机为阻尼模式，同时请求舵机的角度。在旋转舵机的情况下，每隔一段时间就更新一下舵机的角度。串口2每隔一段时间打印一下舵机角度信息。

源代码

```

#include "stm32f10x.h"
#include "usart.h"
#include "sys_tick.h"
#include "fashion_star_uart_servo.h"

// 使用串口1作为舵机控制的端口
// <接线说明>
// STM32F103 PA9(Tx) <----> 串口舵机转接板 Rx
// STM32F103 PA10(Rx) <----> 串口舵机转接板 Tx
// STM32F103 GND <----> 串口舵机转接板 GND
// STM32F103 V5 <----> 串口舵机转接板 5V
// <注意事项>
// 使用前确保已设置usart.h里面的USART1_ENABLE为1

```

```

// 设置完成之后，将下行取消注释
Usart_DataTypeDef* servoUsart = &usart1;
// 使用串口2作为日志输出的端口
// <接线说明>
// STM32F103 PA2(Tx) <----> USB转TTL Rx
// STM32F103 PA3(Rx) <----> USB转TTL Tx
// STM32F103 GND <----> USB转TTL GND
// STM32F103 V5 <----> USB转TTL 5V (可选)
// <注意事项>
// 使用前确保已设置usart.h里面的USART2_ENABLE为1
Usart_DataTypeDef* loggingUsart = &usart2;

// 重定向c库函数printf到串口，重定向后可使用printf函数
int fputc(int ch, FILE *f)
{
    while((loggingUsart->pUSARTx->SR&0x40)==0){}
    /* 发送一个字节数据到串口 */
    USART_SendData(loggingUsart->pUSARTx, (uint8_t) ch);
    /* 等待发送完毕 */
    // while (USART_GetFlagStatus(USART1, USART_FLAG_TC) != SET);
    return (ch);
}

FSUS_STATUS statusCode; // 请求包的状态码
uint8_t servoId = 0; // 连接在转接板上的串口舵机ID号
uint16_t power = 500; // 阻尼模式下的功率，功率越大阻力越大
float angle = 0; // 舵机的角度

int main (void)
{
    // 嘀嗒定时器初始化
    SysTick_Init();
    // 串口初始化
    Usart_Init();
    // 设置舵机为阻尼模式
    FSUS_DampingMode(servoUsart, servoId, power);
    while (1)
    {
        // 读取一下舵机的角度
        statusCode = FSUS_QueryServoAngle(servoUsart, servoId, &angle);

        if (statusCode ==FSUS_STATUS_SUCCESS){
            // 成功的读取到了舵机的角度
            printf("[INFO] servo id= %d ; angle = %f\r\n", servoId, angle);
        }else{
            // 没有正确的读取到舵机的角度
            printf("\r\n[INFO] read servo %d angle, status code: %d \r\n",
servoId, statusCode);
            printf("[ERROR]failed to read servo angle\r\n");
        }
        // 等待1000ms
        SysTick_DelayMs(500);
    }
}

```

