串口总线舵机SDK使用手册(Arduino Mega2560)

串口总线舵机SDK使用手册(Arduino Mega2560)

- 安装串口总线舵机的Arduino库
- 演示例程的操作流程
- 舵机对象的创建与初始化
- 舵机通讯检测
  - API-`ping`
  - 例程源码
- 舵机阻尼模式
  - API-`setDamping`
  - 例程源码
- 舵机角度查询
  - API-`queryAngle`
  - 例程源码
- 舵机轮式模式
  - API-`wheelStop`
  - API-`wheelRun`
  - API-`wheelRunNTime`
  - API-`wheelRunNCircle`
  - 例程源码
- 设置舵机角度
  - API-`setAngle`
  - API-`setRawAngleByInterval`
  - API-`setRawAngleByVelocity`
  - API-`isStop`
  - API-`setRange`
  - 例程源码
- 舵机阻塞式等待
  - API-`wait`
  - 例程源码
- 设置舵机角度-多圈模式
  - API-`setRawAngleMTurn`
  - API-`setRawAngleByInterval`
  - API-`setRawAngleMTurnByVelocity`
  - 例程源码
- 舵机扭力开关
  - API-`setTorque`
  - 例程源码
- 舵机标定
  - API-`calibration`
  - API-`angleReal2Raw`
  - API-`angleRaw2Real`
  - 例程源码
- 舵机转速设置
  - API-`setSpeed`
- 舵机数据读取
  - 示例源码

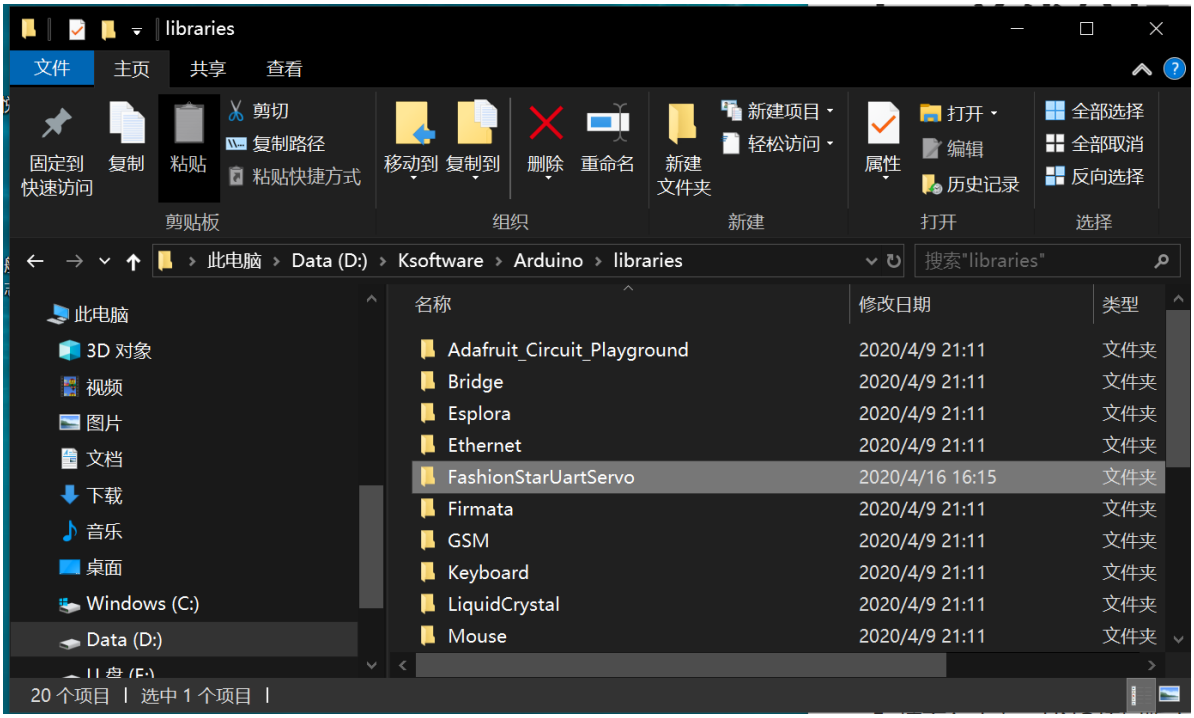
作者: 阿凯|Kyle

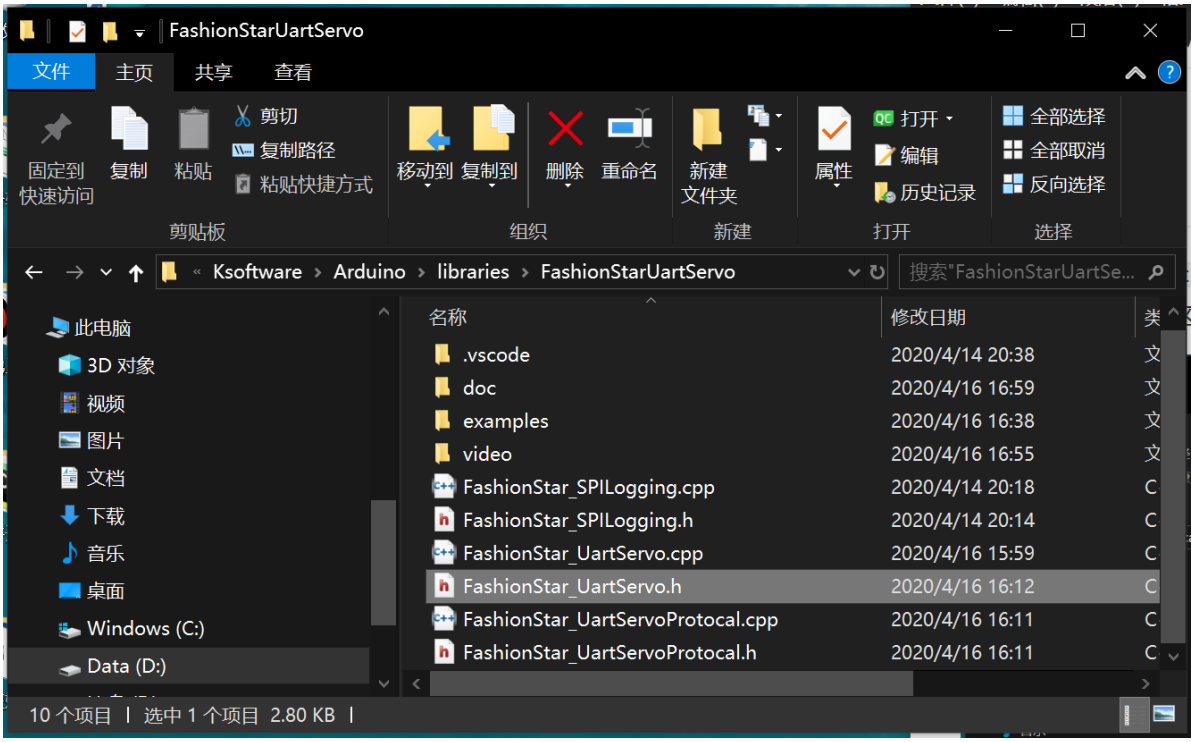
邮箱: kyle.xing@fashionstar.com.hk

更新时间: 2021 / 07 / 15

安装串口总线舵机的Arduino库

将`FashionStar_UartServo`这个工程文件, 整体拷贝到`Arduino IDE`安装路径下的`libraries`这个文件夹.

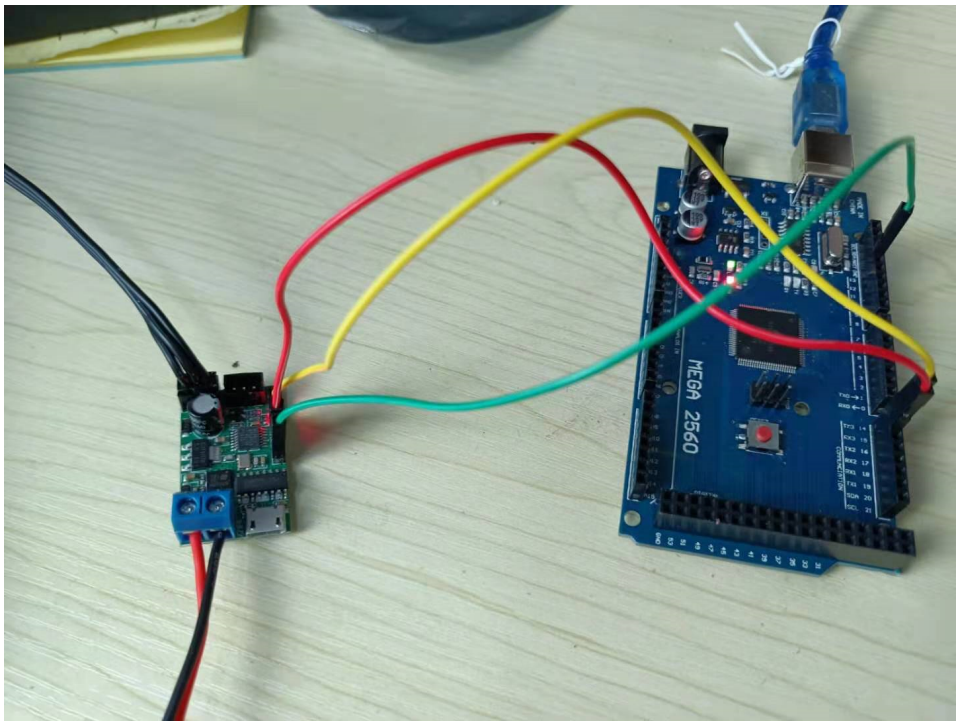




演示例程的操作流程

1. 接线 - Arduino Mega2560跟串口转接板UART接口

Arduino Mega2560	USB转TTL模块
D15(串口3 RX 接收端)	Tx (USB转TTL模块的接收端)
D14 (串口3 Tx 发送端)	Rx （USB转TTL模块的发送端）
GND	GND



2. 接线 - Arduino Mega2560与PC 通过USB线相连接
3. 在PC端打开Arduino IDE, 打开FashionStar串口总线舵机的例程文件.

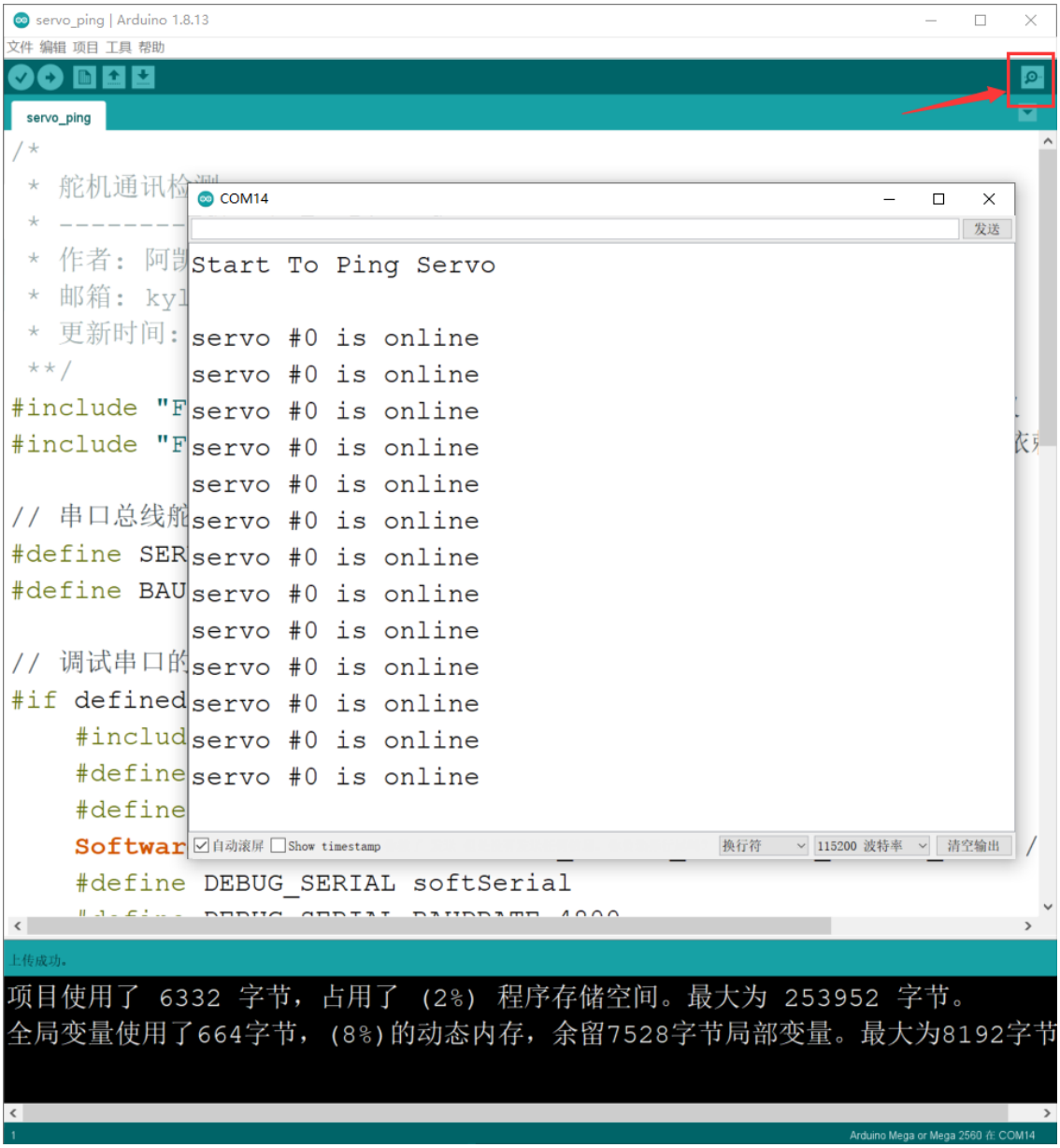
   打开Arduino示例代码-流程演示.mp4

4. 选择开发板型号为Arduino Mega2560, 选择端口号

5. 给串口舵机转接板供电， 电压7.2V.

6. 编译并烧录固件

7. 查看日志输出



舵机对象的创建与初始化

```
#include "FashionStar_UartServoProtocol.h" // 串口总线舵机通信协议
#include "FashionStar_UartServo.h" // 串口总线舵机SDK
```

`FashionStar_UartServoProtocol` 用来处理舵机的底层通信协议的逻辑(数据帧的收发, 数据校验等).

`FashionStar_UartServo` 是舵机的SDK, 是在协议上层的更高一级的封装.

创建一个串口总线舵机通信协议对象`FSUS_Protocol`, 构造器里面需要填写Arduino与串口总线舵机通信的波特率, 默认为`115200`.

```
#define BAUDRATE 115200 // 波特率

FSUS_Protocol protocol(BAUDRATE); //协议
```

创建一个`FSUS_Servo`舵机对象, 创建的时候需要传入舵机的ID, 以及通信协议对象的指针`&protocol`.

舵机的ID取值范围为`0-254

```
#define SERVO_ID 0 //舵机ID号

FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机
```

接下来需要在`setup()`函数里对通信协议对象以及舵机对象进行初始化

```
void setup(){
    ...
    protocol.init(); // 舵机通信协议初始化
    uservo.init(); // 串口总线舵机初始化
    ...
}

```

舵机通讯检测

API-`ping`

调用舵机的`ping()`函数用于舵机的通信检测, 判断舵机是否在线.

```
bool isOnline = uservo.ping(); // 舵机通讯检测
```

例程源码

`servo_ping.ino`

```
/*
 * 舵机通讯检测
 * -------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 **/
#include "FashionStar_UartServoProtocol.h" // 串口总线舵机通信协议
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 串口总线舵机配置
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
```

```
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE);        //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

void setup(){
    protocol.init(); // 舵机通信协议初始化
    uservo.init(); // 串口总线舵机初始化
    // 打印例程信息
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE);
    DEBUG_SERIAL.println("Start To Ping Servo\n");
}

void loop(){
    bool isOnline = uservo.ping(); // 舵机通讯检测
    String message = "servo #"+String(uservo.servoId,DEC) + " is ";  // 日志输出
    if(isOnline){
        message += "online";
    }else{
        message += "offline";
    }
    // 调试串口初始化
    DEBUG_SERIAL.println(message);
    // 等待1s
    delay(1000);
}

```

输出日志

```
Start To Ping Servo


servo #0 is online.

servo #0 is online.

servo #0 is online.

servo #0 is online.

```

舵机阻尼模式

API-`setDamping`

设置舵机为阻尼模式.

```
void FSUS_Servo::setDamping(FSUS_POWER_T power)
```

输入参数

- `power` 舵机的功率，单位为mW. 功率值越大，旋转舵机的时候阻尼力也就越大

使用示例

```
#define DAMPING_POWER 800 // 阻尼模式下的功率(单位mW) 500,800,1000

uservo.setDamping(DAMPING_POWER);
```

例程源码

`servo_dammping.ino`

```
/*
 * 设置舵机为阻尼模式
 * 调整参数`DAMPING_POWER`感受不同的阻尼力
 * --------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 **/
#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 串口总线舵机配置参数
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率
#define DAMPING_POWER 800 // 阻尼模式下的功率(单位mW) 500,800,1000

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

```

```
FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

void setup(){

    protocol.init(); // 通信协议初始化
    uservo.init(); // 舵机初始化
    // 打印日志
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE);
    DEBUG_SERIAL.println("Set Servo Mode To Dammping");
    // 设置电机的阻尼系数
    uservo.setDamping(DAMPING_POWER);
}

void loop(){
    // TODO;
}

```

日志输出

```
Set Servo Mode To Dammping
```

舵机角度查询

API-`queryAngle`

查询舵机当前的真实角度，向舵机发送角度查询指令，并将角度值赋值给舵机对象的`curAngle`属性

```
FSUS_SERVO_ANGLE_T FSUS_Servo::queryAngle()
```

输入参数

- <无>

输出参数

- `curAngle` 舵机当前的真实角度

使用示例

示例1

```
float curAngle = uservo.queryAngle()
```

示例2

```
// 舵机角度查询 (更新角度)
uservo.queryAngle();
// 通过.curAngle访问当前的真实角度
uservo.curAngle
```

例程源码

servo_query_angle.ino

```
/*
 * 舵机角度回读实验
 * 用手掰动舵机, 角度回读并将角度读数通过SPI发送
 * ------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 **/
#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 串口总线舵机配置
#define SERVO_ID 0 //舵机ID号
#define DAMPING_POWER 800 // 阻尼模式下的功率(单位mW) 500,800,1000
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif



FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机


void setup(){
    protocol.init();                    // 通信协议初始化
    uservo.init();                      //舵机角度初始化
    uservo.setDamping(DAMPING_POWER);   // 舵机设置为阻尼模式
    // 打印例程信息
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE);
    DEBUG_SERIAL.println("Query Servo Angle\n");
}

void loop(){
    // 舵机角度查询 (更新角度)
    uservo.queryRawAngle();
    // 日志输出
```

```
    String message = "Status Code: " + String(uservo.protocol->responsePack.recv_status, DEC) + " servo #"+String(uservo.servoId, DEC) + " , Current Angle = "+String(uservo.curRawAngle, 1)+" deg";
    DEBUG_SERIAL.println(message);
    // 等待1s
    delay(1000);
}
```

输出日志

```
Query Servo Angle

Status Code: 0 servo #0 , Current Angle = -99.0

Status Code: 0 servo #0 , Current Angle = -99.0

Status Code: 0 servo #0 , Current Angle = -99.0

```

舵机轮式模式

API-`wheelStop`

轮式模式, 停止旋转

函数原型

```
void FSUS_Servo::wheelStop()
```

输入参数

<无>

API-`wheelRun`

轮子持续旋转

函数原型

```
void FSUS_Servo::wheelRun(uint8_t is_cw)
```

输入参数

- `is_cw` 轮子的旋转方向
  - `0`: 逆时针
  - `1`: 顺时针

API-`wheelRunNTime`

轮子旋转特定的时间

函数原型

```
void FSUS_Servo::wheelRunNTime(uint8_t is_cw, uint16_t time_ms)
```

输入参数

- `is_cw`: 轮子的旋转方向
  - `0`: 逆时针
  - `1`: 顺时针
- `time_ms`: 持续旋转的时间，单位为ms

API-`wheelRunNCircle`

轮子旋转特定的圈数

函数原型

```
void FSUS_Servo::wheelRunNCircle(uint8_t is_cw, uint16_t circle_num)
```

输入参数

- `is_cw`: 轮子的旋转方向
  - `0`: 逆时针
  - `1`: 顺时针
- `circle_num`: 轮子旋转的圈数

例程源码

`servo_wheel_mode.ino`

```
/*
 * 测试舵机轮式模式
 * 提示: 拓展板上电之后，记得按下Arduino的RESET按键
 * -------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */
#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 配置参数
#define BAUDRATE 115200 // 波特率
#define SERVO_ID 0 //舵机ID号

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

/* 轮子持续旋转指令与停止指令测试 */
```

```
void testWheelRunAndStop(){
    uservo.wheelRun(FSUS_CCW); // 轮子持续旋转，方向为逆时针
    delay(2000);             // 等待2s
    uservo.wheelStop();
    delay(2000);             // 等待2s
    uservo.wheelRun(FSUS_CW); // 轮子持续旋转
    delay(2000);             // 等待2s
    uservo.wheelStop();
    delay(2000);             // 等待2s
}

/* 测试轮子旋转特定的时间 */
void testWheelRunNTime(){
    uservo.wheelRunNTime(FSUS_CW, 5000); // 轮子持续旋转5s(顺时针)
    delay(5000);
    uservo.wheelRunNTime(FSUS_CCW, 5000); // 轮子持续旋转5s(逆时针)
    delay(5000);
}

/* 测试轮子旋转特定的圈数 */
void testWheelRunNCircle(){
    uint16_t nCircle = 2; // 旋转圈数
    uint16_t delayMsEstimate = (uint16_t)(360.0 * nCircle / uservo.speed *
1000); // 估计旋转的时间
    uservo.wheelRunNCircle(FSUS_CW, 2); // 轮子持续旋转2圈(顺时针)
    delay(delayMsEstimate);              // 等到轮子旋转到特定的位置

    uservo.wheelRunNCircle(FSUS_CCW, 2);// 轮子持续旋转2圈(逆时针)
    delay(delayMsEstimate);              // 等到轮子旋转到特定的位置}
}

void setup(){
    protocol.init();         // 通信协议初始化
    uservo.init();           //舵机角度初始化
    uservo.setSpeed(100);     // 设置转速为20°/s

    // 测试持续旋转与停止
    // testRunAndStop();

    // 测试旋转特定的时间
    // testWheelRunNTime();

    // 测试旋转特定的圈数
    testWheelRunNCircle();
}

void loop(){
}
```

设置舵机角度

API-`setAngle`

设定舵机的角度

函数原型

```
/* 设置舵机的原始角度 */
void FSUS_Servo::setRawAngle(FSUS_SERVO_ANGLE_T rawAngle, FSUS_INTERVAL_T interval, FSUS_POWER_T power)
```

```
/* 设置舵机的原始角度 */
void FSUS_Servo::setRawAngle(FSUS_SERVO_ANGLE_T rawAngle, FSUS_INTERVAL_T interval)
```

```
/* 设置舵机的原始角度 */
void FSUS_Servo::setRawAngle(FSUS_SERVO_ANGLE_T rawAngle)
```

输入参数

- `rawAngle`：舵机的目标角度，单位°
- `interval` 舵机旋转的周期, 单位ms
- `power` 最大功率, 单位mW

API-`setRawAngleByInterval`

函数原型

```
// 设置舵机的原始角度(指定周期)
void FSUS_Servo::setRawAngleByInterval(FSUS_SERVO_ANGLE_T rawAngle, FSUS_INTERVAL_T interval, FSUS_INTERVAL_T t_acc, FSUS_INTERVAL_T t_dec, FSUS_POWER_T power)
```

输入参数

- `rawAngle`：舵机的目标角度，单位°
- `interval`： 舵机旋转的周期, 单位ms
- `t_acc`：加速时间
- `t_dec`：减速时间
- `power`： 最大功率, 单位mW

API-`setRawAngleByVelocity`

函数原型

```
// 设定舵机的原始角度(指定转速)
void FSUS_Servo::setRawAngleByVelocity(FSUS_SERVO_ANGLE_T rawAngle,
FSUS_SERVO_SPEED_T velocity, FSUS_INTERVAL_T t_acc, FSUS_INTERVAL_T t_dec,
FSUS_POWER_T power)
```

输入参数

- `rawAngle` : 舵机的目标角度，单位 °
- `velocity`: 舵机旋转的转速, 单位°/s
- `t_acc` : 加速时间
- `t_dec` : 减速时间
- `power` : 最大功率, 单位mW

API-`isStop`

判断舵机是否在旋转, 是否是静止.

改函数在执行的时候,会先查询舵机当前的角度, 返回对比跟目标角度`targetAngle`之间的差值是否小于控制死区.

函数原型

```
bool FSUS_Servo::isStop()
```

输入参数

<无>

返回参数

- `is_stop`:
  - `true` 舵机已经到达目标角度, 停下来了
  - `false` 舵机还没有到达目标角度,正在旋转

API-`setRange`

设置舵机的角度范围

函数原型

```
void FSUS_Servo::setAngleRange(FSUS_SERVO_ANGLE_T minAngle,
FSUS_SERVO_ANGLE_T maxAngle)
```

输入参数

- `minAngle` : 舵机角度下限
- `maxAngle` : 舵机角度上限

输出参数

<无>

例程源码

```
/*
 * 设置舵机的角度(单圈模式)
 * 提示: 拓展板上电之后, 记得按下Arduino的RESET按键
 * --------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */

#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 串口总线舵机配置参数
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

uint16_t interval;  // 运行周期 单位ms
uint16_t t_acc;     // 加速时间 单位ms
uint16_t t_dec;     // 减速时间 单位ms
float velocity;         // 目标转速 单位°/s

/* 等待并报告当前的角度*/
void waitAndReport(){
    uservo.wait();          // 等待舵机旋转到目标角度
    DEBUG_SERIAL.println("Real Angle = " + String(uservo.curRawAngle, 1) + "
Target Angle = "+String(uservo.targetRawAngle, 1));
    delay(2000); // 暂停2s

}

void setup(){
    protocol.init(); // 通信协议初始化
    uservo.init(); //舵机角度初始化
    // 打印例程信息
```

```
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE); // 初始化软串口的波特率
    DEBUG_SERIAL.println("Set Servo Angle");
}

void loop(){
    DEBUG_SERIAL.println("Set Angle = 90°");
    uservo.setRawAngle(90.0);  // 设置舵机的角度
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = -90°");
    uservo.setRawAngle(-90);
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = 90° - Set Interval = 500ms");
    interval = 1000;
    t_acc = 100;
    t_dec = 100;
    uservo.setRawAngleByInterval(90, interval, t_acc, t_dec, 0);
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = -90° - Set Velocity = 200°/s");
    velocity = 200.0;
    t_acc = 100;
    t_dec = 100;
    uservo.setRawAngleByVelocity(-90, velocity, t_acc, t_dec, 0);
    waitAndReport();
}
```

输出日志

```
Set Angle = 90°
Real Angle = 89.7 Target Angle = 90.0
Set Angle = -90°
Real Angle = -89.6 Target Angle = -90.0
Set Angle = 90° - Set Interval = 500ms
Real Angle = 89.7 Target Angle = 90.0
Set Angle = -90° - Set Velocity = 200°/s
Real Angle = -89.6 Target Angle = -90.0
```

舵机阻塞式等待

API-`wait`

等待舵机旋转到目标角度, 阻塞式.

函数原型

```
void FSUS_Servo::wait()
```

输入参数

<无>

输出参数

<无>

例程源码

`servo_wait.ino`

```
/*
 * 测试wait()函数,轮询角度直到舵机旋转到目标位置
 * 提示: 拓展板上电之后, 记得按下Arduino的RESET按键
 * --------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */
#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 串口总线舵机配置参数
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

void setup(){

    protocol.init(); // 通信协议初始化
    uservo.init(); //舵机初始化
    // 打印例程信息
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE);
    DEBUG_SERIAL.println("Test Wait");
}

void loop(){
    DEBUG_SERIAL.println("Set Angle = 90.0");
    uservo.setAngle(90.0); // 设置舵机的角度
    uservo.wait();
    DEBUG_SERIAL.println("Real Angle = "+String(uservo.curAngle, 2));

```

```
    DEBUG_SERIAL.println("Set Angle = -90.0");
    uservo.setAngle(-90);
    uservo.wait();
    DEBUG_SERIAL.println("Real Angle = "+String(uservo.curAngle, 2));
}
```

输出日志

```
Set Angle = -90.0
Real Angle = -89.00
Set Angle = 90.0
Real Angle = 89.80
Set Angle = -90.0
Real Angle = -89.00
Set Angle = 90.0
Real Angle = 89.80
Set Angle = -90.0
Real Angle = -89.00
Set Angle = 90.0
Real Angle = 89.80
Set Angle = -90.0
Real Angle = -89.00
```

设置舵机角度-多圈模式

API-`setRawAngleMTurn`

函数原型

```
// 设定舵机的原始角度(多圈)
void FSUS_Servo::setRawAngleMTurn(FSUS_SERVO_ANGLE_T rawAngle,
FSUS_INTERVAL_T_MTURN interval, FSUS_POWER_T power)
```

```
// 设定舵机的原始角度(多圈)
void FSUS_Servo::setRawAngleMTurn(FSUS_SERVO_ANGLE_T rawAngle,
FSUS_INTERVAL_T_MTURN interval)
```

```
// 设定舵机的原始角度(多圈)
void FSUS_Servo::setRawAngleMTurn(FSUS_SERVO_ANGLE_T rawAngle)
```

输入参数

- `rawAngle`：舵机的目标角度，单位°
- `interval` 舵机旋转的周期, 单位ms
- `power` 最大功率，单位mW

输出参数

<无>

API-`setRawAngleByInterval`

函数原型

```
// 设定舵机的原始角度(多圈+指定周期)
void FSUS_Servo::setRawAngleMTurnByInterval(FSUS_SERVO_ANGLE_T rawAngle, FSUS_INTERVAL_T_MTURN interval, FSUS_INTERVAL_T t_acc, FSUS_INTERVAL_T t_dec, FSUS_POWER_T power)
```

输入参数

- `rawAngle`：舵机的目标角度，单位°
- `interval` 舵机旋转的周期, 单位ms
- `t_acc` 加速时间，单位ms
- `t_dec` 减速时间，单位ms
- `power` 最大功率，单位mW

输出参数

<无>

API-`setRawAngleMTurnByVelocity`

函数原型

```
// 设定舵机的原始角度(多圈+指定转速)
void FSUS_Servo::setRawAngleMTurnByVelocity(FSUS_SERVO_ANGLE_T rawAngle, FSUS_SERVO_SPEED_T velocity, FSUS_INTERVAL_T t_acc, FSUS_INTERVAL_T t_dec, FSUS_POWER_T power)
```

输入参数

- `rawAngle`：舵机的目标角度，单位°
- `velocity`： 舵机旋转的速度，单位°/s
- `t_acc`： 加速时间，单位ms
- `t_dec`： 减速时间，单位ms
- `power`： 最大功率，单位mW

输出参数

<无>

例程源码

servo_set_angle_mturn.ino

```
/*
 * 设置舵机的角度(多圈模式)
 * 提示: 拓展板上电之后, 记得按下Arduino的RESET按键
 * -------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */

#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖


// 串口总线舵机配置参数
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

uint32_t interval;  // 运行周期 单位ms
uint16_t t_acc;     // 加速时间 单位ms
uint16_t t_dec;     // 减速时间 单位ms
float velocity;          // 目标转速 单位°/s

/* 等待并报告当前的角度*/
void waitAndReport(){
    uservo.wait();          // 等待舵机旋转到目标角度
    DEBUG_SERIAL.println("Real Angle = " + String(uservo.curRawAngle, 1) + "
Target Angle = "+String(uservo.targetRawAngle, 1));
    delay(2000); // 暂停2s
}

void setup(){
    protocol.init(); // 通信协议初始化
    uservo.init(); //舵机角度初始化
    // 打印例程信息
```

```
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE); // 初始化软串口的波特率
    DEBUG_SERIAL.println("Set Servo Angle");
}

void loop(){
    DEBUG_SERIAL.println("Set Angle = 900°");
    uservo.setRawAngleMTurn(900.0);  // 设置舵机的角度
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = -900.0°");
    uservo.setRawAngleMTurn(-900.0);
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = 900° - Set Interval = 10s");
    interval = 10000;
    t_acc = 100;
    t_dec = 100;
    uservo.setRawAngleMTurnByInterval(900, interval, t_acc, t_dec, 0);
    waitAndReport();

    DEBUG_SERIAL.println("Set Angle = -900° - Set Velocity = 200°/s");
    velocity = 200.0;
    t_acc = 100;
    t_dec = 100;
    uservo.setRawAngleMTurnByVelocity(-900, velocity, t_acc, t_dec, 0);
    waitAndReport();
}
```

输出日志

```
Set Angle = 900°
Set Servo Angle
Set Angle = 900°
Real Angle = 899.0 Target Angle = 900.0
Set Angle = -900.0°
Real Angle = -899.0 Target Angle = -900.0
Set Angle = 900° - Set Interval = 10s
Real Angle = 899.0 Target Angle = 900.0
Set Angle = -900° - Set Velocity = 200°/s
Real Angle = -899.0 Target Angle = -900.0
```

舵机扭力开关

API-`setTorque`

函数原型

```
void FSUS_Servo::setTorque(bool enable)
```

输入参数

- `enable`：扭力是否开启

- `true`: 开启扭力
- `false`: 关闭扭力

使用示例

```
uservo.setTorque(true); // 开启扭力
```

例程源码

`servo_torque.ino`

```
/*
 * 测试舵机扭力开关
 * 提示: 拓展板上电之后, 记得按下Arduino的RESET按键
 * -------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */
#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 配置参数
#define BAUDRATE 115200 // 波特率
#define SERVO_ID 0 //舵机ID号

FSUS_Protocol protocol; //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机
void setup(){
    protocol.init(); // 通信协议初始化
    uservo.init(); //舵机初始

    uservo.setTorque(true); // 开启扭力
    // uservo.setTorque(false); // 开启扭力
}

void loop(){

}
```

舵机标定

API-`calibration`

在`FSUS_Servo`类里面, 有两个跟标定相关的参数:

```
class FSUS_Servo{
public:
    ...
    float kAngleReal2Raw; // 舵机标定数据-舵机角度与位置之间的比例系数
    float bAngleReal2Raw; // 舵机标定数据-舵机角度与位置转换过程中的偏移量
    ...
}
```

舵机真实角度跟原始角度的映射关系如下:

$$angleRaw = kAngleReal2Raw \cdot angleReal + bAngleReal2Raw \tag{1}$$

函数原型

```
void FSUS_Servo::calibration(FSUS_SERVO_ANGLE_T rawA, FSUS_SERVO_ANGLE_T realA, FSUS_SERVO_ANGLE_T rawB, FSUS_SERVO_ANGLE_T realB)
```

输入参数

- `rawA` 在位置A时刻舵机原始的角度
- `realA` 在位置A时刻舵机真实的角度
- `rawB` 在位置B时刻舵机原始的角度
- `realB` 在位置B时刻舵机真实的角度

使用示例

```
// 设置舵机的标定点
// 样本1
#define SERVO_REAL_ANGLE_A 90 // 舵机真实角度
#define SERVO_RAW_ANGLE_A -86.2 // 舵机原始角度
// 样本2
#define SERVO_REAL_ANGLE_B -90 // 舵机真实角度
#define SERVO_RAW_ANGLE_B 91.9 // 舵机原始角度


// 输入舵机标定数据
uservo.calibration(
    SERVO_RAW_ANGLE_A,SERVO_REAL_ANGLE_A,\
    SERVO_RAW_ANGLE_B,SERVO_REAL_ANGLE_B);
```

函数原型

```
void FSUS_Servo::calibration(float kAngleReal2Raw, float bAngleReal2Raw);
```

输入参数

- `kAngleReal2Raw` :舵机标定数据-舵机角度与位置之间的比例系数
- `bAngleReal2Raw` : 舵机标定数据-舵机角度与位置转换过程中的偏移量

API-`angleReal2Raw`

舵机真实角度转换为舵机原始角度

函数原型

```
// 真实角度转化为原始角度
FSUS_SERVO_ANGLE_T FSUS_Servo::angleReal2Raw(FSUS_SERVO_ANGLE_T realAngle);
```

输入参数

- `realAngle`: 舵机真实角度

返回参数

- `rawAngle`: 舵机原始角度

API-`angleRaw2Real`

舵机原始角度转化为真实角度

函数原型

```
 // 原始角度转换为真实角度
FSUS_SERVO_ANGLE_T FSUS_Servo::angleRaw2Real(FSUS_SERVO_ANGLE_T rawAngle);
```

输入参数

- `rawAngle`: 舵机原始角度

返回参数

- `realAngle`: 舵机真实角度

例程源码

```
/*
 * 测试舵机标定
 * 提示: 拓展板上电之后, 记得按下Arduino的RESET按键
 * ---------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 */

#include "FashionStar_UartServoProtocol.h"
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖


// 串口总线舵机配置参数
#define SERVO_ID 0 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 设置舵机的标定点
```

```
// 样本1
#define SERVO_REAL_ANGLE_A 90 // 舵机真实角度
#define SERVO_RAW_ANGLE_A -86.2 // 舵机原始角度
// 样本2
#define SERVO_REAL_ANGLE_B -90 // 舵机真实角度
#define SERVO_RAW_ANGLE_B 91.9 // 舵机原始角度

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

void setup(){
    protocol.init(); // 通信协议初始化
    uservo.init(); //舵机角度初始化
    // 调试串口初始化
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE); // 初始化软串口的波特率
    DEBUG_SERIAL.println("Set Servo Angle");
    // 输入舵机标定数据
    uservo.calibration(
        SERVO_RAW_ANGLE_A,SERVO_REAL_ANGLE_A,\
        SERVO_RAW_ANGLE_B,SERVO_REAL_ANGLE_B);

    // 打印舵机标定数据
    DEBUG_SERIAL.println("kAngleReal2Raw = "+String(uservo.kAngleReal2Raw,2)
+ \
        "; bAngleReal2Raw = " + String(uservo.bAngleReal2Raw, 2));
}

void loop(){
    DEBUG_SERIAL.println("Set Angle = 90°");
    uservo.setAngle(90.0); // 设置舵机的角度
    uservo.wait();
    delay(2000);

    DEBUG_SERIAL.println("Set Angle = -90°");
    uservo.setAngle(-90);
    uservo.wait();
    delay(2000);
}
```

输出日志

```
Set Servo Angle

kAngleReal2Raw = -0.99; bAngleReal2Raw = 2.85

Set Angle = 90

Set Angle = -90
```

舵机转速设置

API-`setSpeed`

函数原型

```
void FSUS_Servo::setSpeed(FSUS_SERVO_SPEED_T speed)
```

输入参数

- `speed` 舵机的平均转速, 单位°/s

返回参数

<无>

舵机数据读取

示例源码

`servo_data_read.ino`

```
/*
 * 舵机数据读取实验
 * -------------------------
 * 作者: 阿凯|Kyle
 * 邮箱: kyle.xing@fashionstar.com.hk
 * 更新时间: 2021/06/02
 **/
#include "FashionStar_UartServoProtocol.h" // 串口总线舵机通信协议
#include "FashionStar_UartServo.h" // Fashion Star串口总线舵机的依赖

// 配置
#define SERVO_ID 4 //舵机ID号
#define BAUDRATE 115200 // 波特率

// 调试串口的配置
#if defined(ARDUINO_AVR_UNO)
    #include <SoftwareSerial.h>
    #define SOFT_SERIAL_RX 6
    #define SOFT_SERIAL_TX 7
    SoftwareSerial softSerial(SOFT_SERIAL_RX, SOFT_SERIAL_TX); // 创建软串口
```

```
    #define DEBUG_SERIAL softSerial
    #define DEBUG_SERIAL_BAUDRATE 4800
#elif defined(ARDUINO_AVR_MEGA2560)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#elif defined(ARDUINO_ARCH_ESP32)
    #define DEBUG_SERIAL Serial
    #define DEBUG_SERIAL_BAUDRATE 115200
#endif

FSUS_Protocol protocol(BAUDRATE); //协议
FSUS_Servo uservo(SERVO_ID, &protocol); // 创建舵机

// 读取数据
uint16_t voltage;       // 电压 mV
uint16_t current;       // 电流 mA
uint16_t power;         // 功率 mW
uint16_t temperature;   // 温度 ℃

void setup(){

    protocol.init(); // 舵机通信协议初始化
    uservo.init(); // 串口总线舵机初始化
    // 打印例程信息
    DEBUG_SERIAL.begin(DEBUG_SERIAL_BAUDRATE);
    DEBUG_SERIAL.println("Start To Test Servo Data Read \n"); // 打印日志

    uservo.setAngle(-25.0,  1000, 200); // 设置舵机角度(限制功率)
}

void loop(){
    voltage = uservo.queryVoltage();
    current = uservo.queryCurrent();
    power = uservo.queryPower();
    temperature = uservo.queryTemperature();
    DEBUG_SERIAL.println("voltage: "+String((float)voltage, 1)+" mV\n");
    delay(100);
    DEBUG_SERIAL.println("current: "+String((float)current, 1)+" mA\n");
    delay(100);
    DEBUG_SERIAL.println("power: "+String((float)power, 1)+" mW\n");
    delay(100);
    DEBUG_SERIAL.println("temperature: "+String((float)temperature, 1)+"
Celsius\n");
    delay(1000);
}
```

输出日志

> 注: 这个数值范围也不对啊， 要么就是测量的数值并不准确。

```

voltage: 9473.0 mV
current: 2.0 mA
power: 3.0 mW
```

```
temperature: 46340.0 Celsius

voltage: 9985.0 mV
current: 2.0 mA
power: 3.0 mW
temperature: 46340.0 Celsius

voltage: 9473.0 mV
current: 2.0 mA
power: 3.0 mW
temperature: 46340.0 Celsius

voltage: 9985.0 mV
current: 2.0 mA
power: 3.0 mW
temperature: 46340.0 Celsius

voltage: 9473.0 mV
current: 2.0 mA
power: 3.0 mW
temperature: 46084.0 Celsius

voltage: 9473.0 mV
current: 2.0 mA
power: 3.0 mW
temperature: 46084.0 Celsius

voltage: 9473.0 mV
current: 2.0 mA
power: 3.0 mW
```