7bot Firmware function Library of robot arm Servo Motor handbook

A New servo firmware Library based on Serial Communication

introduce

The firmware library is a function package that consists of macros, classes, and external functions. It contains each

Through the use of this firmware function library, the user can easily manipulate the steering machine without mastering the details. Therefore, the firmware library can greatly reduce the development time of the user and reduce the development cost. In addition, the use of the firmware can greatly reduce the entry time for the customers who need to learn. Between, reduce the difficulty of programming.

This library function manual is generally divided into three parts

- definition
- Function library overview
- Application description of function library

1. Application layer function of servo(steer.h)

1.0 Variables in Steer class of application layer of servo

This class is defined in Steer.h, see the file for details。

Table0. Gives a list of public variables for Steer

Public variable name(public)	describe
Position_Current[2]	Gets when the value of the previous position is stored in the variable, High bit bytes in Position_Current [0] in units of mm
Position_Target[2]	Gets the value of the target location stored in the variable，High bit bytes in Position_Target [0] in units of mm
RunTime[2]	Gets the time value that runs to the destination location stored in the variable,RunTime [0] stores high bits of bytes, in units MSS, generally with the target Position write together to form speed control
Min_Angle_Limit[2]	Minimum angle limit, range （0~4095）means（0~360°），Byte storage mode ditto
Max_Angle_Limit[2]	Maximum angle limitation，range （0~4095）means（0~360°），Byte storage mode ditto
Voltage	Gets the current voltage value, which is not on yet
Temperature	Gets the current temperature value, which is not yet on
Speed_Current[2]	Gets the current speed value, which is not yet open

Table1. Gives a list of private variables for Steer

1.1 List of public functions in the Steer class

Table2. Gives a list of Steer's public functions

function name	describe
Steer	servo communication application layer constructor: initialization ID and serial communication
Steer_Ping	servo response function to test whether the communication of the servo is normal
Set_Steer_Max_Angle_Limit	Set the maximum angle limit for the servo. It is recommended that the maximum angle limit be set within 4096, that is,Within 360 °
Set_Steer_Min_Angle_Limit	Set the minimum angle limit for the servo. It is recommended to set it at 4096, that is, within 360 degrees
Set_Steer_Torque_On	Set the servo torque switch to open, that is, the servo acquires torque
Set_Steer_Torque_Off	Set the servo torque switch to turn off, that is, the servo loses torque
Set_Steer_position_runtime	The servo running function, which is configured to make the servo move
Change_Steer_ID	Change the ID of the servo
Set_Steer_Reset	Return of servo to factory setting
Get_Steer_All_Inf	Get all the current information on the servo

Get_Steer_Electric_Current_Inf	Gain forward current of rudder， The values obtained are stored in the Electric_Current [] array，For details, refer to the public member variables in the class definition
Get_Steer_voltage_Inf	Gets the current voltage of the servo, the resulting value is stored in the voltage,For details, look at the public member variables in the class definition
Get_Steer_Temperature_Inf	Get the current temperature of the servo, get the current temperature of the servo
Get_Steer_Position_Current_Inf	The obtained value of the rudder is stored in the Position_Current [] array，For details, see the Public member variable
Get_Steer_Position_Target_Inf	The target position of the obtained rudder is obtained and the obtained value is stored in the In the Position_Target [] array,For details, see the Public member variable
Get_Steer_RunTime_Inf	Get the running time of the servo, and the resulting value is stored in RunTime [], For details, look at the public member variables in the class definition
Get_Steer_Speed_Current_inf	Get the forward speed of the rudder, the obtained value is stored in the Speed_Current [] array, see the class definition of the male Member variable
Get_Steer_Angle_Limit_inf	The maximum / minimum angle limits of the servo are obtained, and the values obtained are stored in two arrays, Min_Angle_Limit [] and Max_Angle_Limit [],For details, look at the public member variables in the class definition

1.1.0 Function Steer

Table3.

function name	Steer
function prototype	Steer(byte ID=0 , HardwareSerial *serial = &Serial1)
functional description	Initialize ID and communication serial port
Input parameter 1	ID: ID of servo
Input parameter 2	Serial: serial port selection (need to be determined according to your own development board)
returned value	no
prerequisite	no
Called function	The bottom communication function of servo **Set_Serial_init**

Example:

```
/**Set id = 0, initialize serial port 1, and run the function when the object is created.**/
Steer    steer1( 0, &Serial1);
```

1.1.1 function **Steer_Ping**

Table4.

function name	Steer_Ping
function prototype	boolean Steer_Ping();
functional description	servo response function to test whether the communication of the servo is normal
input parameter	no
returned value	Returns the Boolean value of: ture for success and false for failure
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo ping

Example:

/**Set id = 0, initialize serial port 1, and check that the servo is communicating (answering) properly **/

```
Steer steer1(0,&Serial1);
steer1.Steer_Ping();
```

1.1.2 function **Set_Steer_Max_Angle_Limit**

Table5.

function name	Set_Steer_Max_Angle_Limit
function prototype	void Set_Steer_Max_Angle_Limit(word max_angle);
functional description	Setting maximum angular limits for servo
input parameter	max_angle: word type, setting the maximum value of the servo, range (0 ~ 4 095), that is, within (0 ~ 360 °)
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo write

Example:

/**Set id = 0, initialize serial port 1, and set the maximum deflection angle of the servo to 180 ° **/

```
Steer steer1( 0, &Serial1);
steer1. Set_Steer_Max_Angle_Limit (2047);
```

1.1.3 function **Set_Steer_Min_Angle_Limit**

Table6.

function name	Set_Steer_Min_Angle_Limit
function prototype	void Set_Steer_Min_Angle_Limit(word max_angle);
functional description	Setting minimum angular limits for servo
input parameter	min_angle: word Type, set the maximum value of the rudder, range (0- 4 095), that is, within (0 -360 °)
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo write

example:

/**Set id = 0, initialize serial port 1, and set the minimum deflection angle of the servo to 90 °**/
Steer steer1(0, &Serial1);
steer1. Set_Steer_Min_Angle_Limit (1024);

1.1.4 function **Set_Steer_Torque_On**

function name	Set_Steer_Torque_On
function prototype	void Set_Steer_Torque_On();
functional description	Set the servo torque switch to open, that is, the servo acquires torque
input parameter	no
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo write

Example:

/**Set id = 0, initialize serial port 1, and set the torque of the servo to open**/
Steer steer1(0, &Serial1);
steer1. Set_Steer_Torque_On();

1.1.5 function **Set_Steer_Torque_Off**

Table8.

function name	Set_Steer_Torque_Off
function prototype	void Set_Steer_Torque_Off();
functional description	Set the servo torque switch to open, that is, the servo loses torque
input parameter	no
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo write

example:

/**设Set id = 0, initialize the serial port 1, and set the torque of the servo to turn off, that is, lose the torque **/
Steer steer1(0, &Serial1);
steer1. Set_Steer_Torque_Off();

1.1.6 function **Set_Steer_position_runtime**

Table9..

function name	Set_Steer_position_runtime
function prototype	void Set_Steer_position_runtime(word pos, word runtime)
functional description	Make the servo run to the target position pos within runtime time

Input parameter 1	Poss: word type, target location, range (0-4095), that is, (0-360 °)
Input parameter 2	runtime: Running time, unit MSS, it is recommended that the novice turn this value up,To prevent injury or damage from being exercised too quickly.
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo **sync_write**

example:

```
/**Set id = 0, initialize serial port 1, and set the servo to 180 °in 2000ms**/
Steer    steer1( 0, &Serial1);
steer1. Set_Steer_position_runtime(2047, 2000);
```

1.1.7 function **Change_Steer_ID**

Table10.

function name	Change_Steer_ID
function prototype	void Change_Steer_ID(byte new_id);
functional description	Change the servo ID. note: after changing the servo ID, you can still call the object to control the servo because the id in the private variable will be changed at the same time
input parameter	new_id: New ID of servo
returned value	no
prerequisite	Initialize a Steer object
Called function	The bottom communication function of servo **write**

example:

```
/**Set id = 0, initialize serial port 1, and set the new id of the servo to 2**/
Steer    steer1(0,&Serial1);
steer1. Change_Steer_ID(2);
```

1.1.8 function **Set_Steer_Reset**

Table11.

function name	Set_Steer_Reset
function prototype	void Set_Steer_Reset();
functional description	After restoring the factory setting, the ID of the servo will be set to 0X01 after reset. **Note:** after restoring the factory setting, you can still call the object to control the servo because the id in the private variable is also changed to 0x01
input parameter	no
returned value	no

prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **reset**

example:

```
/**Set id = 0, initialize serial port 1, and restore the servo to factory setting**/
Steer    steer1(0,&Serial1);
steer1.Set_Steer_Reset();
```

1.1.9 function **Get_Steer_Electric_Current_Inf**（功能暂未开放）

Table12.

function name	Get_Steer_Electric_Current_Inf
function prototype	void Get_Steer_Electric_Current_Inf ();
functional description	The current of the servo is obtained and the function is not turned on for the time being. If open, The resulting value is stored in the Electric_Current [] array, see public member variables in the class definition
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

Example:

```
/**Set id = 0, initialize serial port 1, and get the current value of the servo at the same time**/
Steer    steer1( 0, &Serial1);
steer1. Get_Steer_Electric_Current_Inf ();
```

1.1.10 function **Get_Steer_voltage_Inf**（Function not open now）

Table13.

function name	Get_Steer_voltage_Inf
function prototype	void Get_Steer_voltage_Inf ();
functional description	The current voltage of the servo is obtained and the function is not on yet. If open, the obtained values are stored in the voltage, see the class definition in the public member variable
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

Example:

```
/**Set id = 0, initialize serial port 1, and get the voltage of the servo at the same time**/
Steer    steer1( 0, &Serial1 );
steer1. Get_Steer_voltage_Inf ();
```

1.1.11 function **Get_Steer_Temperature_Inf**（Function undeveloped）

Table14.

function name	Get_Steer_Temperature_Inf

function prototype	void Get_Steer_Temperature_Inf ();
functional description	The current temperature of the servo is obtained and the function is not on for the time being. If open, the obtained values are stored in the Temperature， For details, look at the public member variables in the class definition.
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

example:

```
/**Set id = 0, initialize serial port 1, and get the temperature of the servo at the same time**/
Steer    steer1( 0, &Serial1);
steer1. Get_Steer_Temperature_Inf ();
```

1.1.12 function **Get_Steer_Position_Current_Inf**

Table15.

function name	Get_Steer_Position_Current_Inf
function prototype	void Get_Steer_Position_Current_Inf ();
functional description	Obtain the current position of the servo. If open, the obtained values are stored in the In the Position_Current [] array, see the public member variables in the class definition
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

example:

```
/**Set id = 0, initialize serial port 1, and get the current position of the servo**/
Steer    steer1( 0, &Serial1);
steer1. Get_Steer_Position_Current_Inf ();
```

1.1.13 function **Get_Steer_Position_Target_Inf**

Table16.

function name	Get_Steer_Position_Target_Inf
function prototype	void Get_Steer_Position_Target_Inf ();
functional description	The target position of the obtained rudder is obtained and the obtained value is stored in the In the Position_Target [] array, see the public member variables in the class definition
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

example:

/**Set id = 0, initialize serial port 1, and get the target position of the servo at the same time**/
Steer steer1(0, &Serial1);
steer1. Get_Steer_Position_Target_Inf ();

1.1.14 function **Get_Steer_RunTime_Inf**

Table17.

function name	Get_Steer_RunTime_Inf
function prototype	void Get_Steer_RunTime_Inf ();
functional description	Get the running time of the servo to the target position and get the value store. In the RunTime[] array, let's look at the public member variables in the class definition.
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

example:

/**Set id = 0, initialize serial port 1, and get the running time of the servo to the target position**/
Steer steer1(0, &Serial1);
steer1. Get_Steer_RunTime_Inf ();

1.1.15 function **Get_Steer_Speed_Current_inf**（Function not open now）

Table18.

function name	Get_Steer_Speed_Current_inf
function prototype	void Get_Steer_Speed_Current_inf ();
functional description	The current speed of the servo is obtained and the function is not on yet. If open, the obtained value is stored in the Speed_Current [] array, see the public member variable in the class definition
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

例:

/**Set id = 0, initialize serial port 1 and get the current speed of the servo**/
Steer steer1(0, &Serial1);
steer1. Get_Steer_Speed_Current_inf ();

1.1.16 function **Get_Steer_Angle_Limit_inf**

Table19.

function name	Get_Steer_Angle_Limit_inf
function prototype	void Get_Steer_Angle_Limit_inf ();
functional description	Obtain the maximum and minimum angle limits for the servo, and the obtained values are stored in the In two arrays, Min_Angle_Limit [] and Max_Angle_Limit [],

	see the public member variables in the class definition
input parameter	no
returned value	no
prerequisite	Initialize a **Steer** object
Called function	The bottom communication function of servo **read**

Example:

```
/**Set id = 0, initialize serial port 1, and get maximum and minimum angle limits for the servo**/
Steer    steer1( 0, &Serial1);
steer1. Get_Steer_Angle_Limit_inf ();
```