motor.h文档

# 代码规范

使用整数类型时尽量指明位数，如用uint32\_t来代替unsigned int，以避免int字长的不确定性导致的问题。

struct类型名以大写字母开头；如果是const，则前面加c；如果是指针，则前面加p。如：cpMotor表示const Motor \*。

为防止与生成的代码中的名称冲突，函数名全部用小写字母，常量名全部大写，单词间用下划线分隔。对于函数，第一个单词应标明用途，后面加一个动词短语，最后可以加一个修饰词，如pwm\_set\_pulse\_single。以双下划线开头的名称为内部使用，一般情况下请勿在外部使用。

# 外部类型

typedef struct { TIM\_HandleTypeDef \* phtim; uint32\_t channel; } Motor;

储存每个电机端口的TIM\_Handle的指针和TIM\_Channel。

typedef const Motor \* cpMotor;

# 外部常量

const cpMotor

MOTORA\_F = &\_\_MOTORA\_F, MOTORA\_R = &\_\_MOTORA\_R,

MOTORB\_F = &\_\_MOTORB\_F, MOTORB\_R = &\_\_MOTORB\_R,

MOTORC\_F = &\_\_MOTORC\_F, MOTORC\_R = &\_\_MOTORC\_R,

MOTORD\_F = &\_\_MOTORD\_F, MOTORD\_R = &\_\_MOTORD\_R;

电动机A,B,C,D的前进、后退端口的指针。

# 外部函数

void pwm\_set\_pulse\_single(const cpMotor motor, uint16\_t pulse);

设置单个电动机的前进或后退的pulse值。

void pwm\_set\_pulse\_F(uint16\_t pulse);

设置所有电动机前进的pulse值。

void pwm\_set\_pulse\_R(uint16\_t pulse);

设置所有电动机后退的pulse值。

# 内部常量

const Motor

\_\_MOTORA\_F = {&htim1, TIM\_CHANNEL\_3},

\_\_MOTORA\_R = {&htim1, TIM\_CHANNEL\_4}, // A线接反了，已修正

\_\_MOTORB\_F = {&htim1, TIM\_CHANNEL\_2},

\_\_MOTORB\_R = {&htim1, TIM\_CHANNEL\_1},

\_\_MOTORC\_F = {&htim3, TIM\_CHANNEL\_3},

\_\_MOTORC\_R = {&htim3, TIM\_CHANNEL\_4},

\_\_MOTORD\_F = {&htim3, TIM\_CHANNEL\_2},

\_\_MOTORD\_R = {&htim3, TIM\_CHANNEL\_1}; // D线接反了，已修正

各个电机前进后退端口的本体。

# 内部函数

void \_\_pwm\_set(const cpMotor motor, TIM\_OC\_InitTypeDef \* psConfigOC);

设置某个电机前进或后退的配置。