

stepmotor4.c

```

1#include "stepmotor4.h"
2#include "tim.h"
3#include "gpio.h"
4#include "main.h"
5
6
7
8/* inicjalizacja biblioteki */
9void stepper_init_motor4(void)
10{
11    HAL_TIM_Base_Start_IT(&PULSE_TIM_MOTOR4);
12    HAL_Delay(2);
13}
14
15/* ustawia predkosc obrotowa silnika krokowego w rpm */
16void stepper_speed_motor4(uint16_t rpm)
17{
18    if(rpm > 0)
19    {
20        uint16_t arr_val = XT_TIM_CLK_MOTOR4 / (((XT_TIM_PSC_MOTOR4+1) * rpm *
        STEPS_PER_REV_MOTOR4 * MICROSTEP_NUM_MOTOR4) / 60) - 1;
21        uint16_t pulse_val = arr_val / 2;
22        __HAL_TIM_SET_AUTORELOAD(&PULSE_TIM_MOTOR4, arr_val);
23        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR4, PULSE_TIM_CH_MOTOR4, pulse_val);
24    }
25    else
26    {
27        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR4, PULSE_TIM_CH_MOTOR4, 0);
28    }
29}
30
31/* rozpoczyna ruch silnika z kierunkiem obrotu dir */
32void stepper_run_motor4(uint8_t dir)
33{
34    if(dir == STEPPER_CW_MOTOR4)
35    {
36        HAL_GPIO_WritePin(DIR_AXIS4_GPIO_Port, DIR_AXIS4_Pin, GPIO_PIN_SET);
37    }
38    else if (dir == STEPPER_CCW_MOTOR4)
39    {
40        HAL_GPIO_WritePin(DIR_AXIS4_GPIO_Port, DIR_AXIS4_Pin, GPIO_PIN_RESET);
41    }
42    HAL_Delay(2);
43    HAL_TIM_PWM_Start(&PULSE_TIM_MOTOR4, PULSE_TIM_CH_MOTOR4);
44}
45
46/* zatrzymuje silnik - wyłącza wyjścia kontrolera silnika */
47void stepper_stop_motor4(void)
48{
49    HAL_TIM_PWM_Stop(&PULSE_TIM_MOTOR4, PULSE_TIM_CH_MOTOR4);
50}
51
52/* ustawia zadana liczbe krokow do wykonania */
53void stepper_steps_motor4(uint16_t steps, volatile uint16_t *steplimit4)
54{
55    *steplimit4 = steps;
56}
57
58/* obraca wal silnika o zadny kat z zadana predkoscia w kierunku dir */
59void stepper_rot_motor4(uint16_t ang, uint16_t rpm, uint8_t dir, volatile uint16_t
    *steplimit4, volatile int* rotationCounter4, volatile uint16_t* isStop4)
60{

```

stepmotor4.c

```
61     if(dir == STEPPER_CW_MOTOR4)
62     {
63         *isStop4 = 0;
64         *rotationCounter4 += ang;
65     }
66     else if(dir == STEPPER_CCW_MOTOR4)
67     {
68         *isStop4 = 0;
69         *rotationCounter4 -= ang;
70     }
71     stepper_steps_motor4((((STEPS_PER_REV_MOTOR4 * MICROSTEP_NUM_MOTOR4)/40)*ang)/90),
    stepLimit4);
72     stepper_speed_motor4(rpm);
73     stepper_run_motor4(dir);
74 }
75
76 /* obraca wal silnika do pozycji domowej o zadany kat z zadana predkoscia w kierunku dir */
77 void stepper_rot_home_motor4(uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit4,
    volatile int* rotationCounter4, volatile uint16_t* isStop4)
78 {
79     if(*rotationCounter4 < 0)
80     {
81         *isStop4 = 0;
82         uint16_t ang = *rotationCounter4 * (-1);
83         stepper_rot_motor4(ang, rpm, STEPPER_CW_MOTOR4, stepLimit4, rotationCounter4,
    isStop4);
84     }
85     else
86     {
87         *isStop4 = 0;
88         uint16_t ang = *rotationCounter4;
89         stepper_rot_motor4(ang, rpm, STEPPER_CCW_MOTOR4, stepLimit4, rotationCounter4,
    isStop4);
90     }
91 }
92
93
94
```