

stepmotor3.c

```

1#include "stepmotor3.h"
2#include "tim.h"
3#include "gpio.h"
4#include "main.h"
5
6
7
8/* inicjalizacja biblioteki */
9void stepper_init_motor3(void)
10{
11    HAL_TIM_Base_Start_IT(&PULSE_TIM_MOTOR3);
12    HAL_Delay(2);
13}
14
15/* ustawia predkosc obrotowa silnika krokowego w rpm */
16void stepper_speed_motor3(uint16_t rpm)
17{
18    if(rpm > 0)
19    {
20        uint16_t arr_val = XT_TIM_CLK_MOTOR3 / (((XT_TIM_PSC_MOTOR3+1) * rpm *
        STEPS_PER_REV_MOTOR3 * MICROSTEP_NUM_MOTOR3) / 60) - 1;
21        uint16_t pulse_val = arr_val / 2;
22        __HAL_TIM_SET_AUTORELOAD(&PULSE_TIM_MOTOR3, arr_val);
23        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR3, PULSE_TIM_CH_MOTOR3, pulse_val);
24    }
25    else
26    {
27        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR3, PULSE_TIM_CH_MOTOR3, 0);
28    }
29}
30
31/* rozpoczyna ruch silnika z kierunkiem obrotu dir */
32void stepper_run_motor3(uint8_t dir)
33{
34    if(dir == STEPPER_CW_MOTOR3)
35    {
36        HAL_GPIO_WritePin(DIR_AXIS3_GPIO_Port, DIR_AXIS3_Pin, GPIO_PIN_SET);
37    }
38    else if (dir == STEPPER_CCW_MOTOR3)
39    {
40        HAL_GPIO_WritePin(DIR_AXIS3_GPIO_Port, DIR_AXIS3_Pin, GPIO_PIN_RESET);
41    }
42    HAL_Delay(2);
43    HAL_TIM_PWM_Start(&PULSE_TIM_MOTOR3, PULSE_TIM_CH_MOTOR3);
44}
45
46/* zatrzymuje silnik - wyłącza wyjścia kontrolera silnika */
47void stepper_stop_motor3(void)
48{
49    HAL_TIM_PWM_Stop(&PULSE_TIM_MOTOR3, PULSE_TIM_CH_MOTOR3);
50}
51
52/* ustawia zadana liczbe krokow do wykonania */
53void stepper_steps_motor3(uint16_t steps, volatile uint16_t *stepLimit3)
54{
55    *stepLimit3 = steps;
56}
57
58/* obraca wal silnika o zadny kat z zadana predkoscia w kierunku dir */
59void stepper_rot_motor3(uint16_t ang, uint16_t rpm, uint8_t dir, volatile uint16_t
    *stepLimit3, volatile int* rotationCounter3, volatile uint16_t* isStop3)
60{

```

stepmotor3.c

```
61     if(dir == STEPPER_CW_MOTOR3)
62     {
63         *isStop3 = 0;
64         *rotationCounter3 += ang;
65     }
66     else if(dir == STEPPER_CCW_MOTOR3)
67     {
68         *isStop3 = 0;
69         *rotationCounter3 -= ang;
70     }
71     stepper_steps_motor3((((STEPS_PER_REV_MOTOR3 * MICROSTEP_NUM_MOTOR3)/40)*ang)/90),
    stepLimit3);
72     stepper_speed_motor3(rpm);
73     stepper_run_motor3(dir);
74 }
75
76 /* obraca wal silnika do pozycji domowej o zadany kat z zadana predkoscia w kierunku dir */
77 void stepper_rot_home_motor3(uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit3,
    volatile int* rotationCounter3, volatile uint16_t* isStop3)
78 {
79     if(*rotationCounter3 < 0)
80     {
81         *isStop3 = 0;
82         uint16_t ang = *rotationCounter3 * (-1);
83         stepper_rot_motor3(ang, rpm, STEPPER_CW_MOTOR3, stepLimit3, rotationCounter3,
    isStop3);
84     }
85     else
86     {
87         *isStop3 = 0;
88         uint16_t ang = *rotationCounter3;
89         stepper_rot_motor3(ang, rpm, STEPPER_CCW_MOTOR3, stepLimit3, rotationCounter3,
    isStop3);
90     }
91 }
92
93
94
```