```c
#include "stepmotor2.h"
#include "tim.h"
#include "gpio.h"
#include "main.h"



/* inicjalizacja biblioteki */
void stepper_init_motor2(void)
{
    HAL_TIM_Base_Start_IT(&PULSE_TIM_MOTOR2);
    HAL_Delay(2);
}

/* ustawia predkosc obrotowa silnika krokowego w rpm */
void stepper_speed_motor2(uint16_t rpm)
{
    if(rpm > 0)
    {
        uint16_t arr_val = XT_TIM_CLK_MOTOR2 / (((XT_TIM_PSC_MOTOR2+1) * rpm * STEPS_PER_REV_MOTOR2 * MICROSTEP_NUM_MOTOR2) / 60) - 1;
        uint16_t pulse_val = arr_val / 2;
        __HAL_TIM_SET_AUTORELOAD(&PULSE_TIM_MOTOR2, arr_val);
        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR2, PULSE_TIM_CH_MOTOR2, pulse_val);
    }
    else
    {
        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR2, PULSE_TIM_CH_MOTOR2, 0);
    }
}

/* rozpoczyna ruch silnika z kierunkiem obrotu dir */
void stepper_run_motor2(uint8_t dir)
{
    if(dir == STEPPER_CW_MOTOR2)
    {
        HAL_GPIO_WritePin(DIR_AXIS2_GPIO_Port, DIR_AXIS2_Pin, GPIO_PIN_SET);
    }
    else if (dir == STEPPER_CCW_MOTOR2)
    {
        HAL_GPIO_WritePin(DIR_AXIS2_GPIO_Port, DIR_AXIS2_Pin, GPIO_PIN_RESET);
    }
    HAL_Delay(2);
    HAL_TIM_PWM_Start(&PULSE_TIM_MOTOR2, PULSE_TIM_CH_MOTOR2);
}

/* zatrzymuje silnik - wylacza wyjscia kontrolera silnika */
void stepper_stop_motor2(void)
{
    HAL_TIM_PWM_Stop(&PULSE_TIM_MOTOR2, PULSE_TIM_CH_MOTOR2);
}

/* ustawia zadana liczbe krokow do wykonania */
void stepper_steps_motor2(uint16_t steps, volatile uint16_t *stepLimit2)
{
    *stepLimit2 = steps;
}

/* obraca wal silnika o zadny kat z zadana predkoscia w kierunku dir */
void stepper_rot_motor2(uint16_t ang, uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit2, volatile int* rotationCounter2, volatile uint16_t* isStop2)
{
```

```
61      if(dir == STEPPER_CW_MOTOR2)
62      {
63          *isStop2 = 0;
64          *rotationCounter2 += ang;
65      }
66      else if(dir == STEPPER_CCW_MOTOR2)
67      {
68          *isStop2 = 0;
69          *rotationCounter2 -= ang;
70      }
71      stepper_steps_motor2(((((STEPS_PER_REV_MOTOR2 * MICROSTEP_NUM_MOTOR2)/40)*ang)/90),
    stepLimit2);
72      stepper_speed_motor2(rpm);
73      stepper_run_motor2(dir);
74  }
75
76  /* obraca wal silnika do pozycji domowej o zadany kat z zadana predkoscia w kierunku dir */
77  void stepper_rot_home_motor2(uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit2,
    volatile int* rotationCounter2, volatile uint16_t* isStop2)
78  {
79      if(*rotationCounter2 < 0)
80      {
81          *isStop2 = 0;
82          uint16_t ang = *rotationCounter2 * (-1);
83          stepper_rot_motor2(ang, rpm, STEPPER_CW_MOTOR2, stepLimit2, rotationCounter2,
    isStop2);
84      }
85      else
86      {
87          *isStop2 = 0;
88          uint16_t ang = *rotationCounter2;
89          stepper_rot_motor2(ang, rpm, STEPPER_CCW_MOTOR2, stepLimit2, rotationCounter2,
    isStop2);
90      }
91  }
92
93
94
```