```c
#include "stepmotor1.h"
#include "tim.h"
#include "gpio.h"
#include "main.h"



/* inicjalizacja biblioteki */
void stepper_init_motor1(void)
{
    HAL_TIM_Base_Start_IT(&PULSE_TIM_MOTOR1);
    HAL_Delay(2);
}

/* ustawia predkosc obrotowa silnika krokowego w rpm */
void stepper_speed_motor1(uint16_t rpm)
{
    if(rpm > 0)
    {
        uint16_t arr_val = XT_TIM_CLK_MOTOR1 / (((XT_TIM_PSC_MOTOR1+1) * rpm * STEPS_PER_REV_MOTOR1 * MICROSTEP_NUM_MOTOR1) / 60) - 1;
        uint16_t pulse_val = arr_val / 2;
        __HAL_TIM_SET_AUTORELOAD(&PULSE_TIM_MOTOR1, arr_val);
        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR1, PULSE_TIM_CH_MOTOR1, pulse_val);
    }
    else
    {
        __HAL_TIM_SET_COMPARE(&PULSE_TIM_MOTOR1, PULSE_TIM_CH_MOTOR1, 0);
    }
}

/* rozpoczyna ruch silnika z kierunkiem obrotu dir */
void stepper_run_motor1(uint8_t dir)
{
    if(dir == STEPPER_CW_MOTOR1)
    {
        HAL_GPIO_WritePin(DIR_AXIS1_GPIO_Port, DIR_AXIS1_Pin, GPIO_PIN_SET);
    }
    else if (dir == STEPPER_CCW_MOTOR1)
    {
        HAL_GPIO_WritePin(DIR_AXIS1_GPIO_Port, DIR_AXIS1_Pin, GPIO_PIN_RESET);
    }
    HAL_Delay(2);
    HAL_TIM_PWM_Start(&PULSE_TIM_MOTOR1, PULSE_TIM_CH_MOTOR1);
}

/* zatrzymuje silnik - wylacza wyjscia kontrolera silnika */
void stepper_stop_motor1(void)
{
    HAL_TIM_PWM_Stop(&PULSE_TIM_MOTOR1, PULSE_TIM_CH_MOTOR1);
}

/* ustawia zadana liczbe krokow do wykonania */
void stepper_steps_motor1(uint16_t steps, volatile uint16_t *stepLimit1)
{
    *stepLimit1 = steps;
}

/* obraca wal silnika o zadny kat z zadana predkoscia w kierunku dir */
void stepper_rot_motor1(uint16_t ang, uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit1, volatile int* rotationCounter1, volatile uint16_t* isStop1)
{
```

```
61      if(dir == STEPPER_CW_MOTOR1)
62      {
63          *isStop1 = 0;
64          *rotationCounter1 += ang;
65      }
66      else if(dir == STEPPER_CCW_MOTOR1)
67      {
68          *isStop1 = 0;
69          *rotationCounter1 -= ang;
70      }
71      stepper_steps_motor1(((((STEPS_PER_REV_MOTOR1 * MICROSTEP_NUM_MOTOR1)/40)*ang)/90),
    stepLimit1);
72      stepper_speed_motor1(rpm);
73      stepper_run_motor1(dir);
74  }
75
76  /* obraca wal silnika do pozycji domowej o zadany kat z zadana predkoscia w kierunku dir */
77  void stepper_rot_home_motor1(uint16_t rpm, uint8_t dir, volatile uint16_t *stepLimit1,
    volatile int* rotationCounter1, volatile uint16_t* isStop1)
78  {
79      if(*rotationCounter1 < 0)
80      {
81          *isStop1 = 0;
82          uint16_t ang = *rotationCounter1 * (-1);
83          stepper_rot_motor1(ang, rpm, STEPPER_CW_MOTOR1, stepLimit1, rotationCounter1,
    isStop1);
84      }
85      else
86      {
87          *isStop1 = 0;
88          uint16_t ang = *rotationCounter1;
89          stepper_rot_motor1(ang, rpm, STEPPER_CCW_MOTOR1, stepLimit1, rotationCounter1,
    isStop1);
90      }
91  }
92
93
94
```