MECHTRON 2TA4

Lab 5 – Control of a Stepper Motor

Luai Bashar – bashal1 – 400388669

03/04/23

Questions

1. The angular resolution can be found by using the formula $Resolution = \frac{360°}{\#\ of\ steps}$, where 360 degrees represents a full step. A total of 48 steps are made per revolution, so we can sub that into the equation:

$$Resolution = \frac{360°}{\#\ of\ steps} = \frac{360°}{48} = 7.5°$$

The resolution is 7.5 degrees, meaning that each step traverses this number of degrees**.**

2. The last two numbers of my student number is 69. This number is larger than 66, so by subtracting this number by 33, **my real period is 36.**

3. For a **half stepping sequence**, double the number of steps is used. Instead of 48 steps, 96 steps are used. We can calculate the time it takes for a singular step with $\frac{period}{\#\ of\ steps}$:

$$\frac{period}{\#\ of\ steps} = \frac{36}{96} = 0.375\ seconds.$$

A total of 0.375 seconds is taken in a half-stepping sequence. In a **full step sequence**, 48 steps are taken. We can use the same formula to find the time it takes between two steps:

$$\frac{period}{\#\ of\ steps} = \frac{36}{48} = 0.75\ seconds.$$

A total of 0.75 seconds is taken in a full-stepping sequence.

4. We can rearrange the Fclock equation to find a value for the prescaler, $PSC = \frac{Fcl\_PSC}{Fck\_CNT} - 1$.

SYSCLK is configured to 72 MHz and APB1 is configured to 36 MHz, so TIM3CLK is set as 72MHz:

$PSC = \frac{Fcl\_PSC}{Fck\_CNT} - 1 = \frac{72\ MHz}{10\ kHz} - 1 = \ 7199$

Half-Step: $OCR = \frac{36}{96} \times 10\ kHz - 1 = 3749$

Full-Step: $OCR = \frac{36}{48} \times 10\ kHz - 1 = 7499$

5. Here is the code which configures the timer, the output/input pins, and the external buttons:

```
void  TIM3_Config(void) {

  Tim3_PrescalerValue = (uint32_t) ((SystemCoreClock) / 10000) - 1;

  Tim3_Handle.Instance = TIM3; //TIM3 is defined in stm32f429xx.h

  Tim3_Handle.Init.Period = Tim3_CCR - 1;
  Tim3_Handle.Init.Prescaler = Tim3_PrescalerValue;
  Tim3_Handle.Init.ClockDivision = 0;
  Tim3_Handle.Init.CounterMode = TIM_COUNTERMODE_UP;
  HAL_TIM_Base_Init(&Tim3_Handle);
  HAL_TIM_Base_Start_IT(&Tim3_Handle);

}
```

```c
int cycle = 1;
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim) {
  if (clockwise) {
    if (cycle == 1) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_14,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,1);
      cycle = 2;
    }
    else if (cycle == 2) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,1);
      cycle = 3;
    }
    else if (cycle == 3) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_2,1);
      cycle = 4;
    }
    else if (cycle == 4) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_2,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_14,1);
      cycle = 1;
    }
  }
  else {
    if (cycle == 1) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_14,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_2,1);
      cycle = 2;
    }
    else if (cycle == 2) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_2,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,1);
      cycle = 3;
```

```
    else if (cycle == 3) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_4,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,1);
      cycle = 4;
    }
    else if (cycle == 4) {
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,0);
      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_14,1);
      cycle = 1;
    }
  }
}
//#####################################################################

//output configs
//#####################################################################
void Power_Config(void){
  GPIO_InitTypeDef   GPIO_InitStructure;

  __HAL_RCC_GPIOC_CLK_ENABLE();
  GPIO_InitStructure.Mode =  GPIO_MODE_OUTPUT_PP;
  GPIO_InitStructure.Pull = GPIO_NOPULL;

  GPIO_InitStructure.Pin = GPIO_PIN_13;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);

  GPIO_InitStructure.Pin = GPIO_PIN_14;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);

  GPIO_InitStructure.Pin = GPIO_PIN_2;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);

  GPIO_InitStructure.Pin = GPIO_PIN_4;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);
}
```

```c
int full_step = 1;
int clockwise = 0;
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    //changes from full step to half step
    if(GPIO_Pin == KEY_BUTTON_PIN)  //GPIO_PIN_0
    {
      if (full_step == 1) {
        full_step = 0;
        Tim3_CCR = 3750;
        TIM3_Config();
      }
      else {
        full_step = 1;
        Tim3_CCR = 7500;
        TIM3_Config();
      }
    }

    //changes clockwise and counterclockwise
    if(GPIO_Pin == GPIO_PIN_1)
    {
      if (clockwise == 0) clockwise = 1;
      else clockwise = 0;
    }  //end of PIN_1
```