

lab 4 - Timers

Stepper Motor control

M. Briday

December 9, 2021

1 Objective

The objective of this lab is to adapt the timer frequency to control a stepper motor speed. The timer is used in synchronization mode.

2 Control of a stepper motor

The generated sequence allows to control a single pole stepper motor as shown in figure 1. A stepper motor has its rotating part in rotation if square signals are applied in a precise order with respect to each other to the phases of the motor A, B, C and D. The direction of rotation is determined by the order in which these phases are supplied, in relation to each other.

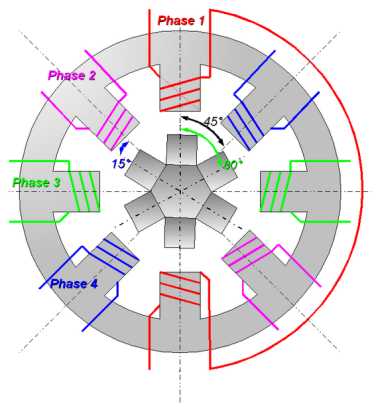


Figure 1: Operating principle of the stepper motor

The ULN2003 chip is a buffer (Darlington transistor array) that is in charge of the power conversion.

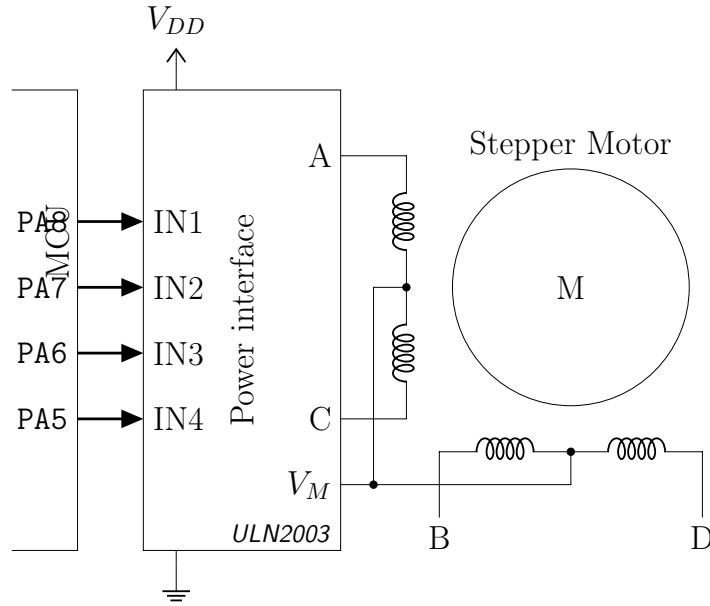


Figure 2: Electronic control interface

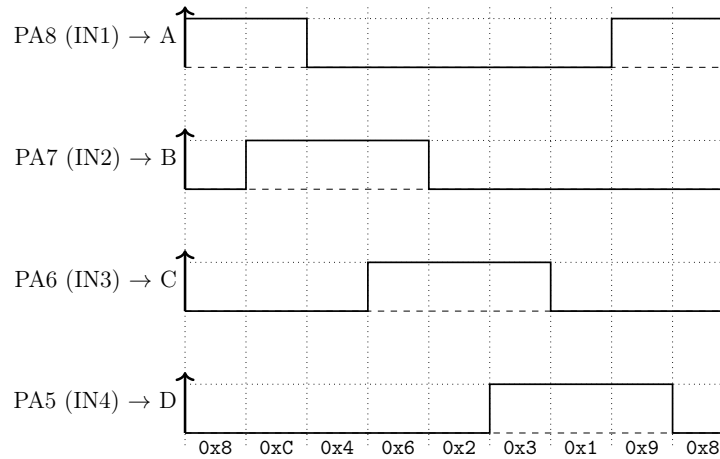


Figure 3: Time sequence for a counter-clockwise rotation

3 Counter-Clockwise Rotation

To rotate the stepper motor counter-clockwise, we use a table that stores the entire sequence. The table is defined as follows:

```
1 const unsigned char seq[]={8,0xc,4,6,2,3,1,9};
```

Question 1 Write the *setup()* function to configure outputs.

We use in the next two question a direct access to the GPIO, using ODR.

Question 2 Write the `stepCCW()` function that updates outputs so that the motor make a single step each time the function is called. Note: a `digitalWrite(...)` is not appropriate, as the 4 outputs should be updated at the same time. Validate the code with a step by step execution.

Question 3 Use a timer so that the `stepCCW()` function is called @10Hz. The synchronization phase should not use a blocking function (active wait for a flag). There preferred way is a function that just look at the timer status (overflow).

Question 4 update the code to use the `BSRR` register of the GPIO. Validate the new solution.

4 Clockwise Rotation

Question 5 Write the `stepCW()` function that works as `stepCCW`, but in the other direction. The function uses the `BSRR` GPIO register directly.

We now mix the 2 directions:

Question 6 The stepper motor should turn clockwise when the button (D6) is pressed, and in the other direction when it is released.

5 Rotation speed

The rotation speed is directly associated to the frequency of the timer. We use here the potentiometer that is associated to the ADC (ADC1, channel 4 on pin PA3). To use the ADC, a set of files `adc.c/h` are provided.

Question 7 Update the timer configuration so that the stepper functions are called from 10Hz (potentiometer set to 0) to 500Hz (potentiometer set to max).

Question 8 Test your solution: the stepper motor should change its direction of rotation each time button (D6) is pressed (Finite State Machine associated to the button state).

6 Just a round...

There are 64 steps for each round, but there is also a reduction factor of 64.

We now want to do just one round. Each time the button D6 is pushed, the stepper performs one round, toggling the direction of rotation.

Question 9 Update your application to add this constraint.