AN3208
Application note

Microstepping motor drive with STM8A and STM8S microcontrollers

Introduction

Stepper motors are electrically powered motors that create rotation from electrical current driven into the motor.

They are used in a wide variety of applications such as printers, automated machine tools, disk drives, automotive dashboard instrument clusters, and other applications requiring precise motion control. They are well-suited for positioning applications since they can achieve very good positional accuracy without complicated feedback loops associated with servomechanism (servo) systems. However, their resolution, when driven in the conventional full or half step modes of operation, is limited by the configuration of the motor.

Many designers today seek methods to increase the resolution of stepper motor drives. Dedicated stepper motor controllers/ICs are available on the market. These controllers contain the special logic and high-current drive circuits necessary to operate the stepper motors. In some applications, for example in automotive dashboards, stepper motors with a lower current rating (20 mA) are used to power the needles or pointers that display parameters such as vehicle speed or the engine RPM.

Stepper motors need to be driven in microstepping mode (see *Section 4: Driving stepper motors using STM8A and STM8S microcontrollers*). However, in this case, the use of dedicated stepper motor controllers may increase the system cost and complexity. As an alternative, the motors can be driven easily using the resources located within a microcontroller (example, pulse-width modulation timers and I/O pins), thus reducing the hardware cost and complexity. CPU load is very low when using internal resources and the microcontroller is not precluded from performing other control activities or driving other external peripherals. For example, the STM8 is able to drive two stepper motors together with an LCD glass containing a high number of segments such as a motorcycle dashboard application.

The application described in this document is a software/low cost solution to drive stepper motors in microstepping mode using the STM8 microcontroller. The main focus of this application note is to explain how to drive the microstepping motor with STM8A and STM8S devices. An overview of the various stepper motor types is given in *Section 2: Types of stepper motor*. Stepper motor basics are explained in*Section 4: Driving stepper motors using STM8A and STM8S microcontrollers*. *Section 5* summarizes stepper motor software.

Reference documents

- STM8A reference manual (RM0009)
- STM8S reference manual (RM0016)
- STM8A/S datasheets

Reference firmware

- STM8A/S firmware library

Contents

List of tables

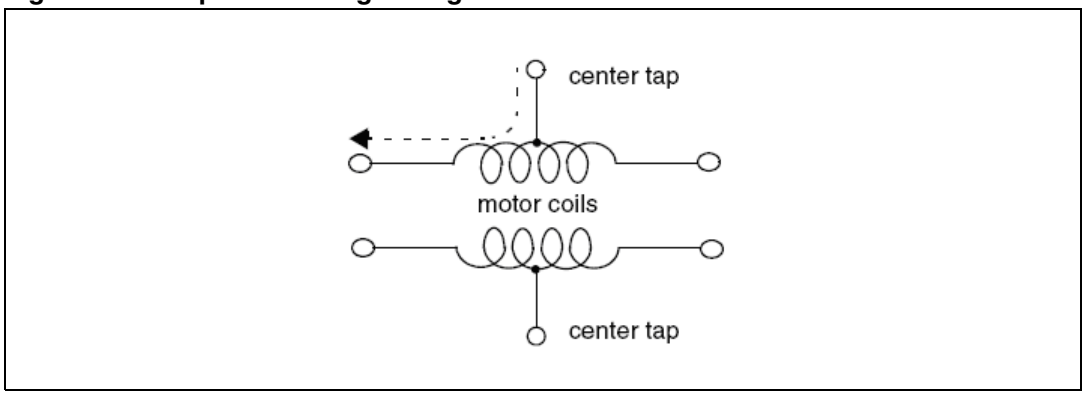
List of figures

1 Winding arrangement in two-phase stepper motors

There are two basic winding arrangements for the electromagnetic coils in a two-phase stepper motor: bipolar and unipolar.

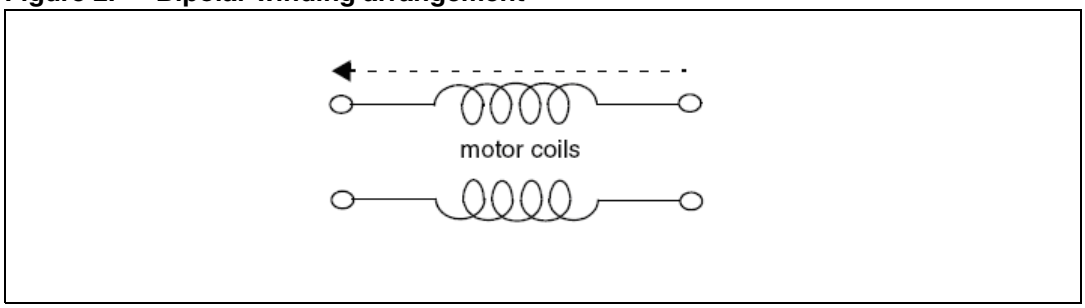
The unipolar stepper motor has two identical coils which are not connected electrically and each coil has a centre tap.

Figure 1. Unipolar winding arrangement



The bipolar stepper motor is the same as the unipolar stepper except the motor coils do not have the center taps. Bipolar motor can produce higher torque in comparison to the unipolar motor as the entire coil is energized and not just half-coils.

Figure 2. Bipolar winding arrangement



2 Types of stepper motor

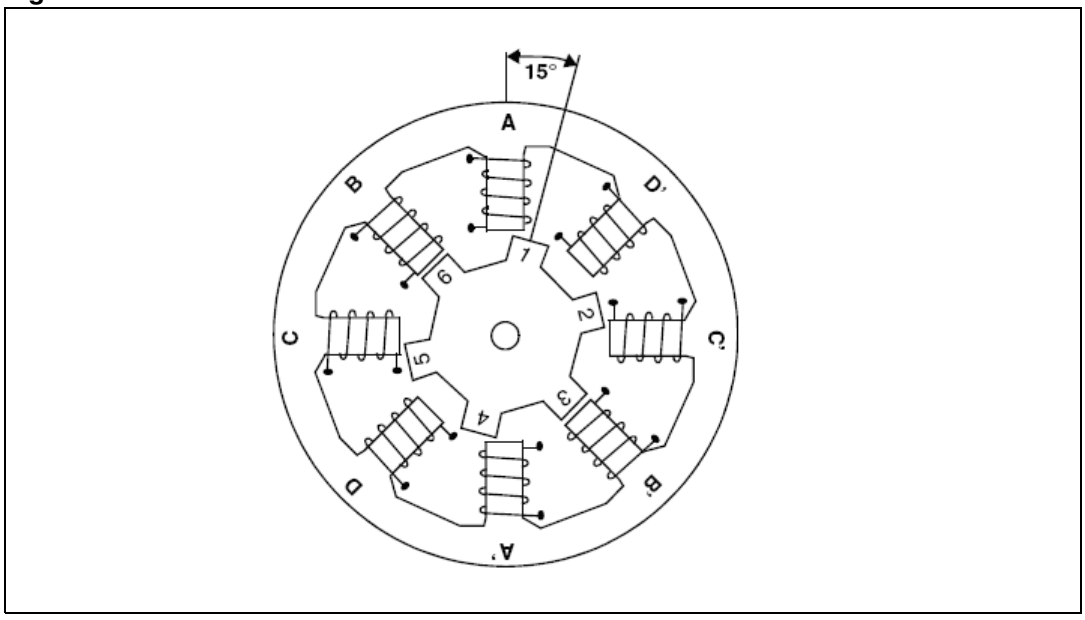
There are three main types of stepper motors:

1. Variable-reluctance stepper motor
2. Permanent magnet stepper motor
3. Hybrid synchronous stepper motor

2.1 Variable-reluctance (VR) motor

This type of motor contains a soft iron multi-toothed rotor and a wound stator. When the stator windings are energized with DC current, it magnetizes the stator poles. Rotor teeth are attracted towards the energized stator poles and the rotation occurs. The variable-reluctance motor does not use permanent magnets, so the field strength can be varied. The VR motor generates less torque so it is generally used for small positioning loads. *Figure 3* shows a cross section of a typical VR stepper motor.

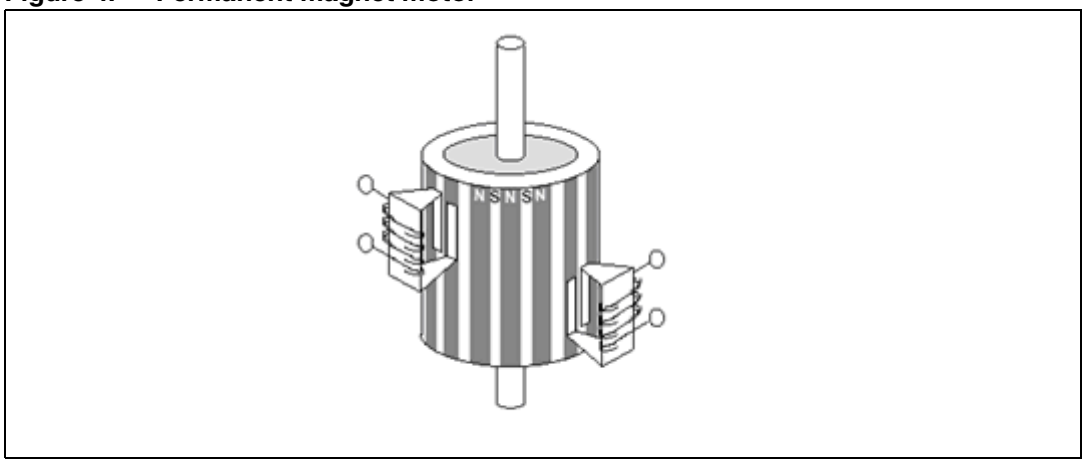
Figure 3. Variable-reluctance motor



2.2 Permanent magnet (PM) motor

This motor is known as a tin-can or can-stock motor. The permanent magnet stepper motor is a low cost and low resolution type motor with a typical step angle of 7.5 ° to 15 °. PM motors use permanent magnets and the rotor does not have the teeth of the VR motor. The PM motor has improved torque characteristics compared with the VR motor.

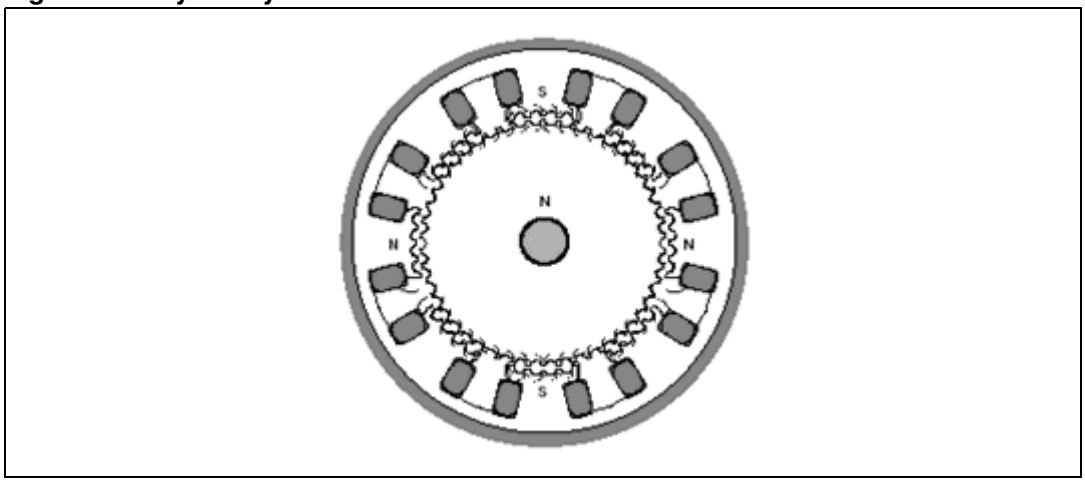
Figure 4. Permanent magnet motor



2.3 Hybrid synchronous motor

The hybrid stepper motor contains features of both the PM and VR motors. Hybrid motors are more expensive than PM stepper motors but provide better performance regarding step resolution, torque and speed. Typical step angles for the hybrid motor range from 3.6 ° to 0.9 °.The rotor is multi-toothed like the VR motor and contains an axially magnetized concentric magnet around its shaft. The teeth on the rotor provide an even better path which helps guide the magnetic flux to preferred locations in the air gap. This further increases the detent, holding, and dynamic torque characteristics of the motor compared with both VR and PM motors.

Figure 5. Hybrid synchronous motor



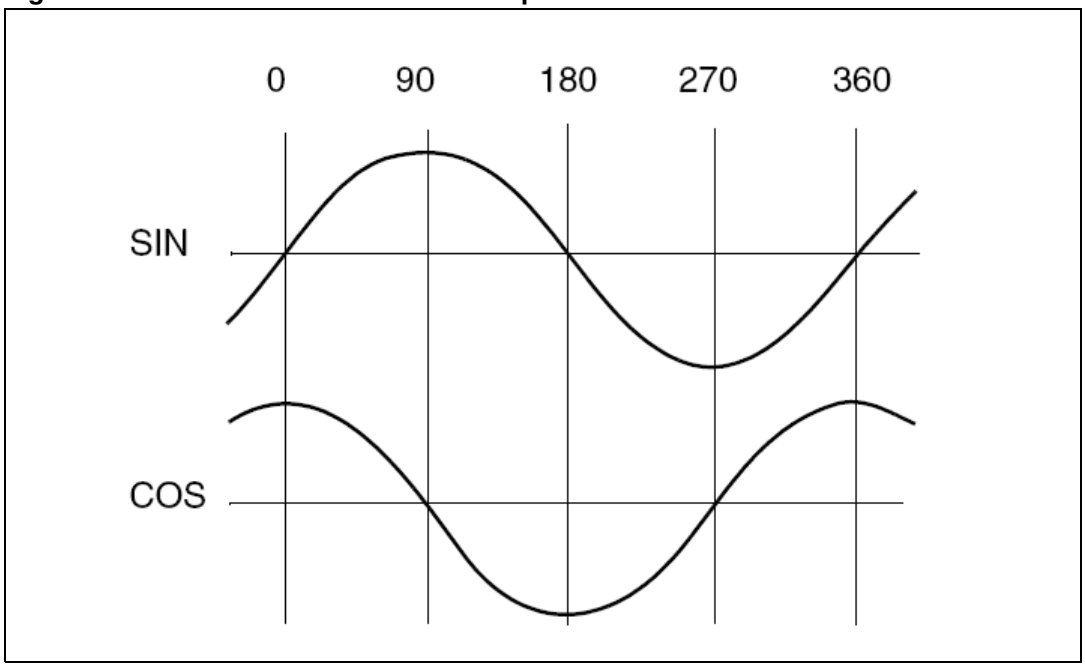
3 Microstepping

Physically, stepper motors can be large but, often they are small enough to be driven by current in the mA range. Current pulses are applied to the motor which generates discrete rotations of the motor shaft. Although it is possible to drive a stepper motor in a manner where it has near continuous rotation, doing so requires more finesse of the input waveform that drives the stepper motor.

Microstepping is a way of moving the motor shaft more smoothly than in full- or half-step drive modes, allowing the stepper motor to stop and hold a position between the full- or half-step positions. The jerky character of low stepping motor operation is reduced. There are fewer vibrations and less problems with resonance which makes noiseless stepping possible down to 0 Hz. With microstepping, smaller step angles and better positioning is possible.

The ideal current waveform for driving a stepper motor is a sinewave. Two sinewaves, 90 ° out of phase (or another angle depending on the motor construction), form the ideal drive current. If the stepper coils follow these current waveforms, the motor runs quietly and smoothly, which is the ideal condition. In fact, the steps associated with stepper motors will disappear.

Figure 6. Current waveforms with 90 ° phase difference

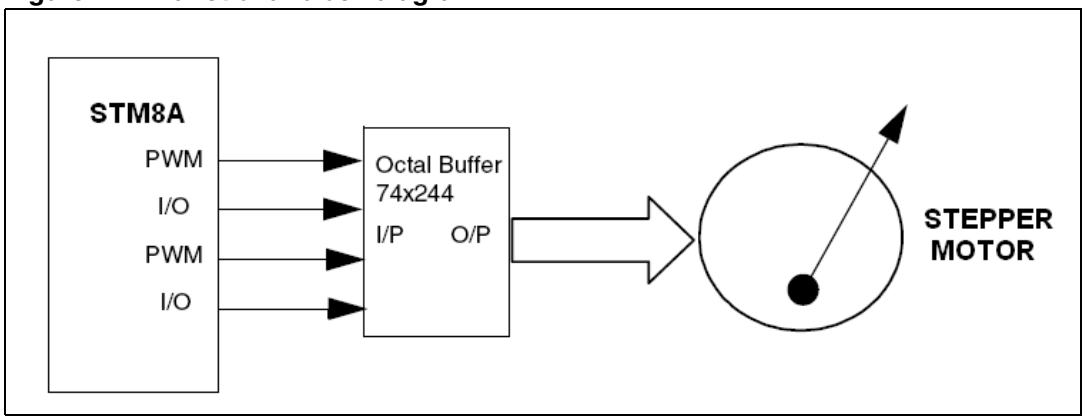


In microstepping mode, the current magnitude in the motor coil has to be controlled in a proper sequence. The current can be controlled using a H-bridge circuit or PWM technique. *Section 4: Driving stepper motors using STM8A and STM8S microcontrollers* describes the software/low-cost solution to drive the stepper motors in microstepping mode using STM8A and STM8S microcontrollers.

4 Driving stepper motors using STM8A and STM8S microcontrollers

This application solution makes use of a PWM current control technique. The stepper motor is driven directly by PWMs, I/O pins and a 74HC/HCT244 buffer/driver (so the position of the motor can be controlled with precision without any feedback mechanism) of an an STM8A or STM8S microcontroller. Two ends of each motor winding are connected to one PWM and one I/O line. TIM1 (16-bit advanced control timer) peripheral on the STM8A or STM8S allows two independent PWM signals to be generated. The PWM signals have the same frequency and are controlled by the counter clock period and the capture/compare register values (TIM1_CCRxH and TIM1_CCRxL). For detailed information about TIM1, refer to the STM8A and STM8S device datasheets. *Figure 7* shows the functional block diagram of a stepper motor, STM8A PWM and I/O signals, and an 74HC244 buffer.

Figure 7. Functional block diagram



The stepper motor used in the current solution is the M-S motor X25.689 from Switec (see *Figure 8* and *Figure 9*). The main features are:

- 1/3 ° resolution per step
- Low current consumption
- High speed (greater than 600 °/s)
- Can be driven directly by a microcontroller

The M-S motor has two types of movement:

- Rotor movement
- Pointer shaft movement

This motor is therefore a bipolar motor. It has four different wires which operate the motor in both clockwise and anti-clockwise directions.

Figure 8. Stepper motor schematic layout

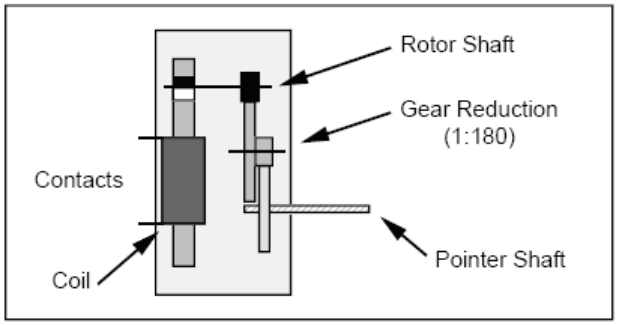
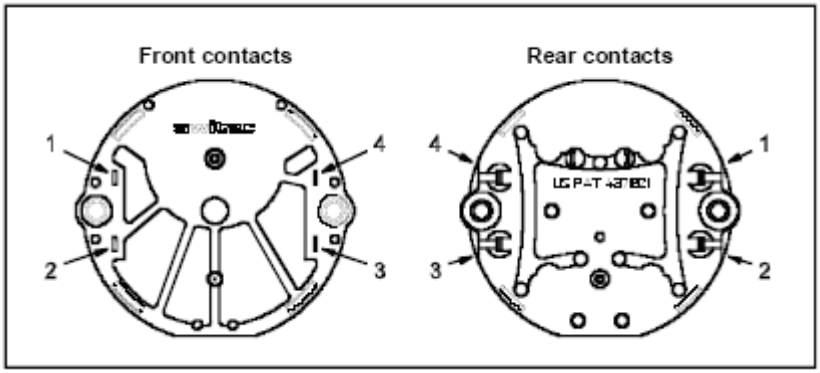


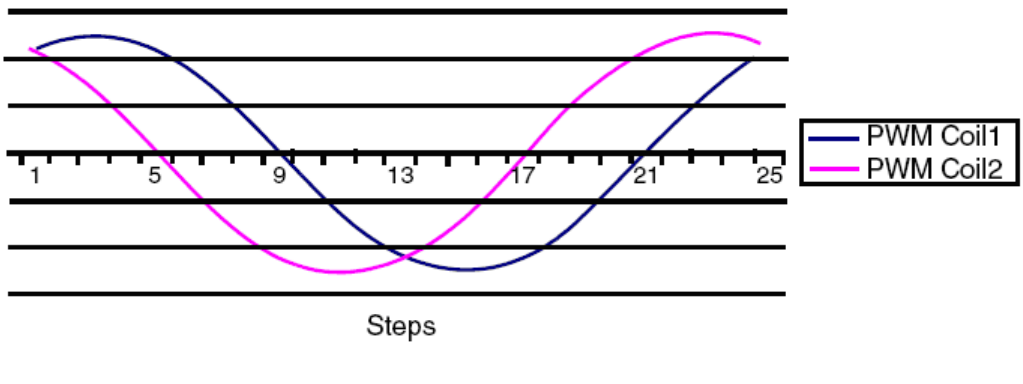
Figure 9. Stepper motor pin configuration



The M-S motor has a gear reduction ratio of 1/180 meaning that 180 ° of rotor movement (defined as a full-step) is converted to a one degree rotation of the M-S shaft. One full-step is divided into three partial steps which in turn are further divided into four microsteps. So, one complete rotation (360 °) of the rotor is equivalent to 24 microsteps of the M-S shaft. One microstep is equivalent to the 1/12 ° movement of the M-S shaft.

In the M-S motor there is a 60 ° phase difference between the two motor winding currents. *Figure 10* shows the current waveforms.

Figure 10. Current waveforms (Switec M-S motor)



The current in the motor windings is controlled by varying the PWM duty cycle values. The PWM duty cycle for each microstep depends on the phase angle of currents in the two windings. The duty cycles of *Table 1* are calculated using the current waveforms of *Figure 10*. The same table can be generated for other motors using the same method. For more details about duty cycle calculation see *Section 5.2: Software description*.

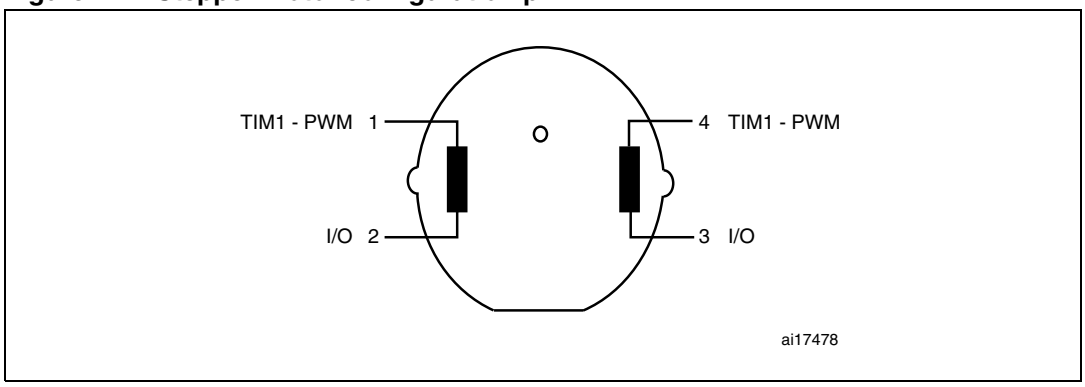
Table 1. PWM duty cycles for an M-S motor

Step no.	Step angle (°)	Phase angle (°) coil 1	Phase angle (°) coil 2	Sine value (°) coil 1	Sine value (°) coil 2	PWM duty cycle (%) coil 1	PWM duty cycle (%) coil 2
1	0	60	120	0.87	0.87	87	87
2	15	75	135	0.97	0.71	97	71
3	30	90	150	1.00	0.50	100	50
4	45	105	165	0.97	0.26	97	26
5	60	120	180	0.87	0.00	87	0
6	75	135	195	0.71	-0.26	71	26
7	90	150	210	0.50	-0.50	50	50
8	105	165	225	0.26	-0.71	26	71
9	120	180	240	0.00	-0.87	0	87
10	135	195	255	-0.26	-0.97	26	97
11	150	210	270	-0.50	-1.00	50	100
12	165	225	285	-0.71	-0.97	71	97
13	180	240	300	-0.87	-0.87	87	87
14	195	255	315	-0.97	-0.71	97	71
15	210	270	330	-1.00	-0.50	100	50
16	225	285	345	-0.97	-0.26	97	26
17	240	300	360	-0.87	0.00	87	0
18	255	315	15	-0.71	0.26	71	26
19	270	330	30	-0.50	0.50	50	50
20	285	345	45	-0.26	0.71	26	71
21	300	360	60	0.00	0.87	0	87
22	315	15	75	0.26	0.97	26	97
23	330	30	90	0.50	1.00	50	100
24	345	45	105	0.71	0.97	71	97
25	360	60	120	0.87	0.87	87	87

Note: Table 1 shows theoretical duty cycle values. Contact the stepper motor supplier for exact/customized values.

A negative sign shows the current flow in the reverse direction. The current flowing through each motor coil is controlled using one PWM output (PWMx) and one I/O port (configured in output push-pull mode) as shown in *Figure 11*. PWM channels are configured to provide the required waveform in synchronization with the GPIOs.

Figure 11. Stepper motor configuration pin



When the I/O output is 0, the current through a coil flows in one direction and when the I/O output is 1, the current flows in the reverse direction.

Step-by-step explanation of *Table 1*:

- Step 1: Coil 1 and coil 2 have the same magnitude (0.87, duty cycle 87 %) and in both coils the current flows in the same direction, I/O value for both coils is 0.
- Step 2: The magnitude of coil 1 is 0.71 (duty cycle 71 %) and the magnitude of coil 2 is 0.97 (duty cycle 97%). The currents in both coils flow in the same direction, I/O value for both coils is 0.

  .

  .
- Step 6: The magnitude of coil 1 is 0.71 (duty cycle 71 %) and the magnitude of coil 2 is -0.26 (duty cycle 26 %). The currents in the two coils flow in the opposite direction. I/O value for coil 1 is 0, I/O value for coil 2 is 1.

For more details about duty cycle calculation see *Section 5.2: Software description*.

5 Software

5.1 Preliminary information

The software described in this section has been implemented using a 32-Kbyte STM8AF6266 device with 32 pins. It performs the complete needle rotation of one stepper motor (in a forward and back forward direction).

Two PWM channels and two I/O lines are necessary for driving the stepper motor.

Thus, four stepper motors can be driven by STM8AF6266 devices as it is able to provide up to eight PWM signals.

Note: Up to four PWM signals are provided by TIM1, two PWM signals are provided by TIMER2 (leaving one channel free), and another two PWM signal are provided by TIMER3.

5.2 Software description

The software code is based on the STM8A/S standard firmware library available from http://www.st.com.

The firmware consists of:

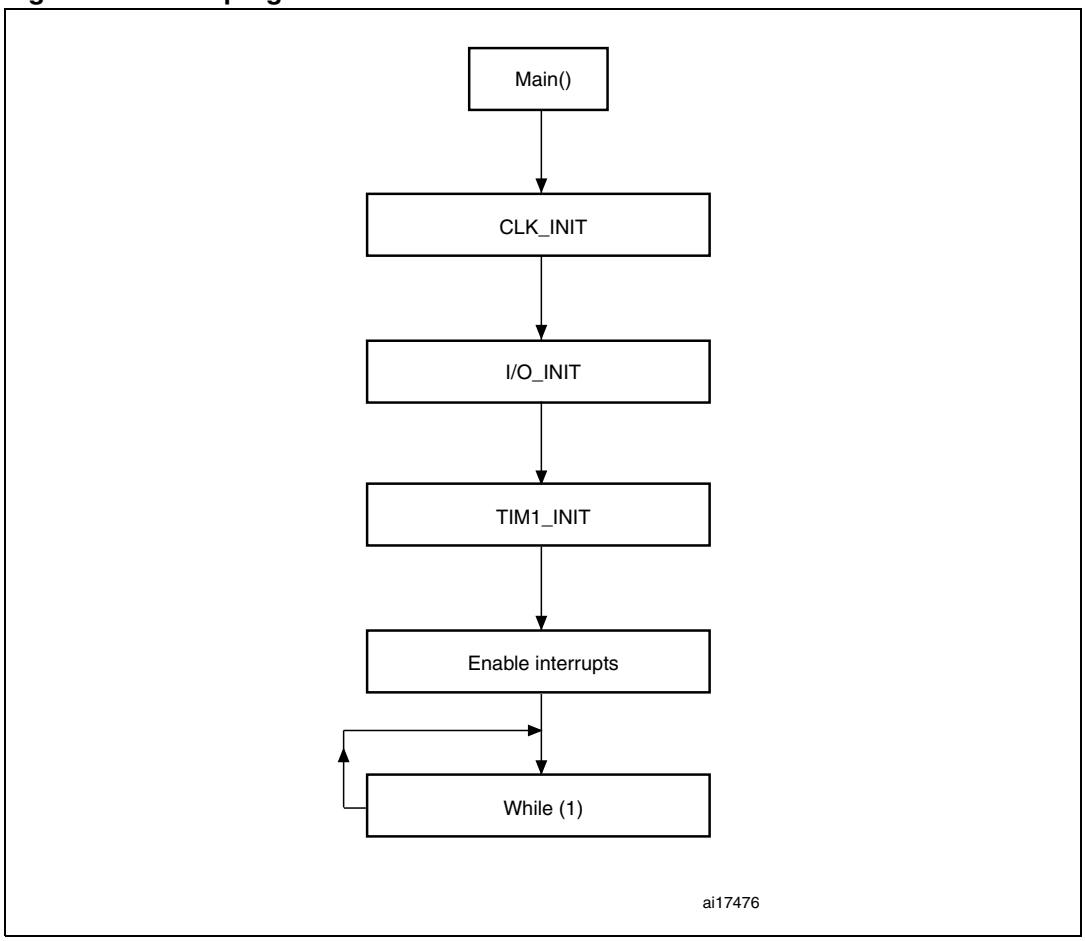
- A main program
- An interrupt routine.

TIM1 interrupt routine, *(TIM1_UPD_OVF_TRG_BRK_IRQHandler()),* is used for providing PWMs signals with correct duty cycles.

5.2.1 Main program

The main program initializes the peripherals (GPIOs, timers, clock CNTRL), the variables, and the interrupt routines. After initialization, it stays in an endless loop (see *Figure 12*).

Figure 12. Main program flow



CLK_Init(): Is the clock initialization routine. For this application solution, an HSI clock has been used. The CPU frequency is 16 Mhz.

I/O Init(): Is the GPIO initialization routine. I/O pins dedicated for stepper motor driving are configured in output push-pull mode.

TIM1_Init(): Is the initialization routine of TIM1. TIM1 is configured in “PWM edge alignment mode” providing four PWM signals at 60 Khz.

5.2.2 TIM1 interrupt routine

The microsteps are output from the TIM1 interrupt routine (see *Figure 13: TIM1 interrupt (microstep output) flow chart*. Each microstep output is obtained by changing the PWM duty cycle according to *Table 1*. *Table 2* provides the TIM1CCRx register values required to obtain the PWMx duty cycle and I/O output values for each microstep position.

Table 2. Specific duty cycles of the TIM1 registers and I/O values

Step no.	Step angle (°)	PWM duty cycle coil 1 (%)	PWM duty cycle coil 2 (%)	TIM1_CCR1 register values (PWM1)	TIM1_CCR2 register values (PWM2)	I/O values coil 1	I/O values coil 2
1	0	87	87	0x74	0x74	0	0
2	15	97	71	0x81	0x5F	0	0
3	30	100	50	0x86	0x43	0	0
4	45	97	26	0x81	0x22	0	0
5	60	87	0	0x74	0x00	0	0
6	75	71	26	0x5F	0x63	0	1
7	90	50	50	0x43	0x43	0	1
8	105	26	71	0x22	0x26	0	1
9	120	0	87	0x00	0x11	0	1
10	135	26	97	0x63	0x04	1	1
11	150	50	100	0x43	0x00	1	1
12	165	71	97	0x26	0x04	1	1
13	180	87	87	0x11	0x11	1	1
14	195	97	71	0x04	0x26	1	1
15	210	100	50	0x00	0x43	1	1
16	225	97	26	0x04	0x63	1	1
17	240	87	0	0x11	0x86	1	1
18	255	71	26	0x26	0x22	1	0
19	270	50	50	0x43	0x43	1	0
20	285	26	71	0x63	0x5F	1	0
21	300	0	87	0x86	0x74	1	0
22	315	26	97	0x22	0x81	0	0
23	330	50	100	0x43	0x86	0	0
24	345	71	97	0x5F	0x81	0	0
25	360	87	87	0x74	0x74	0	0

The TIMCCRx register values in *Table 2* have been calculated under the STM8 configurations given below and using *Equation 1*.

STM8 configurations

- f_Master = 16 MHz (HSIDIV[1:0] = 000)
- TIM1_counter_clock = 8 MHz (TIM1_PSCR = 1)
- PWMx frequency = 60 kHz (TIM1_ARR = 0x85)

Equation 1

$$TIM1\_CCRx = (PWMx\_duty\_cycle\_coil\_x) \times (TIM1\_ARR+1)$$

Example

To obtain PWM_duty_cycle_coil1 = 87 %:

TIM1_CCR1 = (0.87) x (0x85 + 1) = 0x74

The maximum step rate/speed possible for the M-S motor is 600 full steps/s or 7200 microsteps/s or 7200 Hz. So, the duration between two consecutive microstep output interrupts should be ≥ 139 µs. The 74x244 buffer/driver current limitation should also be considered to determine the maximum speed with which the stepper motor can be driven.

In this application the step rate is set to 830 µs. This means that the next microstep is output after 50 PWM interrupts because the PWM frequency is 60 kHz (16.6 µs). This choice of step rate is guaranteed without incurring any problems over the minimum limit of 300 µs.

When an overflow interrupt on TIM1 occurs (@ 60 kHz or 16.6 µs), the TIM1 interrupt routine is executed.

The flow chart of the TIM1 interrupt is shown in *Figure 13*.

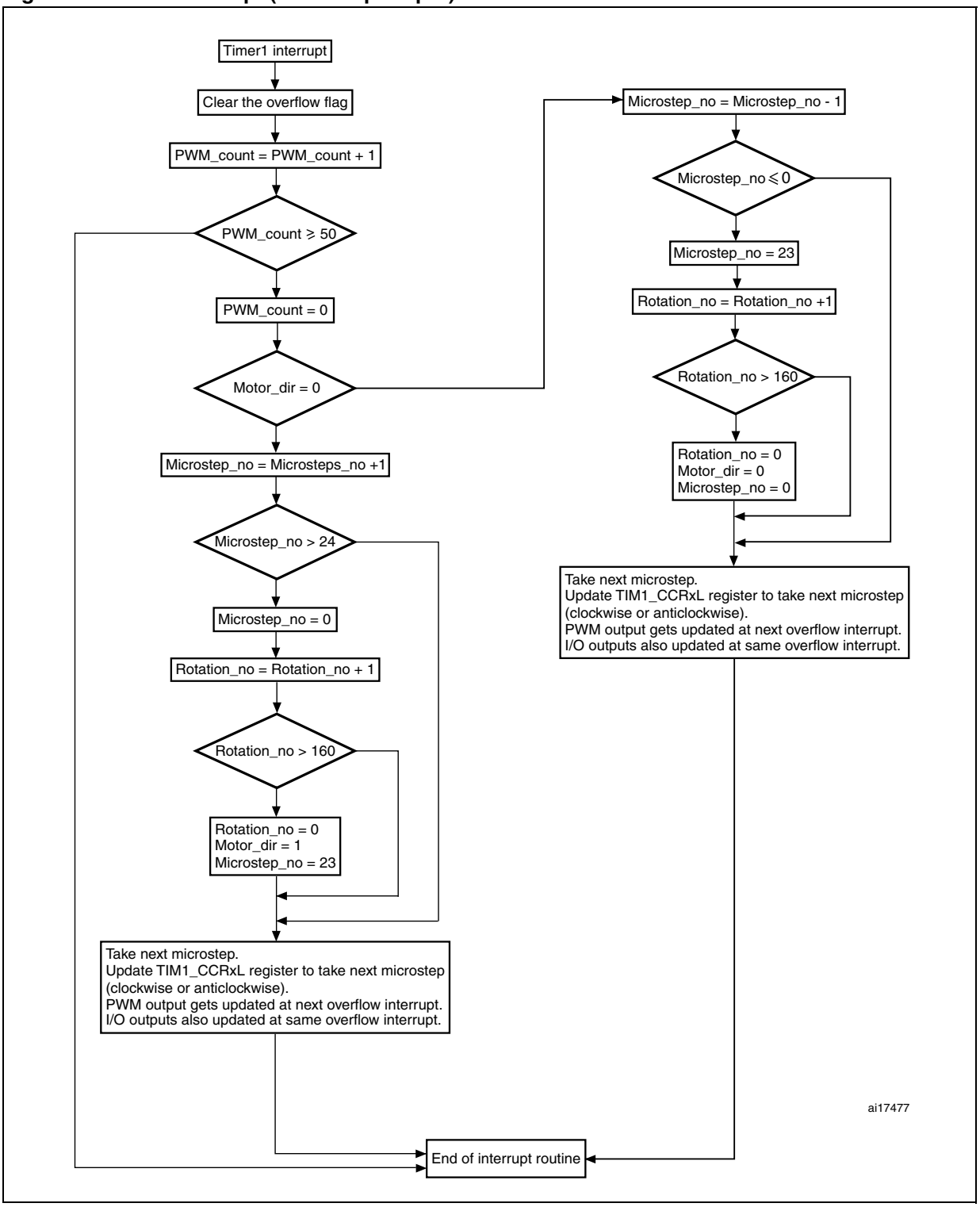
After resetting the overflow flag, the *PWM_Count* is incremented by 1. When this variable reaches a value of 50, the *PWM_Count* is reinitialized to 0, and according to *Motor_dir* information, the needle of the stepper motor moves forwards or backwards (*Motor_dir* = 0 or *Motor_dir* = 1 respectively).

If the stepper needle moves in forwards, the microstepping number (*Microstep_No*) is incremented by 1 (up to 24).

In this application the pointer shaft of the stepper motors moves the stepper needle from 0 to 320 °. Taking into account that after 24 microsteps the pointer shaft has moved by 2°, a complete movement of 320 ° (the *Rotation_No* variable which represents the number of rotations) has to reach 160 (160 rotations multiplied by 2 °).

After one complete movement, *No_rotation* is set to 0, while *Motor_dir* is set to 1. From this point on, the needle moves backwards and, the *Microstep_No* decreased from 23 to 0.

Figure 13. TIM1 interrupt (microstep output) flow chart



6 Revision history

Table 3. Document revision history

Date	Revision	Changes
06-Jul-2010	1	Initial release