Design of Ultra-low-end Controllers for Efficient Stepper Motor Control

Mordecai Raji[1],Akeem Shokanbi[3] and Happy Monday[2]

[1,a]*University of Electronic Science and Tech. of China, Electronics Department,611731West Hi-Tech Zone,Chengdu,Sichuan,China*
[2]*University of Electronic Science and Tech. of China, Electronics Department,611731 West Hi-Tech Zone, Chengdu,Sichuan, China*
[3]*University of Electronic Science and Tech. of China, Automation Department,611731 West Hi-Tech Zone, Chengdu,Sichuan, China*

Abstract. This paper lays emphasis on thedevelopment of low cost controllers for stepper motors in contrast to its resource limitations such as memory size, few I/O pins and computing power compared to High-end designs. The microchip AVR ATtiny45 microcontroller was employed alongside a redesigned (reduced-input pin count) pulse distribution circuit for two H-Bridge drivers. The motor can rotate in both directions as well as possible speed control. The concept of the motor control signals was modeled in Matlab/Simulink, firmware was written in Atmel AVRStudio development environment, while theoverall design was carried out in Proteus software followed by Hardware implementation. Total material cost is about $5 which would be less in commercial production cases.

1Introduction

Since the advent of stepper motors, it has gained widespread use in different applications including medical equipment, robotics, antenna positioning, printers, scanners, industrial applications, disk drives including micro-surgical operations, etc. The widespread acceptance is due to its advantage over conventional DC/AC motors such as precise motion, position,and speed, more so, compared to conventional motors, stepper motors are more flexible to control [1][2][3]. Stepper motors are brushless motors, so they can only be commutated (in the right sequence) by digital signals to achieve motion. For a bipolar stepper motor, signals from the controller are coupled to the coils with two H-Bridge as shown below.

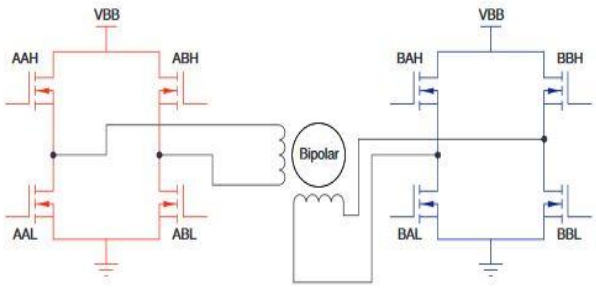

Figure 1.H-Bridge circuit.

The applied signal can either make the motor move or hold a position. Motion can be afull step, half step, and micro-step to higher resolutions of micro-steps, therefore the performance of stepper motors depends largely on the controller. Most stepper motor controllers are based on (mid-end) MCUs, ARMs, CPLDs, DSPs and FPGAs[3] which usually include four kinds of systems; they are microcontroller (MCU) system, professional motion control and programmable logic controller (PLC) system, PC and motion control system respectively. All the above control methods require the user to further develop Windows program [3].

The purpose of this paper is to design and implement a very simple and cost-effective stepper motor controller with micro stepping capabilities. Experimental setup features the Microchip ATtiny45 and a custom driver circuit to achieve comparable features and results as that of high-end controllers. For thefurther sake of simplicity, the control buttons for motor direction of rotations are also used for step functions (and speed control). Arguably, interfacing the ATtiny45 AVR microcontroller (or other 8 pin microcontroller) with many stepper motor driver ICs will leave room for little or no other I/O function because it has only 6 I/O pins. An example; the L298 dual H-bridge monolithic stepper motor driver has 6 control pins.

2Theories of operation

2.1 Concept of PWM and microstepping

Stepper motors can be operated in full step and half step. The need for higher resolution, less vibration,and smoother movement steps gave birth to the concept of microstepping. Therefore to achieve microstepping, the technique of Sinusoidal Pulse Width Modulation (PWM) is usually employed. SPWM is the sinusoidal variation of the Pulse Width of an output Signal. It is typically generated from the output of a comparator; fed in one of it input a carrier signal and the other a modulating signal. Figure 2 depicts AVR PWM timer in Fast PWM mode

using sawtooth waveform as the carrier signal and the compare (reference) signal for modulation. So, SPWM can simply be achieved by sinusoidal variation of the reference signal level using the Sine Look-Up Table stored in the EEPROM of the microcontroller.

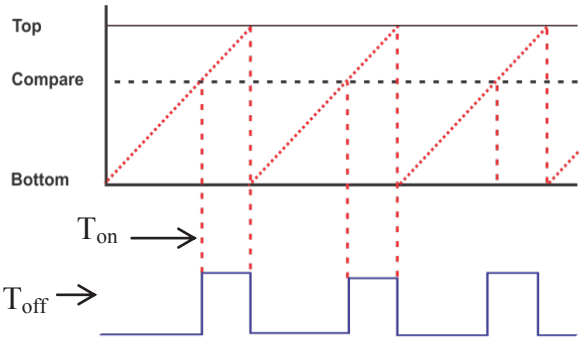

Figure 2. Pulse width modulation.

The Average value of the output waveform is given by [4]

$$\bar{y} = \frac{1}{T}\int_0^T f(t)\,dt. \tag{1}$$

where T is the period and f(t) is the pulse waveformThe above expression is simplified as

$$\bar{y} = D \cdot y_{max} \tag{2}$$

where D is the Duty Cycle and $Y_{min} = 0$.
Bipolar stepper motors require two sine output phased 90 degrees to each other i.e. a sine and a cosine wave[5] as shown in figure 3.

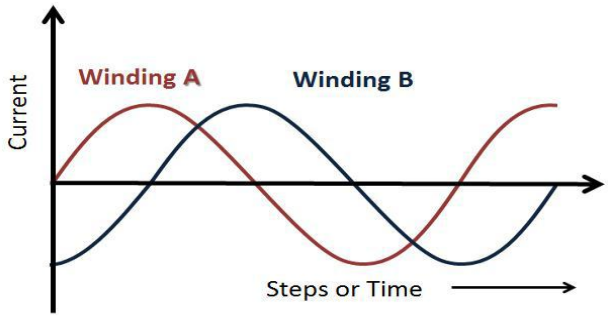

Figure 3. 90-degree current phase difference (for microstepping) through the winding of a stepper motor.

2.2 Control hardware and software

The ATtiny45 is an 8-bit AVR RISC based microcontroller, combines 4KB ISP flash memory, 256-Bytes EEPROM, 256 SRAM and 6 general purpose I/O lines. The figure below reveals the function assigned to each pin. The firmware & Hardware was carefully designed to accommodate all basic functions.

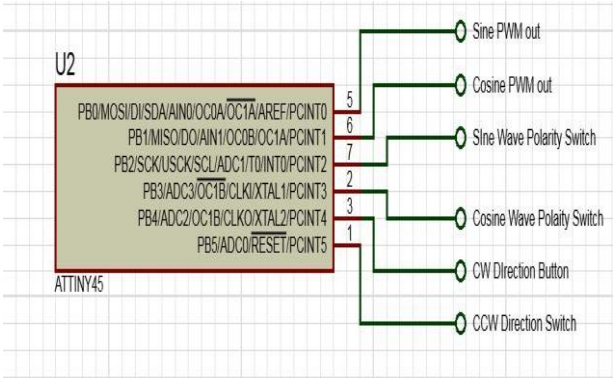

Figure4. Pin-out assignments for the Atiny 45 microcontroller operation.

2.3Control algorithm

The algorithm is optimized for a bipolar stepper motor. The PWM is generated simply from one sine Look-up table(LUT) using two table pointers phased 90 degrees to each other as opposed to using two LUTs which will use up more memory. The outputs of the pointers are sent to the OC0A and OC0B pins (PB0 and PB1 respectively) in fast PWM mode. The direction of the table pointers depends on the button pressed.

In any instance where any of the table pointer indexes is zero, the polarity of the affiliated switch pin will be inverted as to generate anegative sine or negative cosine wave for the required duration. This is essential for the two H-Bridge. At every 90 degrees, the polarity pin changes state: four times in one full rotation. In other to further understand and verify this theory, a model of this concept was created and simulated using Matlab/Simulink. Figure 3 shows the model component setup and the result. The result verifies the appropriate gating signals for the H-Bridge.

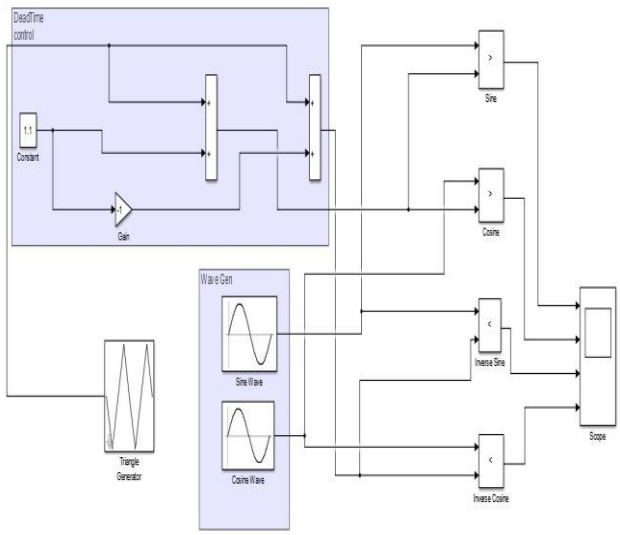

Figure 5. Simulink model.

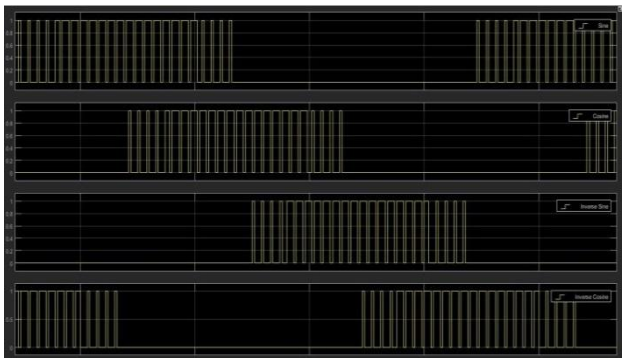

Figure 6. Simulation result of gating signals.

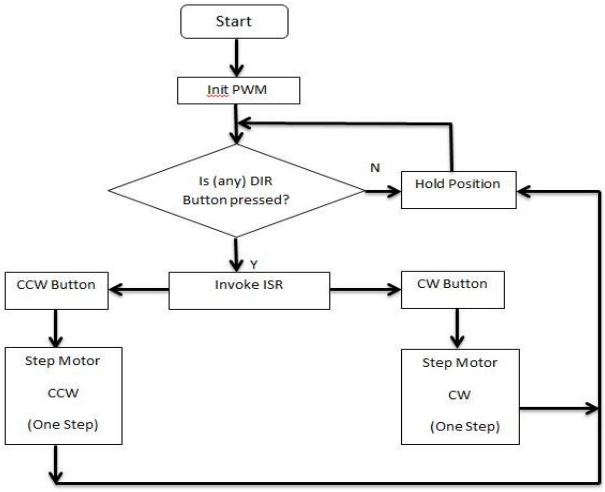

Figure 7. Flowchart.

```
ISR (TIMER0_COMPA_vect)
{
    if((PINB & 0b010000)==0) //if Right button is pressed. Forward direction
    {
        i++; //sine index pointer increment
        j++; //cosine index pointer increment
    }

    if((PINB & 0b100000)==0) //if left button is pressed. Reverse direction
    {
        i--; //sine index pointer decrement
        j--; //cosine index pointer decrement
    }

    if(i==0) // check sine PWM phase angle (180 degrees or 0 degrees)
    {
        PORTB^=(1<<2); //invert polarity
    }

    if(j==0) //check cosine PWM phase angle (180 degrees or 0 degrees)
    {
        PORTB^=(1<<3); //invert polarity
    }

    OCR0A = pgm_read_byte(&sinewave[i]); //update sine PWM output
    OCR0B = pgm_read_byte(&sinewave[j]); //update cosine PWM output
}
```

Figure 8. Control algorithm of the stepper motor.

2.4 H-Bridge driver and control circuitry

Due to the very limited number of GPIO pins of the ATtiny45 (and other 8-pin microcontrollers), driving the two H-Bridge circuits directly would leave little or no room for other control functions. As a result, 'two pre-stage circuits were developed which will serve as adaptors as to reduce the pin count. The circuit is a simple combination of logic gates and also acts as pulse output switch (distribution) pin to the two H-Bridge.

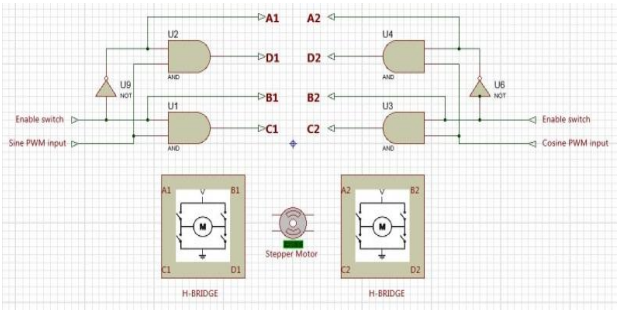

Figure 9. H-Bridge driver circuit.

2.5 Stepper motor

Stepper Motors are classified into Variable Reluctance Motor (VRM), Permanent Magnet (PM) Motor and Hybrid Motors (HM). PM has relatively lower torque than VRM [6] while Hybrid Motors hasan advantages over the other two such as high torque and ability to achieve higher speed. Therefore for the experimental setup, a hybrid stepper motor was selected. The Table below shows the Motor specifications from the manufacturer (NEMA).

Item	Specifications
Step Angle	1.8°
Step Angle Accuracy	± 5% (full step, no load)
Resistance Accuracy	± 10%
Inductance Accuracy	± 20%
Temperature Rise	80°C Max.(rated current,2 phase on)
Ambient Temperature	-20°C~+50°C
Insulation Resistance	100MΩ Min. ,500VDC
Dielectric Strength	500VAC for one minute
Shaft Radial Play	0.02Max. (450 g-load)
Shaft Axial Play	0.08Max. (450 g-load)
Max. radial force	28N (20mm from the flange)
Max. axial force	10N

Figure 10. Stepper motor specifications.

3 Experimental setup, verification and result

For experimental purposes, a high torque hybrid stepper motor is used with the controller. Initially, the setup was put into motor free running mode for easy determination of the controller's efficacy. In this mode, the speed depends on the microcontroller's clock frequency, number of values in the LUT and the number of clock cycle it takes to serve the interrupt each time. For an 8-bit timer, interrupt occurs every 256 clock cycle [7], theClock frequency is 8MHz, no pre-scaling.

Stepper motors have their specific problems; low-frequency oscillations around the synchronism frequency at high speeds [8]; running the motor at higher frequencies (speed) resulted into step loss; primarily due to the drastic current drop in each phase of the motor. A chopper driver should be employed.

At the final test, the pushbuttons were installed; each corresponding to the motor's direction of rotation. The microcontroller was reprogramed to respond to the pushbuttons'low-level signals. The pushbuttons are on

PB4 and PB5 for forward and reverse direction respectively. PB5 serves the same function as the factory-configured reset pin; therefore the Microcontroller's corresponding reset fuse should be disabled for that pin so the pin could function as a typical IO. These pushbuttons could be replaced with a host controller; the host controller can easily determine the motor speed and position by pulse rate and pulse counting respectively without the need for a rotary encoder. However, positioning and tracking would only be efficient by ensuring the host controller is configured to deliver the right pulse length per count and prevention of step loss because feedback is not provided.

The position of the motor shaft is determined by the torque phasor due to the currents applied to the motor coils [8], the torque phasor is determined as desired by the microcontroller by incrementing or decrementing the index pointer value to the LUT to generate sine wave signals phased 90 degrees to each other. The relationship between the electrical position and the current per phase is given by [9]:

$$\begin{cases} i_A = I_0 \cdot \cos\theta_e \\ i_B = I_0 \cdot \sin\theta_e \end{cases}$$

where Θ_e is the electrical position and I_o is the current per phase.

For practical purpose, the popular L298 dual H-bridge Bipolar Stepper Motor driver is used. It has 6 control input pins, therefore, the microcontroller pins need to be adapted to the driver IC. Figure 9 is an excerpt of the above H-Bridge driver circuit reduced to two NOT gates. The L298 accepts TTL inputs. For the experiment, gate ICs was not used; two transistors were wired as inverters to form NOT gates. The L298 dual H-Bridge driver circuit was chosen because of thehigh current capability of 2A DC operation and 2.5A at 80% duty cycle [10] and it is a flexible driver IC. In most stepper motor configurations, The Enable pins are used for activating or deactivating the H-Bridge but in this design, the Sine PWM output and the Cosine PWM output are fed to the EnA (pin 6) and EnB (pin 11) respectively while the pulse distribution is achieved with the In1-In4 (pin 5, 7, 10, and 12).

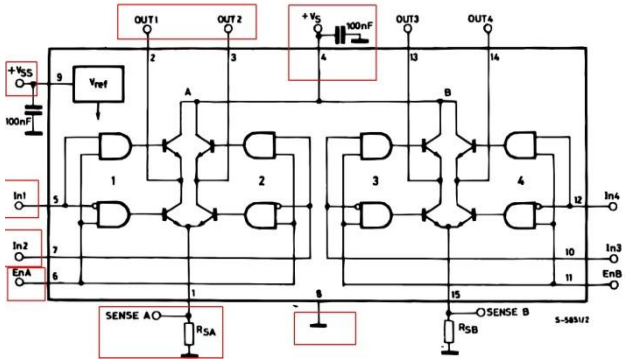

Figure 11. Block diagram of the L298 dual h-bridge monolithic driver.

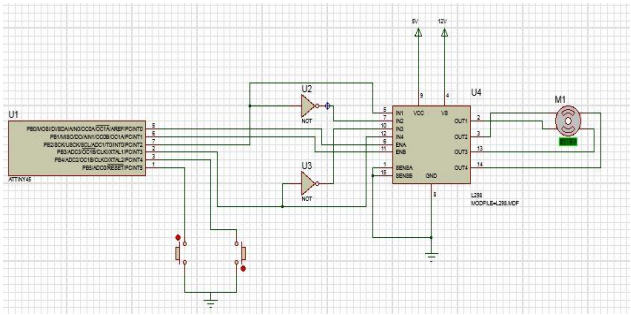

Figure 12. Using with the L298 dual h-bridge driver IC.

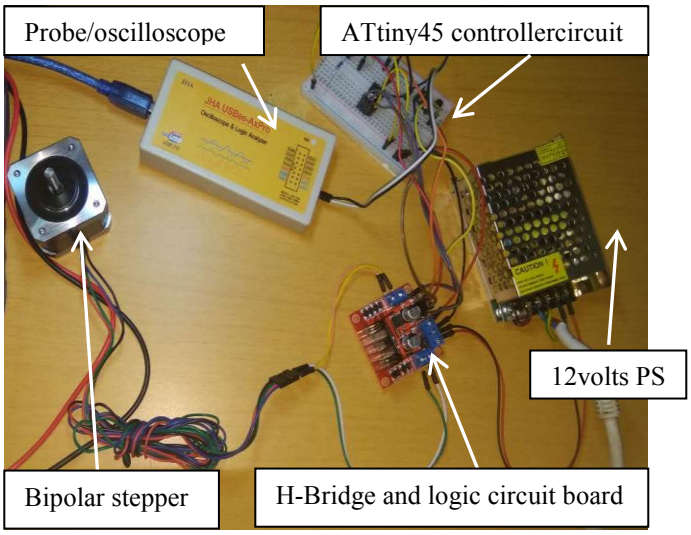

Figure 13. Experimental setup.

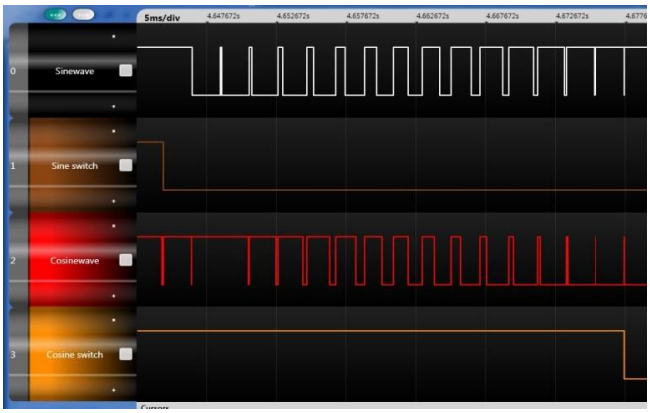

Figure 14.SPWM gating signals fed to the input control pin of the L298 driver IC.

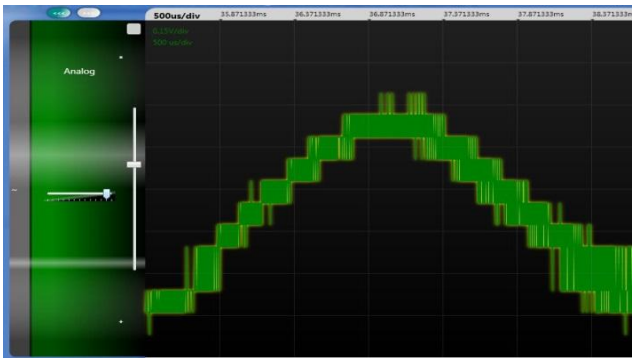

Figure 15. Stepper motor voltage waveform in one phase of the stepper motor (16 micro steps).

4Conclusion

This paper aims to prove that microcontrollers could be put into more effective use (pushing its boundary) in control systems and other embedded system domains by skillful management of its (limited) resources and that design Engineers should not be quick into resorting to costly controllers (DSPs, FPGAs) without fully exploring its capabilities.

Basic control of a stepper motor has been discussed with simple implementations aimed at achieving results comparable to high-end controllers. Some techniques have been presented to make the circuit as simple as possible with the least possible footprint and cost implementation. Vibration and step loss have been reduced with a considerable increase in torque by either adjusting the controller's clock frequency or the number of samples in the LUT at the least. Resonance is minimal, step resolution is high and setup is fairly easy**.** In a commercial purpose, due to the simplicity of the system, the microcontroller and the H-Bridge driver could be integrated to a single chip as a mixed-signal Integrated circuit or on a single circuit board.

Future work would be to squeeze more functions into a controller of the same level without compromising efficiency such as running the pushbuttons on a single pin of the microcontroller by exploring the inbuilt ADC. Thetwo pushbuttons (or even more) will be identified by different voltage levels.This would leave a pin vacant which could be used for other functions such as sensing current through the stepper motor or for positioning.

References

1. M. Y. Tarnini, "Fast and Cheap Stepper Motor Drive," *4th International Conf. Renew. Energy Res. Appl.*, **vol. 5**, no. 2, pp. 689–693 (2015)
2. W. Ruifeng, W. Zhe, and W. Liying, "Stepper Motor Control based on AT89S51 Microcontroller," *2015 8th Int. Conf. Intell. Comput. Technol. Autom.*, pp. 1–4 (2015)
3. G. L. ZHANG Benhua, LI Chenghua, SUN Shiming, "Design on a Unipolar and Unidirectional Stepper Motor Circuit," *Int. Conf. Electron. Mech. Eng. Inf. Technol.*, pp. 1795–1797 (2011)
4. https://en.wikipedia.org/wiki/Pulse-width_modulation
5. A. Nk, D. Krishnan, S. Moorthi, and M. P. Selvan, "FPGA Based Microstepping Scheme for Stepper Motor in Space-Based Solar Power Systems," *2012 IEEE 7th Int. Conf. Ind. Inf. Syst.*, pp. 6–10(2012)
6. Ł. Przeniosło and M. Hołub, "Development of microprocessor,timeoptimized stepper motor driving algorithm," *2017 22nd Int. Conf. Methods Model. Autom. Robot.*, pp. 174–179 (2017)
7. "PWM Sine Wave Generation," https://web.csulb.edu/~hill/ee470/Lab%202d%20-%20Sine_Wave_Generator.pdf.
8. G. Baluta, "Microstepping Mode for Stepper Motor Control," *2007 Int. Symp. Signals, Circuits Syst.*, pp. 1–4(2007)
9. Acarnley, "Stepping Motors: a Guide to Modern Theory and Practice, "4th ed., IEE Control Engineering Series 63, ISBN: 0-85296-029-8, . T E. Michael Faraday House pp.48-51 (2002)
10. http://www.st.com/content/ccc/resource/technical/document/datasheet/82/cc/3f/39/0a/29/4d/f0/CD00000240.pdf/files/CD00000240.pdf/jcr:content/translations/en.CD00000240.pdf