AVR360: Step Motor Controller

Features

- **High-Speed Step Motor Controller**
- **Interrupt Driven**
- **Compact Code (Only 10 Bytes Interrupt Routine)**
- **Very High Speed**
- **Low Computing Requirement**
- **Supports all AVR® Devices**

Introduction

This application note describes how to implement a compact size and high-speed interrupt driven step motor controller. Step motors are typically used in applications like camera zoom/film feeder, fax machines, printers, copying machines, paper feeders/sorters and disk drives.

The high performance of the AVR controller enables the designer to implement high speed step motor applications with low computing requirements of the controller.

Theory of Operation

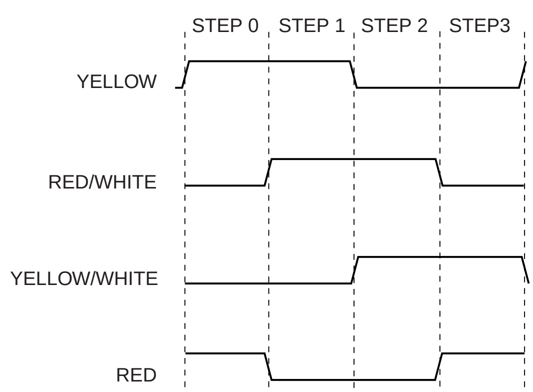
A DC step motor translates current pulses into motor rotation. A typical motor contains four winding coils. The coils are often labeled red, yellow/white, red/white and yellow, but may have other colors. Applying voltage to these coils forces the motor to step one step.

In normal operation, two winding coils are activated at the same time. The step motor moves clockwise one step per change in winding activated. If the sequence is applied in reverse order, the motor will run counterclockwise.

The speed of rotation is controlled by the frequency of the pulses. Every time a pulse is applied to the step motor the motor will rotate a fixed distance. A typical step rotation is 1.8 degrees. With 1.8 degree rotation in each step will a complete rotation of the motor (360 degrees) require 200 steps.

By changing the interval of the timer interrupts, the speed of the motor can be regulated, and by counting the number of steps, the rotation angle can be controlled.

Figure 1. Step Motor Step Sequence



Step Motor Controller

Application Note

Rev. 1181A–11/98

Table 1 shows the hexadecimal values to be output to the step motor to perform each step.

Table 1. Step Motor Values

Step	Yellow	Red/ White	Yellow/ White	Red	Hex Value
0	1	0	0	1	9
1	1	1	0	0	C
2	0	1	1	0	6
3	0	0	1	1	3

Software Description

The software uses a 16 bits timer with capture function to generate interrupt every 100 ms. When the interrupt is executed, a new step value is output to PORTB.

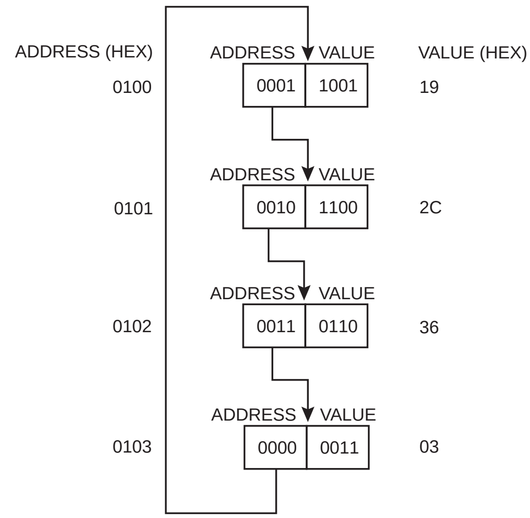
Values for the step motor are stored in flash memory. At startup, the values are copied to SRAM to achieve faster access and maximum speed performance.

In this implementation, the interrupt routine takes 7 cycles + 4 cycles to enter and 4 cycles to exit the interrupt. This totals 15 cycles. With a clock speed of 8 MHz, the step motor control takes less than 2ms. If interrupt is required every 100 ms, the step motor handling takes only 2% of the processing power in the CPU.

In this example the values for the step motor are stored at RAM address 0100 (hex). The upper byte of the RAM address is constant and only the low nibble of the low byte is used to access the address information. See Figure 2.

The lower nibble (4 bits) of the variables is the actual value to control the step motor, the upper nibble holds the address of the next value.

Figure 2. Step Motor Addresses and Values



By using this method, maximum speed can be achieved, combined with a minimum of processor resources.

Resources

Table 2. CPU and Memory Usage

Function	Code Size	Cycles	Register Usage	Interrupt	Description
main	38 words	-	R16, XL, XH, ZL, ZH	-	Initialization and example program
OC1A	10 words	13 + return	R16, XL, XH	Timer 1 output compare A	Output step motor value and calculate next value
TOTAL	48 words	-	R16, XL, XH, ZL, ZH		

Table 3. Peripheral Usage

Peripheral	Description	Interrupts Enabled
4 I/O pins	Step motor output pins	
Timer 1	Generate timer interrupt for step motor frequency generation	Timer 1 output compare A

Code Listing

```
;*******************************************************************
;* A P P L I C A T I O N   N O T E   F O R T H E A V R   F A M I L Y
;*
;* Number:AVR360
;* File Name:          "stepmot.asm”
;* Title               :Simple high speed step motor controller
;* Date                :98.07.02
;* Version             :1.00
;* Support telephone   :+47 72 88 43 88 (ATMEL Norway)
;* Support fax         :+47 72 88 43 99 (ATMEL Norway)
;* Support E-mail      :avr@atmel.com
;* Target MCU          :All AVR devices
;*
;****************************************************************

.include "..\8515def.inc"

;*************** Define global registers ************************
.def    temp       = R16

;*************** Define constants ******************************

.equ    c_value    = 500                          ;Compare value for output compare interrupt
                                                  ; 500 cycles@5Mhz = 100us

;**************************************************************
;*
;*      PROGRAM START - EXECUTION STARTS HERE
;*
;****************************************************************

.cseg
                                                  ;Initialize interrupt vectors
.org    0x00
        rjmp       main
.org    OC1        Aaddr                          ;Init Output compare A interrupt vector
        rjmp       OC1A


;*****************************************************************
;*
;* OC1A- Timer1 Output compare A interrupt routine
;*
;*
;* DESCRIPTION
;*
;*This interrupt routine load new step motor value from the step
```

```
;*motor table in SRAM. The values in the table have two functions,
;*the lower nibble contains the value to output to the step motor.
;*The upper nibble holds the address of the next value. First the
;*step value is output to the port, next the address is moved to
;*the XL register.
;*
;* Number of words    :6 + return
;* Number of cycles   :7 + return
;* Low registers used :None
;* High registers used :3 (temp,XL,XH)
;***************************************************************


OC1A:   in      temp,SREG
        push    temp
        ld      temp,X                          ;Load temp with X pointer value
        mov     XL,temp                         ;Move value to X pointer
        andi    temp,0x0F                       ;Mask away upper nibble
        out     PORTB,temp                      ;Output lower nibble to step motor
        swap    XL                              ;Swap upper and lower nibble
        andi    XL,0x0F                         ;Mask away upper nibble, address is ready
        pop     temp
        out     SREG,temp
        reti

;*************************************************************
;*
;* Main Program
;*
;*This program initialize Timer 1 output compare interrupt to
;*occur with a interval defined with the c_value constant.
;*The step motor lookup table is loaded from the flash and stored
;*in SRAM address 0x0100 to achieve maximum speed.
;*
;******************************************************************
;***** Code

main:   ldi     r16,high(RAMEND)                ;Intialize stackpointer
        out     SPH,r16
        ldi     r16,low(RAMEND)
        out     SPL,r16
        ldi     temp,0x0F                       ;Set PORTB pin3-0 as output
        out     DDRB,temp
        ldi     temp,0x00
        out     PORTB,temp                      ;Write initial value to PORTB
        ldi     temp,high(c_value)              ;Load compare high value
        out     OCR1AH,temp
        ldi     temp,low(c_value)               ;Load compare low value
        out     OCR1AL,temp
        ldi     temp,0x00
```

```
        out     TCNT1H,temp                     ;Clear timer high byte
        out     TCNT1L,temp                     ;Clear timer low byte
        out     TCCR1A,temp                     ;Clear timer control reg A
        ldi     temp,0x40
        out     TIFR,temp                       ;Clear pending timer interrupt
        out     TIMSK,temp                      ;Enable Timer compare interrupt

        ldi     ZH,high(step*2)                 ;Init Z pointer to step table in flash
        ldi     ZL,low(step*2)
        ldi     XH,high(0x0100)                 ;Init X pointer to RAM location
        ldi     XL,low(0x0100)

        ldi     temp,0x04                       ;Load counter value
load:   lpm                                     ;Load step value from flash
        st      X+,R0                           ;Store step value in RAM
        adiw    ZL,0x01                         ;Increment flash pointer
        dec     temp                            ;Decrement counter
        brne    load                            ;Continue until table is loaded

        ldi     XH,high(0x0100)                 ;Initialize X pointer to RAM location
        ldi     XL,low(0x0100)

        ldi     temp,0x9
        out     TCCR1B,temp                     ;Clear timer on compare match,CK/1
        sei                                     ;Enble global interrupt
loop:   rjmp    loop                            ;Do something else

step:      .db     0x19,0x2C,0x36,0x03          ;Step motor lookup table
```

Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe
Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686677
FAX (44) 1276-686697

Asia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road
Tsimshatsui East
Kowloon, Hong Kong
TEL (852) 27219778
FAX (852) 27221369

Japan
Atmel Japan K.K.
Tonetsu Shinkawa Bldg., 9F
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex, France
TEL (33) 4 42 53 60 00
FAX (33) 4 42 53 60 01

Fax-on-Demand
North America:
1-(800) 292-8635
International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

1181A–11/98/xM