Freescale Semiconductor
Application Note

Document Number: AN3410
Rev. 1, 02/2007

Implementing a Sewing Machine Controller with an MC9RS08KA2 Microcontroller

by: Ulises Corrales Salgado
Applications Engineer
RTAC Americas

Contents

1 Introduction

This application note explains how to control motor speed using an MC9RS08KA microcontroller.

Many machines or devices are made entirely with mechanical parts, making maintenance a problem. Electronic devices are easier to maintain and cheaper to produce than mechanical devices.

This document explains how to design a low-cost, digital motor-speed control using an accelerometer and an MC9RS08KA2 microcontroller. This option can replace the pulleys and gears on ordinary mechanical devices, reduce cost, and increase functionality.

1.1 Description

Sewing machines can be classified into three types, depending on their build. The first type is made with mechanical parts and is difficult to use because, to control the motor's speed, you must use both feet to move the pedal (Figure 1).

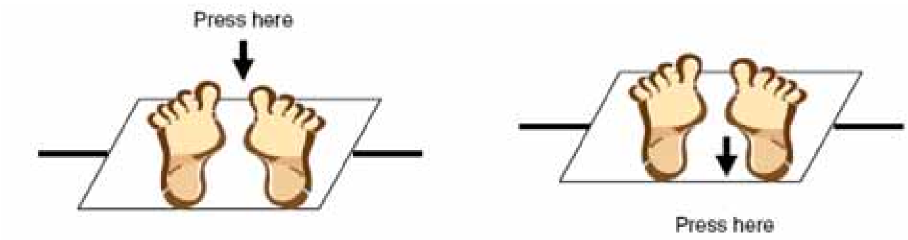


Figure 1. Using a Manual Sewing Machine

The second type has a pedal made with gauges that function like a variable resistor. The resistor values vary depending of the information of the gauge. When you press the gauge, the motor speeds up. When you release it, it slows down (Figure 2).

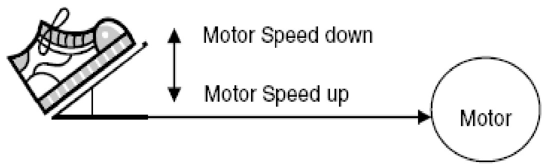


Figure 2. Using an Electric Sewing Machine

The third type of sewing machine has an accelerometer sensor[1] in its pedal. The accelerometer measures the tilt of the pedal. With this, you can speed up or speed down the motor (Figure 3).

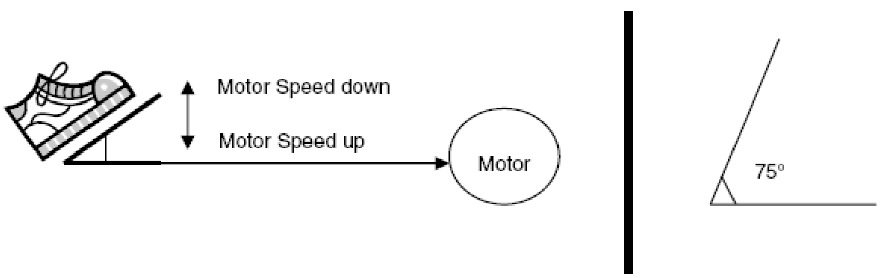


Figure 3. Using a Sewing Machine with an Accelerometer

The accelerometer measures acceleration and angles. In this application, it measures angles from 0° to 75°.

Figure 4 shows the blocks that make up the three types of sewing machines. The yellow blocks (also outlined with thicker lines) indicate the application note’s focus.

1. http://www.freescale.com/files/sensors/doc/data_sheet/MMA7260QT.pdf

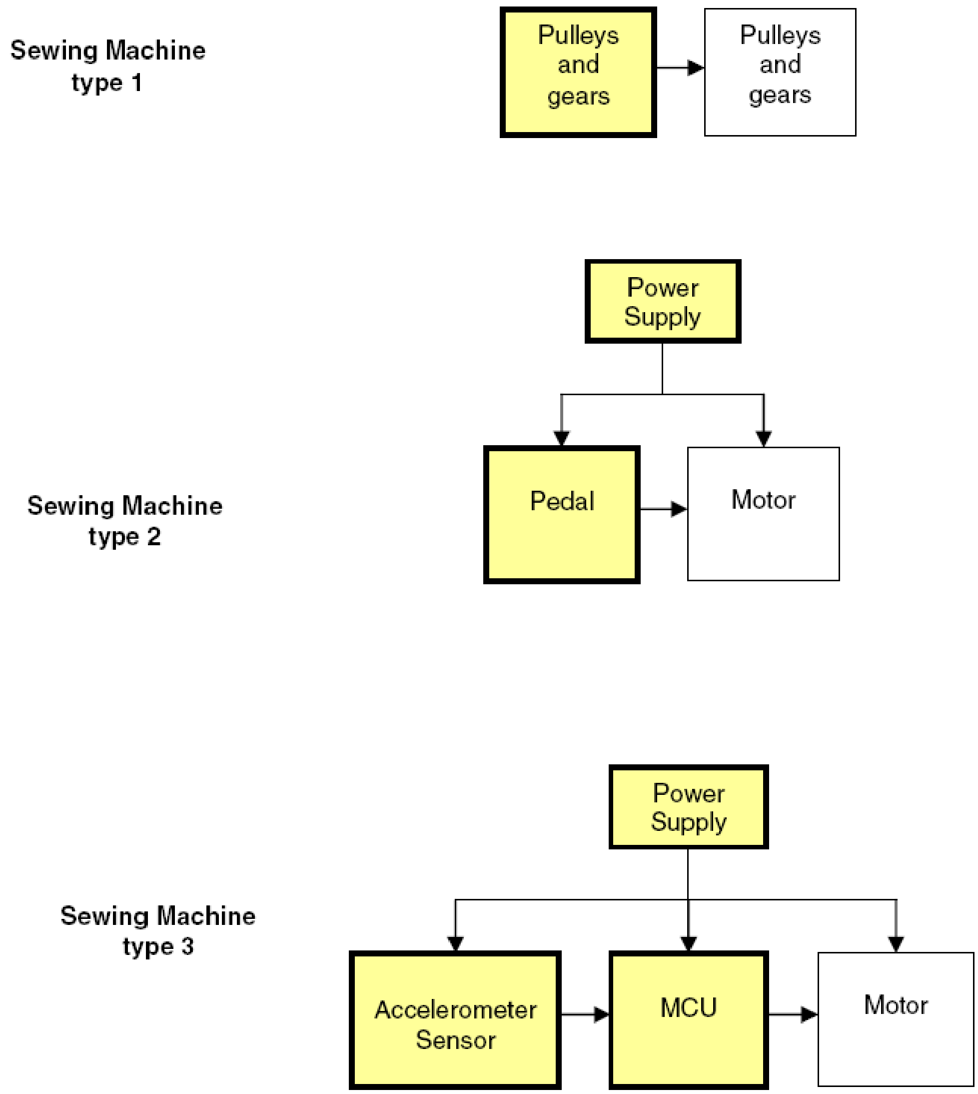


Figure 4. Block Diagrams for Three Sewing Machine Types

See 2.1, "Solution Benefits," for the advantages of our solution.

1.2 Design Requirements

To create the accelerometer controller, you need:

- One MC9RS08KA2 microcontroller
- One MMA7260QT accelerometer sensor
- The power stage from the MCU to the DC motor, depending on the motor characteristics

2 Solution

Our solution depends on the accelerometer sensor's data. The main program reads the analog-to-digital converter (ADC). After the KA2 acquires the data, the microcontroller processes it and generates a pulse-width modulation (PWM), which speeds up or slows the motor. The accelerometer sensor is located in the foot pedal. The motor speed depends on the degree of tilt (Figure 5).

Figure 5. Accelerometer Location

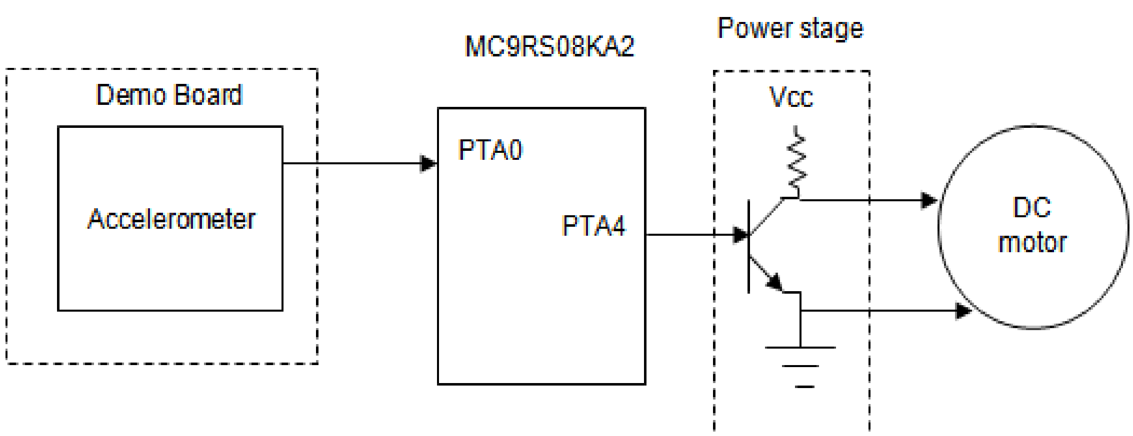


Figure 6. Sewing Machine Block Diagram

NOTE

This device was made using a DC motor. This device will not work using an AC motor. The power-stage block varies depending on the motor characteristics. Also, the MC9RS08KA2 microcontroller does not have an ADC module, so an RC circuit and software obtains the ADC values.

The accelerometer demo board was used in this application note. The accelerometer demo board has an MC9S08QG that configures the sensitivity sensor and also keeps the accelerometer working in active mode. There is no documentation regarding this board except for a schematic (Appendix A, “ Firmware”). You can solve this problem by using hardware such as in Figure 7.

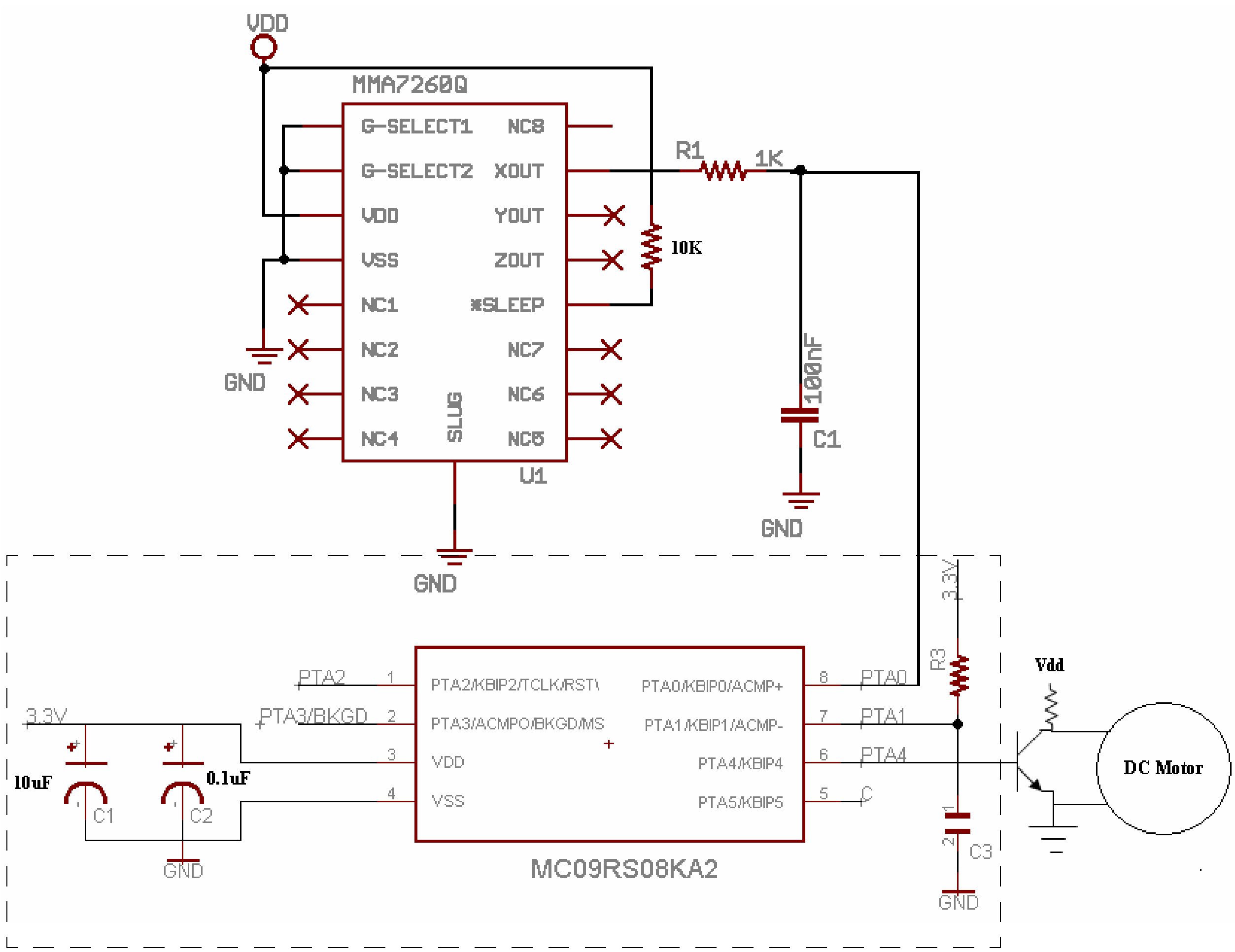


Figure 7. Sewing Machine Schematic Diagram

NOTE

The MCU connection inside the dashed square are part of the KA2 evaluation board. You should make the DC motor connections and accelerometer connections.

2.1 Solution Benefits

Device benefits:

- Smaller foot pedal
- Pedal is not temperature-sensitive. If the temperature varies, the ADC reading is the same.
- The foot pedal is cheaper than a gauge pedal.
- Detects tilt changes more precisely, allowing better control over motor speed.

3 Detailed Description

The MCU firmware first configures the microcontroller (this routine configures the bus clock and disables the COP). After that, the modulo timer is configured. Next, the discharge capacitor is configured.

Data table — This part of the code is used to get values between 0 and 255. The values the SensorReading variable obtains are between 40 and 75. These values cannot be used in the PWM routine because the range values are too short, so adjustments are made to these values.

Lookup table —This part of the code does a page calculation to obtain the data from the data table. In this project, the ADC reads 35 values approximately between 0° to 75°. The values must be extrapolated between 0 and 255 because the lowest value the ADC obtains (0° of tilt) is 40, this value has to extrapolate to 255. You can only obtain the extrapolation values by subtracting seven from the previous value. The maximum extrapolation value is 255.

For example, if you want to use eight values rather than all the values the ADC obtained, extrapolate new values and enter them in the data table in the firmware project. Table 1 and Table 2 show the extrapolation values made in the project and the example extrapolation values obtained.

NOTE

The ADC values are always the same, but you can use the values you want. The ADC values you do not choose are obtained by the extrapolated value of its predecessor. In the example, the values between 40 and 44 are extrapolated to the 255 value.

Table 1. Values from the Project

Values Obtained by ADC	Degree of Tilt (Approximately)	Extrapolation
40	0	255
41	2.2	248
42	4.4	241
43	6.6	234
…	…	…
71	68.2	38

Table 1. Values from the Project (continued)

Values Obtained by ADC	Degree of Tilt (Approximately)	Extrapolation
72	70.4	31
73	72.6	24
74	74.8	17
75	75	0

Table 2. Example Values

Desired ADC values	Extrapolation
40	255
45	220
50	185
55	150
60	115
65	80
70	45
75	0

```
ORG

dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,255,248,241,234,227,220,213,206
dc.b
199,192,185,178,171,164,157,150,143,136,129,122,115,108,101,94
dc.b 87,80,73,66,59,52,45,38,31,24,17,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Figure 8. Extrapolated Values from the Data Table

PWM — The motor speed is controlled in this section of code. There are two main routines: PWMLoopOn and PWMLoopOff.

- The PWMLoopOn routine activates the PTA4 pin, which causes the motor to spin.
- The PWMLoopOff deactivates the PTA4 pin, causing the motor to stop.

The PWM value determines the highest speed a motor can reach and, in this case, the ConvertedValue value.

Configure MTIM — In this section, the MTIM module is configured, and the ACMP module is disabled. A subtraction is executed in this section: 255 – ConvertedValue. The result is saved in the complement variable. This variable is used to turn off the motor in the PWM routine.

User constants — Reserved memory sections that cannot be modified.

User variables — Include values, erase, etc., that can be changed during the program.

Macro definitions — Here is where the internal clock source (ICS) is configured.

Configures port A — This section configures the port A pins. In this program PTA5, PTA4, and PTA1 are configured as outputs.

To understand the project code better, see the program-flow diagram (Figure 9).

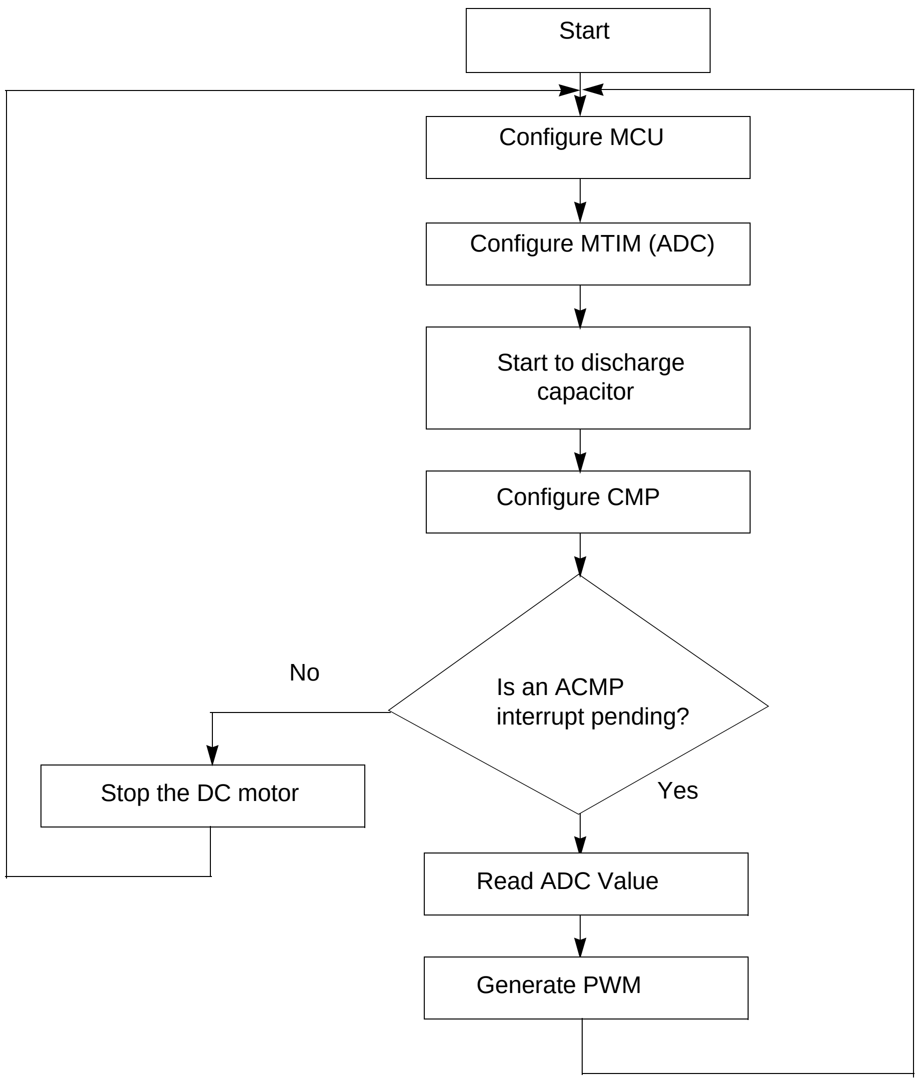


Figure 9. Program Flow

4 How to Download the Program to Flash Memory

To download this project:

1. Open the CodeWarrior™ version 5.1 development tool.
2. Open AppNote.mcp file.
3. Select the option SofTec RS08.
4. Click on the make button.
5. Click on the debug button.
6. Select the DEMO9RS08KA2 board.
7. Click OK.

5 Conclusion

Electronic devices are easier to maintain and support than mechanical or gauge devices. You have better control with an electronic device than a mechanical one. This application note explained how to control a sewing machine's motor speed with an MC9RS08KA microcontroller.

6 References

http://www.freescale.com/files/microcontrollers/doc/data_sheet/MC9RS08KA2.pdf

http://www.freescale.com/files/sensors/doc/app_note/AN3107.pdf?fsrch=1

http://www.freescale.com/files/sensors/doc/data_sheet/MMA7260QT.pdf

7 Glossary

AC — Alternating current

ACMP — Analog comparator

ADC — Analog-to-digital converter

COP — Computer operating properly (watchdog)

DC — Direct current

MCU — Microcontroller unit

MTIM — Modulo timer

PWM — Pulse-width modulation

RC — Resistor capacitor

Appendix A
Firmware

```
;*******************************************************************
;*                          MAIN.ASM                               *
;*******************************************************************
;*          Copyright (c) 2007 Freescale Semiconductor             *
;*          http://www.freescale.com/                              *
;*******************************************************************
;* Implementing a Sewing machine controller with a MC9RS08KA2      *
;* microcontroller. The speed motor depends of tilt pedal.         *
;*******************************************************************
;*  Ulises Corrales Salgado                                         *
;*  Applications Engineer                                           *
;*  RTAC Americas                                                   *
;*******************************************************************


; Include derivative-specific definitions
            INCLUDE 'derivative.inc'
;
; export symbols
;
            XDEF _Startup
            ABSENTRY _Startup

;*****************************************************************************
;*                           User Constants                                  *
;*****************************************************************************

Table_Data EQU $3E00

; Variable declarations
ACMP_ENABLE      SET   $92
ACMP_DISABLED    SET   $20
MTIM_INIT        SET   $50
MTIM_ENABLE      SET   $60
MTIM_STOP_RESET  SET   $30
MTIM_128_DIV     SET   $07
FREE_RUN         SET   $00
```

```
DEBUG_MODE        SET    $00
RUN_MODE          SET    $01

MODE:             EQU    DEBUG_MODE


;******************************************************************************
;*                            User Variables                                  *
;******************************************************************************
;
; variable/data section
;
            ORG    RAMStart         ; Insert your data definition here

ConvertedValue: DS.B   1            ; This varible store converted value
Complement      DS.B   1
Temp_Page       DS.B   1            ; Temporal backup Page
SensorReading   DS.B   1
PcBuffer        DS.B   2            ; Temporal backup SPC

            ORG    ROMStart


;**************************************************************
;*                      MACRO DEFINITIONS                      *
;**************************************************************
TRIM_ICS: MACRO                    ; Macro used to configure the ICS with TRIM
  MOV   #$FF,PAGESEL               ; change to last page
  LDX   #$FA                       ; load the content which TRIM value is store
  LDA   ,x                         ; read D[X]
 CBEQA #$FF,No_Trim     ; Omit the 0xFF value if $3FFA location content

                                    ; the value
  STA   ICSTRM                     ; Store TRIM value into ICSTRM register
No_Trim:
  ENDM

ENTRY_SUB: MACRO                    ; Macro for "stacking" SPC
            SHA
```

```
            STA PcBuffer + 2*(\1)
            SHA
            SLA
            STA PcBuffer + 2*(\1) +1
            SLA
  ENDM

  NOP                            ; Needs to separate MACROS

EXIT_SUB: MACRO                   ; Macro for restore SPC
            SHA
            LDA PcBuffer + 2*(\1)
            SHA
            SLA
            LDA PcBuffer + 2*(\1) +1
            SLA
  ENDM


;***************************************************************
;*                      CONFIGURES PORT A                      *
;***************************************************************
PortA:
  MOV #HIGH_6_13(PTAPE), PAGESEL
  MOV #$FE, MAP_ADDR_6(PTAPE)    ; Enables internal Pulling device

  MOV #HIGH_6_13(PTAPUD), PAGESEL
  MOV #$04,MAP_ADDR_6(PTAPUD)    ;Configures Internal pull up device in PTA  ; 5

  MOV #$32, PTADD                ; PTA5, PTA4 and PTA3 as outputs
  MOV #$00, PTAD

  RTS

;***************************************************************
;*                    CONFIGURES SYSTEM CONTROL                *
;***************************************************************
Init_Config:
  IFNE  MODE
```

```
  MOV #HIGH_6_13(SOPT), PAGESEL
   MOV  #$01,  MAP_ADDR_6(SOPT)      ;  Disables  COP  and  enables  RESET  (PTA2)  pin

  ELSE
  MOV #HIGH_6_13(SOPT), PAGESEL
  MOV #$03, MAP_ADDR_6(SOPT)      ; Disables COP, enables BKGD (PTA3) and RESET
                                  ; (PTA2) pins
  ENDIF
  CLR  ICSC1                      ; FLL is selected as Bus Clock
  TRIM_ICS
  CLR  ICSC2
  RTS

;***************************************************************
;*          Analog Comparator Initial Configuration           *
;***************************************************************
ACMP_Conf:
      MOV #ACMP_ENABLE,ACMPSC     ; ACMP Enabled, ACMP+ pin active, Interrupt
                                  ;enabled, Rising edges detections
      RTS

;***************************************************************
;*            Modulus Timer Configuration for ADC             *
;***************************************************************
 MTIM_ADC_Init:
      MOV #MTIM_128_DIV,MTIMCLK     ; Select bus clock as reference, Set prescaler
                                    ; with 64
      MOV #FREE_RUN,MTIMMOD         ; Configure Timer as free running
      MOV #MTIM_STOP_RESET,MTIMSC
      RTS


;***************************************************************
;*                   Discharge Capacitor                      *
;***************************************************************
Discharge_Cap:
      BSET  1,PTADD                 ; Configure PTA1 as Output
      BCLR  1,PTAD                  ; Start Capacitor discharging
      LDA   #$FE                    ; Set delay time
```

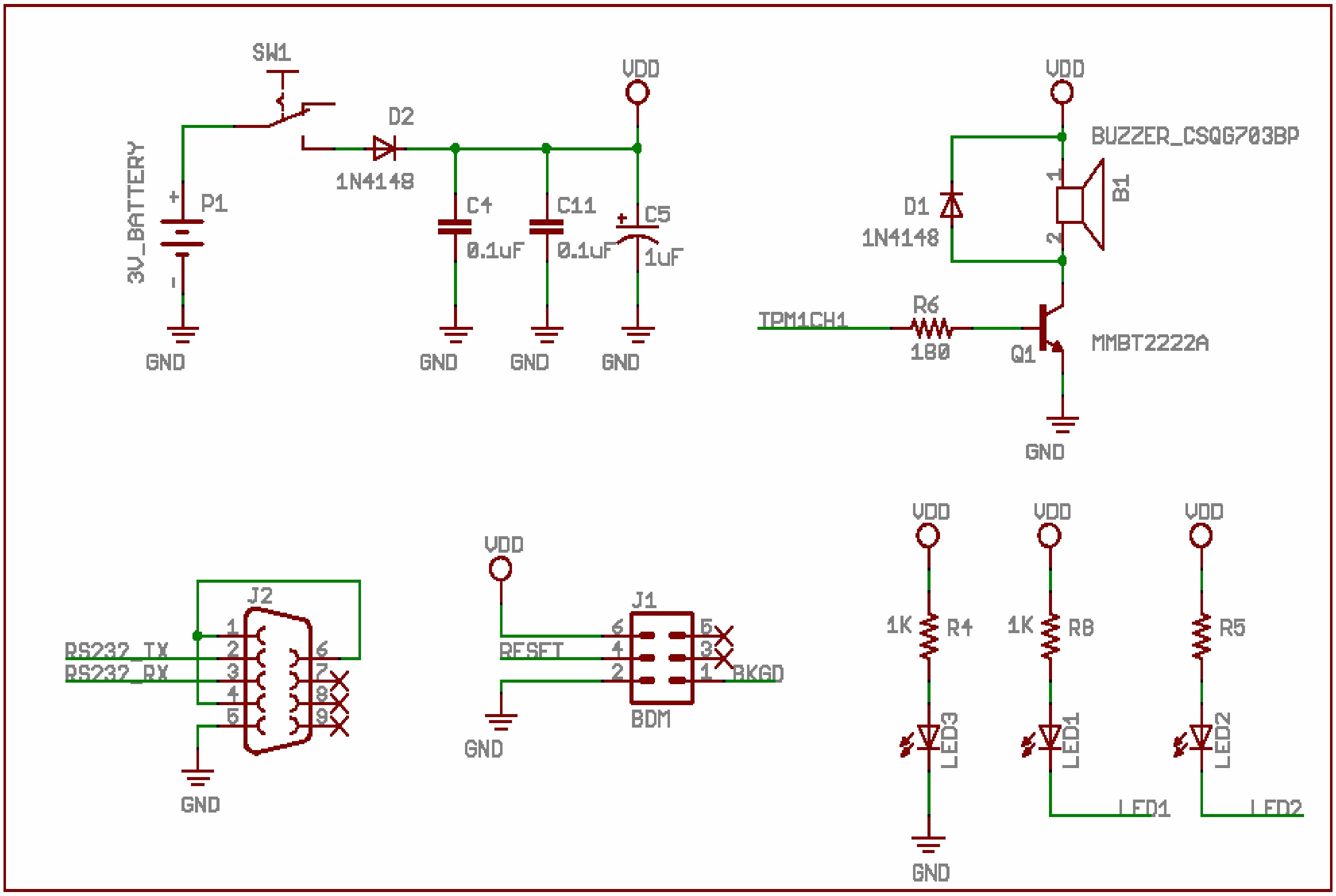
```
Waste_time:
      DBNZA Waste_time              ; Wait until Delay = 0
      RTS


_Startup:

  BSR Init_Config
  BSR PortA
NextValue:
  BSR MTIM_ADC_Init
  BSR Discharge_Cap
  BSR ACMP_Conf                     ; Configure ACMP+ and ACMP-
  MOV #MTIM_ENABLE,MTIMSC           ; Timer Counter Enabled

mainLoop:
  WAIT                              ; Wait for ACMP interrupt
  BSET 4,MTIMSC
  LDA MTIMCNT
  STA SensorReading                 ; Store counter value
  MOV #HIGH_6_13(SIP1), PAGESEL
  BRSET 3, MAP_ADDR_6(SIP1),PWM     ; Branch if ACMP interrupt arrives
  BCLR 7,ACMPSC
  BRA NextValue


PWM:

  MOV #MTIM_STOP_RESET,MTIMSC       ; Stop and reset counter
  MOV #ACMP_DISABLED, ACMPSC        ; ACMP Disabled, Clear Interrupt flag

  LDA SensorReading
  CMP #75
  BHS NextValue
  JSR LookupTable


;***************************************************************
;*                       Configure MTIM                        *
```

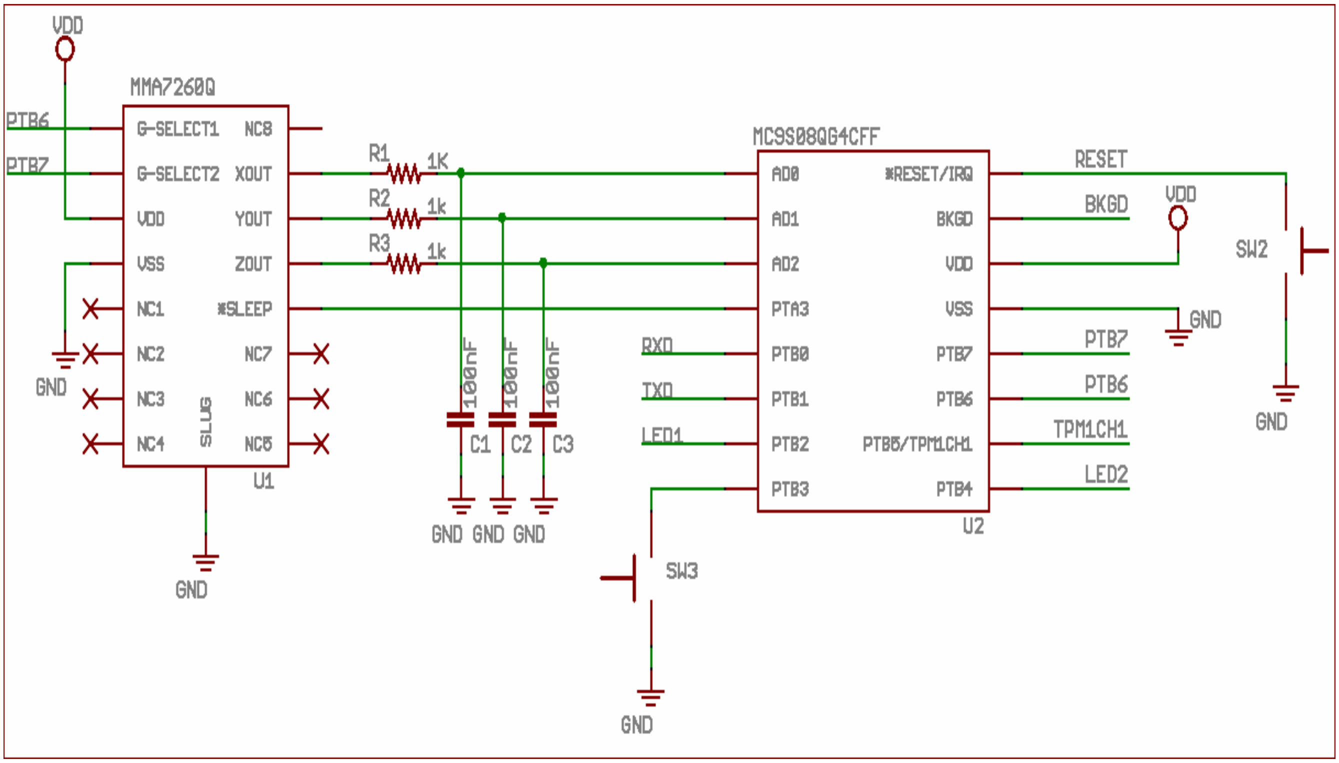
```
;**************************************************************
Activecounter:
  MOV #$00,MTIMCLK          ; Enables interrupt, stops and resets timer counter
  MOV #$01,MTIMMOD          ; MTIM modulo with 1 counts before interrupt.
  MOV #$70,MTIMSC            ; Selects internal clock as reference bus clock (4 MHz)
                            ; with prescaler 1

  MOV #MTIM_STOP_RESET,MTIMSC       ; Stop and reset counter
  MOV #ACMP_DISABLED, ACMPSC        ; ACMP disabled, Clear interrupt flag

  LDA #$FF
  SUB ConvertedValue
  STA Complement

  BCLR 4,MTIMSC                     ; MTIM counter is Active
  BRA PWMLoopOn

  ;**************************************************************
;*                            PWM                                 *
;****************************************************************

PWMLoopOn:
BRSET 7,MTIMSC,PWM_Isr_D            ; Branch if timer interrupt pending
BRA PWMLoopOn
PWM_Isr_D:
BSET 5,MTIMSC                       ; Reset MTIM Counter, Clear overflow flag
BRA PWM_Set_D
PWM_Set_D:
BSET 4,PTAD                         ; Turn on led 4
DBNZ ConvertedValue,PWMLoopOn

PWMLoopOff:
BRSET 7,MTIMSC,PWM_Isr_D1           ; Branch if timer interrupt pending
BRA PWMLoopOff
PWM_Isr_D1:
BSET 5,MTIMSC                       ; Reset MTIM Counter, Clear  overflow flag
BRA PWM_Clear
PWM_Clear:
BCLR 4,PTAD                         ; Turn off led 4
```

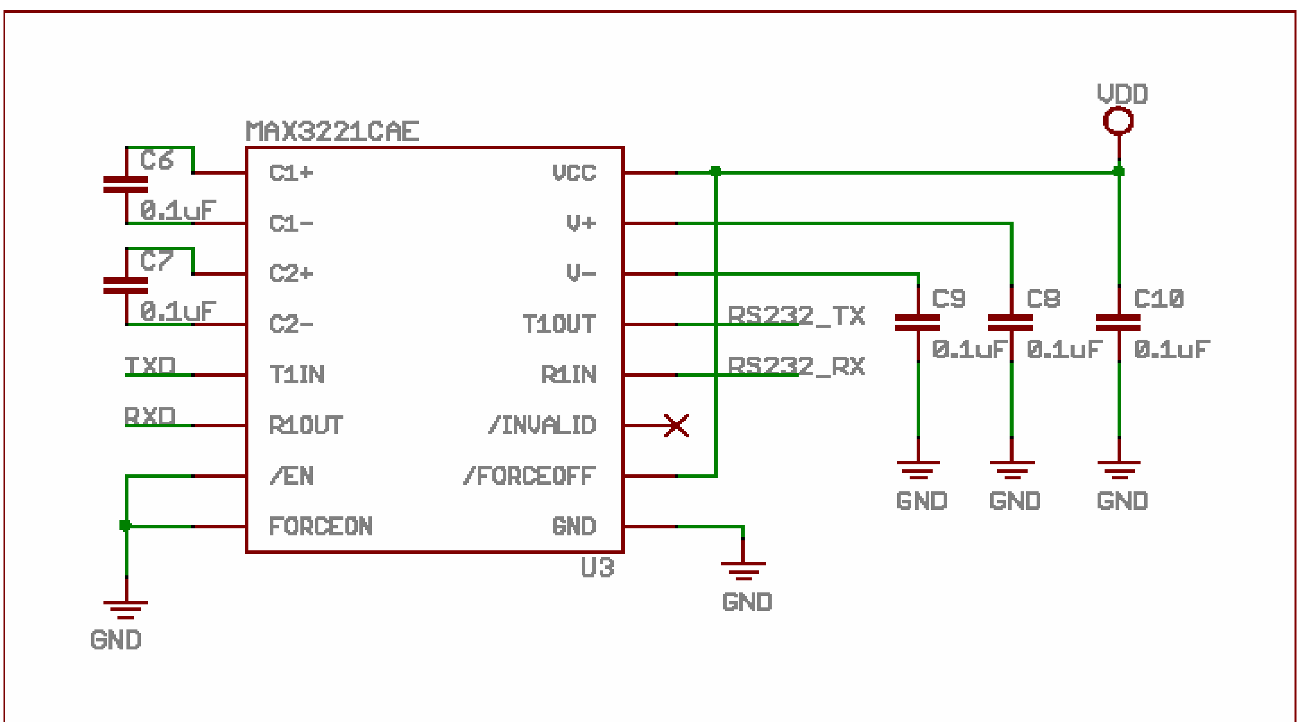
```
DBNZ Complement,PWMLoopOff

BRA NextValue



LookupTable:
      LDA SensorReading
      ROLA                         ; Getting 2 MSB
      ROLA
      ROLA
      AND #$03
      ADD #(Table_Data>>6)         ; Page Calculating
      MOV #PAGESEL,Temp_Page       ; Backup actual page
      STA PAGESEL                  ; Page Change
      LDA SensorReading
      AND #$3F                     ; Extract 6 LSB
      ADD #$C0                     ; Index to paging window
      TAX
      LDA ,x                       ; Load table result
      STA ConvertedValue           ; Store result
      MOV #Temp_Page, PAGESEL       ; Back Page
      RTS

;***************************************************************
;*                  Startup Vector                             *
;***************************************************************
            ORG   $3FFD

            JMP _Startup          ; Reset







;***************************************************************
;*                     Data Table                              *
;***************************************************************
 ORG Table_Data
```

```
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,255,248,241,234,227,220,213,206
dc.b 199,192,185,178,171,164,157,150,143,136,129,122,115,108,101,94
dc.b 87,80,73,66,59,52,45,38,31,24,17,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
dc.b 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Appendix B Accelerometer Demo Board Schematic




VDD
MMA7260Q
PTB6
PTB7
G-SELECT1
G-SELECT2
VDD
VSS
NC1
NC2
NC3
NC4
NC8
XOUT
YOUT
ZOUT
*SLEEP
NC7
NC6
NC5
SLUG
U1
GND
R1
1K
R2
1k
R3
1k
100nF
100nF
100nF
C1
C2
C3
GND GND GND
MC9S08QG4CFF
AD0
AD1
AD2
PTA3
PTB0
PTB1
PTB2
PTB3
*RESET/IRQ
BKGD
VDD
VSS
PTB7
PTB6
PTB5/TPM1CH1
PTB4
U2
RXD
TXD
LED1
RESET
BKGD
VDD
GND
PTB7
PTB6
TPM1CH1
LED2
SW2
GND
SW3
GND


MAX3221CAE
C6
0.1uF
C7
0.1uF
TXD
RXD
C1+
C1-
C2+
C2-
T1IN
R1OUT
/EN
FORCEON
VCC
V+
V-
T1OUT
R1IN
/INVALID
/FORCEOFF
GND
U3
GND
GND
VDD
RS232_TX
RS232_RX
C9
0.1uF
C8
0.1uF
C10
0.1uF
GND
GND
GND

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3410
Rev. 1
02/2007