AN1635

Baseball Pitch Speedometer

**Prepared by: Carlos Miranda, Systems and Applications Engineer and
David Heeley, Systems and Applications Mechanical Engineer**

INTRODUCTION

The Baseball Pitch Speedometer, in its simplest form, consists of a target with acceleration sensors mounted on it, an MCU to process the sensors' outputs and calculate the ball speed, and a display to show the result. The actual implementation, shown in Figure 1, resembles a miniature pitching cage, that can be used for training and/or entertainment. The cage is approximately 6 ft. tall by 3 ft. wide by 6 ft. deep. The upper portion is wrapped in a nylon net to retain the baseballs as they rebound off the target. A natural rubber mat, backed by a shock resistant acrylic plate, serve as the target. Accelerometers, used to sense the ball impact, and buffers, used to drive the signal down the transmission line, are mounted on the back side of the target. The remainder of the electronics is contained in a display box on the top front side of the cage.

Accelerometers are sensors that measure the acceleration exerted on an object. They convert a physical quantity into an electrical output signal. Because acceleration is a vector quantity, defined by both magnitude and direction, an accelerometer's output signal typically has an offset voltage and can swing positive and negative relative to the offset, to account for both positive and negative acceleration. An example acceleration profile is shown in Figure 2. Because acceleration is defined as the rate of change of velocity with respect to time, the integration of acceleration as a function of time will yield a net change in velocity. By digitizing and numerically integrating the output signal of an accelerometer through the use of a microcontroller, the "area under the curve" could be computed. The result corresponds to the net change in velocity of the object under observation. This is the basic principle behind the Baseball Pitch Speedometer.

Figure 1. David Heeley, mechanical designer of the Baseball Pitch Speedometer Demo, tests his skills at Sensors Expo Boston '97.

© Motorola, Inc. 1998

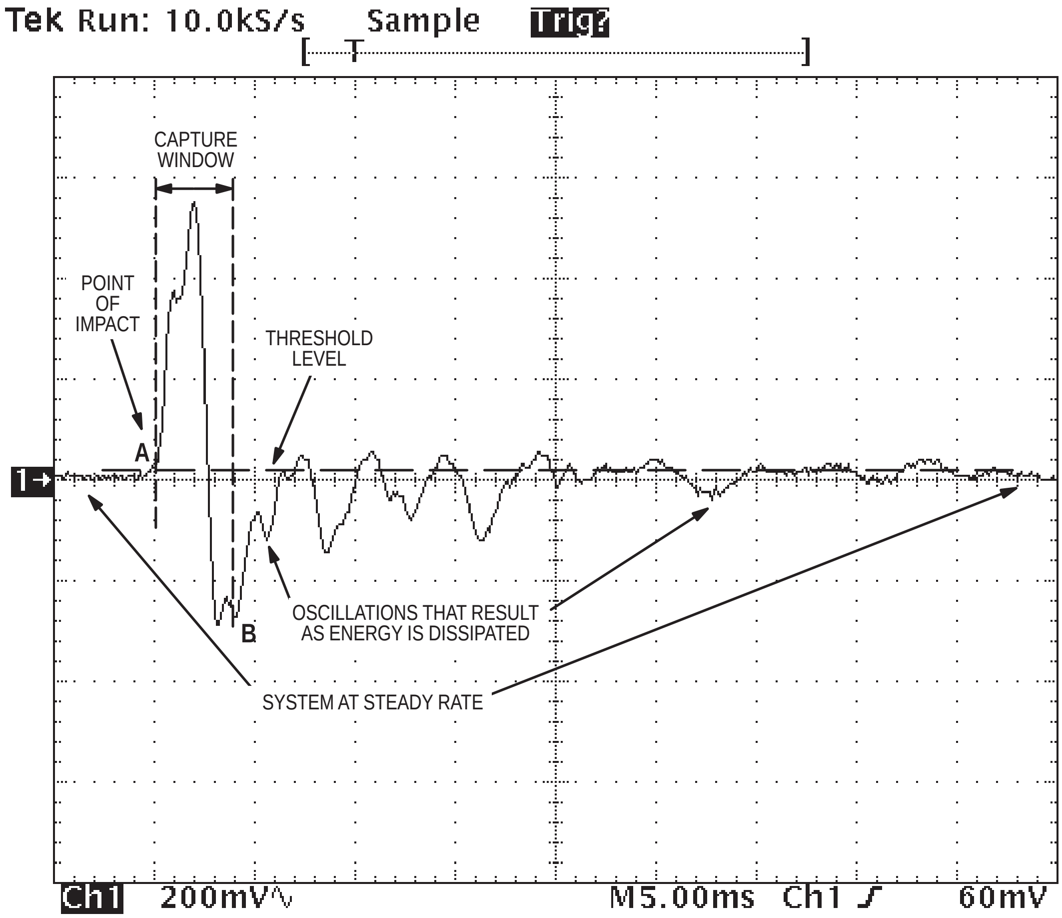

Figure 2. Typical Crash Pattern for the Baseball Pitch Speedometer Demo

THEORY OF OPERATION

When a ball is thrown against the target, the accelerometer senses the impact and produces an analog output signal, proportional to the acceleration measured, resulting in a crash signature. The amplitude and duration of the crash signature is a function of the velocity of the ball. How can this crash signature be correlated to the velocity of the baseball? By making use of the principle of conservation of momentum (see Equation 1). The principle of conservation of momentum states that the total momentum within a closed system remains constant. In our case, the system consists of the thrown ball and the target.

$$m_{ball} * V_{ball,initial} + m_{target} * V_{target,initial} = m_{ball} * V_{ball,final} + m_{target} * V_{target,final} \qquad \text{Eq. 1}$$

When the ball is thrown, it has a momentum equivalent to $m_{ball} * V_{ball,initial}$. The target initially has zero momentum since it is stationary. When the ball collides with the target, part of the momentum of the ball is transferred to the target, and the target will momentarily experience acceleration, velocity, and some finite, though small, displacement before dissipating the momentum and returning to a rest state. The other portion of momentum is retained by the ball as it bounces off the target, due to the elastic nature of the collision. By measuring the acceleration imparted on the target, its velocity is computed through integration. Ideally, if the mass of the ball, the mass of the target, and the final velocity of the ball are known, then the problem could be solved analytically and the initial velocity of the baseball determined.

The analysis of the crash phenomenon is, however, actually quite complex. Some factors that must be taken into account and that complicate the analysis greatly, are the spring constant and damping coefficient of the target. The target will be displaced during impact because it is anchored to the frame by a thick rubber mat. This action effectively causes the system to have a certain amount of spring. Also, though the mat is very dense, it will deform somewhat during impact and will act as shock absorber. In addition, the ball itself also has a spring constant and damping coefficient associated with it, since it bounces off the target and, though not noticeable by the naked eye, will deform during the impact. Finally, and of even greater significance, the mass of the ball, the mass of the target, and the final velocity of the ball are neither known nor measured. So how can the system work?

The Baseball Pitch Speedometer works by exploiting the fact that the final velocity of the target will be, according to Eq. 1, linearly proportional to the initial velocity of the thrown ball. Therefore, by measuring the acceleration response of the system to various ball velocities, which can be measured by independent means such as a radar gun, the system could be calibrated and a linear model developed. To facilitate the characterization and calibration of the system, a pitching machine was used to ensure that the incident ball speed would be repeatable. It also eliminated potential error caused by the variability of location of impact on the target that would inevitably result from several manual throws. Figure 3 shows a linear regression plot of the response of the system as a function of incident velocity. As is indicated by the plot, just a simple constant of proportionality could be used to correlate the measured acceleration response to the incident velocity of the ball, with fairly accurate results.

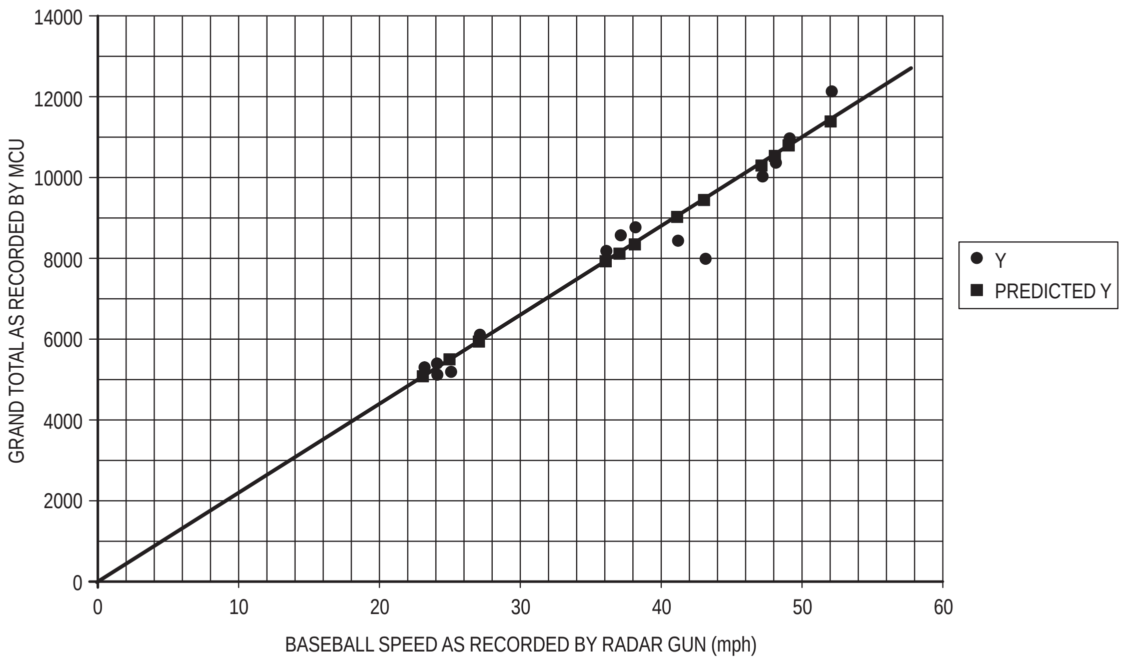

Figure 3. Baseball Pitch Speedometer Characterization Data

IMPLEMENTATION — HARDWARE

The target mat of the Baseball Pitch Speedometer has an area of approximately 9 ft^2 (3 by 3). Even though the rubber material used to construct the target is quite dense and heavy, the transmission of an impact is very poor if the ball strikes the target too far from the sensor. Therefore, to cover such a relatively large area it is necessary to use at least four devices; one centered in each quadrant of the square target. In addition, a shock resistant plate about a quarter inch thick is mounted behind the rubber mat. These features help make the response of the system more uniform and reduce errors that result from the variability of where the ball strikes the target.

The bulk of the circuit hardware is contained in a display box mounted on the top front side of the cage. Since the accelerometers are physically located far away from the mother board (about 10 feet of wiring), op–amps were used to buffer the accelerometers' output and drive the transmission line. The four accelerometer signals are then simultaneously fed into a comparator network and four of the ADC inputs on an MC68HC11 microcontroller. The MC68HC11 was selected because it has the capability of converting four A/D channels in one conversion sequence and operates at a higher clock speed. These two features reduce the overall time interval between digitizations of the analog signal (that result from the minimum required time for proper A/D conversion and from software latency) thus allowing a more accurate representation of the acceleration waveform to be captured. The comparator network serves a similar purpose by eliminating the additional software algorithm and execution time that would be required to continually monitor the outputs of all four accelerometers and determine whether impact has occurred or not. By minimizing this delay (some is still present since the output signal must exceed a threshold, and a finite amount of time is required for this) more of the initial and more significant part of the signal is captured.

The comparator network employs four LM311's configured to provide an OR function, and a single output is fed into an input capture pin on the MCU. A potentiometer and filter capacitor are used to provide a stable reference threshold voltage to the comparator network. The threshold voltage is set as close as possible to the accelerometers' offset voltage to minimize the delay between ball impact and the triggering of the conversion sequence, but enough clearance must be provided to prevent false triggering due to noise. Because the comparator network is wired such that any one of the accelerometer outputs can trigger it, the threshold voltage must be higher than the highest accelerometer offset voltage. Hysteresis is not necessary for the comparator network, because once the MCU goes into the conversion sequence it ignores the input capture pin.

The system is powered using a commercially available 9 V supply. A Motorola MC7805 voltage regulator is used to provide a steady 5 Volt supply for the operation of the MCU, the accelerometers, the comparator network, and the op–amp buffers. The 9 V supply is directly connected to the common anode 8–segment LED displays. Each segment can draw as much as 30 mA of current. Therefore, to ensure proper operation, the power supply selected to build this circuit should be capable of supplying at least 600 mA. Ports B and C on the MCU are used to drive the LED displays. Each port output pin is connected via a resistor to the base of a BJT, which has the emitter tied to ground. A current limiting resistor is connected between the collector of each BJT and the cathode of the corresponding segment on the display. To minimize the amount of board space consumed by the output driving circuitry, MPQ3904s (quad packaged 2N3904s) were selected instead of the standard discrete 2N3904s. The zero bit on Port C is connected to a combination BJT and MOSFET circuit that drives the “Your Speed” and “Best Speed” LED's. The circuit is wired so that the LED's toggle, and only one can be ON at a time.

Figure 4 shows a schematic of the circuit used. Part (a) shows the accelerometers, the op–amps used to buffer the outputs and drive the transmission lines, the comparator network and the potentiometer used to set the detection threshold. Part (b) shows the MCU, with its minimal required supporting circuitry. Part (c) shows the voltage regulator, a mapping of the cathodes to the corresponding segments on the LED displays, the BJT switch circuitry used to drive the seven segment display LEDs (although not shown on the schematic, this circuit block is actually repeated 15 times), and finally, the circuitry used to drive the “Your Speed”/“Best Speed” LEDs.

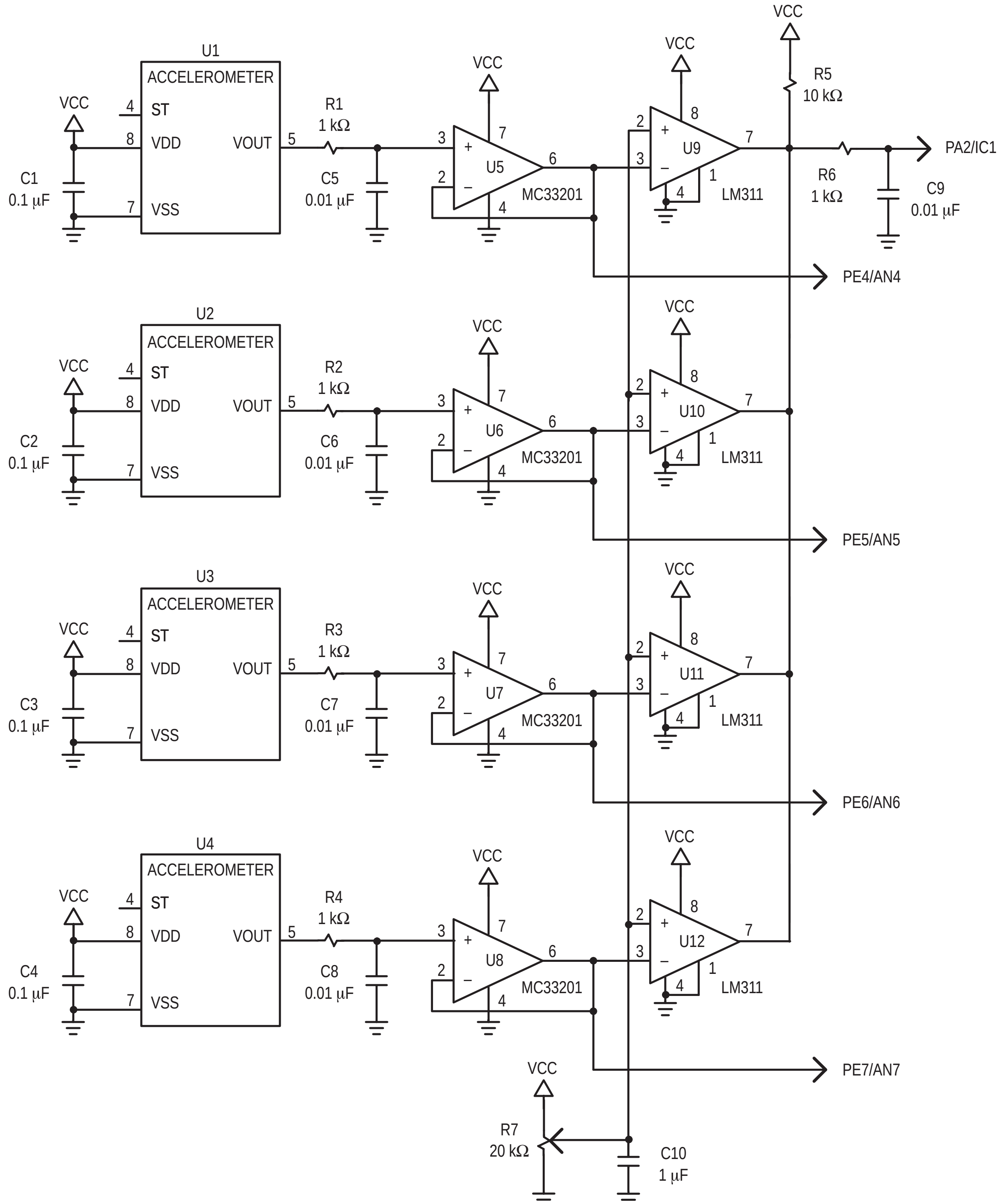

Figure 4a. Accelerometers, Buffer Op–Amps, and Comparator Network

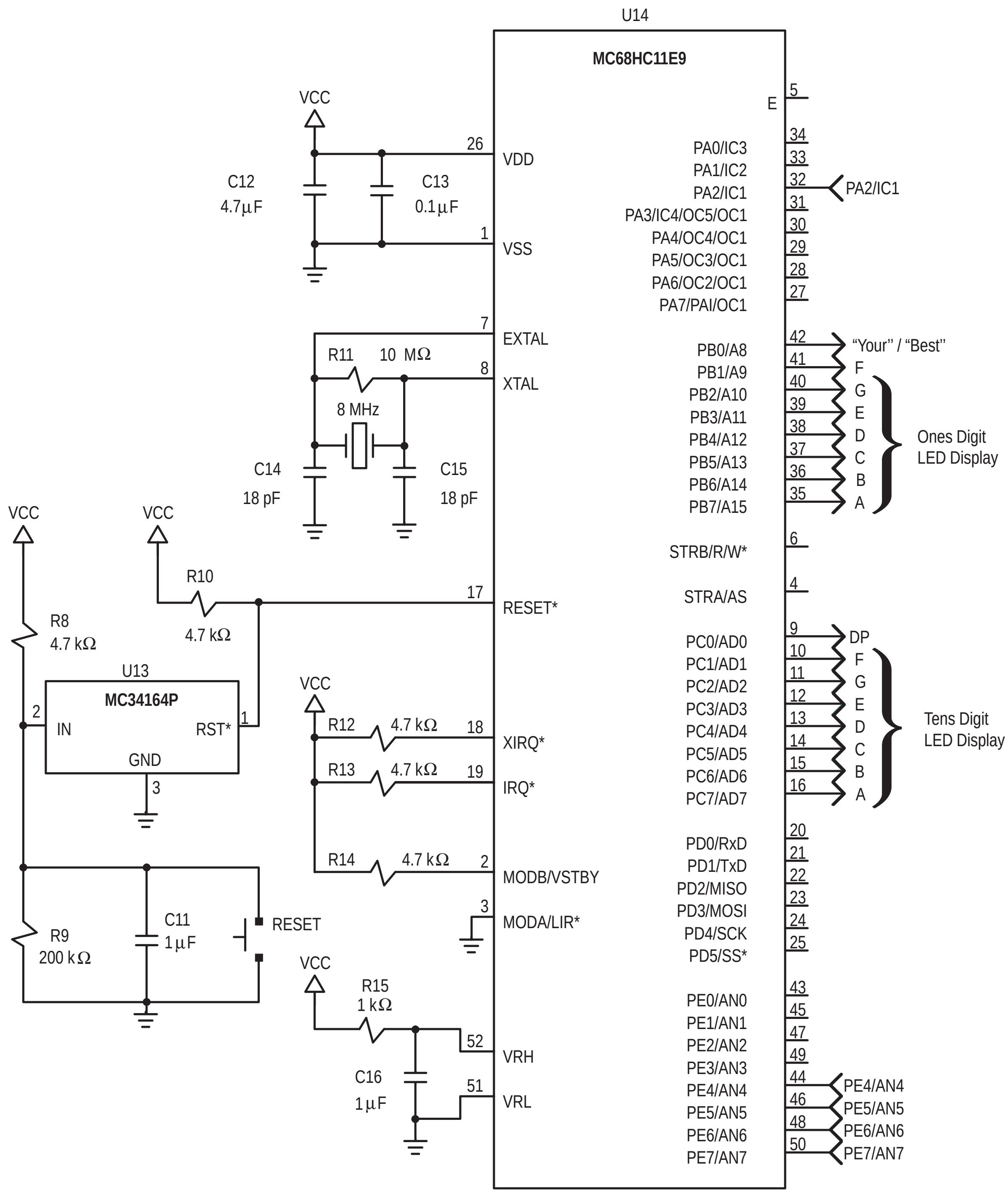

Figure 4b. MC68HC11E9 MCU with Supporting Circuitry

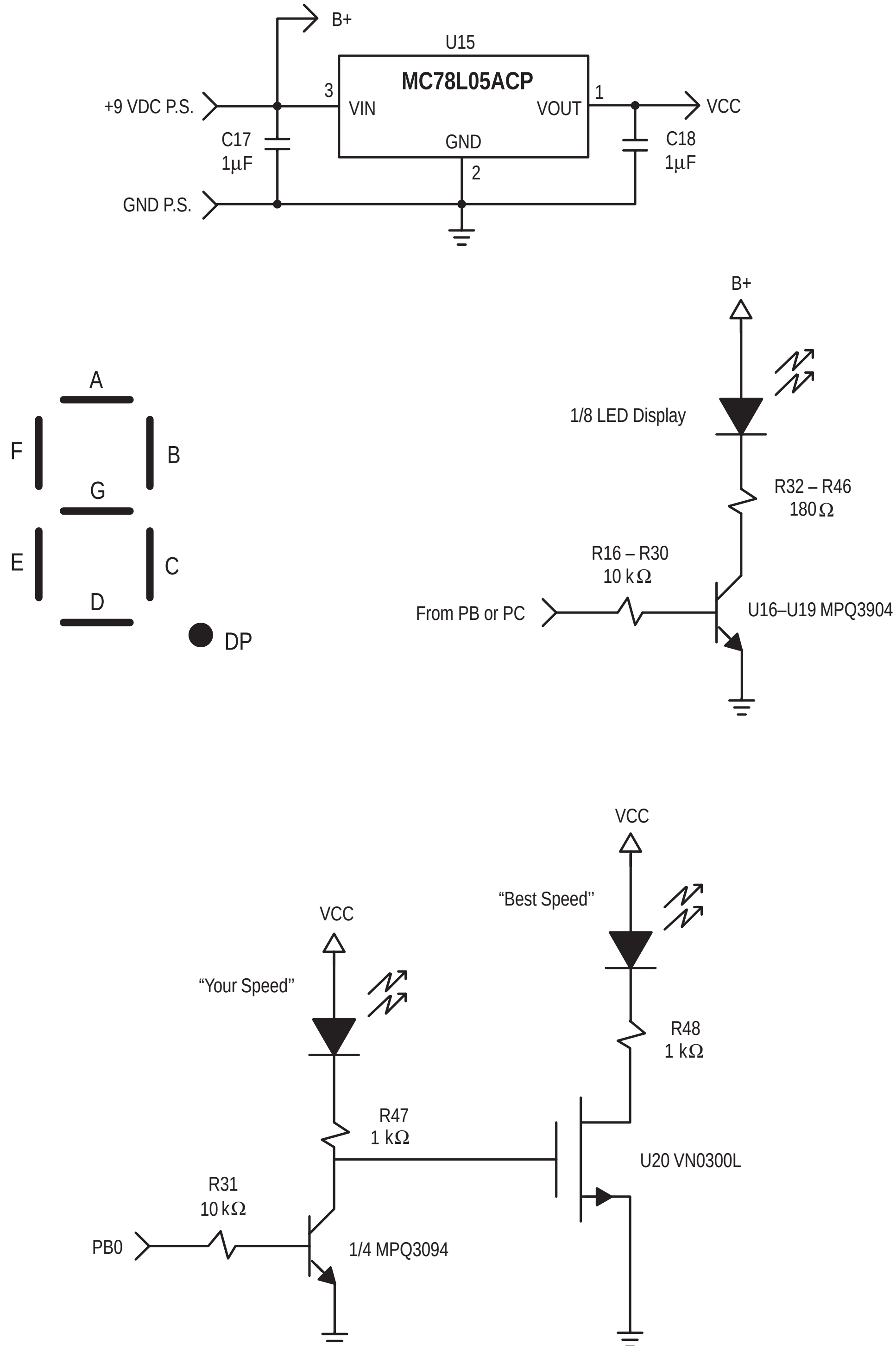

Figure 4c. Voltage Regulator, LED Segment Mapping, and LED Driving Circuitry

IMPLEMENTATION — SOFTWARE

The operation of the Baseball Pitch Speedometer is very simple. Upon power on reset, the output LEDs are initialized to display “00” and “Best Speed.” The analog to digital converter is turned on and the offset voltages of the accelerometers are measured and stored. Finally, all the variables are initialized and the MCU goes into a dormant state, where it will wait for a negative edge input capture pulse to trigger it to begin processing the crash signal.

Once the input capture flag is set, the MCU will immediately begin the analog to digital conversion sequence. As it digitizes the crash signature, it will calculate the absolute difference between the current value and the stored offset voltage value. It will integrate by summing up all the differences. Figure 2 shows a typical crash signature of the Baseball Pitch Speedometer. As illustrated, starting at the point of impact (A), the acceleration will initially ramp up, reaching a maximum, then decrease as the target is displaced. Because the target is constrained to the frame structure, the acceleration will continue to decrease until it reaches a minimum (point B), which correspond to the travel stop of the target. It is difficult to determine exactly when point B will occur, because the amplitude and duration of the initial acceleration pulse will vary with ball speed. Therefore, the capture window duration is set so that it will encompass most typical crash signatures, while rejecting most of the secondary ripples that result as the energy is dissipated by the system.

After integrating the four signals, the results are added together to produce an overall sum. This procedure averages out the individual responses and reduces measurement error due to the variability of where the ball lands on the target. The MCU then divides the grand sum by an empirically predetermined constant of proportionality. The result will then go through a binary to BCD conversion algorithm. A look–up table is used to match the BCD numbers to their corresponding 7–segment display codes. The calculated speed is displayed on the two digit 8–segment displays (one segment corresponds to the decimal point), and the “Your Speed” LED is turned on while the “Best Speed” LED is turned off. After a duration of approximately five seconds, the LEDs are toggled and stored best speed is redisplayed. The five second delay is used to provide enough time for the user to check his/her speed and also to allow the target to return to a rest state. The system is now ready for another pitch. A complete listing of the software is presented in the Appendix.

CONCLUSION

The Baseball Pitch Speedometer works fairly well, with an accuracy of +/– 5 mph. The dynamic range of the system is also worthy of note, measuring speeds from less than 10 mph up to well above the 70 mph range. One key point to emphasize, is that the system is empirically calibrated, and so to maintain good accuracy the system should only be used with balls of mass equal to those used during calibration.

Although intended mainly for training and recreational purposes, the Baseball Pitch Speedometer demonstrates a very important concept concerning the use of accelerometers. Accelerometers can be used not only to detect that an event such as impact or motion has occurred, but more importantly they measure the intensity of such events. They can be used to discern between different crash levels and durations. This is very useful in applications where it is desired to have the system respond in accord with the magnitude of the input being monitored. An example application would be a smart air bag system, where the speed at which the bag inflates is proportional to the severity of the crash. The deployment rate of the airbag would be controlled so that it does not throw the occupant back against the seat, thus minimizing the possibility of injury to the occupant. Another application where this concept may be utilized is in car alarms, where the response may range from an increased state of readiness and monitoring, to a full alarm sequence depending on the intensity of the shock sensed by the accelerometer. This could be used to prevent unnecessary firing of the alarm in the event that an animal or person were to inadvertently bump or brush against the automobile.

APPENDIX — ASSEMBLY CODE LISTING FOR BASEBALL PITCH SPEEDOMETER

```
* Baseball Pitch Speedometer – Rev. 1.0
*
* Program waits for detection of impact via the input capture pin and then reads four A/D channels.
* The area under the Acceleration vs. Time curve is found by subtracting the steady state offsets
* from the digitized readings and summing the results.  The sum is then divided by an empirically
* determined constant of proportionality, and the speed of the ball is displayed.
*
* Written by Carlos Miranda
* Systems and Applications
* Sensor Products Division
* Motorola Semiconductor Products Sector
* May 6, 1997
*
*
*****************************************************************************************************
*       Although the information contained herein, as well as any information provided relative      *
*       thereto, has been carefully reviewed and is believed accurate, Motorola assumes no           *
*       liability arising out of its application or use, neither does it convey any license under    *
*       its patent rights nor the rights of others.                                                  *
*****************************************************************************************************
* These equates assign memory addresses to variables.
EEPROM          EQU             $B600
CODEBGN         EQU             $B60D
REGOFF          EQU             $1000    ;Offset to access registers beyond direct addressing range.
PORTC           EQU             $03
PORTB           EQU             $04
DDRC            EQU             $07
TCTL2           EQU             $21
TFLG1           EQU             $23
ADCTL           EQU             $30
ADR1            EQU             $31
ADR2            EQU             $32
ADR3            EQU             $33
ADR4            EQU             $34
OPTION          EQU             $39
STACK           EQU             $01FF    ;Starting address for the Stack Pointer.
RAM             EQU             $0000
* These equates assign specific masks to variables to facilitate bit setting, clearing, etc.
ADPU            EQU             $80      ;Power up the analog to digital converter circuitry.
CSEL            EQU             $40      ;Select the internal system clock.
CCF             EQU             $80      ;Conversion complete flag.
IC1F            EQU             $04      ;Input Capture 1 flag.
IC1FLE          EQU             $20      ;Configure Input Capture 1 to detect falling edges only.
IC1FCLR         EQU             $FB      ;Clear the Input Capture 1 flag.
CHNLS47         EQU             $14      ;Select channels 4 through 7 with MULT option ON.
SAMPLES         EQU             $0200    ;Number of A/D samples taken.
OC1F            EQU             $80      ;Output Compare 1 flag.
OC1FCLR         EQU             $7F      ;Clear the Output Compare flag.
CURDLY          EQU             $0098    ;Timer cycles to create delay for displaying ”Your Speed.”
RAMBYTS         EQU             $19      ;Number of RAM variables to clear during initialization.
ALLONES         EQU             $FF
YOURSPD         EQU             $01
PRPFCTR         EQU             $00AD    ;This constant of proportionality was empirically determined.
* Variables used for computation.
                ORG             RAM
OFFSET1         RMB             1        ;One for each accelerometer.
OFFSET2         RMB             1
OFFSET3         RMB             1
OFFSET4         RMB             1
SUM1            RMB             2        ;Area under the acceleration vs. time curve.
SUM2            RMB             2
SUM3            RMB             2
SUM4            RMB             2
GRNDSUM         RMB             2
COUNT           RMB             2
CURBIN          RMB             1
TEMPBIN         RMB             1
BCD             RMB             2
CURDSPL         RMB             2
MAXBIN          RMB             1
MAXDSPL         RMB             2
* LED seven segment display patterns table.
                ORG             EEPROM
                JMP             START
SEVSEG          FCB             %11111010
                FCB             %01100000
                FCB             %11011100
                FCB             %11110100
                FCB             %01100110
                FCB             %10110110
                FCB             %10111110
                FCB             %11100000
                FCB             %11111110
                FCB             %11100110
```

```
* This is the main program loop.
                ORG             CODEBGN
START           LDS             #STACK
                LDX             #REGOFF
                JSR             LEDINIT
                JSR             ADCINIT
                JSR             VARINIT
MAIN            JSR             CAPTURE
                JSR             COMPUTE
                JSR             BINTBCD
                JSR             OUTPUT
                BRA             MAIN
* This subroutine initializes ports B & C, and the LED display.
LEDINIT         PSHX
                PSHA
                LDX             #REGOFF
                BSET            DDRC,X,ALLONES          ;Configure port C as an output.
                LDAA            SEVSEG
                STAA            PORTB,X
                STAA            PORTC,X
                PULA
                PULX
                RTS
* This subroutine initializes the analog to digital converter.
ADCINIT         PSHX
                PSHA
                LDX             #REGOFF
                BSET            OPTION,X,ADPU           ;Turn on A/D converter via ADPU bit.
                BCLR            OPTION,X,CSEL           ;Select system e clock via CSEL bit.
                CLRA
DELAY           INCA
                BNE             DELAY
                PULA
                PULX
                RTS
* This subroutine clears all the memory variables.
VARINIT         PSHX
                LDX             #$0000
CLRVAR          CLR             OFFSET1,X
                INX
                CPX             #RAMBYTS                ;Number of RMB bytes.
                BLO             CLRVAR
DONECLR         LDX             #REGOFF
                LDAA            #CHNLS47                ;Measure the offset.
                STAA            ADCTL,X
OFSWAIT         BRCLR           ADCTL,X,CCF,OFSWAIT
                LDD             ADR1,X
                STD             OFFSET1
                LDD             ADR3,X
                STD             OFFSET3
                PULX
                RTS
* This subroutine waits for impact and computes the area under the curve.
CAPTURE         PSHX
                PSHA
                PSHB
                LDX             #REGOFF
                BSET            TCTL2,X,IC1FLE          ;Set IC1 to detect falling edge only.
                BCLR            TFLG1,X,IC1FCLR
MONITOR         BRCLR           TFLG1,X,IC1F,MONITOR
ADCREAD         LDAA            #CHNLS47                ;Select channels 4 – 7 for conversion.
                STAA            ADCTL,X
ADCWAIT         BRCLR           ADCTL,X,CCF,ADCWAIT
CALDLT1         LDAB            ADR1,X
                SUBB            OFFSET1
                BPL             ADDSUM1
                COMB
                INCB
ADDSUM1         CLRA
                ADDD            SUM1
                STD             SUM1
CALDLT2         LDAB            ADR2,X
                SUBB            OFFSET2
                BPL             ADDSUM2
                COMB
                INCB
ADDSUM2         CLRA
                ADDD            SUM2
                STD             SUM2
CALDLT3         LDAB            ADR3,X
                SUBB            OFFSET3
                BPL             ADDSUM3
                COMB
                INCB
```

```
ADDSUM3         CLRA
                ADDD            SUM3
                STD             SUM3
CALDLT4         LDAB            ADR4,X
                SUBB            OFFSET4
                BPL             ADDSUM4
                COMB
                INCB
ADDSUM4         CLRA
                ADDD            SUM4
                STD             SUM4
                LDD             COUNT
                ADDD            #$0001
                STD             COUNT
                CPD             #SAMPLES
                BLO             ADCREAD
                PULB
                PULA
                PULX
                RTS
* This subroutine computes the ball speed by dividing the overall sum by a constant.
COMPUTE         PSHX
                PSHA
                PSHB
                LDD             SUM1
                ADDD            SUM2
                ADDD            SUM3
                ADDD            SUM4
                STD             GRNDSUM
                LDX             #PRPFCTR
                IDIV
                XGDX
                STAB            CURBIN
                PULB
                PULA
                PULX
                RTS
* This subroutine converts from binary to BCD. (Limited to number up to 99 decimal.)
BINTBCD         PSHX
                PSHA
                PSHB
                LDX             #$0000
                LDAA            CURBIN
                STAA            TEMPBIN
                CLRA
                CLRB
BINSHFT         LSL             TEMPBIN
                ROLB
                LSLA
                CMPB            #$10
                BLO             CHKDONE
                INCA
                ANDB            #$0F
CHKDONE         INX
                CPX             #$0008
                BEQ             RAILAT9
CHKFIVE         CMPB            #$05
                BLO             BINSHFT
                ADDB            #$03
                BRA             BINSHFT
RAILAT9         CMPA            #$09            ;Force the display to “99” if speed > 100 mph.
                BLS             DONE
                LDD             #$0909
DONE            STD             BCD
                LDX             #SEVSEG         ;This part finds the seven segment display codes.
                XGDX
                ADDB            BCD
                XGDX
                LDAA            $00,X
                STAA            CURDSPL
                LDX             #SEVSEG
                XGDX
                ADDB            BCD+1
                XGDX
                LDAA            $00,X
                STAA            CURDSPL+1
                PULB
                PULA
                PULX
                RTS
* This subroutine displays the current speed for 5 seconds & then displays the maximum.
OUTPUT          PSHX
                PSHA
                PSHB
```

```
                LDX         #REGOFF
                LDAA        CURBIN
                CMPA        MAXBIN
                BLS         OLDMAX
                STAA        MAXBIN
                LDD         CURDSPL
                STD         MAXDSPL
OLDMAX          LDD         CURDSPL
                STD         PORTC,X
                BSET        PORTB,X,YOURSPD          ;Toggle the ”YOUR”/“BEST” LEDs.
                LDD         #$0000
LEDWAIT         BCLR        TFLG1,X,OC1FCLR          ;Clear output compare 1 flag.
DSPLDLY         BRCLR       TFLG1,X,OC1F,DSPLDLY
                ADDD        #$0001
                CPD         #CURDLY                  ;Decimal 152. (152 * 33ms = 5.0 sec)
                BLO         LEDWAIT
                LDX         #$0000
RECLEAR         CLR         SUM1,X          ;Clear 12 RAM bytes beginning at address ”SUM1”.
                INX                         ;Clears SUM1 thru SUM4, GRNDSUM, and COUNT.
                CPX         #$000C
                BLO         RECLEAR
                LDX         #REGOFF
                LDD         MAXDSPL
                STD         PORTC,X         ;The ”YOUR”/“BEST” LEDS are automatically toggled.
                PULB
                PULA
                PULX
                RTS
```

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

Mfax is a trademark of Motorola, Inc.

How to reach us:
USA / EUROPE / Locations Not Listed: Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217. 1–303–675–2140 or 1–800–441–2447

JAPAN: Nippon Motorola Ltd.; SPD, Strategic Planning Office, 141,
4–32–1 Nishi–Gotanda, Shinagawa–ku, Tokyo, Japan. 81–3–5487–8488

Customer Focus Center: 1–800–521–6274

Mfax™: RMFAX0@email.sps.mot.com – TOUCHTONE 1–602–244–6609
Motorola Fax Back System – US & Canada ONLY 1–800–774–1848
– http://sps.motorola.com/mfax/

ASIA / PACIFIC: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

HOME PAGE: http://motorola.com/sps/

MOTOROLA