intel®

AP-427

APPLICATION NOTE

Using the 4th Entry Interrupt in ABS Designs

STEVE McINTYRE
AUTOMOTIVE APPLICATIONS ENGINEER

July 1989

Order Number: 270692-001

Information in this document is provided in connection with Intel products. Intel assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of Intel products except as provided in Intel's Terms and Conditions of Sale for such products.

Intel retains the right to make changes to these specifications at any time, without notice. Microcomputer Products may have minor variations to this specification known as errata.

*Other brands and names are the property of their respective owners.

†Since publication of documents referenced in this document, registration of the Pentium, OverDrive and iCOMP trademarks has been issued to Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

USING THE 4th ENTRY INTERRUPT IN ABS DESIGNS

CONTENTS

In four-wheel Anti-Lock Braking (ABS) systems, wheel speeds of 150 MPH are a common design consideration. With this high wheel speed comes the problem of high wheel speed detection. The faster the CPU can detect wheel speeds, the faster it can determine wheel lock-ups. At 150 MPH the wheel speed sensor is generating a sinusoidal waveform that is converted to a square wave. This square wave frequency is in the 0 Hz to 6000 Hz range. This means that the number of positive edges that the processor needs to detect is about 120 during each 5 ms processing loop. *The 5 ms processing loop time was calculated to be the minimum loop time with the maximum effect on ABS performance*.

This means that the processor must interrupt what it is doing, service that high speed input interrupt (save the time away for that wheel), and return the CPU to the previous state. At 150 MPH, this could be as much as 72% of its time in that interrupt service routine.

HIGH SPEED INPUTS

The High Speed Inputs on the 8096 family of product is perfect for detecting high speed input frequencies. In four-wheel ABS designs all four High Speed Inputs (HSI) are used. In the past the ABS software designer had to interrupt the processor every time a rising edge occurred on each wheel speed output. 120 interrupts in a 5 ms processing loop time. In worst case conditions, 72% of the processing loop was spent in the interrupt service routine.

NOTE:
Using a 90–100 tooth pulse wheel, that generates an output frequency 0 Hz–6000 Hz, and operating the 8X9XBH/JC at 12 MHz).

The 8X96JC device has been modified slightly to assist the ABS designer. The FIFO full bit (IOS0.6) has been changed in meaning. Instead of being set when the FIFO is almost full (6th FIFO entry, seventh event), the bit is set when the 4th FIFO entry (5th event) has occurred. *There is no longer a FIFO full bit since the 4th entry bit replaced it*. This means that when interrupts are used with the FIFO (IOC0.7), there is a holding register loaded interrupt (IOC0 bit 7 = "0") and a 4th entry interrupt (IOC0 bit 7 = "1").

The ABS software designer can take advantage of this new feature. Instead of interrupting on the first (or every event), he can now utilize the buffering of events in the FIFO, and interrupt on the 4th entry in the FIFO. Leaving 3 more slots in the FIFO for future event storage. NO LOST EVENTS WILL OCCUR.

Using this method saves over 8.8% in the processing loop time. Instead of 72% of the time spent in the interrupt service routine, only 63.2% is spent.

OTHER CONSIDERATIONS

At low wheel speeds the number of edges decreases, making it difficult to detect wheel speeds. Less information is available to the CPU if the 4th entry interrupt method is used instead of the 1st event interrupt method. The software interrupt routine for the 1st event interrupt (holding register full) and the 4th event interrupt is identical. Switching between the these two is a simple matter of writing to a control register bit (IOC0 bit 7).

When wheel speeds below 1000 Hz are detected the software could switch from the 4th entry method to the

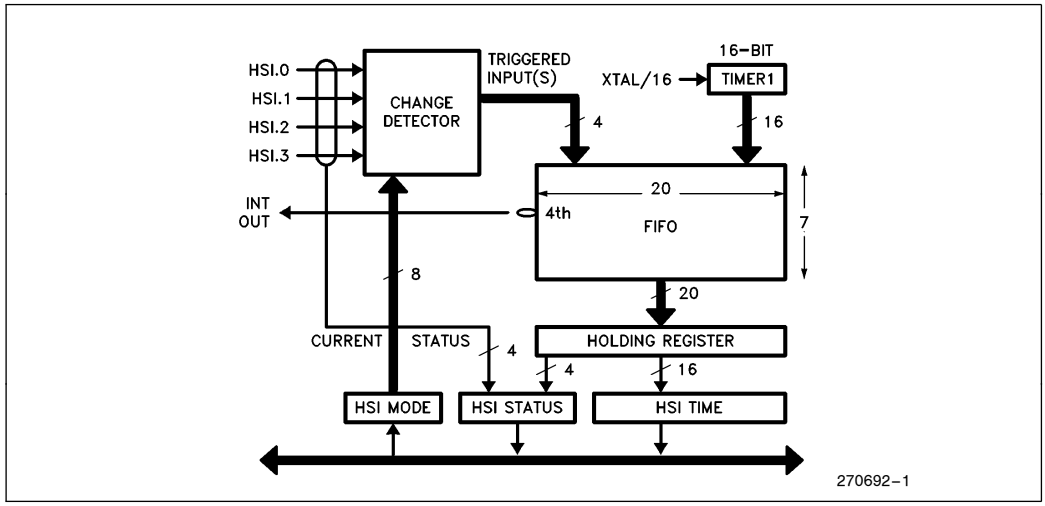

Figure 1. HSI with FIFO

1st entry method. This way more information is gathered quicker at the slow wheel speeds when processing loop time is less critical.

This unique feature of interrupt flexibility translates to the maximum CPU utilization in an application starving for raw CPU power. For these reasons, the 8X96JC is a perfect fit for four-wheel ABS designs.

CODED EXAMPLE

Listed at the end of this AP-NOTE is the acual service routine software used for the four-wheel speed inputs connected to the HSI pins 0–3.

Figure 2 shows a graphic representation of the high speed input pins (labeled Wheel Speed Outputs). Shown is a worst case input on the HSI pins, since one pass through the HSI interrupt service routine takes about 34.5 μs. This includes the interrupt response time, decoding or HSI bits, storage of ITIME or FTIME values, and incrementing pulse counters. If the signals on the high speed inputs are placed about 40 μs apart, the interrupt service routine would only have enough time to exit the routine just in time to enter it again, "back-to-back execution".

The top signal in Figure 2 is the IOPORT1 pin 5. The high time represents the time within the interrupt service routine, the low time is the time between service routines.

If the worst case input frequency of 6000 Hz is used, only 15.5 μs is left between processing the HSI interrupt service routines. Not much time for the processor to complete other tasks besides the HSI interrupt routine.

But, if the fourth entry interrupt is used on the same or slightly similar input signals, the outcome is greatly different.

In Figure 3 similar signals are shown. Instead, if 40 μs apart, the signals are closer together (about 3 μs apart). This shows that about 181.5 μs is spent in the interrupt service routine, and 150.5 μs is saved for processing other than interrupt service instructions. Yet no information is lost. Eight times less interrupts are handled because the entire FIFO and holding register is utilized.

Also notice that the processor will interrupt on the fifth event, not the fourth. When edges are detected by the

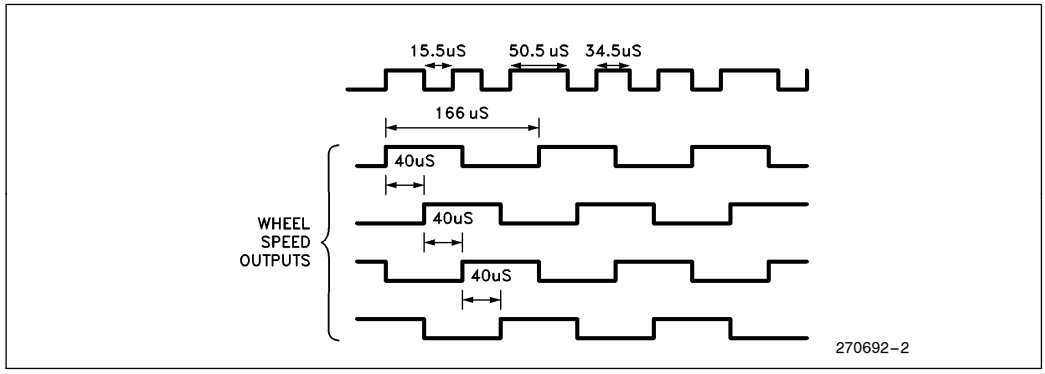

Figure 2. 1st Entry Interrupt Response Time

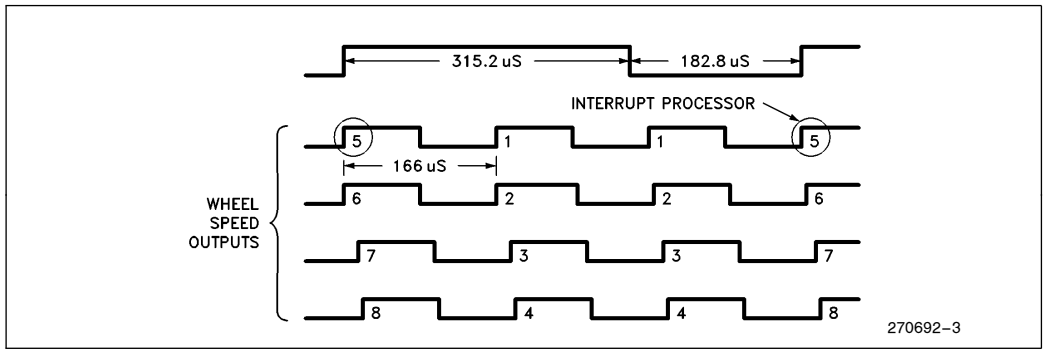

Figure 3. 4th Entry Interrupt Response Time

high speed input unit, the event and time is placed in the FIFO (see Figure 1). Within eight device states the event will be loaded into the holding register (provided there is room, if the holding register is full the device will not allow the transfer into the holding register).

Following this technique, the fourth event will be in the FIFO with the third and second, and the first event will be in the holding register. Upon detection of the fifth event (fourth entry in the FIFO, the device will recognize the event and interrupt the processor.

Three more slots in the FIFO are free for event storage. Since all four inputs are used in four-wheel ABS, the worst case input signal is where each input is at its maximum frequency (6000 Hz) and placement is where each are about 3 μs apart. (The HSI resolution is 2 μs. If the HSI unit sees another transition on another input within that 2 μs window, only one entry will be in the FIFO, but 2 event bits set. If the placement is just outside the 2 μs window, two events must be stored. Hence, worst case.)

ABOUT THE ROUTINE

The HSI interrupt routine supports a frequency detection approach. The number of positive edges detected is noted as well as the beginning and ending time that the first and last positive edges occurred.

Through these (ITIME, FTIME, and TC_x), the incoming frequency can be calculated relatively easily:

$$f = \frac{1}{(FTIME - ITIME) * TC * tick}$$

TC represents the number of positive edges within the ITIME and FTIME limits and *tick* stands for the value of one increment of the timer. (For example: at 12 MHz, one increment of the TIMER1 happens every 2 μs. So, *tick* would equal 2 μs.

Of course, in an ABS application there are other considerations such as wheel rolling radius, types of pavement, and relationships to the other wheels. These things are performed on a periodic basis within the software timer interrupt routine. After the wheel speed information is gathered.

CONCLUDING COMMENTS

Although the saving in the wheel speed detection of a four-wheel ABS system is an 8.8% value, this number came from some pretty aggressive wheel speed frequencies.

The 0 Hz–6000 Hz frequency would typically be around 0 Hz–3000 Hz using a 40–50 tooth pulse wheel. This makes the CPU savings a bit better, because of the interrupt overhead.

With the 3000 Hz number, the savings is over 16.8%. This translates to added functions within the system.

With the added memory of the 8X97JC device and the added CPU savings, such things as Traction control or suspension control could be added easily.

REFERENCES

Bosch Automotive Handbook; 2nd Edition; "Suspensions", "Braking Equipment"; SAE Isbn 0-89883-518-6.

Reinecke, E.; "An Anti-Lock System with Extended Safety and Control System Functions"; Int. J. of Vehicle Design, Vol. 6, Nos. 4/5, pp. 561–566; 1985.

Petersen, E. and Quicke, K.; "New Anti-Lock Systems for Commercial Vehicles Realized with Single-Chip Microcomputers", Wabco Westinghouse GmbH, Hanover; IMechE Paper #C205/81.

Maisch, W. and Schramm, H.; "Further Development of the Anti-Lock Braking Systems for Commercial Vehicles with Compressed Air Brakes"; IMechE Paper #C191/85.

Klein, Hans-Christof; "Anti-Lock Brake Systems for Passenger Cars, State of the Art 1985"; SAE Paper #865139.

Gerstenmeier, Jurgen; "ABS Electronics, Current Status and Future Prospects"; IMechE Paper #C239/85; "Electronic Control Unit for Passenger Car Anti-Skid (ABS)"; IMechE Paper #C186/81; "Traction Control (ASR)—An Extension of the Anti-Lock Braking Systems (ABS)"; SAE Paper #861033.

Hussain, Seyd; "The Application of High Speed Integrated Digital Microcontrollers in Modern Anti-Lock Systems"; SAE Paper 1985.

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 1
DOS 3.20 (038-N) MCS-96 MACRO ASSEMBLER, V1.1
SOURCE FILE: WHEELSPD.A96
OBJECT FILE: WHEELSPD.OBJ
CONTROLS SPECIFIED IN INVOCATION COMMAND: DB
ERR LOC OBJECT              LINE        SOURCE STATEMENT
                             1  $TITLE ("Program for MCS96-HSI-WHEEL SPEED
                                    Calculations.")
                             2  ;
                             3  ;
                             4  ;      TITLE: WHEELSPD.A96
                             5  ;      AUTHOR: Steve McIntyre
                             6  ;      PROJECT:       ABS/ASR
                             7  ;
                             8  ;      LANGUAGE:        iNTEL ASM96 assembler
                             9  ;
                            10  ;      MODIFIED BY:   WHEN:   CHANGES MADE:
                            11  ;      Steve McIntyre  12.7.88 Initial program
                                ;      creation.
                            12  ;
                            13  ;
                            14  WHEEL_SPEED    MODULE MAIN
                            15  ;
                            16  ;      The following program uses the 8X96JC 4th
                                ;      entry interrupt to manipulate the HSI wheel
                            17  ;      speed indicators. The program is used to
                            18  ;      measure the interrupt processing time for
                            19  ;      four-wheel ABS with 100 teeth pulse wheel
                                ;      (0-6000 Hz) signals on all 4 HSI inputs.
                            20  ;
                            21  $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 2
ERR LOC OBJECT        LINE    SOURCE STATEMENT
               =1     22     $include (8096BH.inc)
               =1     23  ;***************************************
               =1     24  ;
               =1     25  ; 8096.INC-DEFINITION OF SYMBOLIC NAMES FOR
               =1     26  ;          THE I/O REGISTERS OF THE 8096
                          ;          (C) INTEL CORPORATION 1983
               =1     27  ;***************************************
               =1     28  ;
    0000       =1     29  R0            EQU  00H:WORD   ; R  ZERO REGISTER
    0002       =1     30  AD_COMMAND    EQU  02H:BYTE   ;  W
    0002       =1     31  AD_RESULT     EQU  02H:WORD   ; R
    0003       =1     32  HSI_MODE      EQU  03H:BYTE   ;  W
    0004       =1     33  HSO_TIME      EQU  04H:WORD   ;  W
    0004       =1     34  HSI_TIME      EQU  04H:WORD   ; R
    0006       =1     35  HSO_COMMAND   EQU  06H:BYTE   ;  W
    0006       =1     36  HSI_STATUS    EQU  06H:BYTE   ; R
    0007       =1     37  SBUF          EQU  07H:BYTE   ; R/W
    0008       =1     38  INT_MASK      EQU  08H:BYTE   ; R/W
    0009       =1     39  INT_PENDING   EQU  09H:BYTE   ; R/W
    000A       =1     40  WATCHDOG      EQU  0AH:BYTE   ;  W WATCHDOG TIMER
    000A       =1     41  TIMER1        EQU  0AH:WORD   ; R
    000C       =1     42  TIMER2        EQU  0CH:WORD   ; R
    000E       =1     43  BAUD_RATE     EQU  0EH:BYTE   ;  W
    000E       =1     44  IOPORT0       EQU  0EH:BYTE   ; R
    000F       =1     45  IOPORT1       EQU  0FH:BYTE   ; R/W
    0010       =1     46  IOPORT2       EQU  10H:BYTE   ; R/W
    0011       =1     47  SP_CON        EQU  11H:BYTE   ;  W
    0011       =1     48  SP_STAT       EQU  11H:BYTE   ; R
    0015       =1     49  IOC0          EQU  15H:BYTE   ;  W
    0015       =1     50  IOS0          EQU  15H:BYTE   ; R
    0016       =1     51  IOC1          EQU  16H:BYTE   ;  W
    0016       =1     52  IOS1          EQU  16H:BYTE   ; R
    0017       =1     53  PWM_CONTROL   EQU  17H:BYTE   ;  W
    0017       =1     54  IOS2          EQU  17H:BYTE   ; R
    0018       =1     55  SP            EQU  18H:WORD   ; R/W
    1FFE       =1     56  IOPORT3       EQU  1ffeH:word ; R/W
    1FFF       =1     57  IOPORT4       EQU  1fffH:byte
               =1     58  $EJECT
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 3
ERR LOC OBJECT        LINE    SOURCE STATEMENT
                        59    ; ******************
                        60    ; ** program equates **
                        61    ; ******************
    0100                62    STK     EQU      100H
    0013                63    INT_MASK1        EQU     13H
    0012                64    INT_PEND1        EQU     12H
                        65    ;
    0007                66    HOLDING_REG_FULL         EQU      7
    0080                67    HSI_HOLDING_FULL         EQU      10000000B
    0040                68    HSI_FIFO_FULL            EQU      01000000B
    0020                69    HSI_FIFO_ROUTINE         EQU      00100000B
    0055                70    ENABLE_ALL_HSI           EQU      01010101B
    0080                71    ENABLE_4TH_ENTRY         EQU      10000000B
    0055                72    HSI_POS_EDGES            EQU      01010101B
    0004                73    ALLOW_4TH_ENTRY_INT      EQU      00000100B
                        74    ;
    0001                75    MASK_SOFTWARE_TIMER_0    EQU      00000001B
    0020                76    ALLOW_SOFTWARE_TIMER_INT          EQU      00100000B
    000F                77    ALL_NEW_CALC     EQU     00001111B
    09C4                78    SW_TIMER_5MS     EQU     2500
    0018                79    SW_TIMER_0_CMD   EQU     18H
                        80    ;
    0000                81    CH_0    EQU      0
    0002                82    CH_1    EQU      2
    0004                83    CH_2    EQU      4
    0006                84    CH_3    EQU      6
    0001                85    MASK_CH_0        EQU     1
    0002                86    MASK_CH_1        EQU     2
    0004                87    MASK_CH_2        EQU     4
    0008                88    MASK_CH_3        EQU     8
                        89    ;
    61A8                90    MPH1    EQU      25000   ; This is a magical number
                                                       ; 3000 Hz = 150 MHz
    0000                91    BIT_0   EQU      0
    0001                92    BIT_1   EQU      1
    0002                93    BIT_2   EQU      2
    0003                94    BIT_3   EQU      3
    0004                95    BIT_4   EQU      4
    0005                96    BIT_5   EQU      5
    0006                97    BIT_6   EQU      6
    0007                98    BIT_7   EQU      7
    0010                99    MASK_0  EQU      10H
    0020               100    MASK_1  EQU      20H
    0040               101    MASK_2  EQU      40H
    0080               102    MASK_3  EQU      80H
                       103    ;
    0080               104    MAIN_LOOP_CONTROL        EQU      80H
                       105    $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 4
ERR LOC OBJECT        LINE    SOURCE STATEMENT
              106     ;
              107     ;       General Purpose RAM
              108     ;
    0080      109               rseg  at  80H
    0080      110     IOC0_SAVE:     DSB    1     ; save IOC0 status
    0081      111     IOS1_SAVE:     DSB    1     ; save IOS1 status
    0082      112     HSI_EVENT_STATUS:     DSB  1     ; for HSI_status reads
    0083      113     NEW_CALC_FLAG:  DSB   1          ; flag to determine
                                                       ; if Itime/Ftime to
                                                       ; be loaded
    0084      114     TC_0:          DSB    1     ; number of edges for HSI0
    0085      115     TC_1:          DSB    1     ; number of edges for HSI1
    0086      116     TC_2:          DSB    1     ; number of edges for HSI2
    0087      117     TC_3:          DSB    1     ; number of edges for HSI3
              118     ;
    0088      119     EVENT_TIME:    DSW    1     ; for HSI_time reads
    008A      120     ITIME_0:       DSW    1     ;
    008C      121     ITIME_1:       DSW    1     ; Initial time for
                                                  ; all four HSI's
    008E      122     ITIME_2:       DSW    1     ;
    0090      123     ITIME_3:       DSW    1     ;
              124     ;
    0092      125     FTIME_0:       DSW    1     ;
    0094      126     FTIME_1:       DSW    1     ;
    0096      127     FTIME_2:       DSW    1     ; Final time for
                                                  ; all four HSI's
    0098      128     FTIME_3:       DSW    1     ;
              129     ;
    009A      130     MPH_0:         DSW    1
    009C      131     MPH_1:         DSW    1
    009E      132     MPH_2:         DSW    1
    00A0      133     MPH_3:         DSW    1
              134     ;
    00A2      135     SW_TIMER_0:    DSW    1
              136     ;
    00A4      137     TEMP:          DSL    1
    00A8      138     TEMP_0:        DSB    1
    00A9      139     TEMP_1:        DSB    1
    00AA      140     TEMP_2:        DSB    1
    00AB      141     TEMP_3:        DSB    1
              142     ;
              143     $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 5
ERR LOC OBJECT           LINE          SOURCE STATEMENT
                  144  ;
                  145  ;     Interrupt Vectors
                  146  ;
    2004          147         cseg    at     2004H
    2004 D620     148         DCW       MSI_SERVICE_ROUTINE
    200A          149         cseg    at     200AH
    200A 5221     150         DCW         SOFTWARE_TIMER_ISR
                  151  $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 6
ERR LOC OBJECT       LINE    SOURCE STATEMENT
                     152 ; ******************
                     153 ; ** MAIN ROUTINE **
                     154 ; ******************
    2080             155     cseg   at         2080H
    2080             156 INITIAL_START:
    2080 FA          157     DI                ; Disable Interrupts
    2081 A1000118    158     LD     SP,#STK; Initialize Stack pointer
                     159
    2085             160 CLEAR_RAM:
    2085 C90000      161     PUSH   #0         ; Clear RAM from SP to 70H
    2088 89700018    162     CMP    SP, #70H
    208C D9F7        163     JH     CLEAR_RAM
                     164
    208E A1000118    165     LD     SP,#STK    ; Reinitialize stack pointer
    2092 B0000F      166     LDB    IOPORT1, R0; clear port 1
    2095 1115        167     CLRB   IOC0       ; Disable all HSI's
    2097 1116        168     CLRB   IOC1       ; Disable and select 1st entry
    2099 1109        169     CLRB   INT_PENDING; Clear all previous
                                                ;  interrupts
    209B 1108        170     CLRB   INT_MASK
    209D 71AA80      171     ANDB   IOC0_SAVE, #NOT ENABLE_ALL_HSI
    20A0 C41580      172     STB    IOC0_SAVE, IOC0 ; Disable HSI's
                     173
    20A3 281F        174     SCALL  EMPTY_FIFO  ; Clear FIFO
    20A5 A00AA2      175     LD     SW_TIMER_0, TIMER1  ; Initialize
                                                ; SWT =timer1
    20A8 28E0        176     SCALL  RESTART_SOFTWARE_TIMER_0
                                                ; Initialize SW_INIT
                     177
    20AA 910408      178     ORB    INT_MASK, #ALLOW_4TH_ENTRY_INT
    20AD 912008      179     ORB    INT_MASK, #ALLOW_SOFTWARE_TIMER_INT
                                                ;Init int_mask
    20B0 B15503      180     LDB    HSI_MODE, #HSI_POS_EDGES  ; Select only
                                                ; positive edges
                     181
                     182 ; New_calc_flag is used for determining when
                         ; SWT interrupt has expired.
                     183
    20B3 B10F83      184     LDB    NEW_CALC_FLAG, #ALL_NEW_CALC
    20B6 11A8        185     CLRB   TEMP_0
    20B8 918016      186     ORB    IOC1, #ENABLE_4th_ENTRY;
                     187
    20BB FB          188     EI     ; Enable Interrupt
                     189
    20BC 915580      190     ORB    IOC0_SAVE, #ENABLE_ALL_HSI
                                    ; Enable HSI interrupts
    20BF C41580      191     STB    IOC0_SAVE, IOC0
                     192
    20C2             193 LOOP_FOREVER:
    20C2 27FE        194     SJMP   LOOP_FOREVER; Let interrupts
                                                ; take over
                     195 $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 7
ERR LOC OBJECT        LINE    SOURCE STATEMENT
                      196  ; ******************
                      197  ; ** Subroutines **
                      198  ; ******************
                      199  ;
                      200  ;   Subroutine to empty the FIFO. (initialization)
                      201  ;
    20C4              202  EMPTY_FIFO:
    20C4 901681       203      ORB IOS1_SAVE, IOS1    ; Check for data available
    20C7 378108       204      JBC IOS1_SAVE, HOLDING_REG_FULL, FIFO_EMPTY
                      205
    20CA 7171F81      206      ANDB IOS1_SAVE, #NOT HSI_HOLDING_FULL  ; clear
                                                                      ; data avail bit
    20CD A00400       207      LD  R0, HSI_TIME   ; Clear FIFO entry
    20D0 27F2         208      SJMP EMPTY_FIFO    ; Keep checking
                      209
    20D2              210  FIFO_EMPTY:
    20D2 71BF81       211      ANDB IOS1_SAVE, #NOT HSI_FIFO_FULL     ; Clear fifo
                                                                      ; full bit
    20D5 F0           212      RET
                      213  $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 1
ERR LOC OBJECT        LINE    SOURCE STATEMENT
                      214   ;
                      215   ;   Subroutine to service the HSI FIFO, when it
                      216   ;   has data available from the 4th entry on up.
                      217   ;
    20D6              218   HSI_SERVICE_ROUTINE:
    20D6 F2           219       PUSHF
                      220
                      221   ;   IOPORT1.5 is used to monitor time spent in
                                HSI ISR routine
    20D7 91200F       222       ORB   IOPORT1,#HSI_FIFO_ROUTINE
                      223
    20DA              224   CHECK_FOR_HSI_ENTRIES:
    20DA 901681       225       ORB   IOS1_SAVE, IOS1 ; Check IOS1 for HSI events
    20DD 3F8102       226       JBS   IOS1_SAVE, HOLDING_REG_FULL, HOLDING_FULL
    20E0 206B         227       SJMP  EXIT_ROUTINE   ; When no more events exit
                                                     ; routine
                      228
    20E2              229   HOLDING_FULL:
    20E2 717F81       230       ANDB  IOS1_SAVE, #NOT HSI_HOLDING_FULL
    20E5 B00682       231       LDB   HSI_EVENT_STATUS, HSI_STATUS  ; Get status and
                                                                    ; time
    20E8 A00488       232       LD    EVENT_TIME, HSI_TIME
                      233
    20EB              234   SCAN_HSI:
                      235   ; Check for activity on each HSI input pin. If activity,
                      236   ; check flags. If the bit is set then HSI time is put
                      237   ; into FTIME and increment TC, else if bit is cleared,
                            ; put HSI time into ITIME and TC=0.
    20EB              238   CHECK_0:
    20EB 308215       239       JBC   HSI_EVENT_STATUS, CH_0, CHECK_1
    20EE 911083       240       ORB   NEW_CALC_FLAG, #MASK_0
                      241
    20F1 30830A       242       JBC   NEW_CALC FLAG, BIT_0, PROCESS_FTIME_0
    20F4 1184         243       CLRB  TC_0   ; Clear HSI pulse counter
    20F6 A0888A       244       LD    ITIME_0, EVENT_TIME   ;Load itime with HSI time
    20F9 71FE83       245       ANDB  NEW_CALC_FLAG, #NOT MASK_CH_0  ; SET ftime flag
    20FC 2005         246       SJMP  CHECK_1 ; check HSI.1
                      247
    20FE              248   PROCESS_FTIME_0:
    20FE 1784         249       INCB  TC_0  ; Increment HSI pulse counter
    2100 A08892       250       LD    FTIME_0, EVENT_TIME  ; load ftime with HSI time
                      251
    2103              252   CHECK_1:
    2103 328215       253       JBC   HSI_EVENT_STATUS, CH_1, CHECK_2
    2106 912083       254       ORB   NEW_CALC_FLAG, #MASK_1
                      255
    2109 31830A       256       JBC   NEW_CALC_FLAG, BIT_1, PROCESS_FTIME_1
    210C 1185         257       CLRB  TC_1
    210E A0888C       258       LD    ITIME_1, EVENT_TIME
    2111 71FD83       259       ANDB  NEW_CALC_FLAG, #NOT MASK_CH_1
    2114 2005         260       SJMP  CHECK_2
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 8
ERR LOC OBJECT        LINE    SOURCE STATEMENT
                      261
    2116              262   PROCESS_FTIME_1:
    2116 1785         263       INCBTC_1
    2118 A08894       264       LD  FTIME_1, EVENT_TIME
                      265
    211B              266   CHECK_2:
    211B 348215       267       JBC HSI_EVENT_STATUS, CH_2, CHECK_3
    211E 914083       268       ORB NEW_CALC_FLAG, #MASK_2
                      269
    2121 32830A       270       JBC NEW_CALC_FLAG, BIT_2, PROCESS_FTIME_2
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88     09:02:09 PAGE 9
ERR LOC OBJECT        LINE      SOURCE STATEMENT
    2124 1186     271     CLRB  TC_2
    2126 A0888E   272     LD    ITIME_2, EVENT_TIME
    2129 71FB83   273     ANDB  NEW_CALC_FLAG, #NOT MASK_CH_2
    212C 2005     274     SJMP  CHECK_3
                  275
    212E          276 PROCESS_FTIME_2:
    212E 1786     277     INCB  TC_2
    2130 A08896   278     LD    FTIME_2, EVENT_TIME
                  279
    2133          280 CHECK_3:
    2133 368215   281     JBC   HSI_EVENT_STATUS, CH_3, SCAN_COMPLETE
    2136 918083   282     ORB   NEW_CALC_FLAG, #MASK_3
                  283
    2139 33830A   284     JBC   NEW_CALC_FLAG, BIT_3, PROCESS_FTIME_3
    213C 1187     285     CLRB  TC_3
    213E A08890   286     LD    ITIME_3, EVENT_TIME
    2141 71F783   287     ANDB  NEW_CALC_FLAG, #NOT MASK_CH_3
    2144 2005     288     SJMP  SCAN_COMPLETE
                  289
    2146          290 PROCESS_FTIME_3:
    2146 1787     291     INCB  TC_3
    2148 A08898   292     LD    FTIME_3, EVENT_TIME
                  293
    214B          294 SCAN_COMPLETE:
    214B 278D     295     SJMP  CHECK_FOR_HSI_ENTRIES  ; Continue until no
                                                       ; more events
                  296
    214D          297 EXIT_ROUTINE:
    214D 71DF0F   298     ANDB  IOPORT1, #NOT HSI_FIFO_ROUTINE ; clear P1.5
    2150 F3       299     POPF
    2151 F0       300     RET
                  301 $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88    09:02:09 PAGE 10
ERR LOC OBJECT           LINE      SOURCE STATEMENT
                  302 ;
                  303 ;   Subroutine to handle software timer interrupts
                      ; (every 5 ms)
                  304 ;
    2152          305 SOFTWARE_TIMER_ISR:
    2152 F2       306    PUSHF  ; Push PSW and INT_mask on to Stack
                  307
    2153 B10408   308    LDB  INT_MASK, #ALLOW_4TH_ENTRY_INT
    2156 FB       309    EI   ; Allow HSI interrupts during the software
                          ; timer ISR
    2157 95800F   310    XORB  IOPORT1, #MAIN_LOOP_CONTROL   ; Toggle P1.7
    215A 901681   311    ORB IOS1_SAVE, IOS1 ; Get ST status bits
    215D 308128   312    JBC  IOS1_SAVE, 0, EXIT_SOFTWARE_TIMER_ISR  ; Check if
                                                                     ; ST_0 set
                  313
    2160 71FE81   314    ANDB  IOS1_SAVE, #NOT MASK_SOFTWARE_TIMER_0 ; Clear, set
                                                                     ; bit
    2163 2825     315    SCALL  RESTART_SOFTWARE_TIMER_0  ; Restart Software Timer
                  316
                  317  ;
                  318  ; Do ABS wheel speed calculations. Each HSI input is
                  319  ; calculated for a wheel speed.
                  320    ;
    2165          321    CALC_0:
    2165 348305   322    JBC NEW_CALC_FLAG, BIT_4, CALC_1  ; check for any edges
                                                           ; on HSI.0
    2168 FA       323    DI  ; don't allow any interrupts during info gathering
                  324  ;     .
                  325  ;     .
                  326  ;     .
    2169 910183   327    ORB  NEW_CALC_FLAG, #MASK_CH_0  ; Clear flag, next
                                                        ; will = ITIME
    216C FB       328    EI   ; ReENABLE interrupts
                  329  ;
                  330  ; Do the same calculation for each HSI input
                  331  ;
    216D          332    CALC_1:
    216D 358305   333    JBC  NEW_CALC_FLAG, BIT_5, CALC_2
    2170 FA       334    DI
                  335  ; .
                  336  ; .
                  337  ; .
    2171 910283   338    ORB  NEW_CALC_FLAG, #MASK_CH_1
    2174 FB       339    EI
    2175          340    CALC_2:
    2175 368305   341    JBC NEW_CALC_FLAG, BIT_6, CALC_3
    2178 FA       342    DI
                  343  ; .
                  344  ; .
                  345  ; .
    2179 910483   346    ORB NEW_CALC_FLAG, #MASK_CH_2
    217C FB       347    EI
    217D          348    CALC_3:
    217D 378305   349    JBC  NEW_CALC_FLAG, BIT_7, _CALC_NONE
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88    09:02:09 PAGE 10
ERR LOC OBJECT        LINE      SOURCE STATEMENT
    2180 FA       350 DI
                  351  ; .
                  352  ; .
                  353  ; .
    2181 910883   354 ORB NEW_CALC_FLAG, #MASK_CH_3
    2184 FB       355 EI
                  356
    2185          357 CALC_NONE:
    2185 B10F83   358 LDB NEW_CALC_FLAG, #ALL_NEW_CALC   ; Tell HSI routine
                                                         ; time = Itime
                  359
    2188          360 EXIT_SOFTWARE_TIMER_ISR:
    2188 F3       361     POPF ; restore PSW
    2188 F0       362     RET ; Return from SW_timer_ISR
                  363 $eject
```

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88    09:02:09 PAGE 11
ERR LOC OBJECT        LINE      SOURCE STATEMENT
                  364 ;
                  365 ;   Subroutine to restart software timer 0
                  366 ;
    218A          367 RESTART_SOFTWARE_TIMER_0:
    218A 65C409A2 368     ADD   SW_TIMER_0, #SW_TIMER_5MS     ; Add 5 ms to old
                                                              ; timer value
    218E B11806   369     LDB   HSO_COMMAND, #SW_TIMER_0_CMD  ;Load SWT_0 command
    2191 A0A204   370     LD    LSO_TIME, SW_TIMER_0  ; and the time to trigger
                                                      ; interrupt
    2194 F0       371     RET   ; And return
    2195          372 END
```

MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations. 10/13/88 09:02:09 PAGE 12

SYMBOL TABLE LISTING

NAME	VALUE	ATTRIBUTES
AD_COMMAND	0002H	NULL ABS BYTE
AD_RESULT	0002H	NULL ABS WORD
ALL_NEW_CALC	000FH	NULL ABS
ALLOW_4TH_ENTRY_INT	0004H	NULL ABS
ALLOW_SOFTWARE_TIMER_INT	0020H	NULL ABS
BAUD_RATE	000EH	NULL ABS BYTE
BIT_0	0000H	NULL ABS
BIT_1	0001H	NULL ABS
BIT_2	0002H	NULL ABS
BIT_3	0003H	NULL ABS
BIT_4	0004H	NULL ABS
BIT_5	0005H	NULL ABS
BIT_6	0006H	NULL ABS
BIT_7	0007H	NULL ABS
CALC_0	2165H	CODE ABS ENTRY
CALC_1	216DH	CODE ABS ENTRY
CALC_2	2175H	CODE ABS ENTRY
CALC_3	217DH	CODE ABS ENTRY
CALC_NONE	2185H	CODE ABS ENTRY
CH_0	0000H	NULL ABS
CH_1	0002H	NULL ABS
CH_2	0004H	NULL ABS
CH_3	0006H	NULL ABS
CHECK_0	20EBH	CODE ABS ENTRY
CHECK_1	2103H	CODE ABS ENTRY
CHECK_2	211BH	CODE ABS ENTRY
CHECK_3	2133H	CODE ABS ENTRY
CHECK_FOR_HSI_ENTRIES	20DAH	CODE ABS ENTRY
CLEAR_RAM	2085H	CODE ABS ENTRY
EMPTY_FIFO	20C4H	CODE ABS ENTRY
ENABLE_4TH_ENTRY	0080H	NULL ABS
ENABLE_ALL_HSI	0055H	NULL ABS
EVENT_TIME	0088H	REG ABS WORD
EXIT_ROUTINE	214DH	CODE ABS ENTRY
EXIT_SOFTWARE_TIMER_ISR	2188H	CODE ABS ENTRY
FIFO_EMPTY	20D2H	CODE ABS ENTRY
FTIME_0	0092H	REG ABS WORD
FTIME_1	0094H	REG ABS WORD
FTIME_2	0096H	REG ABS WORD
FTIME_3	0098H	REG ABS WORD
HOLDING_FULL	20E2H	CODE ABS ENTRY
HOLDING_REG_FULL	0007H	NULL ABS
HSI_EVENT_STATUS	0082H	REG ABS BYTE
HSI_FIFO_FULL	0040H	NULL ABS
HSI_FIFO_ROUTINE	0020H	NULL ABS
HSI_HOLDING_FULL	0080H	NULL ABS
HSI_MODE	0003H	NULL ABS BYTE
HSI_POS_EDGES	0055H	NULL ABS
HSI_SERVICE_ROUTINE	20D6H	CODE ABS ENTRY
HSI_STATUS	0006H	NULL ABS BYTE
HSI_TIME	0004H	NULL ABS WORD
HSO_COMMAND	0006H	NULL ABS BYTE
HSO_TIME	0004H	NULL ABS WORD

NAME	VALUE	ATTRIBUTES
INITIAL_START	2080H	CODE ABS ENTRY
INT_MASK	0008H	NULL ABS BYTE
INT_MASK1	0013H	NULL ABS
INT_PEND1	0012H	NULL ABS
INT_PENDING	0009H	NULL ABS BYTE
IOC0	0015H	NULL ABS BYTE
IOC0_SAVE	0080H	REG ABS BYTE
IOC1	0016H	NULL ABS BYTE
IOPORT0	000EH	NULL ABS BYTE
IOPORT1	000FH	NULL ABS BYTE
IOPORT2	0010H	NULL ABS BYTE
IOPORT3	1FFEH	NULL ABS WORD
IOPORT4	1FFFH	NULL ABS BYTE
IOS0	0015H	NULL ABS BYTE
IOS1	0016H	NULL ABS BYTE
IOS1_SAVE	0081H	REG ABS BYTE
IOS2	0017H	NULL ABS BYTE
ITIME_0	008AH	REG ABS WORD
ITIME_1	008CH	REG ABS WORD
ITIME_2	008EH	REG ABS WORD
ITIME_3	0090H	REG ABS WORD
LOOP_FOREVER	20C2H	CODE ABS ENTRY
MAIN_LOOP_CONTROL	0080H	NULL ABS
MASK_0	0010H	NULL ABS
MASK_1	0020H	NULL ABS
MASK_2	0040H	NULL ABS
MASK_3	0080H	NULL ABS
MASK_CH_0	0001H	NULL ABS
MASK_CH_1	0002H	NULL ABS
MASK_CH_2	0004H	NULL ABS
MASK_CH_3	0008H	NULL ABS
MASK_SOFTWARE_TIMER_0	0001H	NULL ABS
MPH_0	009AH	REG ABS WORD
MPH_1	009CH	REG ABS WORD
MPH_2	009EH	REG ABS WORD
MPH_3	00A0H	REG ABS WORD
MPH1	61A8H	NULL ABS
NEW_CALC_FLAG	0083H	REG ABS BYTE
PROCESS_FTIME_0	20FEH	CODE ABS ENTRY
PROCESS_FTIME_1	2116H	CODE ABS ENTRY
PROCESS_FTIME_2	212EH	CODE ABS ENTRY
PROCESS_FTIME_3	2146H	CODE ABS ENTRY
PWM_CONTROL	0017H	NULL ABS BYTE
R0	0000H	NULL ABS WORD
RESTART_SOFTWARE_TIMER_0	218AH	CODE ABS ENTRY
SBUF	0007H	NULL ABS BYTE
SCAN_COMPLETE	214BH	CODE ABS ENTRY
SCAN_HSI	20EBH	CODE ABS ENTRY
SOFTWARE_TIMER_ISR	2152H	CODE ABS ENTRY
SP	0018H	NULL ABS WORD
SP_CON	0011H	NULL ABS BYTE
SP_STAT	0011H	NULL ABS BYTE
STK	0100H	NULL ABS
SW_TIMER_0	00A2H	REG ABS WORD
SW_TIMER_0_CMD	0018H	NULL ABS
SW_TIMER_5MS	09C4H	NULL ABS

```
MCS-96 MACRO ASSEMBLER
Program for MCS96-HSI-WHEEL SPEED Calculations.      10/13/88    09:02:09 PAGE 14
   NAME                  VALUE      ATTRIBUTES


TC_0                     0084H      REG ABS BYTE
TC_1                     0085H      REG ABS BYTE
TC_2                     0086H      REG ABS BYTE
TC_3                     0087H      REG ABS BYTE
TEMP                     00A4H      REG ABS LONG
TEMP_0                   00A8H      REG ABS BYTE
TEMP_1                   00A9H      REG ABS BYTE
TEMP_2                   00AAH      REG ABS BYTE
TEMP_3                   00ABH      REG ABS BYTE
TIMER1                   000AH      NULL ABS WORD
TIMER2                   000CH      NULL ABS WORD
WATCHDOG                 000AH      NULL ABS BYTE
WHEEL_SPEED              ------     MODULE MAIN STACKSIZE(0)


ASSEMBLY COMPLETED, NO ERROR(S) FOUND.
```