MOTOROLA
SEMICONDUCTOR
APPLICATION NOTE

Order this document
as AN431/D

AN431

Temperature measurement and display using the MC68HC05B4 and the MC14489

By Jeff Wright,
Motorola Ltd., East Kilbride

INTRODUCTION

This application note is intended to show the basic building blocks of a temperature control system based on the MC68HC05Bx family of MCUs. Software routines in the application include look-up table interpolation, binary to BCD conversion, DegC to DegF conversion and the basis of a real time counter/clock. For temperature display the Multi-character LED display driver MC14489 is used, driven from the B4's SCI, resulting in simple hardware with a low component count. The temperature sensing element used here is a thermistor to allow easy interfacing to the A/D converter of the HC05B4, but the software principles shown would be the same for many other types of sensors. A software listing is included at the end of this application note.

TEMPERATURE MEASUREMENT

A pre-calibrated thermistor was chosen as the temperature sensing element. Its characteristic curve over the temperature range of -40 to 80 °C is shown in Figure 1. To get the best accuracy from the HC05B4's on-board A/D, the input signal should be scaled to use as much of the available VRH-VRL range as possible. Here VRH is connected to Vdd and VRL is tied to Vss. In this case, using the thermistor as potential divider with a 20kΩ resistor results in a signal range of approximately 0.3V to 4.7V over the -40 to 80 °C temperature range. The voltage across the thermistor (input to the A/D), plotted against temperature, is shown in Figure 2.

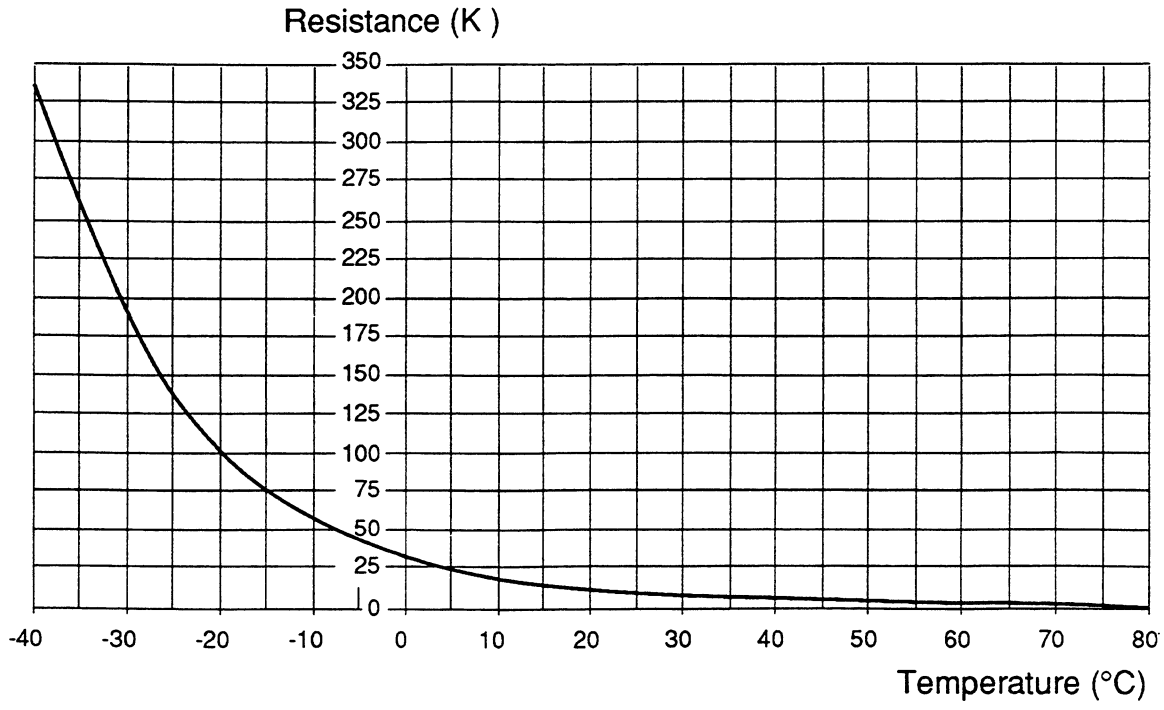

Figure 1. Thermistor resistance vs Temperature

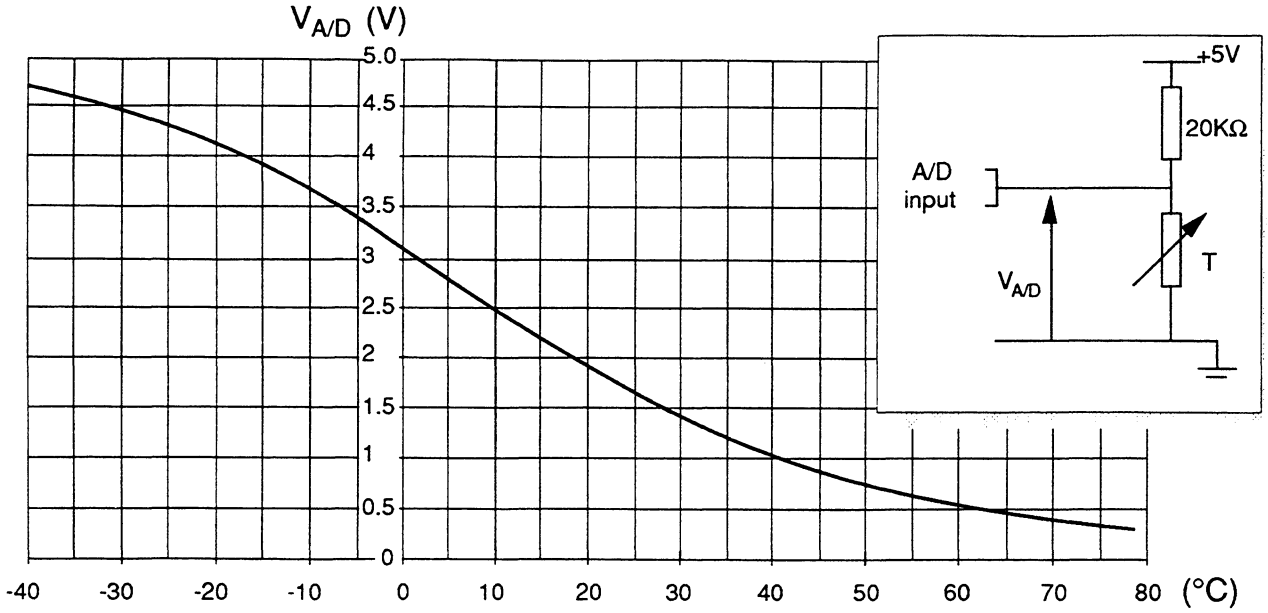

Figure 2. A/D input voltage vs Temperature (inset: circuit used)

As can be seen from Figure 2, the response is non-linear and so a look-up table approach is the simplest way of obtaining the required accuracy. The thermistor characteristics are stored as a series of points in a table in ROM and a linear interpolation between adjacent points is used to obtain the temperature that corresponds to a given A/D reading. The number of points that must be stored depends on how non-linear the response is and the required accuracy of the result. In this case 16 points were chosen; in order to keep the software simple (and therefore fast), they are spread at intervals of 16 through the A/D result range of 0-255. For each point (16, 32, 48 etc.), the voltage on the A/D input was calculated and the corresponding temperature was obtained from the graph of Figure 2. These points were then used to form the look-up table shown in Figure 3, resulting in a temperature range of -40 to 79 °C. Figure 4 shows the reconstructed response of the thermistor obtained by linear interpolation of the points in the look-up table.

A/D RESULT	A/D (volts)	TEMP (°C)	TEMP (°C 2s Compl)
0	0	-	-
16	0.31	79	4F
32	0.63	56	38
48	0.94	43	2B
64	1.26	34	22
80	1.57	27	1B
96	1.88	21	15
112	2.20	15	0F
128	2.51	10	0A
144	2.82	5	05
160	3.14	-1	FF
176	3.45	-6	FA
192	3.77	-11	F5
208	4.08	-18	EE
224	4.39	-26	E6
240	4.71	-40	D8
255	5.0	-	-

Figure 3. Interpolated A/D input voltage vs Temperature

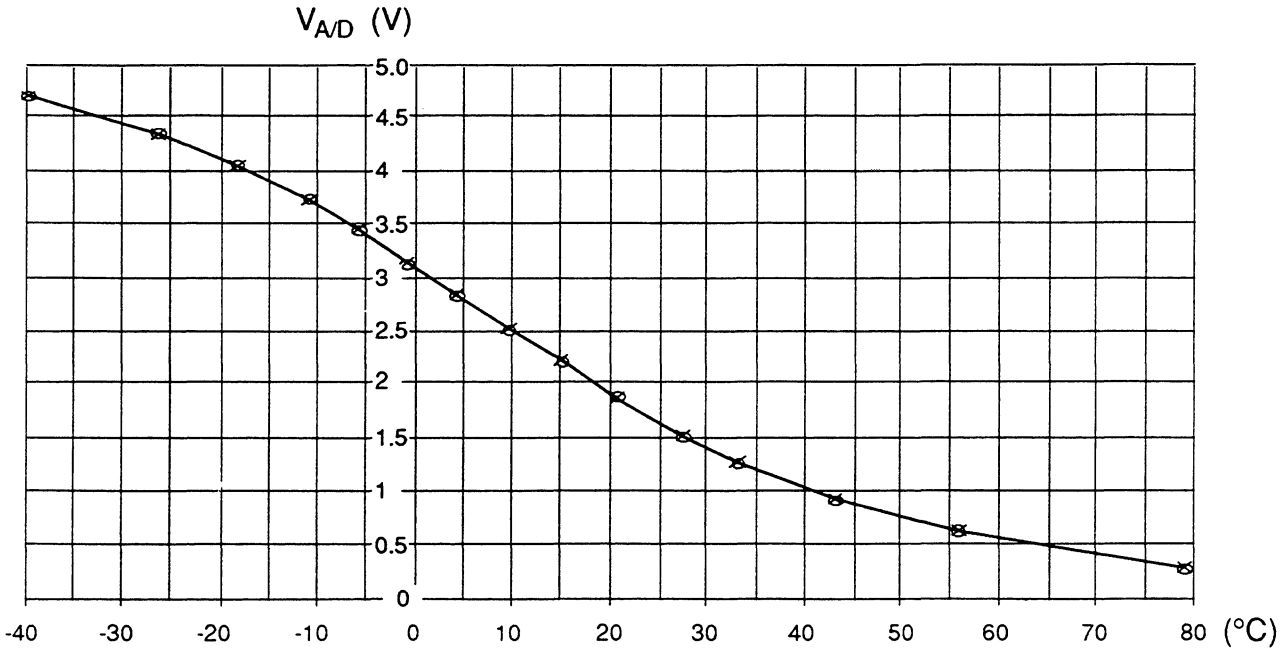

Figure 4. Interpolated A/D input voltage vs Temperature

The temperature reading is updated every second; the software to accomplish this is relatively simple:

The timer is set to overflow every 125 mS with a 4.1934 MHz crystal. The timer overflow interrupt routine updates the real time counters TICKS, SECS, MINS & HRS and sets the flag bit SEC every time a second has elapsed.

The main program loop is executed every second (via the SEC flag bit) and after checking the metric/imperial selector switch the temperature is measured by the subroutine ADCONV. This routine starts by reading the thermistor selector switch and setting up the A/D control register accordingly. An A/D conversion is then carried out four times on the selected channel and the results accumulated in the accumulator and the temporary register TEMP. This result is then divided by 4 by rotating, to obtain the average A/D result. The averaging technique is employed to try and reduce the effect of noise on the A/D input. The number of conversions to average is determined by time constraints and the noise levels in the surrounding environment. The upper nibble of the result is then used to access the look-up table to obtain the 'base' temperature value. If the temperature limit is exceeded then the TLIMIT flag is set before exiting from the routine.

Temperature table entries are stored in 2's complement form so that the interpolation between positive and negative values will work successfully. The interpolation is carried out by obtaining the difference between the base value and the next in the table, multiplying this by the lower nibble of the A/D result and then dividing by 16. This result is then subtracted from the base value to obtain the real temperature in 2's complement °C which is stored in the register NEWTMP before exiting from the routine. The difference information is subtracted from the base value rather than added because the thermistor has a negative temperature co-efficient (NTC) so that an increase in the A/D result corresponds to a drop in temperature.

If the imperial mode is selected (°F) then the next stage before updating the display is to convert from °C to °F and this is carried out in the subroutine CTOF.

Converting from °C to °F is accomplished by multiplying by 1.8 and adding 32. First the sign of the temperature in °C is stored via the flag bit NEGNUM, then the maximum °F limit (53 °C) is checked before the magnitude is multiplied by 1.8 (multiply by 115 and divide by 64). Again, use is made of rotating to do the dividing, in order to increase execution speed. The sign of the result is then restored and 32 added to obtain the temperature in 2's complement °F.

TEMPERATURE DISPLAY

An MC14489 multi-character display driver was chosen for this purpose as it can be easily interfaced to a wide range of Motorola MCUs, requires almost no external components and has a character set that includes the degree symbol (°). The MC14489 can also be cascaded if the application was expanded to require a larger display. The MC14489 would normally be driven from an SPI on the MCU but here, since the the 68HC05B family does not have an SPI, use is made of the SCI clock output feature that is available on this family.

Before the temperature can be written to the display driver it has to be converted into the correct data format.

The first stage of this is to convert from 2's complement binary to BCD. This is carried out in the routine CONBCD which is called from SETDISP. The sign of the temperature is stored in the flag bit NEGNUM before SETDISP is called; then, after first checking if the TLIMIT flag is set, the temperature is converted to BCD in DEC0-2 by CONBCD. This is accomplished by rotating left the binary number followed immediately by a rotate left of the BCD result; this has the effect of multiplying the current BCD result by 2 and adding in the new binary bit at the same time. After each rotate the BCD registers are checked and adjusted for overflow (>$09) before the bit counter contained in the index register is decremented. This process of rotate then adjust is continued until all the binary bits have been used; the BCD result will then be resident in the registers DEC0, 1 & 2.

The rest of the routine SETDISP is concerned with setting up the display registers DISP1, 2, 3 and the display control register DISPC. The MC14489 data format is msb first whereas the 68HC05B4 SCI transmits lsb first; this means that the bit order of the data stream has to be stored in reverse in the display registers. This can be confusing when trying to work out the codes that have to be stored in the B4 to generate a specific character.

Figures 5a and 5b show the 14489 data format and the corresponding bit positions in the B4 registers DISP1, 2, 3 & C. The sign of the temperature is restored and the numeric display registers are configured to display '–' if the temperature limit has been exceeded before exiting from the SETDISP routine.

The main program loop then calls the subroutine DISPL which actually transmits the contents of the display registers to the MC14489 via the SCI. The MC14489 contains special Bit Grabber circuitry that allows either the internal display registers or the configuration register to be updated without address or steering bits so that updating the display involves a simple transmission of either 3 bytes for the display registers or 1 byte for the configuration register. Even for cascaded 14489s there is no need for address bits – see the MC14489 data sheet for more details.

The MC14489 can be clocked at up to 4 MHz at 5 volts so here the maximum transmit baud rate of the SCI is used – 131.072 KHz with a 4.19304 MHz crystal. The transmission of the display data only takes place if there has been a change in the data since the last time. If there has been a change, the 3 data registers are transmitted in turn starting with DISP3 and the OLD registers are updated ready for the change check next time round. After the last byte has gone, the SCI and 14489 are disabled before returning to the main loop.

The last subroutine called from the main program is the 14489 configuration update routine DISCON. This routine operates in a similar manner to DISPL, checking to see if there has been a change to the config. data before transmitting it.

This completes the operation of the program which now jumps back to the start of the main loop and waits for the SEC bit to be set again before repeating the temperature measurement and display sequence.

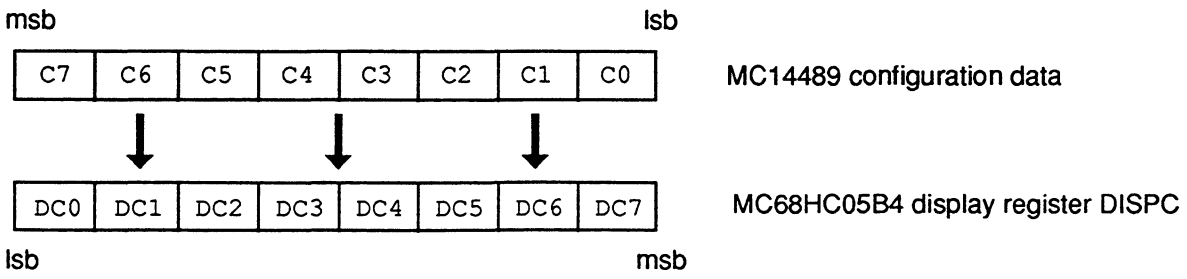

Figure 5a. MC14489 to MC68HC05B4 display register mapping

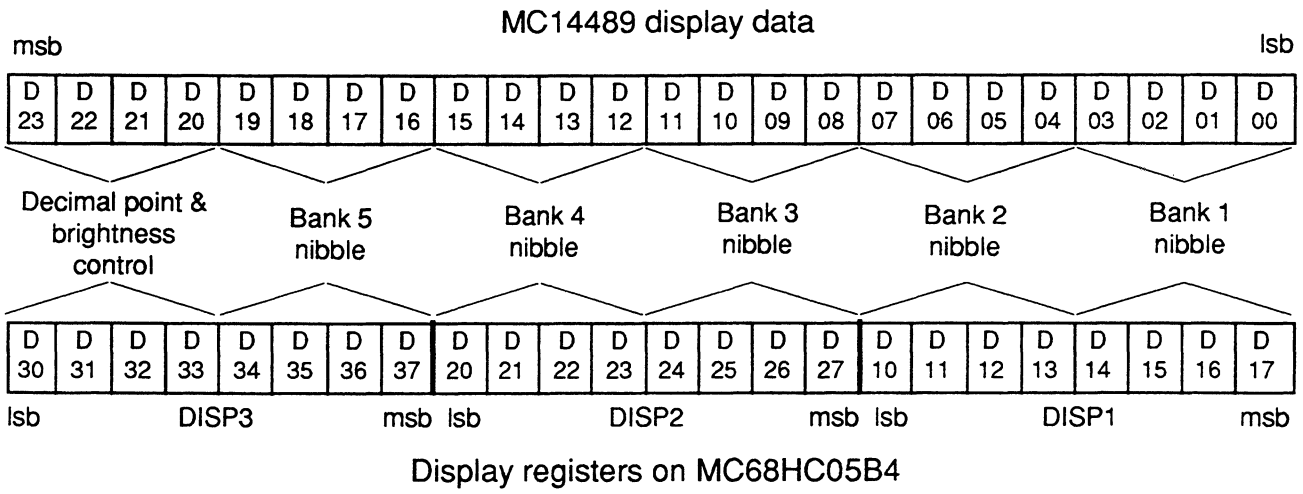

Figure 5b. MC14489 to MC68HC05B4 display register mapping

HARDWARE

As already mentioned, the use of the MC14489 results in a very low component count for the application; the hardware schematic can be seen in Figure 6. The only I/O pins required are for reading the option switches and for controlling the enable of the MC14489. Pull-downs are required on the clock and data pins as these become high impedance when the SCI is disabled. The LED displays are common cathode; a single external resistor is all that is required to set the brightness level of the displays. In this case though, a light dependent resistor, R12 (ORP12), has been used to control the display brightness for a variety of background lighting conditions. The resistance of R12 decreases with increasing light and so R11 must be incorporated to ensure that the maximum source current spec. of the MC14489 is not exceeded in very bright lighting conditions. R13 ensures there is still enough drive current for the LEDs in dark conditions.

APPLICATION AREAS

As mentioned in the introduction, this application note is designed only to show some fundamental building blocks of a temperature control system based on the 68HC05Bx family of MCUs. Where possible, the software has been written in a modular fashion, so that the routines can easily be transported to another application and the binary to BCD routine could be expanded to handle larger numbers. The large number of I/O, PWMs and timer functions unused show that the 68HC05B family has plenty of functionality left to perform other control functions. For example, in process control, fluid flow or speed sensors could be connected to the timer input capture pins, pressure sensors to the other A/D pins, a keypad to the I/O lines and the other I/O & PWMs used to perform output control functions.

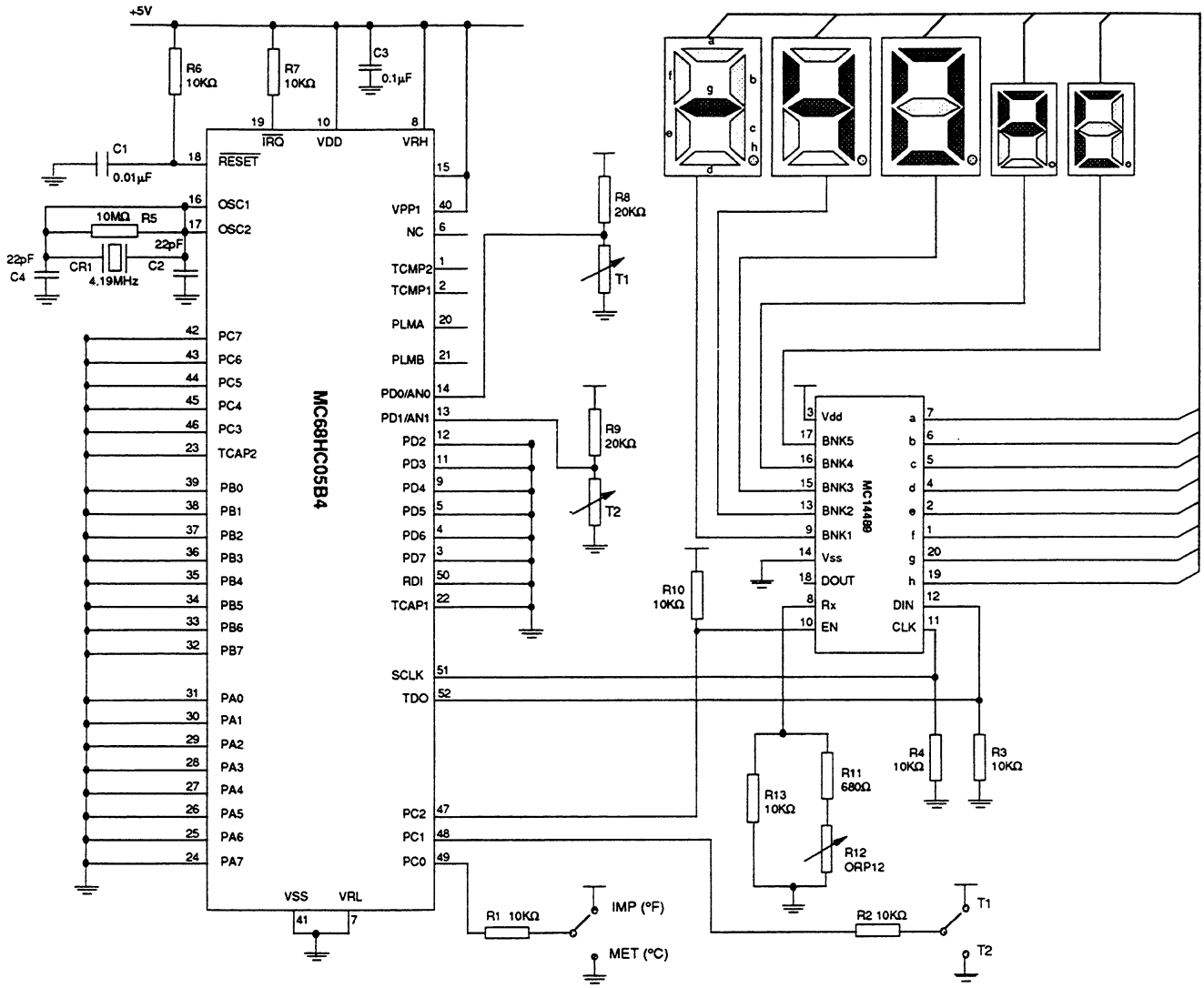

Figure 6. Hardware schematic

```
 1                      ************************************************************************
 2                      *%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*
 3                      *%                                                                      %*
 4                      *%     68HC05B4  TEMPERATURE MEASUREMENT & DISPLAY                      %*
 5                      *%                                                                      %*
 6                      *%                                                                      %*
 7                      *%  Jeff Wright, Motorola East Kilbride.   Last Updated  22/02/90       %*
 8                      *%                                                                      %*
 9                      *%  This software was written by Motorola for demonstration             %*
10                      *%  purposes only. Motorola does not assume any liability arising       %*
11                      *%  out of the application or use of this software and does not         %*
12                      *%  guarantee its functionality                                         %*
13                      *%                                                                      %*
14                      *%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*
15                      ************************************************************************
16
17
18                      *********    I/O and INTERNAL registers definition   ****************
19                      *
20                      *
21                      *      I/O registers
22                      *
23 00000000             PORTA   EQU     $00      port A.
24 00000001             PORTB   EQU     $01      port B.
25 00000002             PORTC   EQU     $02      port C.
26 00000003             PORTD   EQU     $03      port D.
27 00000004             DDRA    EQU     $04      port A DDR.
28 00000005             DDRB    EQU     $05      port B DDR.
29 00000006             DDRC    EQU     $06      port C DDR.
30
31                      *
32                      *      A/D registers
33                      *
34 00000008             ADDATA  EQU     $08      A/D data register.
35 00000009             ADSTCT  EQU     $09      A/D status and control register.
36 00000007             COCO    EQU     7             Conversion complete flag.
37
38                      *
39                      *
40                      *
41                      *      SCI registers
42                      *
43 0000000d             BAUD    EQU     $0D      SCI baud register.
44 0000000e             SCCR1   EQU     $0E      SCI control register 1.
45 0000000f             SCCR2   EQU     $0F      SCI control register 2.
46 00000010             SCSR    EQU     $10      SCI status register.
47 00000007             TDRE    EQU     7
48 00000006             TC      EQU     6
49 00000011             SCDAT   EQU     $11      SCI data register.
50
51                      *
52                      *      TIMER registers
53                      *
54 00000012             TCR     EQU     $12      Timer control register.
55 00000005             TOIE    EQU     5        Timer overflow interrupt enable.
56 00000006             OCIE    EQU     6        Timer output compares interrupt enable.
57 00000007             ICIE    EQU     7        Timer input captures interrupt enable.
58
59 00000013             TSR     EQU     $13      Timer status register.
60 00000003             OCF2    EQU     3             Timer output compare 2 flag.
61 00000004             ICF2    EQU     4             Timer input capture 2 flag.
62 00000005             TOF     EQU     5             Timer overflow flag.
63 00000006             OCF1    EQU     6             Timer output compare 1 flag.
64 00000007             ICF1    EQU     7             Timer input capture 1 flag.
```

```
 65
 66 00000014                TIC1HI  EQU     $14     Timer input capture register 1 (16-bit).
 67 00000015                TIC1LO  EQU     $15
 68 00000016                TOC1HI  EQU     $16     Timer output compare register 1 (16-bit).
 69 00000016                TOC1LO  EQU     $16
 70 00000018                TIMHI   EQU     $18     Timer free running counter (16-bit).
 71 00000019                TIMLO   EQU     $19
 72 0000001a                TIMAHI  EQU     $1A     Timer alternate counter register (16-bit).
 73 0000001b                TIMALO  EQU     $1B
 74 0000001c                TIC2HI  EQU     $1C     Timer input capture register 2 (16-bit).
 75 0000001d                TIC2LO  EQU     $1D
 76 0000001e                TOC2HI  EQU     $1E     Timer output compare register 2 (16-bit).
 77 0000001f                TOC2LO  EQU     $1F
 78
 79                         *
 80                         *
 81                         *       MEMORY MAP DEFINITION
 82                         *
 83                         *
 84 00000020                TEST    EQU     $20             TEST register
 85 00000020                ROM0    EQU     $0020   Start address of ROM0.
 86 00000050                RAM     EQU     $0050   Start address of RAM.
 87 00000f00                UROM    EQU     $0F00   Start address of main user ROM.
 88                         *
 89
 90
 91
 92                         ******************  RAM ALLOCATION  ******************************
 93
 94                                 SECTION.S .RAM,ADDR=$50
 95
 96
 97 00000050                TICKS   RMB     1
 98 00000051                SECS    RMB     1
 99 00000052                MINS    RMB     1
100 00000053                HRS     RMB     1
101
102 00000054                FLAG    RMB     1
103 00000000                OVERFL  EQU     0
104 00000001                NEGNUM  EQU     1
105 00000002                TLIMIT  EQU     2
106 00000003                SEC     EQU     3
107
108 00000055                MODE    RMB     1
109 00000000                IMP     EQU     0
110
111 00000056                BIN0    RMB     1
112 00000057                DEC2    RMB     1
113 00000058                DEC1    RMB     1
114 00000059                DEC0    RMB     1
115
116 0000005a                NEWTMP  RMB     1
117 0000005b                TEMP    RMB     1
118 0000005c                TEMP1   RMB     1
119 0000005d                TEMP2   RMB     1
120
121 0000005e                DISP1   RMB     1
122 0000005f                DISP2   RMB     1
123 00000060                DISP3   RMB     1
124 00000061                DISPC   RMB     1
125 00000062                OLDD1   RMB     1
126 00000063                OLDD2   RMB     1
```

```
00000064                    OLDD3   RMB     1
00000065                    OLDDC   RMB     1


                                        SECTION .PAGE0,ADDR=$020

00000020 004f382b221b150f   ADTAB   FCB     $00,$4F,$38,$2B,$22,$1B,$15,$0F
00000028 0a05fffaf5eee6d8           FCB     $0A,$05,$FF,$FA,$F5,$EE,$E6,$D8

                            ********************************************************
                            *                                                      *
                            *       START OF CODE                                  *
                            *                                                      *
                            ********************************************************

                                        SECTION .USROM,ADDR=$F00

00000f00                    RESET   EQU     *
00000f00  a600                      LDA     #$0     Initialise Ports.
00000f02  b700                      STA     PORTA
00000f04  b701                      STA     PORTB
00000f06  b704                      STA     DDRA
00000f08  b705                      STA     DDRB
00000f0a  ae65                      LDX     #OLDDC
00000f0c  f7                INIRAM  STA     ,X      Initialise all used RAM locations.
00000f0d  5a                        DECX
00000f0e  a350                      CPX     #RAM
00000f10  26fa                      BNE     INIRAM

00000f12  a604                      LDA     #$04
00000f14  b702                      STA     PORTC
00000f16  a604                      LDA     #$04    PC2 output high.
00000f18  b706                      STA     DDRC

00000f1a  b613              TIMINT  LDA     TSR     clr any pending flags.
00000f1c  b619                      LDA     TIMLO
00000f1e  a620                      LDA     #$20    Enable timer overflow
00000f20  b712                      STA     TCR             interrupt.
00000f22  9a                        CLI

                            *--------- START OF MAIN PROGRAM LOOP ----------*

00000f23                    MAINLUP EQU     *
00000f23  0754fd                    BRCLR   SEC,FLAG,MAINLUP
00000f26  1754                      BCLR    SEC,FLAG
00000f28  1155                      BCLR    IMP,MODE        Check metric/imperial selector.
00000f2a  010202                    BRCLR   0,PORTC,NOIMP   Check degC/degF switch.
00000f2d  1055                      BSET    IMP,MODE
00000f2f  1354              NOIMP   BCLR    NEGNUM,FLAG     Clear sign indicator.
00000f31  cd0ffd                    JSR     ADCONV          Go measure temperature
00000f34  015503                    BRCLR   IMP,MODE,GOMETR           (in degC - 2s compl)
00000f37  cd0f4e                    JSR     CTOF            Convert to degF.
00000f3a  b65a              GOMETR  LDA     NEWTMP
00000f3c  2a03                      BPL     GOMORE
00000f3e  40                        NEGA                    Only use magnitude to do BCD conv.
00000f3f  1254                      BSET    NEGNUM,FLAG     Remember the sign of the number.
00000f41  b756              GOMORE  STA     BIN0            Store temperature for conv to BCD.
00000f43  cd0f78            GODISP  JSR     SETDISP         Set-up display bytes.
00000f46  cd1085                    JSR     DISPL           Update display if neccessary.
00000f49  cd10ca                    JSR     DISCON          Update 14489 config if neccessary.
00000f4c  20d5                      BRA     MAINLUP
```

```
190                          *=================================================================*
191                          *=                                                               =*
192                          *=      CTOF            - Converts NEWTMP from degC to degF      =*
193                          *=                                                               =*
194                          *=================================================================*
195
196 00000f4e                CTOF    EQU     *
197 00000f4e  b65a                   LDA     NEWTMP
198 00000f50  2a05                   BPL     NONEG
199 00000f52  1254                   BSET    NEGNUM,FLAG     Remember if No is negative or not.
200 00000f54  40                     NEGA
201 00000f55  2007                   BRA     MUL1P8
202 00000f57  a135           NONEG   CMP     #53             Check for max degF limit of 127F.
203 00000f59  2503                   BLO     MUL1P8
204 00000f5b  1454                   BSET    TLIMIT,FLAG     Set limit and return if over range.
205 00000f5d  81                     RTS
206
207 00000f5e  ae73           MUL1P8  LDX     #115
208 00000f60  42                     MUL                     Multiply by 115 and divide by 64.
209 00000f61  56                     RORX
210 00000f62  46                     RORA                    (same as multiplying by 1.8)
211 00000f63  56                     RORX
212 00000f64  46                     RORA
213 00000f65  56                     RORX
214 00000f66  46                     RORA
215 00000f67  56                     RORX
216 00000f68  46                     RORA
217 00000f69  56                     RORX
218 00000f6a  46                     RORA
219 00000f6b  56                     RORX
220 00000f6c  46                     RORA
221
222 00000f6d  035401                 BRCLR   NEGNUM,FLAG,NONEG1
223 00000f70  40                     NEGA                    Return sign of number.
224 00000f71  1354           NONEG1  BCLR    NEGNUM,FLAG
225 00000f73  ab20                   ADD     #32             Add 32 to get degF.
226 00000f75  b75a                   STA     NEWTMP
227 00000f77  81                     RTS
228
229
230                          *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*
231                          *X                                                               X*
232                          *X      SETDISP         - Sets up display registers with BCD     X*
233                          *X                                                               X*
234                          *XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*
235
236 00000f78                SETDISP EQU     *
237 00000f78  04543e                 BRSET   TLIMIT,FLAG,FORCE     If temp out of range, force to -
238 00000f7b  ae08                   LDX     #$8
239 00000f7d  cd1052                 JSR     CONBCD          Convert 8 bit binary to 3 digit BCD.
240 00000f80  ae04                   LDX     #4
241 00000f82  4f                     CLRA
242 00000f83  3458           LUPDIS1 LSR     DEC1            Shuffle bit order of digits to allow
243 00000f85  49                     ROLA                    for SCI lsb first and 14489 msb first
244 00000f86  5a                     DECX                    incompatability.
245 00000f87  26fa                   BNE     LUPDIS1
246 00000f89  be57                   LDX     DEC2
247 00000f8b  2704                   BEQ     TSTNEG
248 00000f8d  aa80                   ORA     #$80            If over 100deg, add the 100 digit.
249 00000f8f  2005                   BRA     STD1
250 00000f91  035402         TSTNEG  BRCLR   NEGNUM,FLAG,STD1
251 00000f94  aab0                   ORA     #$B0            Add code for a - if temp is negaive.
252 00000f96  b75e           STD1    STA     DISP1           Store in 1st display register.
```

```
00000f98  ae08            LDX     #8
00000f9a  4f              CLRA
00000f9b  3459     LUPDIS2 LSR     DEC0
00000f9d  49              ROLA
00000f9e  5a              DECX                    Shuffle bit order of digits as above.
00000f9f  26fa            BNE     LUPDIS2
00000fa1  aa0f            ORA     #$0F            add code for the deg symbol.
00000fa3  b75f            STA     DISP2           Store in second display register.
00000fa5  a631            LDA     #$31            Big C, all d.ps off.
00000fa7  015502          BRCLR   IMP,MODE,STDIS3
00000faa  a6f1            LDA     #$F1            Big F, all d.ps off.
00000fac  b760     STDIS3  STA     DISP3
00000fae  a6cb            LDA     #$CB
00000fb0  be57            LDX     DEC2
00000fb2  2702            BEQ     STDISC
00000fb4  a68b            LDA     #$8B
00000fb6  b761     STDISC  STA     DISPC
00000fb8  81              RTS
00000fb9  a6bb     FORCE   LDA     #$BB            FORCE DISPLAY TO -^C
00000fbb  b75e            STA     DISP1
00000fbd  a6bf            LDA     #$BF            or -^F
00000fbf  b75f            STA     DISP2
00000fc1  a631            LDA     #$31
00000fc3  015502          BRCLR   IMP,MODE,STDI3
00000fc6  a6f1            LDA     #$F1
00000fc8  b760     STDI3   STA     DISP3
00000fca  a6fb            LDA     #$FB
00000fcc  b761            STA     DISPC
00000fce  81              RTS


                  *0000000000000000000000000000000000000000000000000000000000000000*
                  *0                                                              0*
                  *0      TOVINT          - Timer Overflow IRQ routine            0*
                  *0                                                              0*
                  *0000000000000000000000000000000000000000000000000000000000000000*

00000fcf  0b132a  TOVINT  BRCLR   TOF,TSR,NOOVF   Check Tim overflow has really happened.
00000fd2  3c50            INC     TICKS
00000fd4  b650            LDA     TICKS           UPDATE REAL TIME CLOCK COUNTERS
00000fd6  a108            CMP     #8
00000fd8  2520            BLO     NOINC
00000fda  3f50            CLR     TICKS
00000fdc  3c51            INC     SECS
00000fde  1654            BSET    SEC,FLAG
00000fe0  b651            LDA     SECS
00000fe2  a13c            CMP     #60
00000fe4  2514            BLO     NOINC
00000fe6  3f51            CLR     SECS
00000fe8  3c52            INC     MINS
00000fea  b652            LDA     MINS
00000fec  a13c            CMP     #60
00000fee  250a            BLO     NOINC
00000ff0  3f52            CLR     MINS
00000ff2  b653            LDA     HRS
00000ff4  a1ff            CMP     #$FF
00000ff6  2702            BEQ     NOINC
00000ff8  3c53            INC     HRS

00000ffa  b619     NOINC   LDA     TIMLO           Clear TOF flag.
00000ffc  80       NOOVF   RTI
```

```
                                *^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^*
                                *^                                                           ^*
                                *^ ADCONV - A/D Conversion & Temperature table interpolation.^*
                                *^                                                           ^*
                                *^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^*

00000ffd           ADCONV   EQU     *
00000ffd  1554              BCLR    TLIMIT,FLAG
00000fff  3f5b              CLR     TEMP
00001001  020204            BRSET   1,PORTC,CONT1   Check Thermistor selector switch.
00001004  a621              LDA     #$21
00001006  2002              BRA     SETAD
00001008  a620     CONT1    LDA     #$20
0000100a  b709     SETAD    STA     ADSTCT          Start first conversion.
0000100c  ae04              LDX     #4              Init counter.
0000100e  4f                CLRA
0000100f  0f09fd   ADLUP1   BRCLR   COCO,ADSTCT,ADLUP1      Wait for end of conversion.
00001012  bb08              ADD     ADDATA
00001014  2402              BCC     DECCX           Convert 4 times and accumulate to help
00001016  3c5b              INC     TEMP            eliminate noise.
00001018  5a       DECCX    DECX
00001019  26f4              BNE     ADLUP1
0000101b  365b              ROR     TEMP
0000101d  46                RORA                    Now divide by 4 to get average and
0000101e  365b              ROR     TEMP
00001020  46                RORA                    store in TEMP.
00001021  b75b              STA     TEMP

00001023  44       TABL     LSRA
00001024  44                LSRA                    Isolate upper 4 bits of result,
00001025  44                LSRA
00001026  44                LSRA
00001027  97                TAX
00001028  e620              LDA     ADTAB,X         and use them to access the look-up table
0000102a  2723              BEQ     TRANGE
0000102c  a1d8              CMP     #$D8            Check table entry limits.
0000102e  271f              BEQ     TRANGE
00001030  b75c              STA     TEMP1           Store "base" value.
00001032  5c                INCX
00001033  e020              SUB     ADTAB,X         Get the diff between the base and next entry.
00001035  b75d              STA     TEMP2
00001037  b65b              LDA     TEMP
00001039  a40f              AND     #$0F            Now get the lower 4 bits of the A/D result.
0000103b  be5d              LDX     TEMP2
0000103d  42                MUL                     Multiply by the difference.
0000103e  49                ROLA
0000103f  59                ROLX
00001040  49                ROLA                    Divide answer by 16 and leave in TEMP2.
00001041  59                ROLX
00001042  49                ROLA
00001043  59                ROLX
00001044  49                ROLA
00001045  59                ROLX
00001046  bf5d              STX     TEMP2
00001048  b65c              LDA     TEMP1           Retrieve base value,
0000104a  b05d              SUB     TEMP2           subtract the difference value
0000104c  b75a              STA     NEWTMP          and store answer in NEWTMP.
0000104e  81                RTS
0000104f  1454     TRANGE   BSET    TLIMIT,FLAG
00001051  81                RTS
```

```
379                     *&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&*
380                     *&                                                                       &*
381                     *&      CONBCD  - Converts Binary in BIN0 to BCD in DEC0-2               &*
382                     *&                                                                       &*
383                     *&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&*
384
385 00001052            CONBCD  EQU     *
386 00001052  4f                CLRA
387 00001053  b759              STA     DEC0            Clear BCD result bytes.
388 00001055  b758              STA     DEC1
389 00001057  b757              STA     DEC2
390
391
392 00001059  3956      LUPBCD  ROL     BIN0            Put the next binary bit in carry.
393 0000105b  3959              ROL     DEC0            Multiply current result by 2 and
394 0000105d  3958              ROL     DEC1            add in new bit at same time.
395 0000105f  3957              ROL     DEC2
396 00001061  b659              LDA     DEC0
397 00001063  a00a              SUB     #$0A
398 00001065  2b04              BMI     TSTD1           Now check the BCD bytes
399 00001067  b759              STA     DEC0            for overflow.
400 00001069  3c58              INC     DEC1
401 0000106b  b658      TSTD1   LDA     DEC1
402 0000106d  a00a              SUB     #$0A
403 0000106f  2b04              BMI     TSTD2
404 00001071  b758              STA     DEC1
405 00001073  3c57              INC     DEC2
406 00001075  b657      TSTD2   LDA     DEC2
407 00001077  a00a              SUB     #$0A
408 00001079  2b06              BMI     NOOVR
409 0000107b  a609              LDA     #9
410 0000107d  b757              STA     DEC2            BCD number has overflowed so set flag
411 0000107f  1054              BSET    OVERFL,FLAG     and set upper digit to 9.
412 00001081  5a        NOOVR   DECX
413 00001082  26d5              BNE     LUPBCD          Any more bits to do?
414 00001084  81                RTS
415
416
417                     *@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*
418                     *@                                                                       @*
419                     *@      DISPL   -  Updates 14489 Display registers via SCI               @*
420                     *@                                                                       @*
421                     *@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*
422
423 00001085            DISPL   EQU     *
424 00001085  b65e              LDA     DISP1           Only update display registers if any
425 00001087  b162              CMP     OLDD1           of them have changed since the last
426 00001089  260d              BNE     UPDATE          time.
427 0000108b  b65f              LDA     DISP2
428 0000108d  b163              CMP     OLDD2
429 0000108f  2607              BNE     UPDATE
430 00001091  b660              LDA     DISP3
431 00001093  b164              CMP     OLDD3
432 00001095  2601              BNE     UPDATE
433 00001097  81                RTS
434
435 00001098  a601      UPDATE  LDA     #$01
436 0000109a  b70e              STA     SCCR1           Clock idle low, edge in mid data, last clk.
437 0000109c  4a                DECA
438 0000109d  b70d              STA     BAUD            131.072KHz baud with 4.19etc XTAL.
439 0000109f  a608              LDA     #$08
440 000010a1  b70f              STA     SCCR2           Transmit enabled.
441 000010a3  0d10fd    PREAM   BRCLR   TC,SCSR,PREAM   Wait for preamble to finish.
442 000010a6  1502              BCLR    2,PORTC         Enable transmission to 14489.
```

```
000010a8  b660            LDA     DISP3
000010aa  b764            STA     OLDD3
000010ac  b711            STA     SCDAT            Send first byte.
000010ae  0f10fd  DWAIT1  BRCLR   TDRE,SCSR,DWAIT1 Wait until it has been transfered
000010b1  b65f            LDA     DISP2            - then load second.
000010b3  b763            STA     OLDD2
000010b5  b711            STA     SCDAT
000010b7  0f10fd  DWAIT2  BRCLR   TDRE,SCSR,DWAIT2
000010ba  b65e            LDA     DISP1
000010bc  b762            STA     OLDD1
000010be  b711            STA     SCDAT
000010c0  0d10fd  DWAIT3  BRCLR   TC,SCSR,DWAIT3  Wait until 3rd byte has actually gone
000010c3  a600            LDA     #$00
000010c5  b70f            STA     SCCR2            Dissable SCI transmissions,
000010c7  1402            BSET    2,PORTC          then disable 14489.
000010c9  81              RTS


                  *?????????????????????????????????????????????????????????????????*
                  *?                                                                 ?*
                  *?      DISCON  -  Updates 14489 Config register via SCI           ?*
                  *?                                                                 ?*
                  *?????????????????????????????????????????????????????????????????*


000010ca          DISCON  EQU     *
000010ca  b661            LDA     DISPC            Only update config register if it has
000010cc  b165            CMP     OLDDC            changed since last time.
000010ce  2601            BNE     UPDCON
000010d0  81              RTS

000010d1  a601    UPDCON  LDA     #$01
000010d3  b70e            STA     SCCR1            Clock idle low, edge in mid data, last clk.
000010d5  4a              DECA
000010d6  b70d            STA     BAUD             131.072KHz baud with 4.19etc XTAL.
000010d8  a608            LDA     #$08
000010da  b70f            STA     SCCR2            Transmit enabled.
000010dc  0d10fd  PREAM1  BRCLR   TC,SCSR,PREAM1   Wait for preamble to finish.
000010df  1502    DOCONF  BCLR    2,PORTC          Enable transmission to 14489.
000010e1  b661            LDA     DISPC
000010e3  b765            STA     OLDDC
000010e5  b711            STA     SCDAT
000010e7  0f10fd  DWAIT4  BRCLR   TDRE,SCSR,DWAIT4  Wait until config byte has transfered.
000010ea  a600            LDA     #$00
000010ec  b70f            STA     SCCR2            Now disable SCI transmission.
000010ee  0d10fd  DWAIT5  BRCLR   TC,SCSR,DWAIT5   Wait until config byte has actually gone.
000010f1  1402            BSET    2,PORTC          Disable 14489 & return.
000010f3  81              RTS


                  ************************************************************************
                  *                                                                      *
                  *               VECTOR ADDRESSES                                       *
                  *                                                                      *
                  ************************************************************************
                            SECTION .VECT,ADDR=$1FF2

00001ff2  0f00    SCIINT  FDB     RESET
00001ff4  0fcf    TOVFLW  FDB     TOVINT
00001ff6  0f00    TOCMP   FDB     RESET
00001ff8  0f00    TICAP   FDB     RESET
00001ffa  0f00    EXTINT  FDB     RESET
00001ffc  0f00    SOFTI   FDB     RESET
00001ffe  0f00    POR     FDB     RESET
```

```
                    Section synopsis

  1 00000016 (          22) .RAM
  2 00000010 (          16) .PAGE0
  3 000001f4 (         500) .USROM
  4 0000000e (          14) .VECT

                    Symbol table

.PAGE0  2 00000000 | DISPC   1 00000061 | LUPBCD   3 00001059 | OLDD3   1 00000064 | TEMP    1 0000005b
.RAM    1 00000000 | DOCONF  3 000010df | LUPDIS1  3 00000f83 | OLDDC   1 00000065 | TEMP1   1 0000005c
.USROM  3 00000000 | DWAIT1  3 000010ae | LUPDIS2  3 00000f9b | POR     4 00001ffe | TEMP2   1 0000005d
.VECT   4 00000000 | DWAIT2  3 000010b7 | MINS     1 00000052 | PREAM   3 000010a3 | TICAP   4 00001ff8
ADLUP1  3 0000100f | DWAIT3  3 000010c0 | MODE     1 00000055 | PREAM1  3 000010dc | TICKS   1 00000050
ADTAB   2 00000020 | DWAIT4  3 000010e7 | MUL1P8   3 00000f5e | SCIINT  4 00001ff2 | TIMINT  3 00000f1a
BIN0    1 00000056 | DWAIT5  3 000010ee | NEWTMP   1 0000005a | SECS    1 00000051 | TOCMP   4 00001ff6
CONT1   3 00001008 | EXTINT  4 00001ffa | NOIMP    3 00000f2f | SETAD   3 0000100a | TOVFLW  4 00001ff4
DEC0    1 00000059 | FLAG    1 00000054 | NOINC    3 00000ffa | SOFTI   4 00001ffc | TOVINT  3 00000fcf
DEC1    1 00000058 | FORCE   3 00000fb9 | NONEG    3 00000f57 | STD1    3 00000f96 | TRANGE  3 0000104f
DEC2    1 00000057 | GODISP  3 00000f43 | NONEG1   3 00000f71 | STDI3   3 00000fc8 | TSTD1   3 0000106b
DECCX   3 00001018 | GOMETR  3 00000f3a | NOOVF    3 00000ffc | STDIS3  3 00000fac | TSTD2   3 00001075
DISP1   1 0000005e | GOMORE  3 00000f41 | NOOVR    3 00001081 | STDISC  3 00000fb6 | TSTNEG  3 00000f91
DISP2   1 0000005f | HRS     1 00000053 | OLDD1    1 00000062 | TABL    3 00001023 | UPDATE  3 00001098
DISP3   1 00000060 | INIRAM  3 00000f0c | OLDD2    1 00000063 | TEMP    1 0000005b

                    Symbol cross-reference

.PAGE0   *131
.RAM     *94
.USROM   *142
.VECT    *498
ADLUP1   *334   334   339
ADTAB    *133   351   357
BIN0     *111   183   392
CONT1     327  *330
DEC0     *114   255   387   393   396   399
DEC1     *113   242   388   394   400   401   404
DEC2     *112   246   266   389   395   405   406   410
DECCX     336  *338
DISP1    *121   252   272   424   451
DISP2    *122   260   274   427   447
DISP3    *123   264   278   430   443
DISPC    *124   269   280   469   482
DOCONF   *481
DWAIT1   *446   446
DWAIT2   *450   450
DWAIT3   *454   454
DWAIT4   *485   485
DWAIT5   *488   488
EXTINT   *504
FLAG     *102   170   171   175   182   199   204   222   224   237   250   298   325   376   411
FORCE     237  *271
GODISP   *184
GOMETR    177  *179
GOMORE    180  *183
HRS      *100   308   311
INIRAM   *151   154
LUPBCD   *392   413
LUPDIS1  *242   245
LUPDIS2  *255   258
MINS     *99    303   304   307
MODE     *108   172   174   177   262   276
```

```
MUL1P8    201   203  *207
NEWTMP   *116   179   197   226   374
NOIMP     173  *175
NOINC     295   301   306   310  *313
NONEG     198  *202
NONEG1    222  *224
NOOVF     291  *314
NOOVR     408  *412
OLDD1    *125   425   452
OLDD2    *126   428   448
OLDD3    *127   431   444
OLDDC    *128   150   470   483
POR      *506
PREAM    *441   441
PREAM1   *480   480
SCIINT   *500
SECS     *98    297   299   302
SETAD     329  *331
SOFTI    *505
STD1      249   250  *252
STDI3     276  *278
STDIS3    262  *264
STDISC    267  *269
TABL     *346
```

Symbol cross-reference

```
TEMP     *117   326   337   340   342   344   359
TEMP1    *118   355   372
TEMP2    *119   358   361   371   373
TICAP    *503
TICKS    *97    292   293   296
TIMINT   *161
TOCMP    *502
TOVFLW   *501
TOVINT   *291   501
TRANGE    352   354  *376
TSTD1     398  *401
TSTD2     403  *406
TSTNEG    247  *250
UPDATE    426   429   432  *435
UPDCON    471  *474
```


Literature Distribution Centers:

USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.
EUROPE: Motorola Ltd.; European Literature Center; 88 Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.
JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141 Japan.
ASIA-PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong.

MOTOROLA

A30048-0 PRINTED IN USA 3/91 IMPERIAL LITHO 77974 5,000 ASIC YNAROO