

## AN477

# Simple A/D for MCUs without built-in A/D converters

By Åke Webjörn,  
Motorola AB,  
Sweden

### 1.0 Introduction

Non-critical measurement of resistance is needed in many applications. Examples are temperature, light, pressure and position measurements using devices where the sensor is a variable resistance. Those measurements can be made at minimal cost using existing MCUs, if a simple A/D-converter is added.

This application note describes a method of measuring an unknown resistance with an MC68HC05 type MCU that does not have a built-in analog-to-digital converter. Both the theoretical and the practical aspects of the method are covered.

The method requires two digital I/O lines on the MCU, one external capacitor, and two known and one unknown external resistors. The values of the external components can be selected for the desired performance. In the example outlined below, typically 6 bits of resolution can be achieved. The size of the entire program is about 450 bytes, with the actual A/D-conversion taking 250 bytes.

The MCU used in the example is the MC68HC705J2. For more information on this device, see the technical reference manual, MC68HC705J2/D.

### 2.0 Background

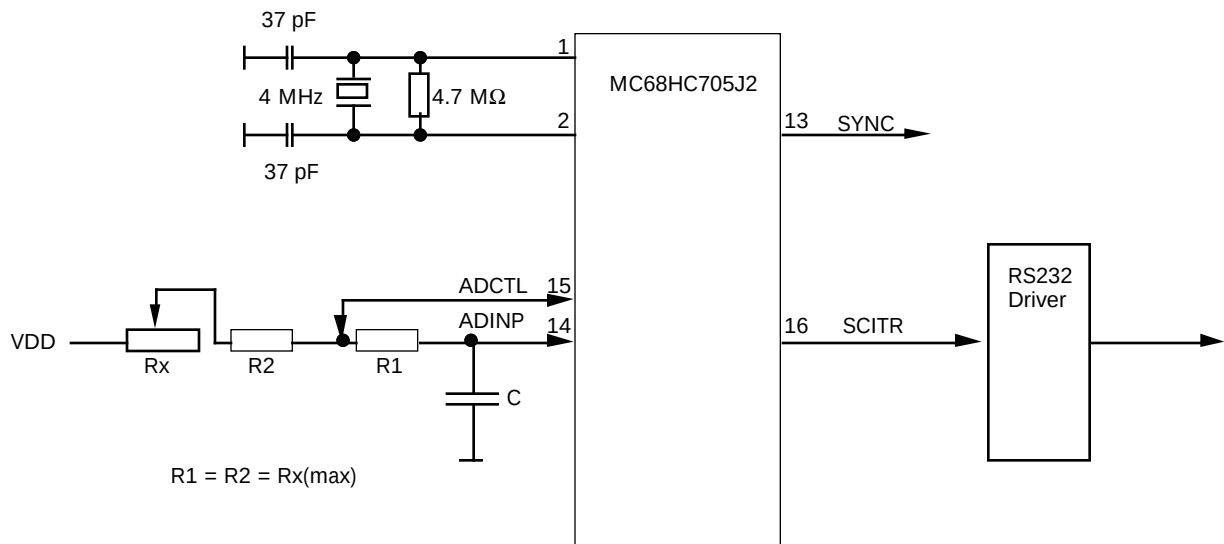
If the MCU had a digital input port with an accurately defined threshold level between high and low voltage, then the task of measuring an external resistance would be quite simple. As this is not the case, it is necessary to use an extra calibration cycle where a fixed resistor is used instead of an accurate reference voltage. In this method the unknown resistor is compared with the known one. Thus a reference resistance is measured and the calibration value is stored. Then, the actual measurement is made and finally, the result is modified according to the calibration value.

#### 2.1 System design

See [Figure 1](#) for details of the system schematic.

- There are four I/Os used on the MCU.
- ADCTL is used both as an output and as an input.
- ADINP is always a high impedance input.
- Two outputs are used for debugging the system.
- SCITR is used to transmit the result serially.
- SYNC is a triggered signal for an external oscilloscope.





**Figure 1** Block diagram

As stated above, the conversion is divided into a calibration phase and a measurement phase, as shown in [Figure 2](#).

#### *Calibration phase*

The calibration phase finds the  $R1 \cdot C$  time constant. This is done by first discharging the external capacitor C by pulling ADCTL low. Notice that the resistors  $R_x + R2$  have no effect at all at this time.

#### *Measurement phase*

Then ADCTL is pulled high towards Vdd to charge the capacitor C. The time,  $T_c$  depending on  $R1 \cdot C$ , is measured to reach the threshold voltage,  $V_{ref}$ , of ADINP. When the input signal passes the threshold level, the elapsed time is measured and stored. In other words the calibration time is:

$$T_c = R1 \cdot C \cdot \ln(V_{dd}/(V_{dd} - V_{ref}))$$

After the measurement, ADCTL is pulled low to discharge the capacitor C again.

The second measurement step makes the ADCTL go high impedance. The capacitor C immediately starts to charge. The time taken is given by the formula:

$$T_m = (R1 + R2 + R_x) \cdot C \cdot \ln(V_{dd}/(V_{dd} - V_{ref}))$$

The time to reach the threshold voltage is measured again.

To make things simple we shall set  $R1$  and  $R2$  each equal to  $R$ .

$$T_m = (R + R + R_x) \cdot C \cdot \ln(V_{ref}/(V_{ref} - V_r))$$

The final result is calculated to get correct scaling.

$$T_f = T_m - 2 \cdot T_c$$

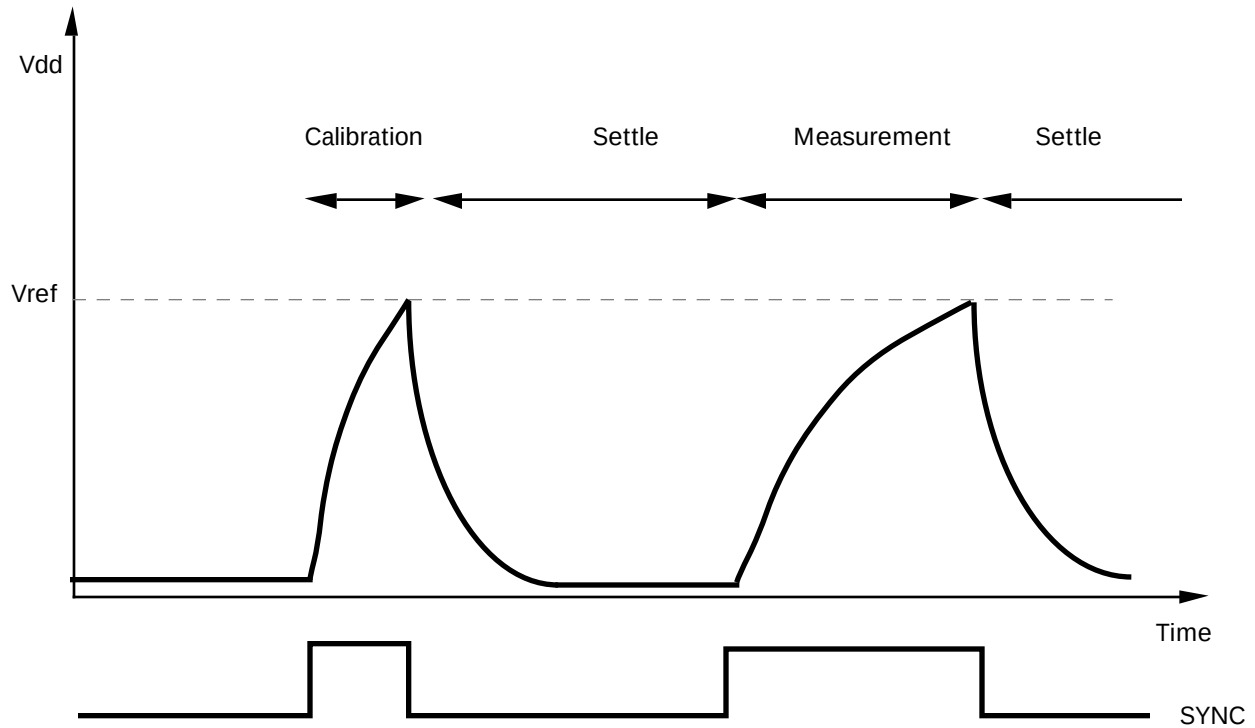
Substitute with the values calculated before and replace the  $\ln(V_{ref}/(V_{ref} - V_r))$  with K.

$$T_f = ((2 * R + R_x) - 2 * R) * K$$

$$T_f = R_x * K$$

Which gives the final result

$$R_x = T_f/K$$



**Figure 2** Signal waveform

## 2.2 Assumptions

The conversion method is based on the following assumptions:

- The I/O lines can be changed from input to output under software control.
- The digital inputs are high impedance with low leakage current.
- The digital output will source and sink current very close to the supply voltage and ground.
- The MCU is fast enough to handle calculations in the background.

## 3.0 Practical design

The MC68HC705J2 device is available in a 20-pin package. It has a built-in timer system and two 8-bit I/O ports. The timer system comprises a free-running 8-bit timer with a readable Timer Control Register (TCR). So if a 4 MHz crystal is used, the TCR runs at 500 kHz. As the counter is 8-bits long, it will overflow about every 5 mS. By using a Timer Overflow Interrupt (TOI) it is easy to let the program extend the counter to 16-bits. This gives the software timer register an overflow value of 1.28 S.

Two I/O pins, ADCTL and ADINP, are used for the resistance measurement. See [Figure 1](#).

The third pin, SCITR, is used to transmit the measured values on a 9600 baud RS232 line to an external computer.

Pin four is an I/O pin that is used to output a SYNC pulse which can be used to examine the accuracy of the system.

### 3.1 Component selection

We can now calculate realistic values for the components.

To avoid overflow in the timer register, the external RC-timing network must be limited to the maximum time. The RC-equation is as follows:

$$(R_x + R_1 + R_2) * C * \ln(V_{dd}/(V_{dd}-V_{ref})) < T_{max} \text{ mS}$$

To make things a bit simpler, use the value R for the maximum resistance value. Also use maximum Vref and Tmax values. In other words:

$$R_x \leq R_1 = R_2 = R, V_{ref} = 0.7 * V_{dd} \text{ and } T_{max} = 1.28 \text{ S}$$

gives the values with

$$3 * RC * \ln(V_{dd}/(V_{dd}-0.7V_{dd})) = 3 * RC * \ln(3.33) =$$

$$3.6 * RC < 1.28 \text{ S}$$

$$RC < 355 \text{ mS}$$

Two limits apply for the selection of the resistors R.

The output impedance of the I/O port should be much smaller than the load resistance R. But if R is made too large then leakage current from the input port will affect the result. From [Figure 3](#) it can be seen that the MCU will easily source in excess of 1 mA close to the power and ground rails.

R should then be larger than :

$$V_{cc}/I_{cc} > 5 \text{ V}/1 \text{ mA} = 5 \text{ K}\Omega.$$

Let's set R to four times as large, or 20 KΩ.

The maximum leakage current from an I/O pin is specified to be less than 10 μA. With a 20 KΩ resistor, the induced error is maximum 0.2 V. This means that C should be in the range

$$C < 355 \text{ mS}/R = 355 \text{ mS}/20 \text{ K}\Omega = 17.8 \mu\text{F}$$

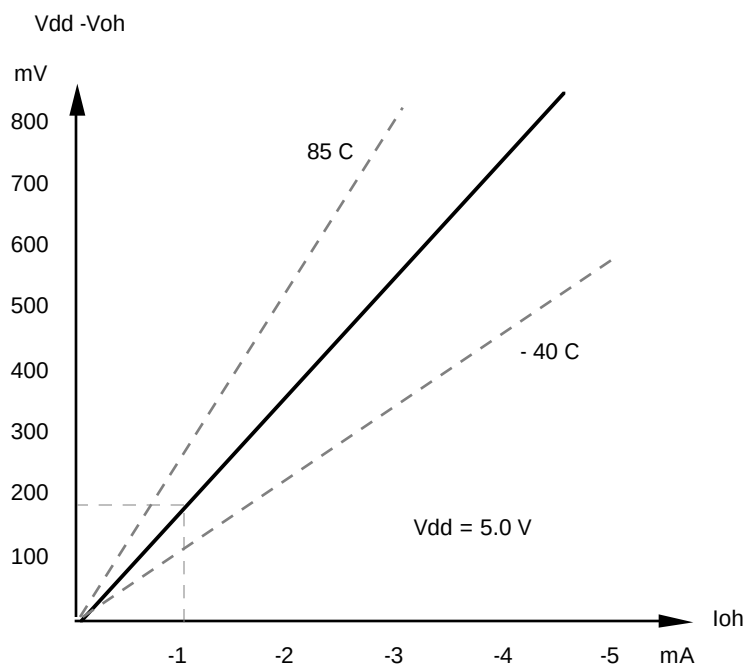
# Freescale Semiconductor, Inc.

If the counter uses only the lowest 8-bits to count, we have:

$$17.8 \mu\text{F}/256 < C < 17.8 \mu\text{F}$$

$$0.07 \mu\text{F} < C < 17.8 \mu\text{F}$$

Note that capacitor C can vary a lot, but if it is small, the time measurement becomes more critical if high resolution is required. So, for the following experiments, a 0.22  $\mu\text{F}$  capacitor was selected.



**Figure 3** Typical high driver characteristic

## 3.2 Software

The software is written to be easy to understand. It is divided into three modules:

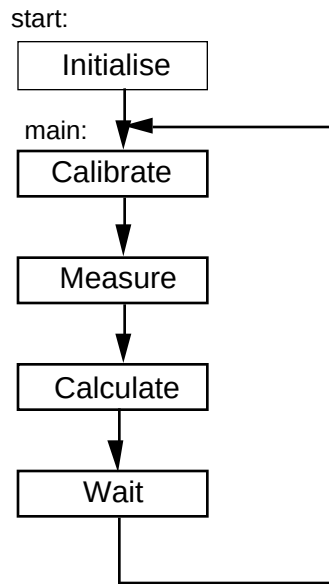
- MAC05.ASM      Macro routines
- UTIL05.ASM    General utilities
- RESADC.ASM    Main program

MAC05.ASM contains a set of simple macros for handling 16-bit calculations. The code can easily be optimised for speed or for code density.

UTIL05.ASM consists of a couple of subroutines. They are used by the debugging part of the program.

RESADC.ASM contains the main loop and the specific routines used.

The flow of the program is shown in [Figure 4](#).



**Figure 4** Program flow

At power up the MCU starts to execute at the label 'start'. After a few instructions to set up the peripherals, the program enters the 'main' loop.

First, it calls 'calib' to calibrate the system with the known external resistor R1. This is done by pulling ADCTL high and measuring the time until ADINP goes high. The time is stored in the RAM cell called 'ref'.

```

calib:      Settlelo()
            Clrtime()
            ADCTL = 1
            Loop until ADINP = 1
            Gettime(ref)
  
```

Second, it does a 'measur' to get the unknown value Rx. First ADCTL is pulled into high-impedance state. After a while ADINP will change. The time until this change is called 'res' and represents R1 + R2 + Rx.

```

measur:     Settlelo()
            Clrtime()
            Inport(ADCTL)
            Loop until ADINP = 1
            Gettime(res)
  
```

Third, it calculates the result in the routine 'calc' and outputs the final value.

So the 'calc' routine begins with scaling the result:

```

calc:      res = res/2 - ref
  
```

Then it applies automatic gain control by shifting *ref* and *res* until the most significant bit is set in *ref*.

```

Repeat
    shl16 res
    shl16 ref
Until MSB in ref is set
  
```

# Freescale Semiconductor, Inc.

Divide by 256 and multiply with the fullscale factor. In the program example *fullscale* is set to 100.

```
res = res/256 * fullscale
```

To obtain *final* result, divide by *ref*.

```
final = res/ref
```

Now let a macro display the result.

```
msgdec(final, finmsg)
```

The result is as shown in [Figure 5](#).

```
delay(100)
jmp      main
```

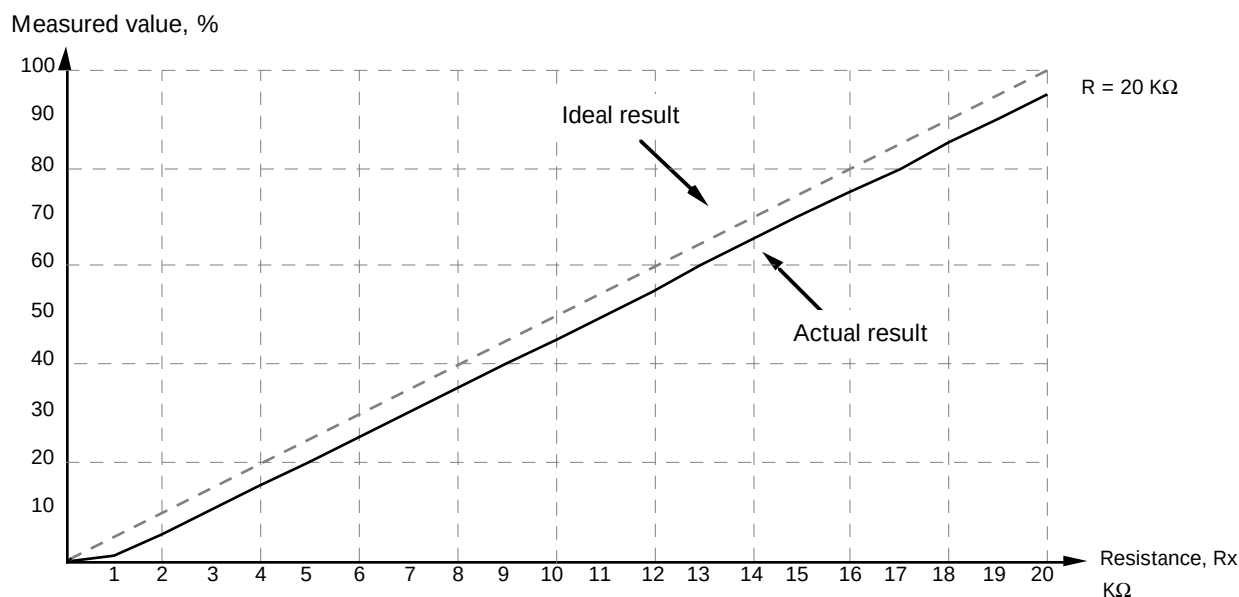
At the end there is a short delay before the program restarts. The complete program is given in listing (1).

```
REF = 053B  RES = 0B50  FINAL = 015
REF = 053B  RES = 0B51  FINAL = 015
REF = 053C  RES = 0B51  FINAL = 015
```

**Figure 5** Output example

## 3.3 Performance

[Figure 6](#) shows the typical performance of the system. The conversion is quite linear in the 6- to 7-bit range and there is an offset error of about 5 %. This offset error is not affected by the values of the external components. The reason is that the ADCTL pin will not go higher than typically 4.9 V during the calibration phase which gives this offset value.



**Figure 6** Measurement of variable resistance, Rx

## 3.4 Suggested improvements

Below we outline two methods for dealing with the problem of the offset voltage.

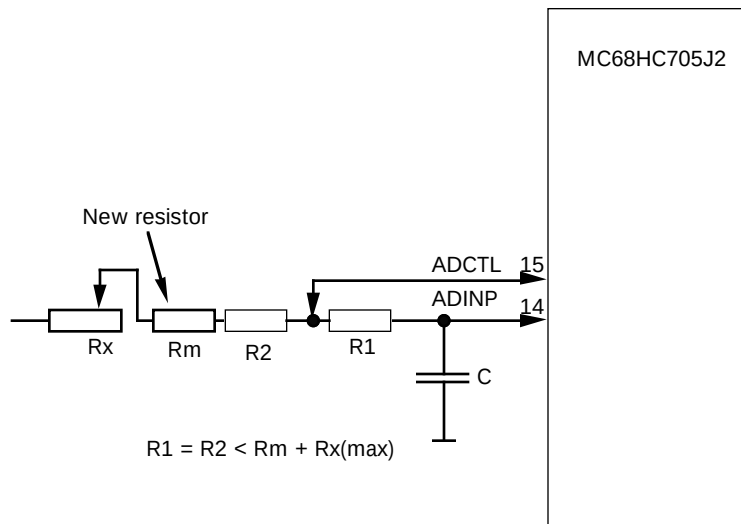
### 3.4.0 Improvement A

Let's see what happens when a small interior resistor,  $R_c$  is included in the output calibration driving circuitry of the MC68HC705J2.

$$T_c = (R + R_c) * C * \ln(V_{dd}/(V_{dd}-V_{ref}))$$

To eliminate  $R_c$ , a resistor  $R_m$  is included in the measuring circuitry. See [Figure 7](#).

$$T_m = (R + R + R_x + R_m) * C * \ln(V_{dd}/(V_{dd}-V_{ref}))$$



**Figure 7** Block diagram with adjusted source impedance

Then put the result in the following equation and replace the timing figure with K.

$$T_f = T_m - 2 * T_c$$

$$T_f = ((2 * R + R_x + R_m) - 2(R + R_c)) * K$$

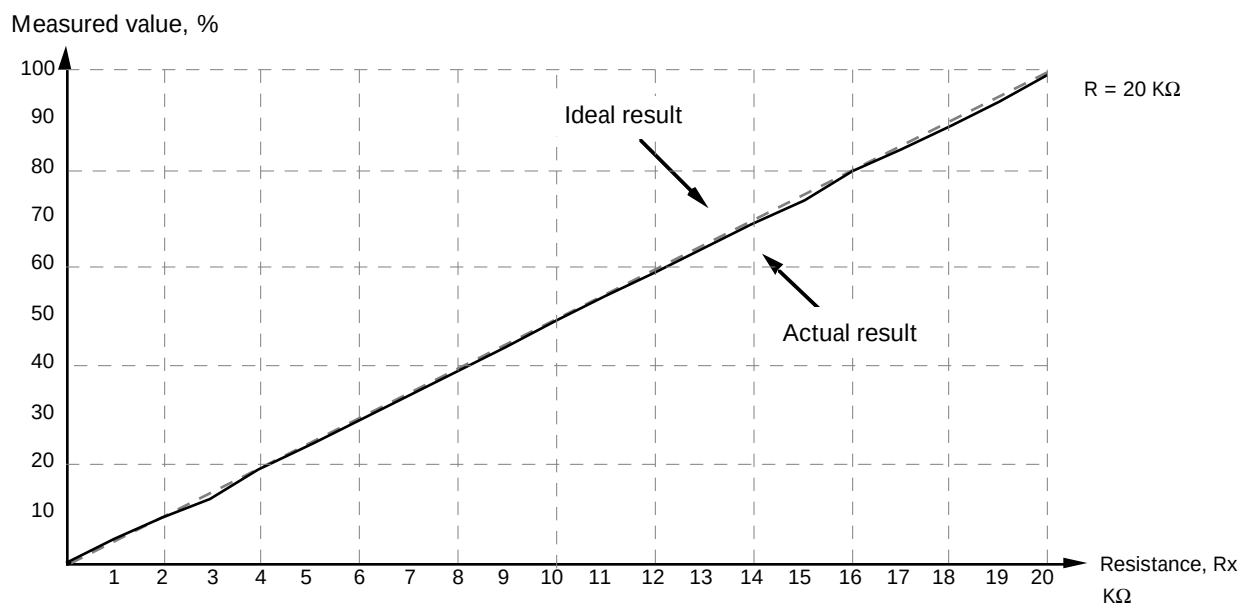
$$T_f = (R_x + R_m - 2 * R_c) * K$$

To eliminate the effect of  $R_c$  then

$$R_m = 2 * R_c$$

To find the resistance  $R_c$ , program the pin ADCTL high and connect an external resistance to ground. It turns out that the internal resistance  $R_c$  is between 0.2 and 0.35 K $\Omega$ . This means that a small  $2 * R_c$ , or 0.4 to 0.7 K $\Omega$  resistor  $R_m$  is inserted in series with  $R_x$ . See [Figure 8](#) for the result.

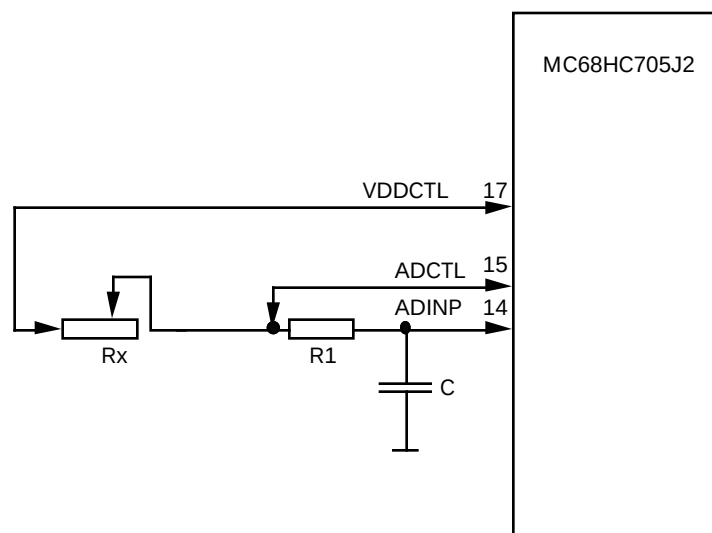




**Figure 8** Improvement A – extra external resistor,  $R_m = 680\Omega$

### 3.4.1 Improvement B

Another way to compensate for the drive characteristic of the output pin on the MC68HC705J2, is to use an extra I/O pin, VDDCTL, instead of  $V_{dd}$ . See Figure 9. VDDCTL is made smart so it is idle during calibration and active during measurement. Resistor R2 is not needed and is removed.



**Figure 9** Block diagram with built-in source impedance

The timing figures turns out as follows:

$$T_c = (R + R_c) * C * \ln(V_{dd}/(V_{dd}-V_{ref}))$$

To eliminate  $R_c$ , a resistor  $R_m$  is included in the output normally used to supply  $V_{dd}$ .

$$T_m = (R + R_x + R_m) * C * \ln(V_{dd}/(V_{dd}-V_{ref}))$$

Then put the result in the following equation and replace the timing figure with  $K$ .

$$T_f = T_m - T_c$$

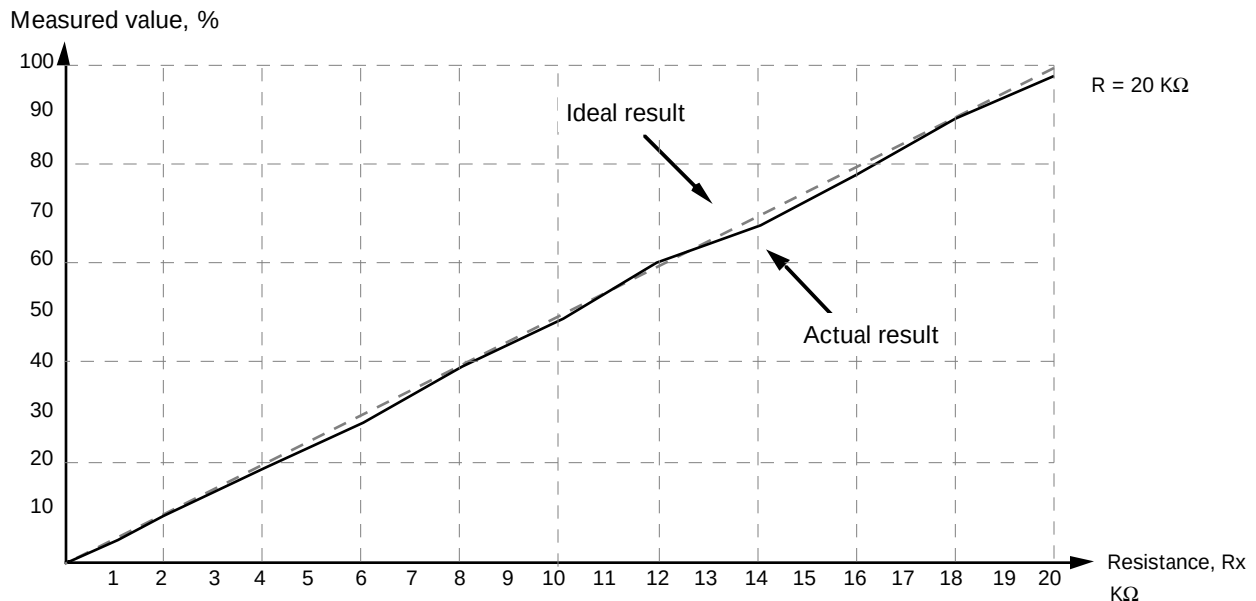
$$T_f = ((R + R_x + R_m) - (R + R_c)) * K$$

$$T_f = (R_x + R_m - R_c) * K$$

To eliminate the effect of  $R_c$  then

$$R_m = R_c$$

The new routine handles  $V_{DDCTL}$  and also takes care of the final result. It is called *RESADC1.ASM*. See program listing (2) and [Figure 10](#).



**Figure 10** Improvement B – use of  $V_{ddctl}$

## 4.0 Conclusion

This application note shows that it is quite simple to add A/D conversion to measure resistance to MC68HC05 microcomputer applications. Three solutions have been discussed, each of which has its own advantages. These may provide some new ideas on how to solve the A/D converter problem.

## Acknowledgements

The author acknowledges the help and assistance of his colleagues Peter McGinn, Ross Mitchell, Owe Mellberg and Berndt Lehnert.

## RESADC.LST

```

M6805 Portable Cross Assembler  0.05  MS-DOS/PC-DOS  Page 1
Sun Mar 15 20:56:45 1992
Command line:
C:\PASM05.EXE -d -l RESADC.lst RESADC.asm
Options list:
ON  - b - Printing of macro definitions
ON  - c - Printing of macro calls
ON  - d - Placing of symbolic debugging information in COFF  (changed)
OFF - e - Printing of macro expansions
ON  - f - Printing of conditional directives
OFF - g - Printing of generated constants list
OFF - q - Expanding and printing of structured syntax
OFF - s - Printing of symbol table
OFF - u - Printing of conditional unassembled source
OFF - x - Printing of cross reference table
OFF - m - Suppress printing of error messages
ON  - w - Printing of warning messages
OFF - v - Suppress printing of updated status
OFF - y - Enabling of sgs extensions
ON  - o - Create object code
ON  -   - Formatting of source line listing
Create listing file - l - RESADC.lst

Xdefs:
  NONE

Xrefs:
  NONE

Input file(s): RESADC.asm (250 lines)
               mac05.asm (63 lines) util05.asm (257 lines)

Output file: RESADC.o
Listing file: RESADC.lst

Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00001
00002                      OPT  LLE=132
00003                      OPT  MUL

```

—continued overleaf—

## Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 3  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00005
00001          include mac05.asm
00002          * Macro definition file for MC68HC05
00003 P 0000      0004    A DDR      EQU   4          * data direction register offset
00004
00005          MACRO DEF      inport    MACR
00006                      BCLR      \0,\1+DDR
00007                      ENDM
00008
00009          MACRO DEF      outport   MACR      ! make \0 an output pin
00010                      BSET      \0,\1+DDR
00011                      ENDM
00012
00013          MACRO DEF      add16     MACR      !\0:=\1+\2
00014                      LDA       \1+1
00015                      ADD       \2+1
00016                      STA       \0+1
00017                      LDA       \1
00018                      ADC       \2
00019                      STA       \0
00020                      ENDM
00021
00022          MACRO DEF      sub16     MACR      ! \0:=\1-\2
00023                      LDA       \1+1
00024                      SUB       \2+1
00025                      STA       \0+1
00026                      LDA       \1
00027                      SBC       \2
00028                      STA       \0
00029                      ENDM
00030
00031          MACRO DEF      cmp16     MACR      ! \0 > \1 ?
00032                      LDA       \0+1
00033                      SUB       \1+1
00034                      LDA       \0
00035                      SBC       \1
00036                      ENDM
00037
00038          MACRO DEF      shl16     MACR
00039                      LSL       \0+1
00040                      ROL       \0
00041                      ENDM
00042
00043          MACRO DEF      shr16     MACR
00044                      LSR       \0
00045                      ROR       \0+1
00046                      ENDM
00047
00048          MACRO DEF      msghex    MACR
00049                      LDX       #\1-msg
00050

```

M6805 Portable Cross Assembler 0.05 mac05.asm Page 4  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00051          JSR      xmitmsg
00052          LDA       \0
00053          JSR      tohex
00054          LDA       \0+1
00055          JSR      tohex
00056          ENDM
00057
00058          MACRO DEF      msgdec    MACR
00059                      LDX       #\1-msg
00060                      JSR      xmitmsg
00061                      LDA       \0
00062                      JSR      todec
00063          ENDM

```

—continued overleaf—

# Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 5  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S | LABEL     | MNEMO | OPERANDS | COMMENT                                 |
|-------|---|------|------|----------|---|-----------|-------|----------|-----------------------------------------|
| 00007 |   |      |      |          |   |           |       |          | *=====                                  |
| 00008 |   |      |      |          |   |           |       |          | * Resistance measurement for HC705J2    |
| 00009 |   |      |      |          |   |           |       |          | *=====                                  |
| 00010 |   |      |      |          |   |           |       |          |                                         |
| 00011 | P | 0000 |      | 0064     | A | fullscale | EQU   | 100      | * this determines the full scale result |
| 00012 |   |      |      |          |   |           |       |          |                                         |
| 00013 | A | 0000 |      |          |   |           | ORG   | \$0      |                                         |
| 00014 | A | 0000 |      |          | A | porta     | RMB   | 1        |                                         |
| 00015 | A | 0001 |      | 0001     | A | pullup    | EQU   | 1        | * pa1 pull up                           |
| 00016 | A | 0001 |      | 0002     | A | scitr     | EQU   | 2        | * pa2 sci transmitter                   |
| 00017 | A | 0001 |      | 0003     | A | adctl     | EQU   | 3        | * pa3 potentiometer control             |
| 00018 | A | 0001 |      | 0004     | A | adinp     | EQU   | 4        | * pa4 potentiometer input               |
| 00019 | A | 0001 |      | 0005     | A | sync      | EQU   | 5        | * pa5 for oscilloscope sync             |
| 00020 |   |      |      |          |   |           |       |          |                                         |
| 00021 | A | 0001 |      | 01       | A | portb     | RMB   | 1        |                                         |
| 00022 | A | 0002 |      | 0002     | A | key       | EQU   | 2        | * key                                   |
| 00023 |   |      |      |          |   |           |       |          |                                         |
| 00024 |   |      |      |          |   |           |       |          |                                         |
| 00025 |   |      |      |          |   |           |       |          | * Timer registers                       |
| 00026 | A | 0008 |      |          |   |           | ORG   | \$8      |                                         |
| 00027 | A | 0008 |      | 01       | A | tcsr      | RMB   | 1        | * Timer Count Status Register           |
| 00028 | A | 0009 |      | 0007     | A | tof       | EQU   | 7        | * Timer Overflow Flag                   |
| 00029 | A | 0009 |      | 0006     | A | rtif      | EQU   | 6        | * Real Time Interrupt Flag              |
| 00030 | A | 0009 |      | 0005     | A | tofe      | EQU   | 5        | * Timer OverFlow Enable                 |
| 00031 | A | 0009 |      | 0004     | A | rtie      | EQU   | 4        | * Real Time Interrupt Enable            |
| 00032 |   |      |      |          |   |           |       |          |                                         |
| 00033 | A | 0009 |      | 01       | A | tcr       | RMB   | 1        | * Timer Counter Register                |
| 00034 |   |      |      |          |   |           |       |          |                                         |
| 00035 |   |      |      |          |   |           |       |          | *=====                                  |
| 00036 |   |      |      |          |   |           |       |          | * Start of RAM area                     |
| 00037 |   |      |      |          |   |           |       |          | *=====                                  |
| 00038 | A | 0090 |      |          |   |           | ORG   | \$90     |                                         |
| 00039 | A | 0090 |      | 01       | A | time      | RMB   | 1        | * elapsed time in 0.5 mS div            |
| 00040 | A | 0091 |      | 02       | A | ref       | RMB   | 2        | * time for rising signal to pass limit  |
| 00041 | A | 0093 |      | 02       | A | res       | RMB   | 2        | * measured result                       |
| 00042 | A | 0095 |      | 01       | A | final     | RMB   | 1        | * 8 bits final result                   |

—continued overleaf—

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 7  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00044
00001          include util05.asm
00002          * Utility program for MC68HC05
00003          * SCI data
00004 A 0096      01      A bitcount RMB 1      * bit counter for transmit
00005 A 0097      01      A char RMB 1      * tmp storage for transmit
00006 A 0098      01      A hex RMB 1      * tmp storage for tohex
00007 A 0099      01      A decimal RMB 1      * tmp storage for todec
00008 A 009a      01      A strptr RMB 1      * string pointer
00009
00010
00011          *=====
00012          *          Symbolic absolute values
00013          *=====
00014 A 009b      0005      A rc EQU 5      * RC time constant in mS
00015 A 009b      001e      A del96 EQU 30      * bitwait for 9600 baud
00016 A 009b      003c      A stopbit EQU 2*del96      * two stop bits
00017 A 009b      000d      A cr EQU $0D
00018 A 009b      000a      A lf EQU $0A
00019
00020          *=====
00021          *          Start of ROM area
00022          *=====
00023
00024 A 0700          ORG $700
00025
00026          *=====
00027          *          Real time interrupt
00028          *=====
00029          * Function: increments the variable time at each interrupt
00030          * Input: none
00031          * Uses: time
00032          * Output: none
00033          * Note: runs every 2 mS with a 4 MHz clock
00034 A 0700 1f 08      A realt BCLR tof,tsr *clear tof flag
00035 A 0702 3c 90      A INC time * time:=time+1
00036 A 0704 80          RTI
00037
00038          *=====
00039          *          Bitwait delay routine
00040          *=====
00041          * Function: Delay for asynchronous transmission
00042          * Input: delay in reg A
00043          * Uses: none
00044          * Output: none
00045          * Note: bitwait formula: bitwait = 32 + 6 * A cycles
00046          *          bit time for 9600 baud is 104 µS or 208 cycles at 4 MHz
00047          *          A = 30 gives a bit time of 106 µS, or an error of < 2%
00048          *          minimum baudrate is about 1300 baud
00049
00050 A 0705 a6 1e      A bitwait LDA #del96 2 cycles

```

—continued overleaf—

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 util05.asm Page 8  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S    | LABEL    | MNEMO | OPERANDS    | COMMENT        |
|-------|---|------|------|----------|------|----------|-------|-------------|----------------|
| 00051 |   |      |      |          |      |          |       |             |                |
| 00052 |   |      |      |          |      | bitwait1 |       |             |                |
| 00053 | A | 0707 | 4a   |          |      |          | DECA  |             | 3 cycles       |
| 00054 | A | 0708 | 26   | fd       | 0707 |          | BNE   | bitwait1    | 3              |
| 00055 | A | 070a | 81   |          |      |          | RTS   |             | 6              |
| 00056 |   |      |      |          |      |          |       |             |                |
| 00057 |   |      |      |          |      |          |       |             |                |
| 00058 |   |      |      |          |      |          |       |             |                |
| 00059 |   |      |      |          |      |          |       |             |                |
| 00060 |   |      |      |          |      |          |       |             |                |
| 00061 |   |      |      |          |      |          |       |             |                |
| 00062 |   |      |      |          |      |          |       |             |                |
| 00063 |   |      |      |          |      |          |       |             |                |
| 00064 |   |      |      |          |      |          |       |             |                |
| 00065 |   |      |      |          |      |          |       |             |                |
| 00066 |   |      |      |          |      |          |       |             |                |
| 00067 | A | 070b | b7   | 97       | A    |          | STA   | char        |                |
| 00068 | A | 070d | a6   | 09       | A    |          | LDA   | #9          |                |
| 00069 | A | 070f | b7   | 96       | A    |          | STA   | bitcount    |                |
| 00070 | A | 0711 | 15   | 00       | A    |          | BCLR  | scitr,porta | send start bit |
| 00071 |   |      |      |          |      |          |       |             |                |
| 00072 |   |      |      |          |      |          |       |             |                |
| 00073 | A | 0713 | ad   | f0       | 0705 | tra3     | BSR   | bitwait     | 6              |
| 00074 | A | 0715 | 3a   | 96       | A    |          | DEC   | bitcount    | 5              |
| 00075 | A | 0717 | 27   | 0c       | 0725 |          | BEQ   | tra2        | 3              |
| 00076 | A | 0719 | 36   | 97       | A    |          | ROR   | char        | 5              |
| 00077 | A | 071b | 25   | 04       | 0721 |          | BCS   | tra1        | 3              |
| 00078 |   |      |      |          |      |          |       |             |                |
| 00079 |   |      |      |          |      |          |       |             |                |
| 00080 | A | 071d | 15   | 00       | A    |          | BCLR  | scitr,porta | send 0         |
| 00081 | A | 071f | 20   | f2       | 0713 |          | BRA   | tra3        |                |
| 00082 |   |      |      |          |      |          |       |             |                |
| 00083 | A | 0721 | 14   | 00       | A    | tra1     | BSET  | scitr,porta | or send 1      |
| 00084 | A | 0723 | 20   | ee       | 0713 |          | BRA   | tra3        |                |
| 00085 |   |      |      |          |      |          |       |             |                |
| 00086 | A | 0725 | 14   | 00       | A    | tra2     | BSET  | scitr,porta | send stop bit  |
| 00087 | A | 0727 | ad   | dc       | 0705 |          | BSR   | bitwait     |                |
| 00088 | A | 0729 | ad   | da       | 0705 |          | BSR   | bitwait     |                |
| 00089 | A | 072b | 81   |          |      |          | RTS   |             |                |
| 00090 |   |      |      |          |      |          |       |             |                |
| 00091 |   |      |      |          |      |          |       |             |                |
| 00092 |   |      |      |          |      |          |       |             |                |
| 00093 |   |      |      |          |      |          |       |             |                |
| 00094 |   |      |      |          |      |          |       |             |                |
| 00095 |   |      |      |          |      |          |       |             |                |
| 00096 |   |      |      |          |      |          |       |             |                |
| 00097 |   |      |      |          |      |          |       |             |                |
| 00098 |   |      |      |          |      |          |       |             |                |
| 00099 |   |      |      |          |      |          |       |             |                |
| 00100 |   |      |      |          |      |          |       |             |                |
| 00101 | A | 072c |      | 20       | A    | refmsg   | FCC   | ' REF = '   |                |
| 00102 | A | 0733 |      | 00       | A    |          | FCB   | 0           |                |

—continued overleaf—

## Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 util05.asm Page 9  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S    | LABEL    | MNEMO | OPERANDS    | COMMENT                   |
|-------|---|------|------|----------|------|----------|-------|-------------|---------------------------|
| 00103 | A | 0734 | 20   |          | A    | resmsg   | FCC   | ' RES = '   |                           |
| 00104 | A | 073b | 00   |          | A    |          | FCB   | 0           |                           |
| 00105 | A | 073c | 20   |          | A    | t2himsg  | FCC   | ' T2HI = '  |                           |
| 00106 | A | 0744 | 00   |          | A    |          | FCB   | 0           |                           |
| 00107 | A | 0745 | 20   |          | A    | t2lomsg  | FCC   | ' T2LO = '  |                           |
| 00108 | A | 074d | 00   |          | A    |          | FCB   | 0           |                           |
| 00109 |   |      |      |          |      |          |       |             |                           |
| 00110 | A | 074e | 20   |          | A    | finmsg   | FCC   | ' FINAL = ' |                           |
| 00111 | A | 0757 | 00   |          | A    |          | FCB   | 0           |                           |
| 00112 |   |      |      |          |      |          |       |             |                           |
| 00113 | A | 0758 | 0d   |          | A    | nlmsg    | FCB   | cr          |                           |
| 00114 | A | 0759 | 0a   |          | A    |          | FCB   | lf          |                           |
| 00115 | A | 075a | 00   |          | A    |          | FCB   | 0           |                           |
| 00116 |   |      |      |          |      |          |       |             |                           |
| 00117 | A | 075b | bf   | 9a       | A    | xmitmsg  | STX   | strptr      |                           |
| 00118 |   |      |      |          |      |          |       |             |                           |
| 00119 |   |      |      |          |      | xmitmsg2 |       |             |                           |
| 00120 | A | 075d | d6   | 072c     | A    |          | LDA   | msg,X       | * get character           |
| 00121 | A | 0760 | 27   | 09       | 076b |          | BEQ   | xmitmsg1    | * done if 0               |
| 00122 | A | 0762 | cd   | 070b     | A    |          | JSR   | transmit    | * else send one character |
| 00123 | A | 0765 | 3c   | 9a       | A    |          | INC   | strptr      | * move pointer            |
| 00124 | A | 0767 | be   | 9a       | A    |          | LDX   | strptr      | * get pointer to X        |
| 00125 | A | 0769 | 20   | f2       | 075d |          | BRA   | xmitmsg2    |                           |
| 00126 | A | 076b | 81   |          |      | xmitmsg1 | RTS   |             |                           |
| 00127 |   |      |      |          |      |          |       |             |                           |
| 00128 |   |      |      |          |      |          |       |             |                           |
| 00129 |   |      |      |          |      |          |       |             |                           |
| 00130 |   |      |      |          |      |          |       |             |                           |
| 00131 |   |      |      |          |      |          |       |             |                           |
| 00132 |   |      |      |          |      |          |       |             |                           |
| 00133 |   |      |      |          |      |          |       |             |                           |
| 00134 |   |      |      |          |      |          |       |             |                           |
| 00135 |   |      |      |          |      |          |       |             |                           |
| 00136 | A | 076c | ae   | 2f       | A    | todec    | LDX   | #'0'-1      |                           |
| 00137 |   |      |      |          |      |          |       |             |                           |
| 00138 |   |      |      |          |      |          |       |             |                           |
| 00139 | A | 076e | 5c   |          |      |          |       |             |                           |
| 00140 | A | 076f | a0   | 64       | A    |          | SUB   | #100        |                           |
| 00141 | A | 0771 | 24   | fb       | 076e |          | BCC   | todec1      |                           |
| 00142 |   |      |      |          |      |          |       |             |                           |
| 00143 | A | 0773 | ab   | 64       | A    |          | ADD   | #100        | * adjust A                |
| 00144 | A | 0775 | b7   | 99       | A    |          | STA   | decimal     | * save what is left       |
| 00145 | A | 0777 | 9f   |          |      |          | TXA   |             |                           |
| 00146 | A | 0778 | cd   | 070b     | A    |          | JSR   | transmit    | * transmit 100:s          |
| 00147 |   |      |      |          |      |          |       |             |                           |
| 00148 | A | 077b | b6   | 99       | A    |          | LDA   | decimal     |                           |
| 00149 | A | 077d | ae   | 2f       | A    |          | LDX   | #'0'-1      |                           |
| 00150 | A | 077f | 5c   |          |      | todec2   | INCX  |             |                           |
| 00151 | A | 0780 | a0   | 0a       | A    |          | SUB   | #10         |                           |
| 00152 | A | 0782 | 24   | fb       | 077f |          | BCC   | todec2      |                           |
| 00153 | A | 0784 | ab   | 0a       | A    |          | ADD   | #10         |                           |
| 00154 | A | 0786 | b7   | 99       | A    |          | STA   | decimal     |                           |

—continued overleaf—



## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 util05.asm Page 10  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S    | LABEL  | MNEMO | OPERANDS           | COMMENT                                                            |
|-------|---|------|------|----------|------|--------|-------|--------------------|--------------------------------------------------------------------|
| 00155 |   |      |      |          |      |        |       |                    |                                                                    |
| 00156 | A | 0788 | 9f   |          |      |        | TXA   |                    |                                                                    |
| 00157 | A | 0789 | cd   | 070b     | A    |        | JSR   | transmit           | * transmit 10:s                                                    |
| 00158 |   |      |      |          |      |        |       |                    |                                                                    |
| 00159 | A | 078c | b6   | 99       | A    |        | LDA   | decimal            |                                                                    |
| 00160 | A | 078e | ae   | 2f       | A    |        | LDX   | #'0'-1             |                                                                    |
| 00161 | A | 0790 | 5c   |          |      | todec3 | INCX  |                    |                                                                    |
| 00162 | A | 0791 | a0   | 01       | A    |        | SUB   | #1                 |                                                                    |
| 00163 | A | 0793 | 24   | fb       | 0790 |        | BCC   | todec3             |                                                                    |
| 00164 | A | 0795 | 9f   |          |      |        | TXA   |                    |                                                                    |
| 00165 | A | 0796 | cd   | 070b     | A    |        | JSR   | transmit           | * transmit 1:s                                                     |
| 00166 |   |      |      |          |      |        |       |                    |                                                                    |
| 00167 | A | 0799 | a6   | 20       | A    |        | LDA   | #' '               |                                                                    |
| 00168 | A | 079b | cd   | 070b     | A    |        | JSR   | transmit           | * transmit ' '                                                     |
| 00169 | A | 079e | 81   |          |      |        | RTS   |                    |                                                                    |
| 00170 |   |      |      |          |      |        |       |                    |                                                                    |
| 00171 |   |      |      |          |      |        |       |                    | *=====                                                             |
| 00172 |   |      |      |          |      |        |       |                    | * Convert to hexadecimal                                           |
| 00173 |   |      |      |          |      |        |       |                    | *=====                                                             |
| 00174 |   |      |      |          |      |        |       |                    | * Function: Transmits byte as a 2 digit hexadecimal value          |
| 00175 |   |      |      |          |      |        |       |                    | * Input: A contains byte to convert                                |
| 00176 |   |      |      |          |      |        |       |                    | * Uses: hex                                                        |
| 00177 |   |      |      |          |      |        |       |                    | * Output: none                                                     |
| 00178 | A | 079f |      | 30       | A    | hexstr | FCC   | '0123456789ABCDEF' |                                                                    |
| 00179 |   |      |      |          |      |        |       |                    |                                                                    |
| 00180 | A | 07af | b7   | 98       | A    | tohex  | STA   | hex                | * save hex value                                                   |
| 00181 | A | 07b1 | 44   |          |      |        | LSRA  |                    | * shift right 4 times to get high nibble                           |
| 00182 | A | 07b2 | 44   |          |      |        | LSRA  |                    |                                                                    |
| 00183 | A | 07b3 | 44   |          |      |        | LSRA  |                    |                                                                    |
| 00184 | A | 07b4 | 44   |          |      |        | LSRA  |                    |                                                                    |
| 00185 | A | 07b5 | 97   |          |      |        | TAX   |                    | * put result in x                                                  |
| 00186 | A | 07b6 | d6   | 079f     | A    |        | LDA   | hexstr,x           | * translate to ASCII                                               |
| 00187 | A | 07b9 | cd   | 070b     | A    |        | JSR   | transmit           | * transmit result                                                  |
| 00188 |   |      |      |          |      |        |       |                    |                                                                    |
| 00189 | A | 07bc | b6   | 98       | A    |        | LDA   | hex                | * get hex value again                                              |
| 00190 | A | 07be | a4   | 0f       | A    |        | AND   | #\$F               | * mask low nibble                                                  |
| 00191 | A | 07c0 | 97   |          |      |        | TAX   |                    |                                                                    |
| 00192 | A | 07c1 | d6   | 079f     | A    |        | LDA   | hexstr,x           | * translate to ASCII                                               |
| 00193 | A | 07c4 | cd   | 070b     | A    |        | JSR   | transmit           | * transmit low nibble                                              |
| 00194 | A | 07c7 | 81   |          |      |        | RTS   |                    |                                                                    |
| 00195 |   |      |      |          |      |        |       |                    |                                                                    |
| 00196 |   |      |      |          |      |        |       |                    | *=====                                                             |
| 00197 |   |      |      |          |      |        |       |                    | * Delay routine                                                    |
| 00198 |   |      |      |          |      |        |       |                    | *=====                                                             |
| 00199 |   |      |      |          |      |        |       |                    | * Function: Uses the timer overflow interrupt to count time        |
| 00200 |   |      |      |          |      |        |       |                    | * Input: A contains desired delay in 0,5 mS increments             |
| 00201 |   |      |      |          |      |        |       |                    | * Uses: time                                                       |
| 00202 |   |      |      |          |      |        |       |                    | * Output: no registers are destroyed                               |
| 00203 |   |      |      |          |      |        |       |                    | * Note: because this routine is not synchronised with the realtime |
| 00204 |   |      |      |          |      |        |       |                    | * interrupt there can be 0,5 mS jitter.                            |
| 00205 |   |      |      |          |      |        |       |                    |                                                                    |
| 00206 |   |      |      |          |      |        |       |                    |                                                                    |

—continued overleaf—

## Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 util05.asm Page 11  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00207  A 07c8 3f  90      A delay  CLR  time
00208  A 07ca 9a          CLI          enable interrupts
00209  A 07cb b1  90      A delay1  CMPA  time  loop until A = time
00210  A 07cd 26  fc  07cb  BNE  delay1
00211  A 07cf 9b          SEI          turn off interrupts
00212  A 07d0 81          RTS
00213
00214
00215
00216  *=====
00217  * Get time
00218  *=====
00219  * Function: Delay for voltage on input capacitor to settle
00220  * Input: X points at 16 bit result
00221  * Uses: time
00222  * Output: result in [X]
00223  * Note: because the timer counter is clocked with 500 Hz
00224  * special adjustments to the result is needed
00225  * interrupts are disabled when leaving
00226  * The sync signal is for debugging only
00227
00228  A 07d1 1b  00      A gettime  BCLR  sync,porta * pull sync signal low
00229  A 07d3 9b          SEI          * stop interrupts
00230  A 07d4 b6  09      A          LDA  tcr  * read timer counter register
00231  A 07d6 0f  0803  07dc  BRCLR  tof,tcsr,gettime1 * if the tof is set
00232  A 07d9 3c  90      A          INC  time  * then adjust time
00233  A 07db 4f          CLRA          * and clear A
00234  gettime1
00235  A 07dc e701        STA  1,X  * store low result
00236  A 07de b6  90      A          LDA  time  * get high byte
00237  A 07e0 f7          STA  ,X  * and store it as well
00238  A 07e1 81          RTS
00239
00240  *=====
00241  * Synchronise with time
00242  *=====
00243  * Function: Returns when the timer counter overflows
00244  * Input: none
00245  * Uses: time
00246  * Output: none
00247  * Note: interrupts are enabled when leaving
00248  clrtime
00249  A 07e2 3f  90      A          CLR  time  * time:= 0
00250  A 07e4 1f  08      A          BCLR  tof,tcsr *clear tof
00251  A 07e6 0f  08fd  07e6  BRCLR  tof,tcsr,* *loop until tof is set
00252  A 07e9 1f  08      A          BCLR  tof,tcsr *clear tof
00253  A 07eb 1a  00      A          BSET  sync,porta * pull sync high
00254  A 07ed 9a          CLI          * enable interrupts
00255  A 07ee 81          RTS          * and return
00256
00257

```

—continued overleaf—

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 12  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMON OPERANDS COMMENT
00046                                     *=====
00047                                     * Settle measurement voltages
00048                                     *=====
00049                                     * Function: Delay for voltage on input capacitor to settle
00050                                     * Input: none
00051                                     * Uses: adctl, delay
00052                                     * Output: none
00053                                     * Note: three entry labels depending on desired function
00054                                     *       interrupts are disabled when leaving
00055                                     *       modify the rc EQU to get sufficient delay
00056                                     settlehi
00057  A 07ef 16  00      A      BSET  adctl,porta * pull adctl low
00058  A 07f1 20  02      07f5      BRA   settle0
00059                                     settlelo
00060                                     settle0
00061  A 07f3 17  00      A      BCLR  adctl,porta * pull adctl low
00062                                     settle0
00063                                     settle0
00064                                     MACRO INV      output adctl,porta * make adctl an output
00065                                     settle
00066  A 07f7 a6  64      A settle  LDA   #rc*2*10 *wait 10 * RC
00067  A 07f9 ad  cd      07c8      BSR   delay
00068  A 07fb 81                                     RTS
00069                                     *=====
00070                                     * Check limits
00071                                     *=====
00072                                     * Function: IF res > [x] THEN res := [x]
00073                                     * Input: X points at 16 bit value to check against res
00074                                     * Uses: res
00075                                     * Output: none
00076                                     limit
00077  A 07fc e601      LDA   1,x
00078  A 07fe b0  94      A      SUB   res+1
00079  A 0800 f6      LDA   ,x
00080  A 0801 b2  93      A      SBC   res
00081  A 0803 25  07      080c      BCS   limit1
00082  A 0805 f6      LDA   ,x
00083  A 0806 b7  93      A      STA   res
00084  A 0808 e601      LDA   1,x
00085  A 080a b7  94      A      STA   res+1
00086                                     limit1
00087  A 080c 81      RTS
00088                                     *=====
00089                                     * Calibrate
00090                                     *=====
00091                                     calib
00092                                     *=====
00093                                     *=====
00094                                     *=====
00095                                     *=====
00096                                     *=====

```

—continued overleaf—

## Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 13  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMON OPERANDS COMMENT
00097  A 080d cd  07f3  A      JSR    settlelo * wait for input to stabilise lo
00098
00099  A 0810 cd  07e2  A      JSR    clrtime  * synchronise with clock
00100  A 0813 16  00      A      BSET   adctl,porta * pull adctl hi
00101  A 0815 ae  91      A      LDX    #ref      * prepare to get reference value
00102
00103  A 0817 09  00fd  0817      BRCLR  adinp,porta,* * loop until adinp goes high
00104  A 081a cd  07d1  A      JSR    gettime
00105  A 081d 17  00      A      BCLR  adctl,porta * pull adctl lo, not needed !!!!!!!
00106          MACRO INV      msghex ref,refmsg
00107  A 082e 81          RTS
00108
00109
00110
00111          *=====
00112          * Measure
00113          *=====
00114          measure
00115  A 082f cd  07f3  A      JSR    settlelo * wait for input to stabilise lo
00116
00117  A 0832 cd  07e2  A      JSR    clrtime  * synchronise with clock
00118          MACRO INV      inport adctl,porta * release adctl
00119  A 0837 ae  93      A      LDX    #res      * prepare to get result
00120
00121  A 0839 09  00fd  0839      BRCLR  adinp,porta,* * loop until adinp goes high
00122  A 083c cd  07d1  A      JSR    gettime  * get result
00123  A 083f 17  00      A      BCLR  adctl,porta * not needed !!!!!!!!!!!
00124          MACRO INV      outport adctl,porta * make adctl an outport again, ???
00125          MACRO INV      msghex res,resmsg * transmit result
00126  A 0852 81          RTS
00127
00128          *=====
00129          * Calculate final result
00130          *=====
00131
00132          * measurement is ended, now adjust results
00133          calc
00134
00135          *res:=res - 2 * ref
00136          MACRO INV      shr16 res
00137  A 0857 ae  91      A      LDX    #ref
00138  A 0859 cd  07fc  A      JSR    limit    * if res < ref then res := ref
00139
00140          MACRO INV      sub16 res,res,ref
00141
00142          * if res < 0 then res := 0
00143  A 0868 24  04  086e      BCC    calc3
00144  A 086a 3f  93      A      CLR    res
00145  A 086c 3f  94      A      CLR    res+1
00146
00147          * shift ref and res until msb set in ref
00148          MACRO INV      calc3    shl16 res

```

—continued overleaf—

## Freescale Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 14

Sun Mar 15 20:56:45 1992

Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE   S PC   OPC0 OPERANDS S LABEL   MNEMO OPERANDS COMMENT
00149           MACRO INV           shl16 ref      shift until msb set
00150 A 0876 2a   f6      086e      BPL   calc3
00151
00152           * res:= res/256 * fullscale
00153 A 0878 b6   93      A           LDA   res
00154 A 087a ae   64      A           LDX   #fullscale
00155 A 087c 42           MUL
00156 A 087d b7   94      A           STA   res+1
00157 A 087f bf   93      A           STX   res
00158
00159           * final := -1
00160 A 0881 a6   ff      A           LDA   #-1
00161 A 0883 b7   95      A           STA   final
00162
00163           * ref:= ref/256
00164 A 0885 b6   91      A           LDA   ref
00165 A 0887 44           lsra
00166 A 0888 b7   92      A           STA   ref+1
00167 A 088a 3f   91      A           CLR   ref
00168
00169           * final := res/ref
00170           calc5
00171 A 088c 3c   95      A           INC   final
00172           MACRO INV           sub16 res,res,ref ref:=ref-result
00173 A 089a 24   f0      088c      BCC   calc5
00174
00175           MACRO INV           msgdec final,finmsg
00176 A 08a6 81           RTS
00177
00178
00179
00180           *****
00181           *           Main program loop
00182           *****
00183
00184           start
00185           MACRO INV           inport adinp,porta *set up all ports
00186           MACRO INV           inport key,portb
00187           MACRO INV           output adctl,porta
00188           MACRO INV           output scitr,porta
00189           MACRO INV           output sync,porta
00190 A 08b1 1a   08      A           BSET  tofe,tcsr *enable timer interrupt
00191
00192           main
00193
00194 A 08b3 cd   080d      A           JSR   calib   * calibrate
00195
00196 A 08b6 cd   082f      A           JSR   measure * measure
00197
00198 A 08b9 cd   0853      A           JSR   calc
00199
00200 A 08bc ae   2c      A           LDX   #nlmsg-msg * new line

```

—continued overleaf—

# Freescal Semiconductor, Inc.

## (RESADC.LST continued)

M6805 Portable Cross Assembler 0.05 RESADC.asm Page 15  
 Sun Mar 15 20:56:45 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S    | LABEL | MNEMO | OPERANDS | COMMENT             |
|-------|---|------|------|----------|------|-------|-------|----------|---------------------|
| 00201 | A | 08be | cd   | 075b     | A    |       | JSR   | xmitmsg  |                     |
| 00202 |   |      |      |          |      |       |       |          |                     |
| 00203 | A | 08c1 | a6   | 64       | A    |       | LDA   | #100     |                     |
| 00204 | A | 08c3 | cd   | 07c8     | A    |       | JSR   | delay    |                     |
| 00205 | A | 08c6 | 20   | eb       | 08b3 |       | BRA   | main     |                     |
| 00206 |   |      |      |          |      |       |       |          |                     |
| 00207 |   |      |      |          |      |       |       |          | *****               |
| 00208 |   |      |      |          |      |       |       |          |                     |
| 00209 | A | 0f00 |      |          |      |       | ORG   | \$F00    |                     |
| 00210 | A | 0f00 |      | 00       | A    | mor   | FCB   | \$0      | *705J2 mode, no COP |
| 00211 |   |      |      |          |      |       |       |          |                     |
| 00212 | A | 0ff8 |      |          |      |       | ORG   | \$FF8    |                     |
| 00213 | A | 0ff8 |      | 0700     | A    |       | FDB   | realt    |                     |
| 00214 | A | 0ffe |      |          |      |       | ORG   | \$FFE    |                     |
| 00215 | A | 0ffe |      | 08a7     | A    |       | FDB   | start    |                     |
| 00216 |   |      |      |          |      |       |       |          |                     |
| 00217 |   |      |      |          |      |       | END   |          |                     |
| 00218 |   |      |      |          |      |       |       |          |                     |

Total number of errors: 0  
 Total number of warnings: 0  
 Total number of lines: 538

Number of bytes in section ASCT: 476

Number of bytes in program: 476

## RESADC1.LST

M6805 Portable Cross Assembler 0.05 resadc1.asm Page 5  
 Sun Mar 15 21:42:34 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S | LABEL     | MNEMO | OPERANDS | COMMENT                                 |
|-------|---|------|------|----------|---|-----------|-------|----------|-----------------------------------------|
| 00007 |   |      |      |          |   |           |       |          |                                         |
| 00008 |   |      |      |          |   |           |       |          | *=====                                  |
| 00009 |   |      |      |          |   |           |       |          | * Resistance measurement for HC705J2    |
| 00010 |   |      |      |          |   |           |       |          | * Optional version                      |
| 00011 |   |      |      |          |   |           |       |          | *=====                                  |
| 00012 |   |      |      |          |   |           |       |          |                                         |
| 00013 | P | 0000 |      | 0064     | A | fullscale | EQU   | 100      | * this determines the full scale result |
| 00014 |   |      |      |          |   |           |       |          |                                         |
| 00015 | A | 0000 |      |          |   |           | ORG   | \$0      |                                         |
| 00016 | A | 0000 |      | 01       | A | porta     | RMB   | 1        |                                         |
| 00017 | A | 0001 |      | 0001     | A | pullup    | EQU   | 1        | * pa1 pull up                           |
| 00018 | A | 0001 |      | 0002     | A | scitr     | EQU   | 2        | * pa2 sci transmitter                   |
| 00019 | A | 0001 |      | 0003     | A | adctl     | EQU   | 3        | * pa3 potentiometer control             |
| 00020 | A | 0001 |      | 0004     | A | adinp     | EQU   | 4        | * pa4 potentiometer input               |
| 00021 | A | 0001 |      | 0005     | A | sync      | EQU   | 5        | * pa5 for oscilloscop sync              |
| 00022 |   |      |      |          |   |           |       |          |                                         |
| 00023 | A | 0001 |      | 01       | A | portb     | RMB   | 1        |                                         |
| 00024 | A | 0002 |      | 0002     | A | key       | EQU   | 2        | * key                                   |
| 00025 |   |      |      |          |   |           |       |          |                                         |
| 00026 |   |      |      |          |   |           |       |          |                                         |
| 00027 |   |      |      |          |   |           |       |          | * Timer registers                       |
| 00028 | A | 0008 |      |          |   |           | ORG   | \$8      |                                         |
| 00029 | A | 0008 |      | 01       | A | tcsr      | RMB   | 1        | * Timer Count Status Register           |
| 00030 | A | 0009 |      | 0007     | A | tof       | EQU   | 7        | * Timer Overflow Flag                   |
| 00031 | A | 0009 |      | 0006     | A | rtif      | EQU   | 6        | * Real Time Interrupt Flag              |
| 00032 | A | 0009 |      | 0005     | A | tofe      | EQU   | 5        | * Timer OverFlow Enable                 |
| 00033 | A | 0009 |      | 0004     | A | rtie      | EQU   | 4        | * Real Time Interrupt Enable            |
| 00034 |   |      |      |          |   |           |       |          |                                         |
| 00035 | A | 0009 |      | 01       | A | tc        | RMB   | 1        | * Timer Counter Register                |
| 00036 |   |      |      |          |   |           |       |          |                                         |
| 00037 |   |      |      |          |   |           |       |          | *=====                                  |
| 00038 |   |      |      |          |   |           |       |          | * Start of RAM area                     |
| 00039 |   |      |      |          |   |           |       |          | *=====                                  |
| 00040 | A | 0090 |      |          |   |           | ORG   | \$90     |                                         |
| 00041 | A | 0090 |      | 01       | A | time      | RMB   | 1        | * elapsed time in 0.5 mS div            |
| 00042 | A | 0091 |      | 02       | A | ref       | RMB   | 2        | * time for rising signal to pass limit  |
| 00043 | A | 0093 |      | 02       | A | res       | RMB   | 2        | * measured result                       |
| 00044 | A | 0095 |      | 01       | A | final     | RMB   | 1        | * 8 bits final result                   |
| 00045 |   |      |      |          |   |           |       |          |                                         |
| 00020 |   |      |      |          |   |           |       |          | *=====                                  |
| 00021 |   |      |      |          |   |           |       |          | * Start of ROM area                     |
| 00022 |   |      |      |          |   |           |       |          | *=====                                  |
| 00023 |   |      |      |          |   |           |       |          |                                         |

—continued overleaf—

## (RESADC1.LST continued)

M6805 Portable Cross Assembler 0.05 resadc1.asm Page 11  
 Sun Mar 15 21:42:34 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

```

LINE  S PC  OPC0 OPERANDS S LABEL  MNEMO OPERANDS COMMENT
00050                                     *=====
00051                                     * Settle measurment voltages
00052                                     *=====
00053                                     * Function: Delay for voltage on input capacitor to settle
00054                                     * Input: none
00055                                     * Uses: adctl, delay
00056                                     * Output: none
00057                                     * Note: three entry lables depending on desired function
00058                                     * interrupts are disabled when leaving
00059                                     * modify the rc EQU to get sufficient delay
00060 settlehi
00061 A 07ef 16 00 A BSET adctl,porta * pull adctl low
00062 A 07f1 20 02 07f5 BRA settle0
00063
00064 settlelo
00065 A 07f3 17 00 A BCLR adctl,porta * pull adctl low
00066
00067 settle0
00068 MACRO INV output adctl,porta * make adctl an output
00069
00070 A 07f7 a6 64 A settle LDA #rc*2*10 *wait 10 * RC
00071 A 07f9 ad cd 07c8 BSR delay
00072 A 07fb 81 RTS
00073                                     *=====
00074                                     * Check limits
00075                                     *=====
00076                                     * Function: IF res > [x] THEN res := [x]
00077                                     * Input: X points at 16 bit value to check against res
00078                                     * Uses: res
00079                                     * Output: none
00080 limit
00081 A 07fc e601 LDA 1,x
00082 A 07fe b0 94 A SUB res+1
00083 A 0800 f6 LDA ,x
00084 A 0801 b2 93 A SBC res
00085 A 0803 25 07 080c BCS limit1
00086
00087 A 0805 f6 LDA ,x
00088 A 0806 b7 93 A STA res
00089 A 0808 e601 LDA 1,x
00090 A 080a b7 94 A STA res+1
00091
00092 limit1
00093 A 080c 81 RTS
00094
00095
00096                                     *=====
00097                                     * Calibrate
00098                                     *=====
00099
00100 calib

```

—continued overleaf—



## (RESADC1.LST continued)

M6805 Portable Cross Assembler 0.05 resadc1.asm Page 12

Sun Mar 15 21:42:34 1992

Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO  | OPERANDS | S    | LABEL | MNEMO   | OPERANDS      | COMMENT                                    |
|-------|---|------|-------|----------|------|-------|---------|---------------|--------------------------------------------|
| 00101 |   |      | MACRO | INV      |      |       | inport  | pullup,porta  | *<- let pullup float                       |
| 00102 | A | 080f | cd    | 07f3     | A    |       | JSR     | settlelo      | * wait for input to stabilise lo           |
| 00103 |   |      |       |          |      |       |         |               |                                            |
| 00104 | A | 0812 | cd    | 07e2     | A    |       | JSR     | clrtime       | * synchronize with clock                   |
| 00105 | A | 0815 | 16    | 00       | A    |       | BSET    | adctl,porta   | * pull adctl hi                            |
| 00106 | A | 0817 | ae    | 91       | A    |       | LDX     | #ref          | * prepare to get reference value           |
| 00107 |   |      |       |          |      |       |         |               |                                            |
| 00108 | A | 0819 | 09    | 00fd     | 0819 |       | BRCLR   | adinp,porta,* | * loop until adinp goes high               |
| 00109 | A | 081c | cd    | 07d1     | A    |       | JSR     | gettime       |                                            |
| 00110 | A | 081f | 17    | 00       | A    |       | BCLR    | adctl,porta   | * pull adctl lo, not needed !!!!!!!        |
| 00111 |   |      | MACRO | INV      |      |       | msghex  | ref,refmsg    |                                            |
| 00112 | A | 0830 | 81    |          |      |       | RTS     |               |                                            |
| 00113 |   |      |       |          |      |       |         |               |                                            |
| 00114 |   |      |       |          |      |       |         |               |                                            |
| 00115 |   |      |       |          |      |       |         |               |                                            |
| 00116 |   |      |       |          |      |       |         |               | =====                                      |
| 00117 |   |      |       |          |      |       |         |               | * Measure                                  |
| 00118 |   |      |       |          |      |       |         |               | =====                                      |
| 00119 |   |      |       |          |      |       |         |               | measure                                    |
| 00120 | A | 0831 | cd    | 07f3     | A    |       | JSR     | settlelo      | * wait for input to stabilise lo           |
| 00121 |   |      |       |          |      |       |         |               |                                            |
| 00122 | A | 0834 | cd    | 07e2     | A    |       | JSR     | clrtime       | * synchronize with clock                   |
| 00123 |   |      | MACRO | INV      |      |       | inport  | adctl,porta   | * release adctl                            |
| 00124 |   |      | MACRO | INV      |      |       | outport | pullup,porta  | *<- change                                 |
| 00125 | A | 083b | 12    | 00       | A    |       | BSET    | pullup,porta  | *<- set direction                          |
| 00126 | A | 083d | ae    | 93       | A    |       | LDX     | #res          | * prepare to get result                    |
| 00127 |   |      |       |          |      |       |         |               |                                            |
| 00128 | A | 083f | 09    | 00fd     | 083f |       | BRCLR   | adinp,porta,* | * loop until adinp goes high               |
| 00129 | A | 0842 | cd    | 07d1     | A    |       | JSR     | gettime       | * get result                               |
| 00130 |   |      | MACRO | INV      |      |       | inport  | pullup,porta  | *<- change                                 |
| 00131 |   |      | MACRO | INV      |      |       | outport | adctl,porta   | * make adctl an outport again, ???         |
| 00132 |   |      | MACRO | INV      |      |       | msghex  | res,resmsg    | * transmit result                          |
| 00133 | A | 0858 | 81    |          |      |       | RTS     |               |                                            |
| 00134 |   |      |       |          |      |       |         |               |                                            |
| 00135 |   |      |       |          |      |       |         |               | =====                                      |
| 00136 |   |      |       |          |      |       |         |               | * Calculate final result                   |
| 00137 |   |      |       |          |      |       |         |               | =====                                      |
| 00138 |   |      |       |          |      |       |         |               |                                            |
| 00139 |   |      |       |          |      |       |         |               | * measurement is ended, now adjust results |
| 00140 |   |      |       |          |      |       |         |               | calc                                       |
| 00141 |   |      |       |          |      |       |         |               | *                                          |
| 00142 | A | 0859 | ae    | 91       | A    |       | shr16   | res           |                                            |
| 00143 | A | 085b | cd    | 07fc     | A    |       | LDX     | #ref          |                                            |
| 00144 |   |      |       |          |      |       | JSR     | limit         | * if res < ref then res:= ref              |
| 00145 |   |      |       |          |      |       |         |               |                                            |
| 00146 |   |      |       |          |      |       |         |               | *res:=res - ref <- change                  |
| 00147 |   |      | MACRO | INV      |      |       | sub16   | res,res,ref   |                                            |
| 00148 |   |      | MACRO | INV      |      |       | shr16   | res           |                                            |
| 00149 |   |      |       |          |      |       |         |               |                                            |
| 00150 |   |      |       |          |      |       |         |               | * if res < 0 then res := 0                 |
| 00151 | A | 086e | 0f    | 9304     | 0875 |       | BRCLR   | 7,res,calc3   |                                            |
| 00152 | A | 0871 | 3f    | 93       | A    |       | CLR     | res           |                                            |

—continued overleaf—

## (RESADC1.LST continued)

M6805 Portable Cross Assembler 0.05 resadc1.asm Page 13  
 Sun Mar 15 21:42:34 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO  | OPERANDS | S    | LABEL | MNEMONIC | OPERANDS     | COMMENT                                  |
|-------|---|------|-------|----------|------|-------|----------|--------------|------------------------------------------|
| 00153 | A | 0873 | 3f    | 94       | A    |       | CLR      | res+1        |                                          |
| 00154 |   |      |       |          |      |       |          |              |                                          |
| 00155 |   |      |       |          |      |       |          |              |                                          |
| 00156 |   |      |       |          |      |       |          |              | * shift ref and res until msb set in ref |
| 00157 |   |      | MACRO | INV      |      | calc3 | shl16    | res          |                                          |
| 00158 |   |      | MACRO | INV      |      |       | shl16    | ref          | shift until msb set                      |
| 00159 | A | 087d | 2a    | f6       | 0875 |       | BPL      | calc3        |                                          |
| 00160 |   |      |       |          |      |       |          |              |                                          |
| 00161 |   |      |       |          |      |       |          |              | * res:= res/256 * fullscale              |
| 00162 | A | 087f | b6    | 93       | A    |       | LDA      | res          |                                          |
| 00163 | A | 0881 | ae    | 64       | A    |       | LDX      | #fullscale   |                                          |
| 00164 | A | 0883 | 42    |          |      |       | MUL      |              |                                          |
| 00165 | A | 0884 | b7    | 94       | A    |       | STA      | res+1        |                                          |
| 00166 | A | 0886 | bf    | 93       | A    |       | STX      | res          |                                          |
| 00167 |   |      |       |          |      |       |          |              |                                          |
| 00168 |   |      |       |          |      |       |          |              | * final := -1                            |
| 00169 | A | 0888 | a6    | ff       | A    |       | LDA      | #-1          |                                          |
| 00170 | A | 088a | b7    | 95       | A    |       | STA      | final        |                                          |
| 00171 |   |      |       |          |      |       |          |              |                                          |
| 00172 |   |      |       |          |      |       |          |              | * ref:= ref/256                          |
| 00173 | A | 088c | b6    | 91       | A    |       | LDA      | ref          |                                          |
| 00174 | A | 088e | 44    |          |      |       | lsra     |              |                                          |
| 00175 | A | 088f | b7    | 92       | A    |       | STA      | ref+1        |                                          |
| 00176 | A | 0891 | 3f    | 91       | A    |       | CLR      | ref          |                                          |
| 00177 |   |      |       |          |      |       |          |              |                                          |
| 00178 |   |      |       |          |      |       |          |              | * final := res/ref                       |
| 00179 |   |      |       |          |      | calc5 |          |              |                                          |
| 00180 | A | 0893 | 3c    | 95       | A    |       | INC      | final        |                                          |
| 00181 |   |      | MACRO | INV      |      |       | sub16    | res,res,ref  | ref:=ref-result                          |
| 00182 | A | 08a1 | 24    | f0       | 0893 |       | BCC      | calc5        |                                          |
| 00183 |   |      |       |          |      |       |          |              |                                          |
| 00184 |   |      | MACRO | INV      |      |       | msgdec   | final,finmsg |                                          |
| 00185 | A | 08ad | 81    |          |      |       | RTS      |              |                                          |
| 00186 |   |      |       |          |      |       |          |              |                                          |
| 00187 |   |      |       |          |      |       |          |              |                                          |
| 00188 |   |      |       |          |      |       |          |              |                                          |
| 00189 |   |      |       |          |      |       |          |              | *****                                    |
| 00190 |   |      |       |          |      |       |          |              | * Main program loop                      |
| 00191 |   |      |       |          |      |       |          |              | *****                                    |
| 00192 |   |      |       |          |      |       |          |              |                                          |
| 00193 |   |      |       |          |      | start |          |              |                                          |
| 00194 |   |      | MACRO | INV      |      |       | inport   | adinp,porta  | *set up all ports                        |
| 00195 |   |      | MACRO | INV      |      |       | inport   | key,portb    |                                          |
| 00196 |   |      | MACRO | INV      |      |       | output   | adctl,porta  |                                          |
| 00197 |   |      | MACRO | INV      |      |       | output   | scitr,porta  |                                          |
| 00198 |   |      | MACRO | INV      |      |       | output   | sync,porta   |                                          |
| 00199 |   |      | MACRO | INV      |      |       | inport   | pullup,porta | *!option                                 |
| 00200 | A | 08ba | 12    | 00       | A    |       | BSET     | pullup,porta | *!option                                 |
| 00201 | A | 08bc | 1a    | 08       | A    |       | BSET     | tofe,tcsr    | *enable timer interrupt                  |
| 00202 |   |      |       |          |      |       |          |              |                                          |
| 00203 |   |      |       |          |      | main  |          |              |                                          |
| 00204 |   |      |       |          |      |       |          |              |                                          |

—continued overleaf—

# Freescale Semiconductor, Inc.

## (RESADC1.LST continued)

M6805 Portable Cross Assembler 0.05 resadc1.asm Page 14  
 Sun Mar 15 21:42:34 1992  
 Options - MD,MC,NOG,NOU,W,NOMEX,CL,FMT,0

| LINE  | S | PC   | OPCO | OPERANDS | S    | LABEL | MNEMO | OPERANDS   | COMMENT             |
|-------|---|------|------|----------|------|-------|-------|------------|---------------------|
| 00205 | A | 08be | cd   | 080d     | A    |       | JSR   | calib      | * calibrate         |
| 00206 |   |      |      |          |      |       |       |            |                     |
| 00207 | A | 08c1 | cd   | 0831     | A    |       | JSR   | measure    | * measure           |
| 00208 |   |      |      |          |      |       |       |            |                     |
| 00209 | A | 08c4 | cd   | 0859     | A    |       | JSR   | calc       |                     |
| 00210 |   |      |      |          |      |       |       |            |                     |
| 00211 | A | 08c7 | ae   | 2c       | A    |       | LDX   | #nlmsg-msg | * new line          |
| 00212 | A | 08c9 | cd   | 075b     | A    |       | JSR   | xmitmsg    |                     |
| 00213 |   |      |      |          |      |       |       |            |                     |
| 00214 | A | 08cc | a6   | 64       | A    |       | LDA   | #100       |                     |
| 00215 | A | 08ce | cd   | 07c8     | A    |       | JSR   | delay      |                     |
| 00216 | A | 08d1 | 20   | eb       | 08be |       | BRA   | main       |                     |
| 00217 |   |      |      |          |      |       |       |            |                     |
| 00218 |   |      |      |          |      |       |       |            |                     |
| 00219 |   |      |      |          |      |       |       |            |                     |
| 00220 | A | 0f00 |      |          |      |       | ORG   | \$F00      |                     |
| 00221 | A | 0f00 |      | 00       | A    | mor   | FCB   | \$0        | *705J2 mode, no COP |
| 00222 |   |      |      |          |      |       |       |            |                     |
| 00223 | A | 0ff8 |      |          |      |       | ORG   | \$FF8      |                     |
| 00224 | A | 0ff8 |      | 0700     | A    |       | FDB   | realt      |                     |
| 00225 | A | 0ffe |      |          |      |       | ORG   | \$FFE      |                     |
| 00226 | A | 0ffe |      | 08ae     | A    |       | FDB   | start      |                     |
| 00227 |   |      |      |          |      |       |       |            |                     |
| 00228 |   |      |      |          |      |       | END   |            |                     |

Total number of errors: 0  
 Total number of warnings: 0  
 Total number of lines: 548

Number of bytes in section ASCT: 487

Number of bytes in program: 487

**Freescale Semiconductor, Inc.**

All products are sold on Motorola's Terms & Conditions of Supply. In ordering a product covered by this document the Customer agrees to be bound by those Terms & Conditions and nothing contained in this document constitutes or forms part of a contract (with the exception of the contents of this Notice). A copy of Motorola's Terms & Conditions of Supply is available on request.

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 can and do vary in different applications. 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.

The Customer should ensure that it has the most up to date version of the document by contacting its local Motorola office. This document supersedes any earlier documentation relating to the products referred to herein. The information contained in this document is current at the date of publication. It may subsequently be updated, revised or withdrawn.

**Literature Distribution Centres:**

EUROPE: Motorola Ltd., European Literature Centre, 88 Tanners Drive, Blakelands, Milton Keynes, MK14 5BP, England.  
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.  
JAPAN: Nippon Motorola Ltd., 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141, Japan.  
USA: Motorola Literature Distribution, P.O. Box 20912, Phoenix, Arizona 85036.

**MOTOROLA**

**For More Information On This Product,  
Go to: [www.freescale.com](http://www.freescale.com)**

**AN477/D**