Analog and Microprocessor System Design
The Weather Station Project

Peter D. Hiscocks
Department of Electrical and Computer Engineering
Ryerson University
Toronto, Canada

Email: phiscock@ee.ryerson.ca
URL: http://www.ee.ryerson.ca/~phiscock
http://www.syscompdesign.com

Contents

List of Figures

Preface

This is a book about electronic instrumentation, the design and construction of machines for the measurement of physical quantities. It is also about Engineering: the art and science of desiging electronic machinery to perform a particular task.

As an extended example, we shall study the measurement of the physical properties associated with meteorology, the science of weather. In particular, we will design instruments for the measurement of air temperature, humidity, pressure, direction and speed of movement. The circuit designs are not an end in themselves, though they may be useful. Beyond these, which will change with the technology, are engineering skills which outlast changes in the hardware and software: organizing an engineering approach to the design, weighing alternatives, analysing and understanding the behaviour of components and systems, ensuring that performance objectives are met.

Organization of the Text

The text first emphasises the electronic design issues – analogue and digital – that are involved in the design of meterological sensors.

Significant material is devoted to the design of microprocessor software, an area that is difficult and error prone for many students.

The final chapter includes a copy of the Lab Outline and typical Lab Exercises that have been used in this course.

A complete, working weather station could be built using the ideas in this book. However, because this book focusses on electronic design, the mechanical details and the design of some of the more ordinary elements such as the power supply are not addressed here. Anyone reading this book is assumed to be capable of translating the ideas into complete, working hardware.

Course Description

The material in this text was originally used in a one-semester laboratory course in microprocessor-based instrumentation, directed to the following objectives:

- to design an electronic system using analogue and microprocessor components.
- to design circuits requiring care in selection of components and grounding, shielding and decoupling.
- to gain experience in calibrating and verifying the performance of an analogue system.
- to design and build software for an embedded microprocessor system.

Further detail on the lab curriculum is contained in chapter 7.

The Microcomputer System

The Weather Station Project uses a 68HC11 microcomputer board developed by the author for use in this project. The microcomputer board (The MPP Board) is available for commercial sale, along with a 100 page technical manual. Technical details, price and availabilty are listed on the web site

`http://www.syscompdesign.com`

Feedback

Comments, requests for clarification and cash prizes are all encouraged and should be directed to

`phiscock@ee.ryerson.ca`.

Acknowledgements

Special thanks to Jim Koch for designing the MPP circuit board, testing the various circuits, and his endless patience in helping students in the lab.

Thanks to David Smith, maintainer of `xfig` and `transfig` for his help in resolving bugs.

Production Notes

This document was prepared on a Linux system using the RedHat 7.0 release and the `FVWM` desktop. Text editing was done using the `joe` text editor. Diagrams were prepared with the excellent `xfig` drawing program and incorporated as `latex`, `epic`, `pictex` or `eps` formatted files. Images were processed through the `xv` image processing program and incorporated in `eps` format. Graphs were generated with `gnuplot` and then further massaged in `xfig`.

Text processing was done with the LaTeX typesetting program, together with the packages `graphicx`, `epsfig`, `subfigure`, `epic`, `piclatex`, `latexsym`, `afterpage` and `headings`. (The functions of these packages is described in The Latex Companion.)

Document previewing was done with the `xdvi` previewer. When the document was completed, it was formatted to postscript using `dvips` and then further converted to `pdf` format using `ps2pdf`.

Special thanks to the individuals who have provided this excellent software for the use of the technical writing community.

Copyright

Chapter 1

Temperature Measurement

1.1 General Requirements

Our intent is to design a low cost temperature sensor that can be used for weather monitoring in domestic applications. It is expected that the circuits will be used primarily by students and hobbiests. As a result, the the design should be simple, straightforward to design and analyse, use readily available parts, and be simple to construct. It should also be be accurate without calibration or be simple to calibrate, since users are unlikely to have access to sophisticated equipment.

The following are specific performance requirements for the temperature measuring circuit.

Range of Measurement According to the Canadian Encyclopedia (reference [1], the air temperature extremes to be expected in Canada and the world are:

Measurement	**Canadian Record**	**World Record**
Maximum	45.0^{o}C, Midale and Yellow Grass, Saskatchewan, 5 July 1937	58.0^{o}C, Al'azizyah, Libya, 13 Sept 1922
Minimum	-63.0^{o}C, Snag, Yukon Territory, 3 Feb 1947	-89.2^{o}C, Vostok, Antartica, 21 July 1983

Ideally, our temperature measuring circuit should be able to cope with the maximum possible range of ambient temperature on the face of the planet earth. However, as we shall see, this may have to be compromised in the face of practical sensor limitations.

On the other hand, to simplify things for the circuit designer, we assume that all the circuits except the temperature sensor are kept at room temperature, which we define as +15 ^{o}C and +25 ^{o}C [1].

Resolution The resolution of the circuit is the minimum detectable change in temperature. For a domestic thermometer, a resolution of 1^{o}C may be sufficient. For professional weather-monitoring applications, a resolution of 0.1^o is more likely to be required.

Initially, we shall specify the resolution of the temperature measuring system as 1 degree centigrade. We shall also examine a technique for increasing this resolution to one tenth of a degree.

[1] Those with unsympathetic landlords or heritage in the British Isles may wish to choose a lower minimum temperature.

Accuracy The accuracy of the instrument specifies the allowable error in measurement. For example, in a case where the temperature trend is of interest, accuracy may not be all that important. On the other hand, where temperature readings must be compared with other instruments, the accuracy requirements are much more stringent. The resolution of the instrument should reflect its accuracy: it does not make sense to have a readout to six decimal places when the instrument is only accurate to two decimal places.

It is reasonable to make the accuracy of the temperature measurement circuit equal to or better than the resolution. That is, for a temperature resolution of 1 degree, the accuracy should be $+/-1^oC$ over the entire range of operation.

Speed of Response In the measuring of climate data, air temperature changes slowly with respect to the speed of most electronic instruments. However, in the case of a clinical thermometer, a fast response is desireable to minimize the time required for the sensor to change from room to body temperature.

Calibration From the standpoint of manufacturing cost, it is highly desireable that the instrument can be relied upon to meet its accuracy requirement without calibration. Calibration activities are time consuming and therefore expensive, so more expensive components can be justified if they minimize or eliminate calibration requirements.

On the other hand, for a hobbyist, the cost of labour is not a major consideration and the incentive may be to minimize parts cost instead.

Cost The cost should be minimized while meeting the performance specifications.

1.2 The Thermistor

The thermistor (thermal resistor), is so named because it has a strong positive or negative coefficient of resistance with temperature. Thermistors of interest in to us are NTC devices: they have a negative coefficient of resistance. The resistance of the sensor decreases with increasing temperature.

Thermistors are attractive for temperature measuring devices because they are relatively inexpensive and readily available in different physical packages and with different resistance values [10]. Their sensitivity greatly simplifies the electronic display circuitry.

The resistance-temperature characteristic of the thermistor is approximately exponential: the characteristic of a typical thermistor is shown in figure 1.1.

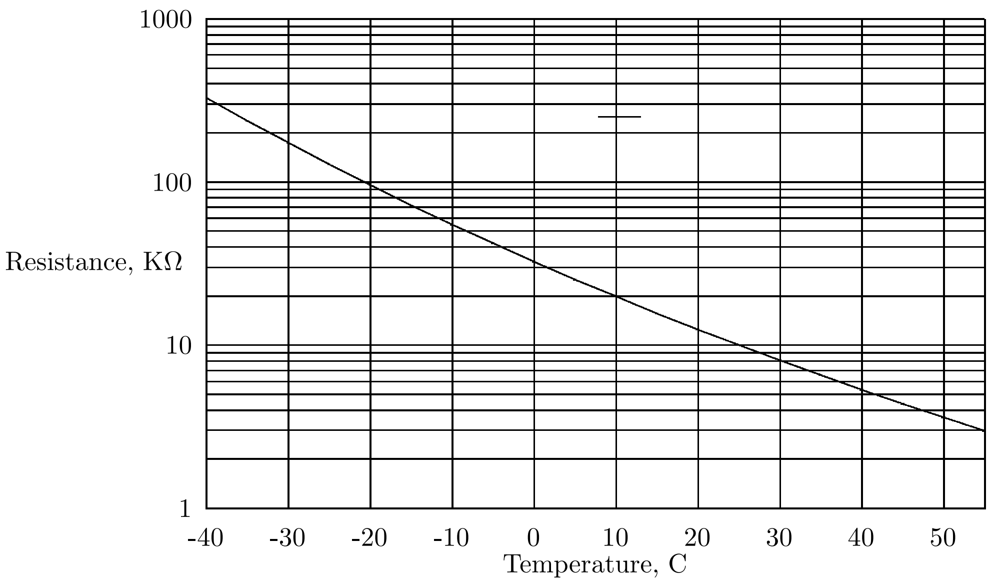


Figure 1.1: Thermistor Resistance-Temperature Characteristic

The data for this thermistor, derived and simplified from the data sheet in reference [10] is given in section1.2.1. We shall refer to this information throughout the design.

1.2.1 Thermistor Electrical Specifications

Dissipation	250 mW, maximum
Dissipation factor	7 mW/^{o}C
Operating Temperature Range	-40^{o}C to +150^{o}C
Thermal Time Constant	11 seconds, in still air
	1.2 seconds, in liquid
Available Resistance Tolerances	$\pm$10%, $\pm$5%, $\pm$3%
$\beta_{25/85}$	3977
β Tolerance at 25^{o}C	0.75 %
Drift in β at 125^{o}C	$\leq$0.5%

Thermistor Resistance-Temperature Characteristic

Thermistor Data, Philips Thermistor 640-6, 10KΩ		
Temperature, ^{o}C	Resistance Deviation due to β tolerance, %	Resistance, KΩ
–40	2.64	328.4
–35	2.40	237.7
–30	2.16	173.9
–25	1.93	128.5
–20	1.58	95.89
–15	1.49	72.23
–10	1.29	54.89
– 5	1.08	42.07
0	0.89	32.51
5	0.70	25.31
10	0.52	19.86
15	0.34	15.69
20	0.17	12.49
25	0.00	10.00
30	0.16	8.060
35	0.32	6.536
40	0.47	5.331
45	0.62	4.373
50	0.77	3.606
55	0.91	2.989

The resistance of the thermistor varies with temperature, and so the resistance of a thermistor is specified at 25^{o}C, and known as R_{25}. In this case, the value of R_{25} is 10KΩ and the thermistor is a 10KΩ device.

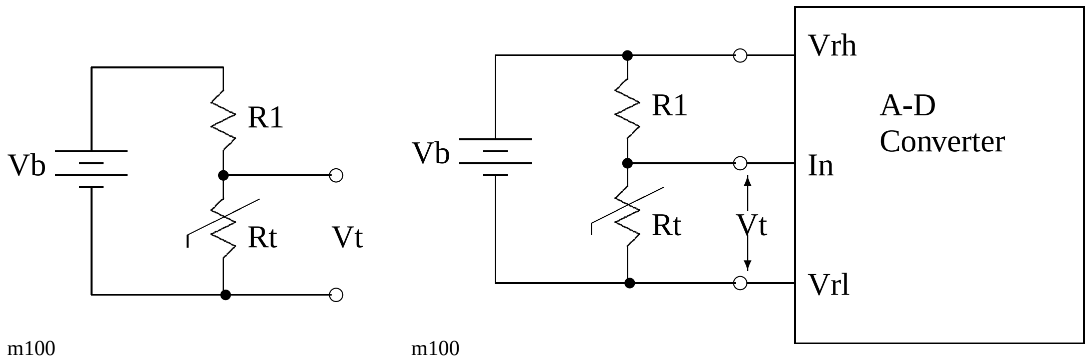
On the other hand, the non-linearity of the temperature-resistance characteristic is a nuisance, though it is possible to fabricate a thermistor sensor with linear resistance-temperature characteristic at somewhat greater cost. It is also relatively difficult to make thermistors accurate at low temperatures.

1.2.2 Thermistor Measuring Circuit

To play on the strengths of the thermistor, we shall design a very simple measuring circuit, using the thermistor of figure 1.1.

Because of the limited accuracy of the chosen thermistor at low temperatures, we shall aim for a temperature range of –30^{o}C to +50^{o}C, adequate for temperate climate zones.

A suitable circuit is shown in figure 1.2(a).



m100
therm-divider-a.latex

(a) Temperature Measuring Circuit

m100
therm-divider-b.latex

(b) Temperature Measurement, Ratiometric A-D

Figure 1.2: Thermistor Temperature Measurement Circuit

As the temperature increases, the resistance R_t of the thermistor decreases, reducing V_t. In this arrangement,

$$V_t = \left(\frac{R_t}{R_t + R_1} \right) \times V_b$$

The magnitude of the supply voltage V_b directly affects the thermistor voltage V_t, so for accurate measurements, the supply voltage V_b must be stable.

If the measurement is by means of a ratiometric analog-digital converter as shown in figure 1.2(b), we can remove the effect of power supply voltage. Recall that a ratiometric A-D converter generates a result which is proportional to the ratio of the input voltage to the two A-D reference voltages, V_{rh} and V_{rl}. If V_{rl} is connected to the power supply reference terminal[2] (the usual case), then the A-D reading N_{AD} is given by

$$N_{AD} = \frac{V_{in}}{V_{rh}} \times N_{max}.$$

In the case of an 8 bit A-D converter, the maximum output value is N_{max} is $2^8 - 1$ or 255.

For thermistor circuit figure 1.2(b), we may substitute 256 for N_{max}, V_t for V_{in} and V_b for V_{rh}:

$$N_{AD} = \frac{V_t}{V_b} \times 256.$$

[2] Usually somewhat loosely referred to as ground

Solving for V_t and then substituting in the thermistor voltage-divider equation, we find that V_b cancels out of the equation, and we have

$$R_t = N_{AD} \times \frac{R_1}{256 - N_{AD}} \tag{1.1}$$

In other words,

- If R_1 and the A-D reading are known, the value of the thermistor resistance may be calculated.
- Within certain limits (ie, assuming the circuits continue to operate) the value of the supply voltage V_b has no effect on the accuracy of the measurement. This is highly desireable for stability and accuracy of the temperature measurement.

A useful form of equation 1.1 gives the value of the A-D converter reading N_{AD} in terms of the resistances:

$$N_{AD} = \frac{R_t}{R_1 + Rt} \times 256 \tag{1.2}$$

Now that we have a method of determining the thermistor resistance from the A-D reading, we need a way to relate the thermistor resistance to its temperature.

There are three possible approaches to this: a lookup table, an approximate formula, and the more accurate Steinhart-Hart approximation.

1.2.3 Lookup Table

Some thermistor data sheets ([10] for example) include a table of resistance-temperature values. The temperature values may then be placed in a lookup table and indexed by the resistance value. For an 8 bit A-D converter, a table with 256 entries would contain all possible temperature readings, as indicated in figure 1.3.

If the output variable can be represented by some value between 0 and 255, then one byte is required per entry of the table. For larger table values, multiple bytes can be used for each entry.

To automate the process of generating the lookup table, it is convenient to write a small program in some readily available computer language such as QBasic, and use that to write a disk file containing the 256 values of the lookup table. The computer program uses the value of the input variable to index into the lookup table, which contains the output variable.

Then the disk file may be loaded into the microprocessor source code to incorporate it into the microprocessor program.

Unfortunately, the lookup table method does not easily allow adjustment of the calibration. In most cases, the entire lookup table must be rebuilt for a different calibration.

1.2.4 Thermistor Equation

The behaviour of an NTC thermistor is often approximated by the equation

$$R_T = R_{25}\, e^{\beta\left(\frac{1}{T} - \frac{1}{T_o}\right)} \tag{1.3}$$

where

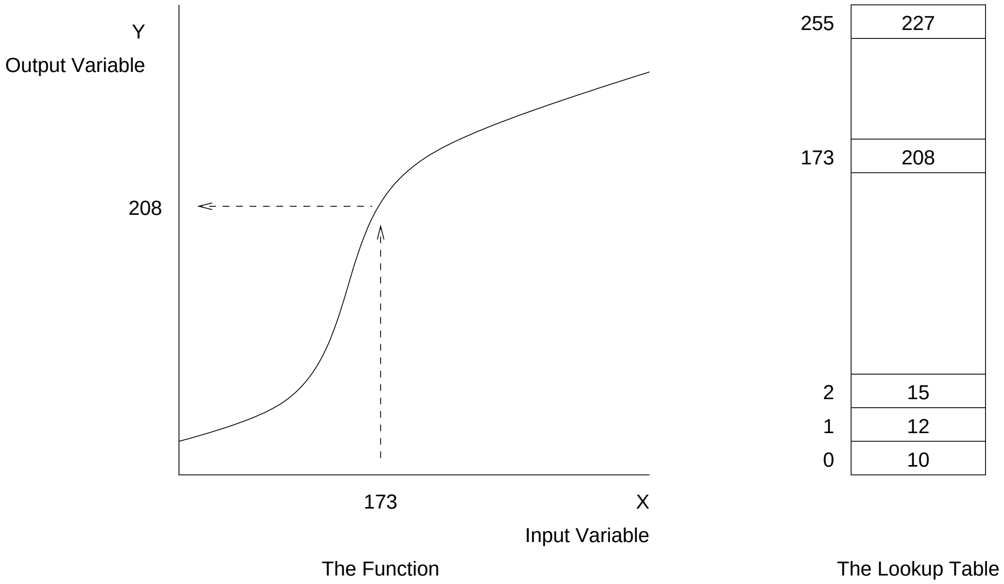


Figure 1.3: Lookup Table

R_T is the resistance of the thermistor at temperature T
R_{25} is the resistance at 25^o C
β is a constant which determines the sensitivity of the thermistor to temperature
T is the temperature in Kelvin
T_o is 25^o C in Kelvin, or 298 K.

The resistance at 25^o C and the value of β are often given on data sheets.

For example, suppose the resistance of a certain thermistor is 10KΩ at 25^oC and the value of β_{25} is given as 3977. Calculate the resistance of the thermistor at 85^oC.

$$\begin{aligned} R_T &= R_{25}\, e^{\beta\left(\dfrac{1}{T} - \dfrac{1}{T_o}\right)} \\ &= 10\, e^{3977\left(\dfrac{1}{358} - \dfrac{1}{298}\right)} \\ &= 1.068 K\Omega \end{aligned}$$

The tabulated value for the thermistor is 1.070KΩ, so the formula is quite close in this case. However, if the same values are used to calculate the resistance of the thermistor at -25^oC, the thermistor equation yeilds $147K\Omega$ and the tabulated value is $128K\Omega$, an error of some 15%. Obviously the thermistor formula must be used with some caution.

The graph of figure 1.4 (plotted on a linear scale to emphasize the discrepancy) shows how the thermistor equation becomes progressively less accurate at low temperatures. The squares show the actual thermistor values, from a table in the data sheet, and the dotted line shows the values generated by the thermistor equation.

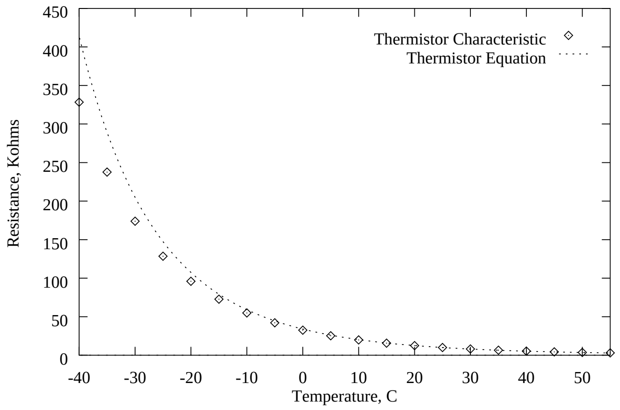


Figure 1.4: Thermistor Equation

1.2.5 Steinhart-Hart Equation

Since a microprocessor will be doing the work of the calculations and speed is not an issue in this project, there is no pressing reason to simplify the calculations. A more complex equation is perfectly acceptable if it generates more accurate results. There is such an equation: the Steinhart-Hart equation, [13]which is more accurate, especially where large changes in temperature are involved:

$$\frac{1}{T} = a + b\ \ln R_T + c\,(\ln R_T)^3 \tag{1.4}$$

where

R_T is the resistance of the thermistor at temperature T
T is the temperature in Kelvin
a, b and c are constants that fit the equation to the behaviour of the thermistor.

To generate a correct equation for a given thermistor, the constants **a**, **b** and **c** must be found by solving three equations in three unknowns. Then the equation may be used to solve for temperature at any intervening value of thermistor resistance.

According to reference [11]:

> When $-40^oC \leq T1, T2, T3 \leq 150^oC$ and $\mid T2 - T1 \mid \leq 50^oC, \mid T3 - T2 \mid \leq 50^oC$ interpolation data generated by this equation will be accurate to $\pm 0.01^oC$ or better.

For example, to determine the constants of the previous 10KΩ thermistor for the range –30^{o}C to +50^{o}C, we might choose the following values of temperature and corresponding thermistor resistance (from the data sheet):

Value	Temperature, ^{o}C	Thermistor Resistance
T1	–30	173.9KΩ
T2	+10	19.86KΩ
T3	+50	3.606KΩ

The three equations are then:

$$\begin{aligned}\frac{1}{243} &= a + b\ \ln 173.9 + c\,(\ln 173.9)^3 \\ \frac{1}{283} &= a + b\ \ln 19.86 + c\,(\ln 19.86)^3 \\ \frac{1}{323} &= a + b\ \ln 3.606 + c\,(\ln 3.606)^3\end{aligned}$$

Using an engineering calculator[3] to solve these equations for **a**, **b** and **c**, we obtain

$$a = 2.772 \times 10^{-3}$$

$$b = 2.524 \times 10^{-4}$$

$$c = 3.042 \times 10^{-7}$$

Now, for example, we can calculate the temperature for a thermistor resistance of 95.89KΩ:

$$\begin{aligned}\frac{1}{T} &= a + b\ \ln R_T + c\,(\ln R_T)^3 \\ &= 2.772 \times 10^{-3} \ + \ 2.524 \times 10^{-4} \times \ln(95.89\Omega) \ + \ 3.042 \times 10^{-7} \times (\ln(95.89\Omega))^3 \\ &= 3.952 \times 10^{-3}\end{aligned}$$

Then

$$\begin{aligned}T &= \frac{1}{3.952 \times 10^{-3}} \\ &= 252.9^o K \\ &= -20.1^o C\end{aligned}$$

From the tabulated values for the thermistor, a resistance of 95.89KΩ should correspond to -20^o Centigrade, so the Steinhart-Hart equation generates an accurate value in this case. It turns out that this is typical and the equation may be relied upon over the entire range of thermistor resistance.

Solving for R_T in the Steinhart-Hart Equation

When the value of temperature is known and the corresponding value of thermistor resistance is to be calculated, the Steinhart-Hart equation (a cubic) must be solved for R_T. A computer program such as Matlab, Maple, or Mathematica or an engineering calculator such as the Texas Instruments TI-85 that is equipped with an equation solving facility will do this. For example, enter the Steinhart-Hart equation as

[3] For example, the Texas Instruments TI-85 calculator has a simultaneous equation solver which makes this calculation quite simple.

$$\frac{1}{T} = a + bx + cx^3 \tag{1.5}$$

where x has been substituted for $\ln(R_T)$.
Use the calculator to solve for x and then find $R_T = e^x$.

1.2.6 Thermistor Circuit Design

Now that we have a method of relating thermistor resistance to temperature, we are in a position to determine the value of the voltage divider resistance R_1 in figure 1.2.

As a starting point, we might make the mid-point of the temperature range correspond to the mid-point of the A-D converter range. The temperature varies from –30^{o}C to +50^{o}C, so the mid-point is +10^{o}C. At this temperature, the thermistor resistance R_t is 19.86KΩ. Then, using equation 1.1, we find that R_1 should be equal to R_t. The nearest standard resistor value is 20KΩ, so that is chosen for R_1. This makes $R_1 = 2 \times R_{25}$.

The graph of figure 1.5 shows in more detail what happens to the transfer characteristic V_t/V_b vs temperature as R_1 is varied with respect to the thermistor resistance R_{25}.

The transfer function for this graph [16] is obtained from the equation

$$\frac{V_t}{V_b} = \frac{R_t}{R_1 + R_t} \tag{1.6}$$

where R_t is given by equation 1.3. (Equation 1.3 is only approximate, as we saw in section 1.2.4, but is adequate for this purpose. It is much easier to use than the Steinhart-Hart formula (equation 1.4) because it yeilds the termistor resistance directly.) Plotting V_t/V_b versus temperature for various values of R_1 yeilds the graph of figure 1.5.

In an ideal world, the graph would be linear, indicating that the output voltage is a linear function of temperature. It's not possible to obtain a linear transfer characteristic for any value of R_1, but the linear region of the graph can be moved to different temperatures. For example, the graph is more linear at low temperatures if R1=8$\times R_{25}$. The graph is more linear at high temperatures if R1=0.5$\times R_{25}$. For our purposes, where operation is between temperatures of -40 and +50^{o}C, a good compromise is R1=2$\times R_{25}$.

As the graph shows, making R_1 equal to R_t in the centre of the temperature region of operation is a satisfactory 'rule of thumb' in designing simple thermistor voltage-divider circuits.

1.2.7 Resolution of the Thermistor Thermometer

The next step is to check the resolution of the temperature measuring circuit. Figure 1.5 shows that the slope of the transfer characteristic varies with temperature so we would expect the resolution to vary with temperature. In order to resolve a change of 1^{o}C, we must ensure that the output of the A-D converter changes by at least one count for each degree change in temperature.

As a quick check, we can determine the change in A-D reading for a 5^o change at the extremes and centre of the operating range by plugging thermistor resistance values into equation 1.2.

For example, the thermistor resistance at –30^{o}C is 173.9KΩ. Using equation 1.2, the corresponding output from the A-D is determined to be 228.

At -25^{o}C, the thermistor resistance is 128.5Ω and the A-D output 220. Thus, in the region of –30^{o}C, a 5^o change in temperature resulted in an A-D reading change of $228 - 220 = 8$, something more than 1 count per degree.

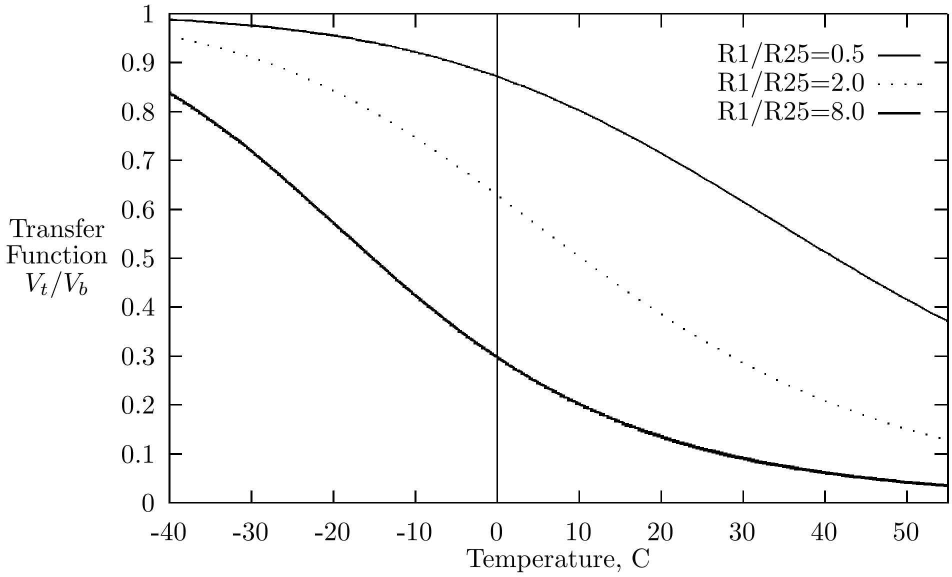


Figure 1.5: Thermistor Divider

Repeating this at the centre of the operating range, +10^{o}C and at the extreme hot temperature, +50^{o}C, we have the following results:

Change in Temperature, ^{o}C ^{o}C	**Change in A-D Reading**	**Approximate Resolution Counts per degree**
–30 to –20	8	1.6
+10 to +15	12	2.4
+45 to +50	7	1.4

Figure 1.6: Temperature Measurement Block Diagram

Throughout the entire temperature range, the A-D reading changes by at least one count per degree. Thus the microprocessor can resolve changes of 1^{o}C throughout the full range of temperature.

In the centre of the operating range, the A-D output changes by more than 2 steps per degree, so it is possible to resolve at least 0.5^{o}C in this region.

This analysis is aprproximate, and to be certain the resolution is sufficient, it should be checked throughout its range. Fortunately, this is easily done with a spreadsheet, plotting routine such as Matlab or GNUPlot, or interactive computer program such as some dialect of BASIC.

1.3 Engineering Tool: BASIC Program

The BASIC programming language has many limitations and in general should not be used to construct production software. However, BASIC has always been popular among engineers for constructing throwaway programs – programs that are used as an investigative tool for some technical problem.

Checking the resolution of the thermistor thermometer is a good example of a problem suited for a BASIC program. It's also a useful technique for generating the entries for a lookup table of A-D readings versus temperature.

A BASIC program of this type is shown below, which was used to generate the values in figure 1.7.

```
REM This program generates a table of values
REM  corresponding to A-D readings from 0
REM  to 255.
REM
REM The thermistor function is represented
REM  by its Steinhart-Hart coefficients.
REM Notice that resistance is in Kilohms
REM
a = .002772
b = 2.524 * 10 ^ (-4)
c = 3.042 * 10 ^ (-7)

REM The series resistance is 20k ohms
R1 = 20

REM Open the disk file for printing
OPEN "thermist.dat" FOR OUTPUT AS #1

REM This is the calculation loop
FOR Nad= 1 TO 255
Rt = R1 * (Nad / (256 - Nad))
Ta = 1 / (a + (b * LOG(Rt)) + (c * (LOG(Rt)) ^ 3))
Tc = Ta - 273
PRINT #1, Nad, INT(Tc);
NEXT Nad

REM Close the disk file
CLOSE
```

This program calculates the value of temperature that corresponds to values of Nad from 1 to 255, and prints them to a disk file, 'thermist.dat'. This file may then be pasted into the source code of the program that reads the A-D converter and calculates temperature.

In practice, there are probably additional requirements of the printing format. For example, for incorporation in 68HC11 assembly language source code, each value would be prefaced by 'FCB'. This mneumonic causes the assembler to recognize a number as a byte-size constant that should be allocated storage. As well, negative numbers cannot be represented directly as bytes, and would have to be converted to some other representation, such as 2's complement form.

1	197	2	161	3	142	4	129	5	120	6	113	7	107	8	102	9	98	10	94
11	91	12	88	13	85	14	83	15	80	16	78	17	76	18	74	19	72	20	71
21	69	22	68	23	66	24	65	25	64	26	62	27	61	28	60	29	59	30	58
31	57	32	56	33	55	34	54	35	53	36	52	37	51	38	50	39	50	40	49
41	48	42	47	43	46	44	46	45	45	46	44	47	44	48	43	49	42	50	42
51	41	52	41	53	40	54	39	55	39	56	38	57	38	58	37	59	37	60	36
61	35	62	35	63	34	64	34	65	33	66	33	67	32	68	32	69	31	70	31
71	31	72	30	73	30	74	29	75	29	76	28	77	28	78	27	79	27	80	27
81	26	82	26	83	25	84	25	85	25	86	24	87	24	88	23	89	23	90	23
91	22	92	22	93	21	94	21	95	21	96	20	97	20	98	20	99	19	100	19
101	18	102	18	103	18	104	17	105	17	106	17	107	16	108	16	109	16	110	15
111	15	112	15	113	14	114	14	115	14	116	13	117	13	118	13	119	12	120	12
121	12	122	11	123	11	124	11	125	10	126	10	127	10	128	9	129	9	130	9
131	8	132	8	133	8	134	7	135	7	136	7	137	6	138	6	139	6	140	5
141	5	142	5	143	4	144	4	145	4	146	3	147	3	148	3	149	3	150	2
151	2	152	2	153	1	154	1	155	1	156	0	157	0	158	0	159	-1	160	-1
161	-1	162	-2	163	-2	164	-2	165	-3	166	-3	167	-3	168	-4	169	-4	170	-4
171	-5	172	-5	173	-5	174	-6	175	-6	176	-6	177	-7	178	-7	179	-7	180	-8
181	-8	182	-8	183	-9	184	-9	185	-10	186	-10	187	-10	188	-11	189	-11	190	-11
191	-12	192	-12	193	-13	194	-13	195	-13	196	-14	197	-14	198	-14	199	-15	200	-15
201	-16	202	-16	203	-17	204	-17	205	-17	206	-18	207	-18	208	-19	209	-19	210	-20
211	-20	212	-21	213	-21	214	-22	215	-22	216	-23	217	-23	218	-24	219	-24	220	-25
221	-25	222	-26	223	-26	224	-27	225	-28	226	-28	227	-29	228	-29	229	-30	230	-31
231	-32	232	-32	233	-33	234	-34	235	-35	236	-35	237	-36	238	-37	239	-38	240	-39
241	-40	242	-41	243	-43	244	-44	245	-45	246	-47	247	-48	248	-50	249	-52	250	-54
251	-57	252	-60	253	-64	254	-69	255	-78										

Figure 1.7: A/D – Temperature Lookup Table

For the lookup table to have sufficient resolution, each entry between the temperatures of -40 to $+50^o$C should change by no more than 1^oC. This is in fact the case, so the thermometer meets its requirement.

Outside this range, the temperature readings jump by more than 1^oC, reflecting the fact that the transfer characteristic, figure 1.5, is less steep at high and low temperatures.

1.3.1 Estimating the Production Accuracy

In a production setting, where units are being assembled from batches of components, we must be able to predict the accuracy of the circuit when different resistors and thermistors of the same type are installed in the circuit.

A number of factors affect the accuracy of the thermistor measuring circuit:

- Tolerance of the thermistor resistance R_{25}, resistance at 25^{o}C
- Tolerance of the thermistor resistance due to variation in the β of the thermistor
- Tolerance in the value of R_1
- Accuracy of the A-D converter
- Accuracy of the equation relating R_t to temperature

To determine the worst case variation in temperature reading in a production batch, we should choose the worst case for each of these variations and add them together. However, this may be unnecessarily stringent.

If the tolerances are independent and uncorrelated, then a good estimate of the worst case variation is found by taking the RMS sum of the individual variations.

Effect of tolerance in thermistor resistance and sensitivity:

Here we assume that the variation in thermistor resistance (a constant 3 %) and the variation in sensitivity (β tolerance from figure 1.2) can both occur together.

Then we calculate the effect on the A-D reading at the lowest, middle and highest operating temperatures. Finally, this gives us the expected variation in A-D reading due to tolerances in the thermistor characteristics. All resistor values are in KΩ.

Temperature, ^{o}C	Nominal R_t	R_t Tolerance	β Tolerance	R_t	N_{AD}	ΔN_{AD}
–30	173.9	+3%	+2.16%	182.8	229.8	
		–3%	–2.16%	164.9	227.4	2.4
+10	19.86	+3%	+2.16%	20.55	129.2	
		–3%	–2.16%	19.16	124.8	4.5
+50	3.606	+3%	+2.16%	3.742	40.2	
		–3%	–2.16%	3.470	37.7	2.5

Effect of tolerance in Divider Resistor R_1

In this case, we assume that R_1 is a resistor with 1% tolerance. Then, following a similar procedure to the above, we have the following table:

Temperature, ^{o}C	Nominal R_1	R_1 Tolerance	R_t	R_t	N_{AD}	ΔN_{AD}
–30	20	+1%	20.2	173.9	228.5	
		-1%	19.8	173.9	228.9	0.4
+10	20	+1%	20.2	19.86	126.4	
		-1%	19.8	19.86	127.7	1.3
+50	20	+1%	20.2	3.606	38.62	
		-1%	19.8	3.606	39.28	0.6

Analogue-Digital Converter Accuracy

The 68HC11 A-D data specifies that the absolute accuracy of the A-D is ± 1 least significant digit. The value of ΔN_{AD} is therefore 1.0 for the A-D converter.

Total Accuracy

Summing up the errors at different temperatures, we have the results shown in the following table:

We previously determined that the sensitivity of the measuring circuit varies with temperature (figure 1.6), so it is reasonable to express the variation in A-D reading as an equivalent temperature. For example, if the accuracy is ± 4.74 counts at +10^{o}C, and the resolution is 2.4 counts per degree, then the accuracy is approximately $2.8/2.4 = \pm 2.8^o$C. This is shown in the last two columns for equivalent Worst Case error and equivalent RMS error.

This is not good news: a thermistor-based temperature measuring circuit using these components will have a variation in accuracy by nearly $\pm 3^o$C.

Temperature, ^{o}C	Worst Case Error, A-D Counts	RMS Error, A-D Counts	Worst Case Error, Equivalent Degrees	RMS Error, Equivalent Degrees
–30	3.8	2.6	2.3	1.6
+10	6.75	4.74	2.8	1.9
+50	4.1	2.75	2.9	1.9

Figure 1.8: Thermistor Circuit Measurement Accuracy

How could we improve the accuracy of the circuit? Most of the error is due to the variation in R_{25} and β, so these must be more tightly controlled to improve the accuracy. More tightly specified thermistors (± 0.1% is available) are available (albeit for a significantly higher cost) and might be suitable.

Alternatively, we can introduce a calibration procedure that measures R_{25} and β, and compensates for their variation. This has attractions for the amateur, but is time consuming and requires additional microprocessor software. Before getting involved in that, it would be wise to investigate other types of temperature transducers, as we do in the next section.

1.3.2 Thermistor Self-Heating and Thermal Runaway

When a current is passed through a thermistor, as it is in this circuit to measure the resistance, a certain amount of heat is dissapated in the thermistor by ohmic heating. Because the thermistor has a Negative Coefficient of Temperature, this heat will lower the resistance and further increase the device current.

There is a potential here for thermal runaway, in which the heat changes the properties of the device so that it dissapates yet more heat until eventually the device incinerates itself.

It would be prudent engineering to ensure first that thermal runaway will not happen in this particular circuit and second, that self-heating does not materially affect the accuracy of the temperature reading.

The voltage-current characteristic of our 10KΩ thermistor is shown in figure 1.9.

A constant resistor would appear as a straight line passing through the origin with slope proportional to the resistance. The thermistor resistance varies with current. At an ambient temperature of 25^{o}C, the resistance actually becomes negative at forward currents greater than about 7mA.

The stability of a circuit containing a thermistor depends on the characteristics of the source, which may be indicated by a load line superimposed on the thermistor characteristic, as indicated in the examples of figure 1.10.

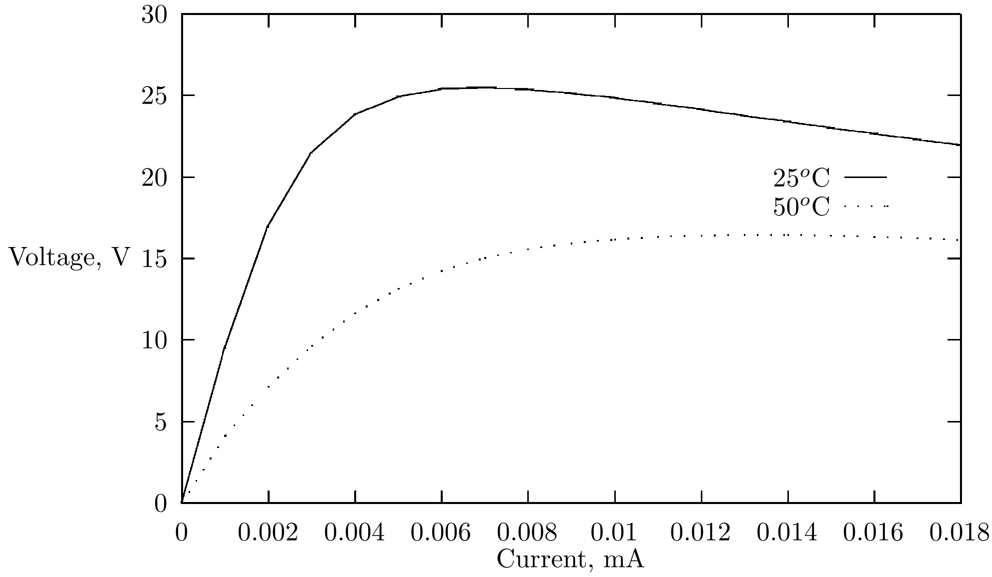
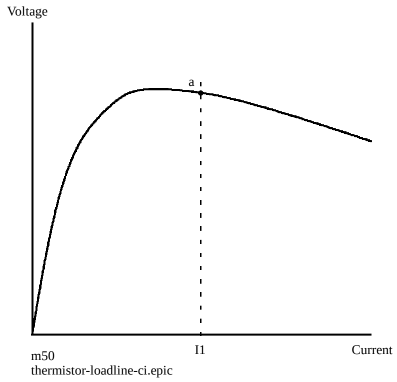
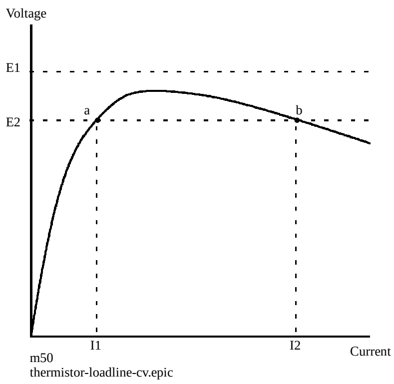


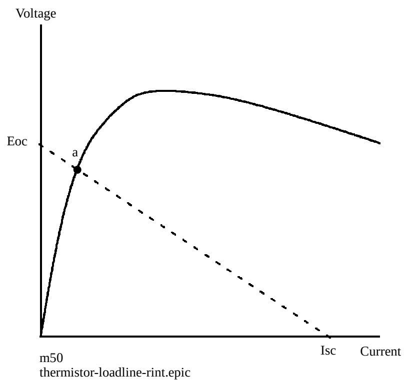
Figure 1.9: Thermistor Voltage-Current Characteristic



(a) Constant Current Thermistor Source



(b) Constant Voltage Thermistor Source



(c) Voltage Source, Internal Resistance

Figure 1.10: Thermistor Sources

The characteristic of a constant current source (figure 1.10(a) intersects the thermistor characteristic at one and only one point so the circuit will be stable.

On the other hand, the voltage source E1 in figure 1.10(b) exceeds the maximum possible voltage of the thermistor and so will certainly cause thermal runaway. The voltage source E2 in figure 1.10(b) is conditionally stable. It intersects the positive resistance portion of the curve at point a, so it is a stable operating point. A second possible stable operating point is at b.

The characteristic of a voltage source with an internal resistance is shown in figure 1.10(c). In this case, the stability of the circuit is dependent on the magnitude of the voltage source and slope of the loadline. Since the loadline intersects the thermistor characteristic at only one point, the values shown would result in a stable circuit.

The situation for our application circuit, with a 5 volt source and 20KΩ resistance, is shown in figure 1.11.

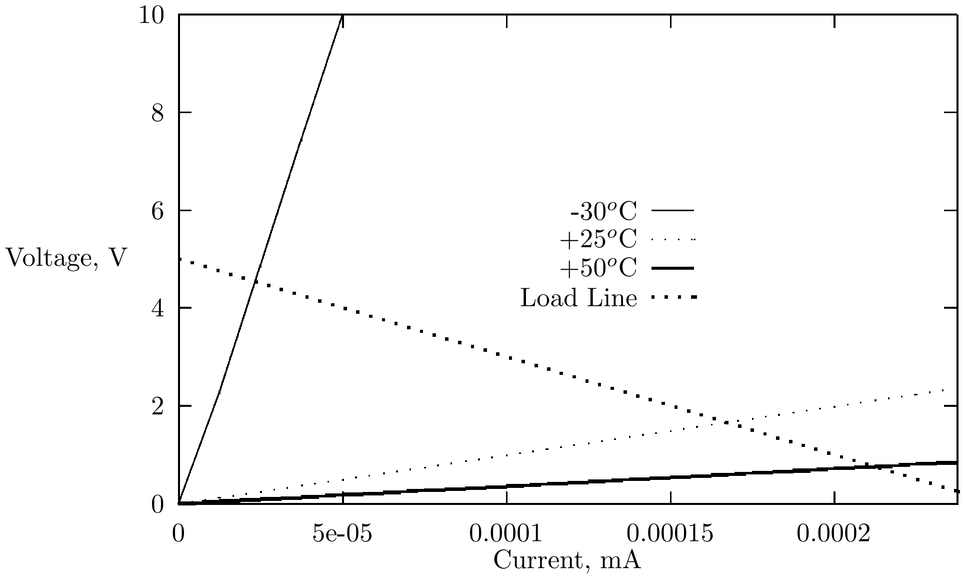


Figure 1.11: Thermistor Circuit Load Line

The load line intersects the thermistor characteristic at one point only for the thermistor at three different temperatures so the circuit will be stable. Further, we can deduce that the self-heating effect is minimal because the thermistor characteristics are essentially straight lines at these voltages and currents.

1.3.3 Plotting the Thermistor VI Characteristic

Determining the voltage-current characteristic of a thermistor presents an interesting challenge. The basic concept is simple: we wish to force a current through the thermistor and measure the resultant voltage across it. However, the current induces heat in the thermistor, which changes its resistance, changing its voltage. Unfortunately, the resulting equations are not amenable to a simple analytical solution.

The relevant equations are as follows:

The voltage across the thermistor is simply

$$v_t = i_t R_T \tag{1.7}$$

where v_t and i_t are the thermistor voltage and current.

The power generated in the thermistor is likewise simply

$$p_t = i_t^2 R_T \tag{1.8}$$

The internal temperature of the thermistor is equal to the ambient temperature T_A plus the heat generated across the thermal resistance θ of the thermistor by the thermistor power P_t, according to equation 1.9.

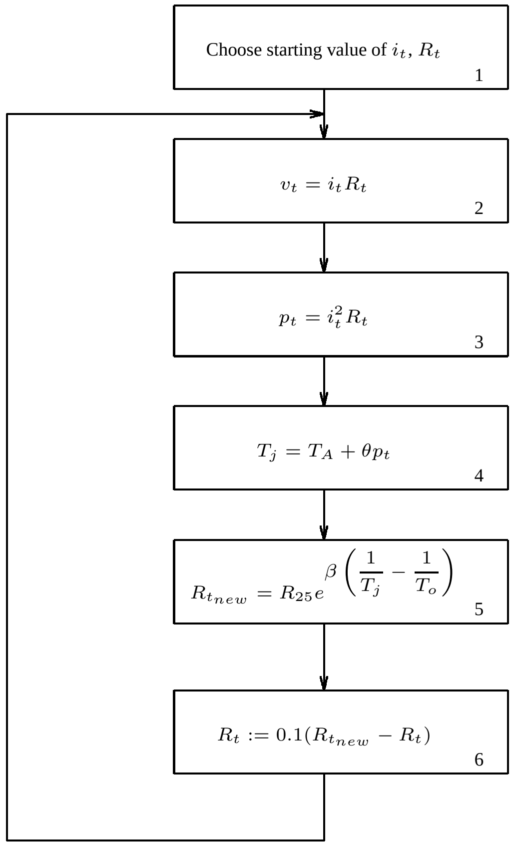
$$T_j = T_A + \theta P_d \tag{1.9}$$

where P_d is P_t in this case.

Finally, from section 1.2.4 we have equation 1.3 for the thermistor resistance:

$$R_T = R_{25}\, e^{\beta\left(\frac{1}{T} - \frac{1}{T_o}\right)} \tag{1.10}$$

Unfortunately, combining equations 1.7 through 1.10 does not lead to a simple algebraic solution. The voltage corresponding to a given current can be found iteratively, however, using the iterative approach of figure 1.12.



m70
therm-vi-fbwchart.epic

Figure 1.12: Thermistor Voltage from Current

We start with a value of current and an approximate value of thermistor resistance. We then determine the heating effect and calculate a new value of resistance. This new value of thermistor resistance is used to improve the original estimate. We repeat the iterative loop until the values of thermistor resistance converge to a solution. Once the thermistor resistance is known the thermistor voltage is simply the current times the thermistor resistance.

In order to ensure that the iteration converges, it may be necessary to adjust the factor by which the estimate of thermistor resistance is improved with each iteration: a factor of 0.1 in the example shown. If the result oscillates with each iteration, reduce this factor.

1.3.4 Engineering Tool: Generating Graphs with GNUPLOT

Plotting of engineering equations is greatly simplified with a computer or programmable graphic calculator. One of the common programs for generating graphs is `GNUPLOT`, a public domain utility with a large number of features. `GNUPLOT` is available for a large number of machines, ranging from DOS computers to engineering workstations.

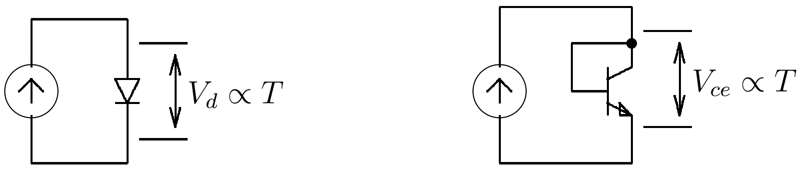
`GNUPLOT` may be used interactively from a command line or may read a file of instructions. The `GNUPLOT` file to generate the graph of figure 1.5 is shown below:

```
  #List of commands to plot thermistor divider curves
set terminal latex # direct graphical output to a file
set output "thermistor-divider.latex" # establish name of the file
set xlabel "Temperature, C" # label x axis
set ylabel "Transfer Function" # label y axis
plot [-40:55]
1.0/(1.0+(5.0/(10.0*(2.71818**(3977.0*(1.0/(273.0+x)-1.0/298.0)))))
title ’R1/R25=0.5’ with lines, \

  1.0/(1.0+(20.0/(10.0*(2.71818**(3977.0*(1.0/(273.0+x)-1.0/298.0))))
title ’R1/R25=2.0’ with lines, \

  1.0/(1.0+(80.0/(10.0*(2.71818**(3977.0*(1.0/(273.0+x)-1.0/298.0))))
title ’R1/R25=8.0’ with lines \
set terminal x11 # show output on terminal
replot # redo the plot
```

This command file contains instructions to plot the function with output first to a disk file **”thermistor-divider.latex”** (for incorporation in another document) and then to the X windows screen. The # character preceeds comments, which are ignored.

1.4 The Diode Temperature Sensor

A forward biased silicon diode or diode-connected transistor (figure 1.13), operated at constant current, will exhibit a change in voltage of some -2.2mV per ^{o}C. In contrast to the non-linear behaviour of the thermistor, this constant coefficient makes the diode a very attractive temperature sensor.



m100
diode-sensor.epic

Figure 1.13: Diode and Diode-Connected Transistor Temperature Sensors

However, there are some design issues that must be dealt with to make a practical temperature measuring circuit.

In the first place, the 8-bit A-D converter of the microprocessor requires $5/255 = 19$ mV per step when operated from reference voltages of +5 volts and ground. The diode 2.2 mV/^{o}C rate of change of voltage with temperature is much less than this, and must be amplified. Second, the forward biased diode has a standing voltage of some 600 millivolts which must be subtracted out from the measurement. Together, these requirements may be met with a simple operational amplifier circuit.

Before launching into a design of suitable circuitry, it is appropriate to have a careful look at the characteristics of the diode-connected transistor to determine whether it will make an accurate temperature sensor.

1.4.1 Diode Temperature Characteristics

Nelson-Jones [17] and Henderson [18] describe analogue thermometers using silicon diodes. In both cases, the authors found that the temperature coefficient was only approximately 2.2mV per ^{o}C and depended on both the type of diode and the the forward voltage. This implies that the diode must be selected for the desired characteristics and/or that the instrument must have facilities for compensating for the different coefficients. Obviously, both these are undesireable if they can be avoided.

According to Sachse [19], page 106):

> The forward current across any p-n junction consists of three major components: diffusion current or drift current, generation recombination current, and surface leakage current. Each of them is determined by a different mechanism and temperature dependence results in complex compound behaviour. Verster [20] has shown that any semiconductor material operated as a transistor simplifies these relationships since only the diffusion component of the emitter is under consideration, while the generation recombination and surface leakage currents are drained away as base current.

This implies that the diode-connected transistor will have a more predictable temperature characteristic. Indeed, Motorola used to manufacture a series of transistors, the MTS102-MTS105 [21] which were specifically intended for temperature measurement. (Cover of the MTS102 Data Sheet is shown in figure 1.14)[4]. Although the MTS102 is no longer available, the other small signal transistors will have similar specifications.

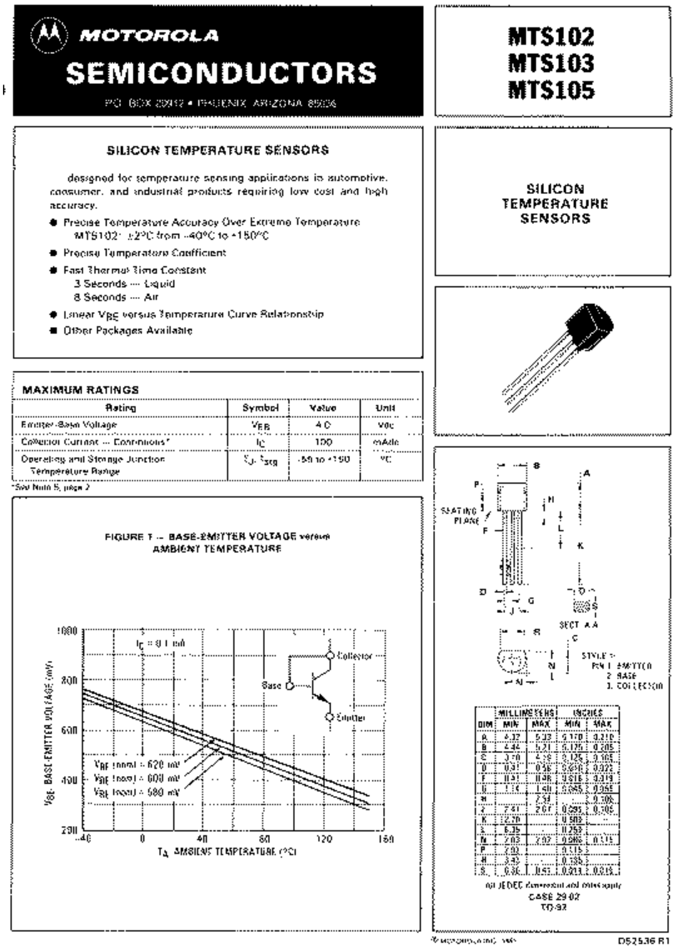


Figure 1.14: The MTS Temperature Measurement Transistor

1.4.2 Transistor Base-Emitter Voltage Function

The classic equation relating the forward voltage to forward current in a forward biased diode or connected transistor is

$$I_c(T) = I_s(T)\, e^{\left(\dfrac{qV_{be}}{nkT}\right)} \tag{1.11}$$

[4]Figures 1.14 and 1.15 are ©Motorola Inc, 1981.

where

T is the absolute temperature in ^{o} Kelvin
I_c is the collector current
I_s is the reverse saturation current
q is the electron charge, 1.6×10^{-19} coulomb
V_{be} is the base-emitter voltage
n is a process constant which depends on the construction technique, typically $\approx$1
K is Boltzmann's constant, $1.38{\times}10^{-23}$ joule/^{o}K

If we have access to values of the base-emitter voltage vs collector-emitter current for a particular transistor, we can solve for the values of n and I_s for that transistor. In this example, we shall use the graph of figure 1.15, adapted from the Motorola data sheet for the MTS102.

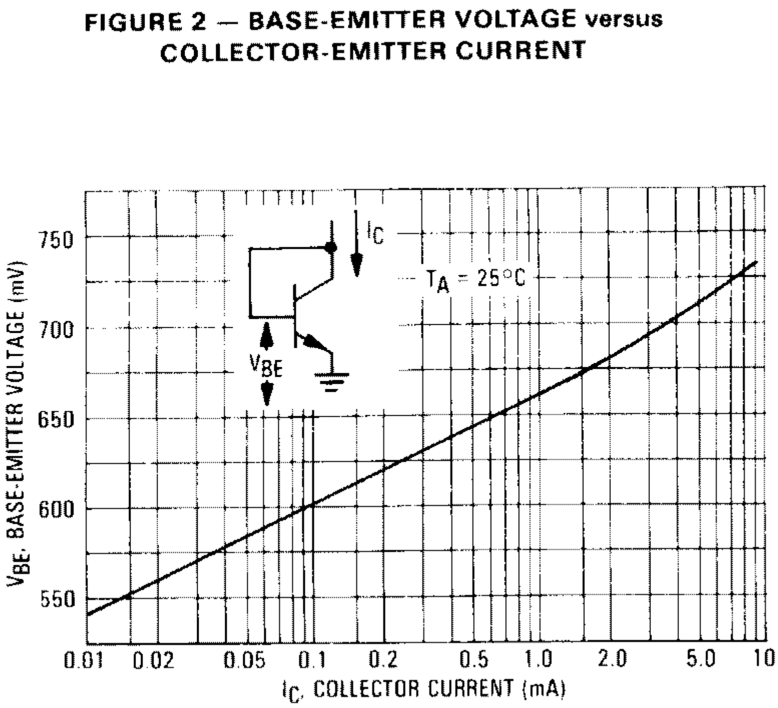


Figure 1.15: V_{be} vs I_C for the MTS102

Calculating the Process Constant n

Assuming that the temperature is constant for this calculation, $I_c(T)$ and $I_s(T)$ can be written simply as I_c and I_s.

Taking the natural log of both sides of equation 1.11, we have

$$\ln I_c = \ln I_s \; + \; \frac{qV_{be}}{nkT}$$

For two different points on the V_{be} I_c characteristic we have

$$\begin{aligned} \ln I_{c1} &= \ln I_s \; + \; \frac{qV_{be1}}{nkT} \\ \ln I_{c2} &= \ln I_s \; + \; \frac{qV_{be2}}{nkT} \end{aligned}$$

Subtracting the two equations,

$$\ln I_{c2} - \ln I_{c1} \quad = \quad \frac{q}{nkT}(V_{be2} - V_{be1})$$

or

$$\ln \frac{I_{c2}}{I_{c1}} \quad = \quad \frac{q}{nkT}(V_{be2} - V_{be1})$$

Two suitable data points from the graph of figure 1.15 are 0.1 mA and 1.0 mA, a decade appart in current. Then we have

I_{c1}, 0.1 mA V_{be1}, 600 mV
I_{c2}, 1.0 mA V_{be2}, 660 mv

Substituting these values in equation 1.12, we have:

$$\ln 10 \quad = \quad \frac{1.6 \times 10^{-19}}{n \times 1.38 \times 10^{-23} \times 298}(0.660 - 0.600)$$

Solving for n,

$$n \quad = 1.014$$

Solving for the Saturation Current I_s

We are now in a position to substitute a pair of V_{be}, I_c values in equation 1.11, along with the newly calculated value of process constant, in order to solve for the saturation current.

$$\begin{aligned} I_c(T) &= I_s \, e^{\left(\frac{qV_{be}}{nkT}\right)} \\ &= I_s \, e^{\left(\frac{1.6 \times 10^{-19} \times 0.6}{1.014 \times 1.38 \times 10^{-23} \times 298}\right)} \end{aligned}$$

Solving for I_s, we have

$$I_s \quad = \quad 1.014 \times 10^{-14} \text{ amps}$$

1.4.3 Base-Emitter Voltage Function of Temperature

When using the diode-connected transistor as a temperature sensor, our primary concern is the temperature coefficient of base-emitter voltage. Taking the natural logarithm of both sides of equation 1.11 and solving for V_{be}, we have

$$V_{be}(T) = \frac{nKT}{q} \ln \frac{I_c(T)}{I_s(T)} \tag{1.12}$$

where the collector current and saturation current have both been shown as functions of temperature T to emphasize that they change when the temperature changes. In other words, we cannot treat I_c and I_s as constants and differentiate this equation with respect to temperature T to determine the temperature coefficient.

However, if the base-emitter voltage is known at some reference temperature T_o, the base-emitter voltage at some other temperature T may be calculated by the following [22, 23, 24]:

$$V_{be}(T) = V_{G0}\left(1 - \frac{T}{T_o}\right) + V_{beo}\left(\frac{T}{T_o}\right) + \frac{nkT}{q} \ln\left(\frac{T_o}{T}\right) + \frac{kT}{q} \ln\left(\frac{I_c}{I_o}\right) \tag{1.13}$$

where the additional terms are

V_{G0}, the bandgap voltage of silicon, typically 1.22V
T_o, a reference temperature in ^{o} Kelvin
V_{beo}, the base-emitter voltage at the reference temperature
I_o, the reference collector current

For example, consider that a diode-connected transistor has a base-emitter voltage of 0.600 volts at a collector current of 0.1 mA and temperature of 20^{o}C. What is the base-emitter voltage at 100^{o}C?

The values of the parameters are:

$$\begin{aligned}
V_{GO} &= 1.22 \text{ volts} \\
T &= 273 + 100 \\
&= 373^o\text{K} \\
T_o &= 273 + 20 \\
&= 293^o K \\
V_{beo} &= 0.600 \text{ volts} \\
n &= 1.014 \\
k &= 1.38 \times 10^{-23} \\
q &= 1.6 \times 10^{-19} \\
I_c &= 0.1 \times 10^{-3} \\
I_o &= 0.1 \times 10^{-3}
\end{aligned}$$

Substituting these in equation 1.13, we have

$$\begin{aligned}
V_{be} &= V_{G0}\left(1-\frac{T}{T_o}\right)+V_{beo}\left(\frac{T}{T_o}\right)+\frac{nkT}{q}\ln\left(\frac{T_o}{T}\right)+\frac{kT}{q}\ln\left(\frac{I_c}{I_o}\right) \\
&= 1.22\left(1-\frac{373}{293}\right)+0.600\left(\frac{373}{293}\right) \\
&\quad +\frac{1.014\times 1.38\times 10^{-23}\times 373}{1.6\times 10^{-19}}\ln\left(\frac{293}{373}\right) \\
&\quad +\frac{1.38\times 10^{-23}\times 373}{1.6\times 10^{-19}}\ln\left(\frac{0.1\times 10^{-3}}{0.1\times 10^{-3}}\right) \\
&= -0.333\ +\ 0.763\ +\ 0.0078\ +0.0 \\
&= 0.437 \text{ volts}
\end{aligned}$$

This agrees with the base-emitter vs temperature value shown on the data sheet for the MTS102 transistor.

Notice that the third term contributes only a very small amount to the final value. The fourth term has no effect because the collector current is equal to the reference current.

1.4.4 Base-Emitter Voltage Temperature Coefficient

The first two terms of equation 1.13 are linear in T and contribute most of the change in base-emitter voltage with temperature. The third term contributes a small non-linear value, and the fourth term may be ignored if the collector current is constant with temperature.

Consequently, if we assume that the diode is driven from a constant current source, the temperature coefficient of base-emitter voltage may be determined by differentiating the first three terms of equation 1.13.

From equation 1.13, the differential of the first two terms is

$$V_{be} \simeq V_{G0}-V_{G0}\left(\frac{T}{T_o}\right)+V_{beo}\left(\frac{T}{T_o}\right)$$

then

$$\frac{dV_{be}}{dT} = 0-\frac{V_{G0}}{T_o}+\frac{V_{beo}}{T_o} \tag{1.14}$$

The differential of the third term is a bit more involved: We wish to find

$$\frac{d}{dT}\,\frac{nkT}{q}\ln\left(\frac{T_o}{T}\right)$$

We begin with the identity

$$\frac{duv}{dx} = u\frac{dv}{dx}+v\frac{du}{dx}$$

and substitute

$$\begin{aligned}
u &= \frac{nkT}{q} \\
v &= \ln\left(\frac{T_o}{T}\right)
\end{aligned}$$

to obtain

$$\frac{d}{dT}\,\frac{nkT}{q}\ln\left(\frac{T_o}{T}\right) \quad = \quad \frac{nkT}{q}\times\frac{d}{dT}\,\ln\frac{To}{T}\,+\,\ln\frac{T_o}{T}\times\frac{d}{dT}\,\frac{nkT}{q} \tag{1.15}$$

Using the identity

$$\frac{d}{dx}\,\ln u = \frac{1}{u}\frac{du}{dx}$$

we have that

$$\begin{aligned}\frac{d}{dT}\ln T_oT &= \frac{1}{T_o/T}\times\frac{d}{dT}\,\frac{T_o}{T}\\ &= \frac{T}{T_o}\times -\frac{T_o}{T^2}\\ &= -\frac{1}{T}\end{aligned}$$

We also have that

$$\frac{d}{dT}\frac{nkT}{q} \quad = \quad \frac{nk}{q}$$

Finally, substituting in equation 1.15, the differential of the third term is given by

$$-\frac{nkT}{q}\times\frac{1}{T}+\ln\frac{T_o}{T}\times\frac{nk}{q} \quad = \quad \frac{nk}{q}\left(\ln\frac{T_o}{T}-1\right)$$

Putting this result together with the linear coefficient terms, we have the complete expression for the rate of change of V_{be} with temperature:

$$\frac{dV_{be}}{dT} = -\frac{V_{G0}}{T_o}+\frac{V_{beo}}{T_o}+\frac{nk}{q}\left(\ln\frac{T_o}{T}-1\right) \tag{1.16}$$

For example, the data sheet for the MTS102 temperature sensor transistor gives V_{beo} as 0.595 volts at 25^{o}C. Determine the temperature coefficient of V_{be}.

Substituting V_{GO}=1.22V, V_{beo}=0.595, T_o=298^o and T=298^o Kelvin in equation 1.16, we have

$$\begin{aligned}\frac{dV_{be}}{dT} &= -\frac{1.22}{298}+\frac{0.595}{298}+\frac{1.014\times 1.38\times 10^{-23}}{1.6\times 10^{-19}}\left(\ln\frac{298}{298}-1\right)\\ &= -4.094\times 10^{-3}+1.997\times 10^{-3}-0.087\times 10^{-3}\\ &= -2.184\,\text{mV}/^o\text{C}\end{aligned}$$

The value from the data sheet is -2.265 mV/^{o}C, which is within 3% of the calculated value.

Plugging temperature values of -40^{o}C and +50^{o}C into equation 1.16, we can calculate the transistor temperature coefficients at the extremes of operation:

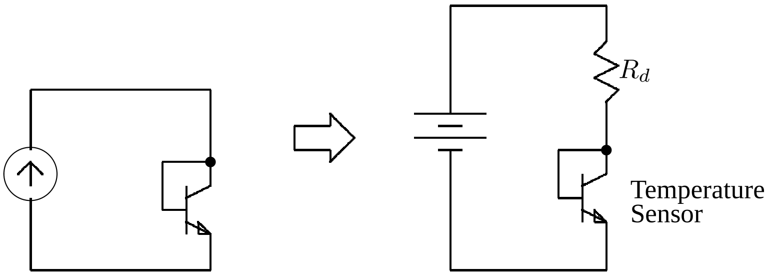
Temperature	Coefficient of Temperature	Error related to coefficient at 25^{o}C
–40	-2.163 mV/^{o}C	-0.1%
+25	-2.184 mV/^{o}C	+0.0%
+50	-2.191 mV/^{o}C	+0.3%

Evidently the temperature coefficient of base-emitter voltage may be relied upon to be essentially constant over the temperature range we intend to use it.

Sensor Current Source

In section 1.4.4, we assumed that the diode would be driven from a constant current source, and that the current through the diode would not change with temperature. Ideally, then, our circuit should match these assumptions.

The simplest possible circuit is a resistor from a positive voltage source as shown in figure 1.16.



m100
simple-diode-source.epic

Figure 1.16: Simple Diode Current Source

If the voltage across the transistor sensor is more or less constant at 0.6 volts, then if the supply voltage is +5 volts, the voltage across the resistor will be more or less constant at 4.4 volts and the current through the sensor will be more or less constant as well. Now, as we know, the voltage across the sensor does vary, so we have to determine whether the variation is enough to affect the accuracy of the sensor voltage.

First of all, we should calculate the value of the series resistor R_d:

$$R_d = \frac{V_{cc} - V_{be}}{I_c} \tag{1.17}$$

For the purpose of calculating the resistance, assume I_c is 100μA and V_{be} is 0.6 volts. Then

$$\begin{aligned} R_d &= \frac{5\text{V} - 0.6\text{V}}{100 \times 10^{-6} \text{ amps}} \\ &= 44K\Omega \end{aligned}$$

Now we need to determine the worst case variation in current due to changes in the value of V_{be}. From figure 1.17, the worst case change in V_{be} is to 0.742 volts, which occurs at a temperature of –40^{o}C. Repeating the calculation of equation 1.17 for V_{be}= 0.742 volts, we find that $I_c \approx 96\mu$amps. This value is only approximate because its change further affects the assumed value of V_{be}. However, as we shall show, the effect is slight enough to be neglected.

Now, how much of an effect does a change of transistor current from 100 μamps to 96 μamps have on the value of V_{be}? From equation 1.13, the current affects the base-emitter voltage according to the expression

$$\Delta V_{be} = \frac{kT}{q} \ln\left(\frac{I_c}{I_o}\right)$$

where I_c is the collector current and I_o is the original reference collector current. Thus the change in V_{be} due to change in collector current is

$$\begin{aligned}
\Delta V_{be} &= \frac{kT}{q} \ln\left(\frac{I_c}{I_o}\right) \\
&= \frac{1.38 \times 10^{-23} \times 233}{1.6 \times 10^{-19}} \ln \frac{96 \times 10^{-6}}{100 \times 10^{-6}} \\
&= -0.803\,\mathrm{mV}
\end{aligned}$$

This is not enough to materially effect our original assumption of 0.742 volts for V_{be}, so our assumption of I_c=96μamps is unchanged.

The temperature coefficient of the transistor sensor is in the order of -2.2 mV/^{o}C, so this change of -0.8 mV in V_{be} is equivalent to approximately a third of a degree C. For greater accuracy, a different current source would be required.

1.4.5 Interface Circuit for a Transistor Temperature Sensor

We are now in a position to design the signal conditioning circuitry to match the output of the temperature sensor to the input of the microprocessor A-D converter.

The change in voltage across the transistor sensor is small compared to the input signal required to actuate the A-D converter. The change across the sensor is about 2.2 millivolts per degree C. Each step of the A-D converter output requires a change of 5 volts$/256$ steps $= 19.5$ millivolts per step. Moreover, the sensor has a standing voltage of about 600 millivolts across it. Simply amplifying the voltage across the sensor will generate a signal outside the range of the A-D converter.

Consequently, the interface circuit must both amplify the change in voltage across the temperature sensor and subtract the standing voltage.

The interface circuit couples the sensor output signal to the A/D input signal, so the first step is to establish their ranges.

Range of the sensor signal

For the measurement of ambient temperature in Canada, a suitable range might be -40^{o}C to +60^{o}C. The total change in temperature is thus a nice round number: 100^{o}C.

Range of the A-D signal

The 8 bit A-D converter has a total range of $2^8 = 256$ steps. A suitable match between the sensor and the A-D would be to assign 2 A-D steps per degree of temperature change. This would then use up 200 of the 255 steps, leaving a margin of 27 steps at each end of the range, and the input to the A-D converter ranges between step 27 and step 227. This arrangement provides a resolution of 2 A-D steps per degree, or 0.5^{o}C per A-D step, quite acceptable for a domestic thermometer.

From the data sheet for the MTS102, the temperature coefficient is -2.265 mV/^{o}C and the value of V_{be} is 595mV at 25^{o}C.

Using these specs, at -40^oC, the sensor voltage is

$$\begin{aligned} & (-40+25) \times -2.265 + 595 \\ = \; & 742 \text{ millivolts} \end{aligned}$$

At $+60^o$C the sensor voltage is

$$\begin{aligned} & (60-25) \times -2.265 + 595 \\ = \; & 515 \text{ millivolts} \end{aligned}$$

The corresponding voltages at the input to the A-D converter are: At step 27 (-40^oC):

$$\begin{aligned} & 27/255 \times 5 \\ = \; & 0.529 \text{ volts} \end{aligned}$$

At step 227 ($+60^o$C):

$$\begin{aligned} & 227/255 \times 5 \\ & 4.451 \text{ volts} \end{aligned}$$

These signal swings are summarized in figure 1.17.

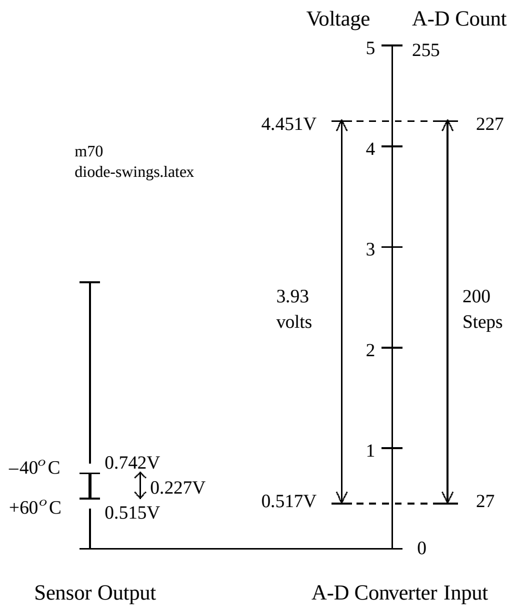
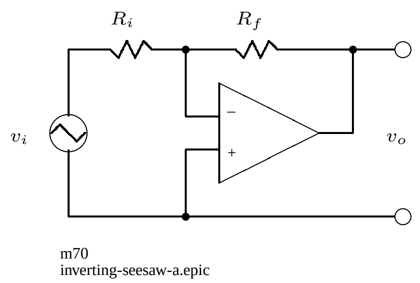
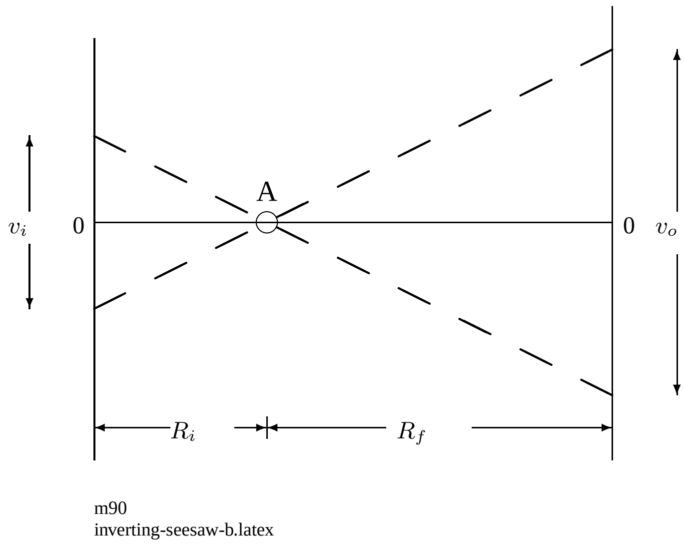


Figure 1.17: Sensor and A-D Signals

To design a suitable amplifier circuit, consider the operational amplifier circuit of figure 1.18(a).



(a) Inverting Amplifi er



(b) Inverting Amplifi er Seesaw Diagram

Figure 1.18: Inverting Amplifier

The inverting operational amplifier circuit has a voltage gain of

$$\frac{v_o}{v_i} = -\frac{R_f}{R_i} \text{ volts/volt} \tag{1.18}$$

We can represent this on the see-saw diagram as shown in figure 1.18(b). Point A on the diagram represents the virtual earth point at the inverting input to the op-amp. As the voltage gain of the circuit is increased, point A moves to the left so that a smaller change in v_i causes a larger change in v_o. Also notice that a positive input results in a negative output and vice-versa, as expected from an inverting amplifier.

Now consider the offset inverting amplifier shown in figure 1.19(a).

The effect of the offset or bias voltage V_b is to move the whole see-saw diagram up by an amount V_b volts, as shown in figure 1.19(b). By comparison with figure 1.17, this is exactly the configuration required to obtain the combination of amplification and bias required to condition the temperature sensor signal output for the A-D converter input.

The resulting seesaw diagram for the temperature sensor is shown in figure 1.20.

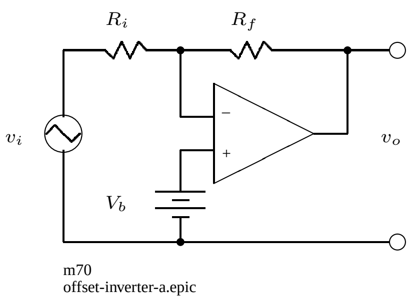
If the diagram is carefully done to scale, it is possible to read the values of A_v and V_b directly off the diagram. However, we shall use an algebraic approach, developing an expression for the transfer function of the offset inverting amplifier.

Refer to figure 1.21.

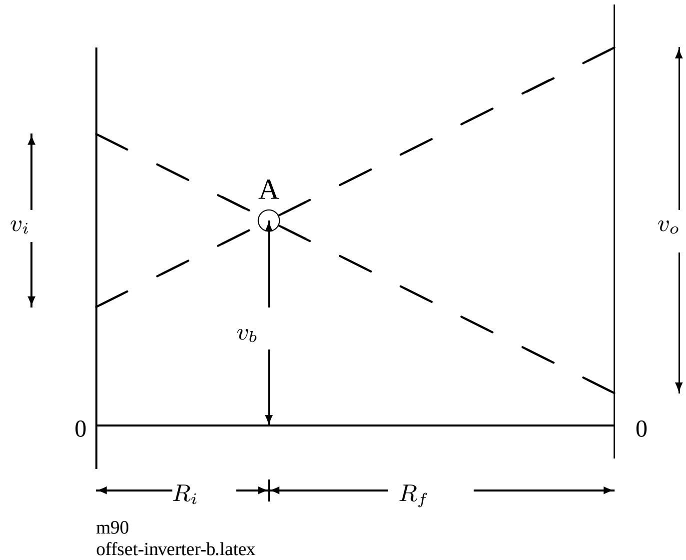
As usual, the rules to keep in mind for negative feedback operational amplifiers are

1. Feedback action keeps the inverting and non-inverting terminals at essentially the same potential.

2. No current flows into the input of the operational amplifier.



(a) Offset Inverting Amplifi er



(b) Offset Inverter Seesaw Diagram

Figure 1.19: Inverting Amplifier

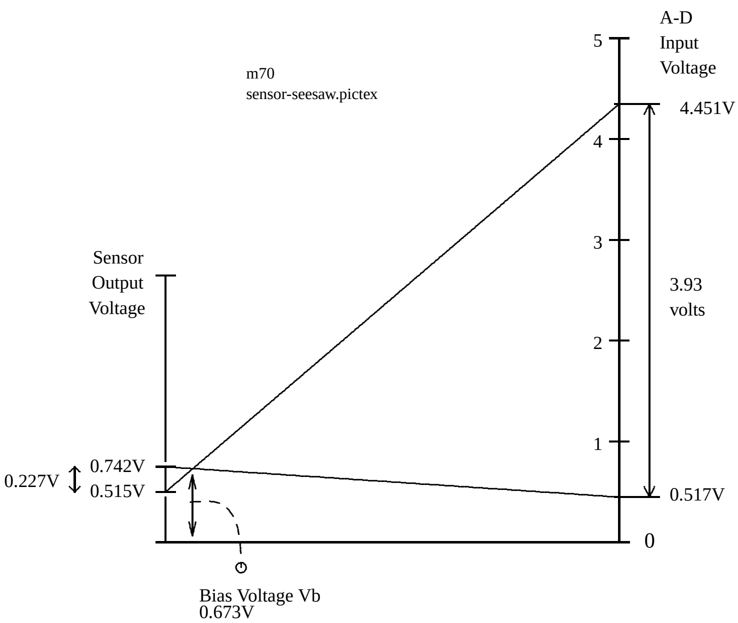
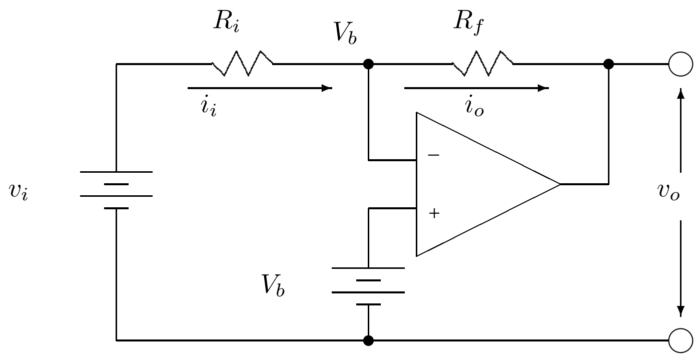


Figure 1.20: Sensor and A-D Signal Seesaw Diagram



m100
offset-inverter-large.latex

Figure 1.21: Sensor and A-D Signal Seesaw Diagram

Referring to figure 1.21, we can proceed as follows:

By rule 1, the voltage at the inverting terminal is equal to V_b. Then the input current i_i is

$$i_i = \frac{v_i - V_b}{R_i} \tag{1.19}$$

and the output voltage v_o is

$$v_o = V_b - i_o R_f \tag{1.20}$$

By rule 2, no current can flow into the op amp inputs and so i_o must equal i_i. Substitute i_i for i_o in equation 1.20 and then substitute equation 1.19 for i_i, and we have

$$v_o = V_b - \left(\frac{v_i - V_b}{R_i}\right) R_f$$

Grouping terms with V_b and v_i, we have

$$v_o = \left(1 + \frac{R_f}{R_i}\right) V_b - \left(\frac{R_f}{R_i}\right) v_i \tag{1.21}$$

This equation says that the output voltage is composed of two components: the first is a non-inverting gain times the bias voltage V_b, and the second is the inverting gain times the input signal, v_i[5]

Now, we could plug two pairs of v_i, v_o values into equation 1.21 and solve two equations in two unknowns to determine V_b and the ratio R_f/R_i. Once R_f/R_i is known, one or other of these resistors can be chosen and the other determined.

However, more understanding may be obtained by inspecting the seesaw diagram, figure 1.20. Notice that the change in op amp output voltage is entirely determined by the change in input voltage and the inverting gain R_f/R_i. The bias voltage simply shifts the whole diagram up or down the page. With that observation, we can solve very simply for R_f/R_i:

$$\begin{aligned}
A_{v_{inverting}} &= -\frac{R_f}{R_i} \\
&= -\frac{\Delta V_o}{\Delta V_i} \\
&= -\left(\frac{v_{o_{low}} - v_{o_{high}}}{v_{i_{high}} - v_{i_{low}}}\right) \\
&= -\left(\frac{0.517 - 4.451}{0.742 - 0.515}\right) \\
&= 17.3
\end{aligned}$$

Now we can substitute 17.3 for R_f/R_i along with a pair of v_i v_o values in equation 1.21, and solve for V_b.

$$v_o = V_b(1 + 17.3) - 17.3 v_i$$

[5]The same result could be obtained using the superposition theorem. Determine first the output due to v_i with V_b replaced by its internal resistance (a short circuit). Then determine the output due to V_b with v_i shorted. Then add these two outputs together to get the output due to both V_b and v_i.

Substitute $v_i = 0.742$ and $v_o = 0.517$, in equation 1.22 and solve for V_b to find that $V_b = 0.673$ volts. This coincides with the value shown for V_b on figure 1.20.

We now know the ratio of R_f/R_i, and we still have to calculate their actual values. Their actual values depend on the operational amplifier bias current and is calculation is deferred to section 1.4.8.

Sourcing the Bias Voltage

A tentative final circuit for the temperature sensor is shown in figure 1.22.

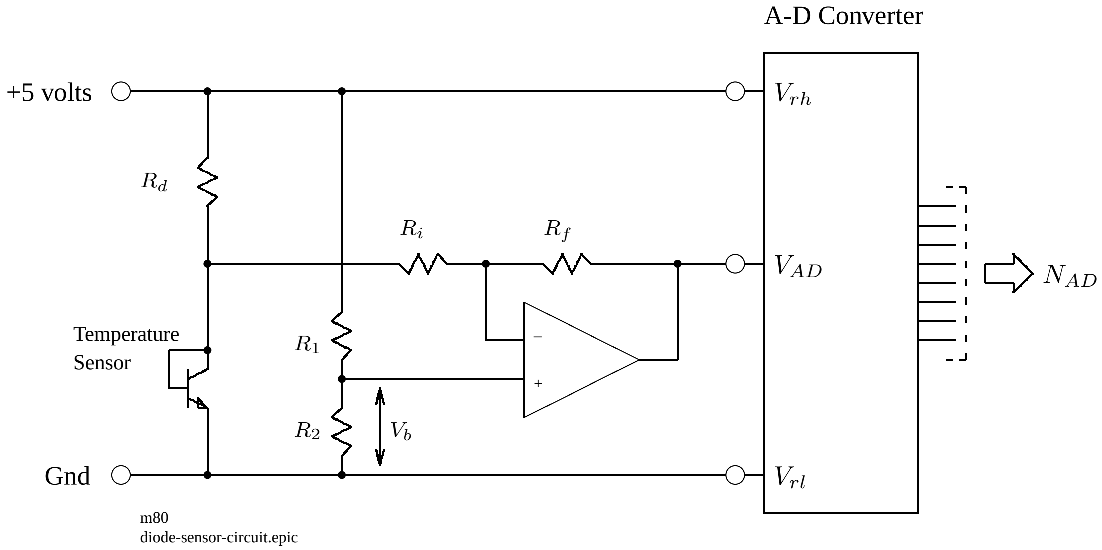


Figure 1.22: Temperature Sensor Circuit

The bias voltage V_b is created by a voltage divider from the +5 volt supply. However, there is a potential problem with this. According to the geometry of the seesaw diagram figure 1.20, any change in the bias voltage V_b will have a dramatic effect on the A-D input voltage. More precisely, equation 1.21 indicates that changes in V_b will appear at the output of the amplifier magnified by a factor of 18.48.

However, we found in section 1.2.2 that deriving the reference voltage V_{RH} for the A-D converter from the same source as the bias voltage caused the A-D reading to be immune to drift or ripple of the supply voltage. Let us see if the same effect applies here.

The A-D reading R is given by

$$N_{AD} = \frac{V_{AD}}{V_{RH}} \times 255 \tag{1.22}$$

where

N_{AD} is the A-D output, 0 to 255
V_{AD} is the input voltage to the A-D converter
V_{RH} is the reference voltage of the A-D converter, the positive supply of the system (+5 volts).

The output of the op amp is the input to the A-D converter, so we can write V_{AD} for v_o in equation 1.21.

$$V_{AD} = V_b \left(1 + \frac{R_f}{R_i}\right) - v_i \left(\frac{R_f}{R_i}\right) \tag{1.23}$$

Now, if the bias voltage V_b is derived from a voltage divider connected to the voltage source V_{RH}, then it will be some fraction K of that voltage. For example, in our case V_b has been calculated to be 0.741 volts, and V_{RH} is +5 volts, K would be $0.741/5 = 0.148$. Then we can write

$$V_b = K \times V_{RH} \tag{1.24}$$

Substituting $K \times V_{RH}$ for V_b and equation 1.23 for V_{AD} in equation 1.22, we have

$$N_{AD} = \frac{255}{V_{RH}} K\, V_{RH} \left(1 + \frac{R_f}{R_i}\right) - v_i \left(\frac{R_f}{R_i}\right) \tag{1.25}$$

$$= 255\, K \left(1 + \frac{R_f}{R_i}\right) - v_i \left(\frac{R_f}{R_i}\right) \tag{1.26}$$

Since V_{RH} and V_b disappear, the A-D reading is independent of the A-D reference voltage and op-amp bias voltage. However, the A-D reading is still dependent on the stability of the coefficient K. A stable value of K should not be difficult to achieve since K depends on the ratio of two resistors.

The actual values of R_1 and R_2 depends on the operational amplifier parameters and is deferred to section 1.4.8.

1.4.6 The Operational Amplifier

The real skill of engineering is designing to a budget, a maxim that applies to the power supplies in this circuit. If +/-15 volt power supplies were available, we could simply drop in any operational amplifier and the circuit would work. However, in our case, we have a single +9 volt unregulated supply, +5 volts regulated and -10 volts at a few milliamps. Ideally, we should design a circuit that works from these voltage supplies.

The primary limitation is the restricted input and output voltage range of a conventional operational amplifier such as the 741. When this type of op amp is operated from power supply voltages of +15 V and -15V, the input and output voltages are typically limited to a range of +/-12 volts. In other words, the input and output voltages cannot go closer than to within 3 volts of the supply rails. If the op amp is operated from +9 volts and ground, its input and output would be then limited to the range +3 to +6 volts. In fact, because it is designed to work from +/-15 volt power supplies, a conventional op-amp may not work at all at the lower supply voltages.

Recognizing this problem, the semiconductor manufacturers have provided us with operational amplifiers such as the National LM324 which are designed to work from a single low voltage power supply such as +5 volts.

For the LM324, the input voltage can extend from ground to $V^+ - 1.5$ volts. In our circuit, both inputs of the op amp are held at 0.741 volts by negative feedback action, and so the inputs of the LM324 can accomodate this voltage.

Referring to figure 1.17, we see that the output signal must range between 0.431 volts and 4.35 volts. The spec sheet for the LM324 shows that the output can go to within half a volt of the positive power supply rail for output currents of less than 10mA. In our circuit, the output of the op amp drives the input to the A-D converter, which is effectively an open circuit load. Thus we can expect an adequate output swing when the op amp is operated from a +5 volt regulated supply.

In addition to its ability to operate from a single supply, the LM324 is also attractive for this application because it is low cost (under $1 in single quantities), includes 4 op-amps per package, and is readily available from a variety of suppliers.

1.4.7 The effect of Power Supply Ripple and Drift

Modern operational amplifiers have a high degree of Power Supply Rejection Ratio, or PSRR. That is, ripple and drift on the power supply lines appears as a much smaller equivalent signal at the input to the operational amplifier. If the PSRR is large, the amplifier has a high degree of immunity to voltage changes on the power supply lines.

The PSRR of an op amp is defined as

$$\text{PSRR(db)} = 20\log\frac{\Delta V_{ps}}{\Delta V_{ips}} \tag{1.27}$$

where

ΔV_{ps} is the variation in power supply voltage ΔV_{ips} is the equivalent variation in input voltage due to variations in supply voltage

The question immediately at hand is whether the LM324 op-amp has sufficient power supply rejection to ignore ripple on the power supply line. If it does, we can use the +9 volt unregulated power supply voltage as a supply for the op-amp. To determine this, we must develop a suitable circuit for analysis of the effect of variations in the power supply voltage.

In this case, the first step, shown in Step 1 of figure 1.23, is to replace all the voltage sources by their internal impedances (short circuits).

In step 2, the circuit is rearranged.

In step 3, resistors R_1 and R_2 have no current flowing through them, so they may be removed. R_d is bypassed by a short circuit, so it may be removed. The power supply circuit is shown explicitly, with its steady component V_s and its varying component ΔV_{ps}.

Step 4 shows the equivalent input voltage v_{ips}, due to the variation in supply voltage ΔV_{ps}. The v_{ips} voltage source could be shown in series with either input: the result is the same. Putting it in series with the inverting input is a little easier to analyse.

This final equivalent circuit is a non-inverting amplifier configuration, so the output voltage due to power supply variations is:

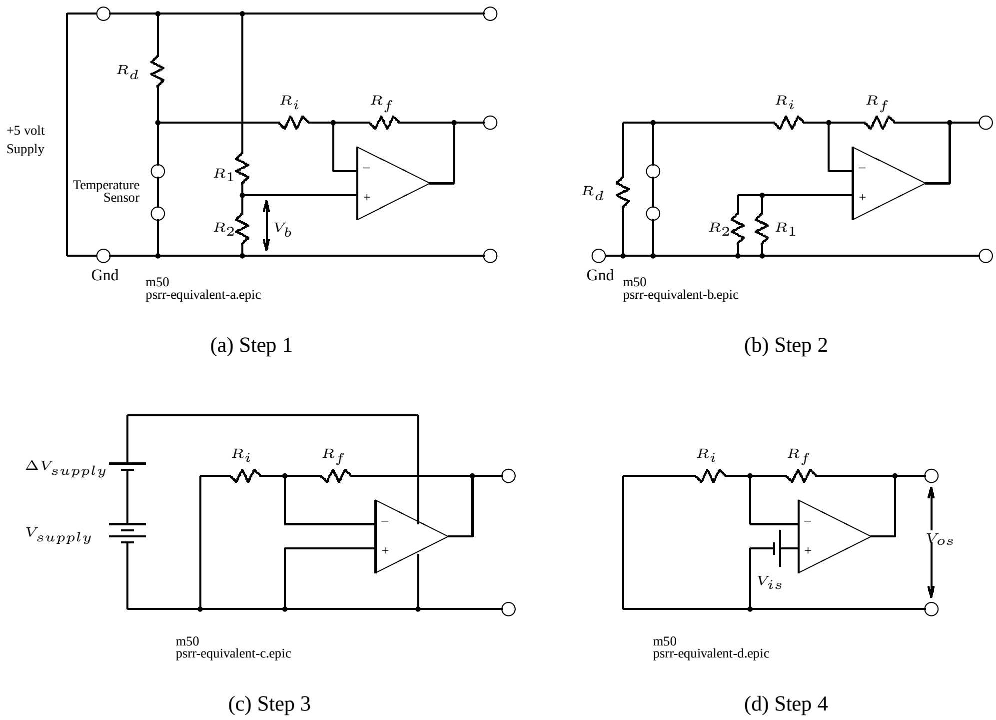
$$v_{ops} = v_{ips}\left(1 + \frac{R_f}{R_i}\right) \tag{1.28}$$

Combining equations 1.27 and 1.28 we have

$$v_{ops} = \frac{\Delta V_p s}{PSRR}\left(1 + \frac{R_f}{R_i}\right) \tag{1.29}$$

Now for an example: If the power supply ripple is 1 volt peak-peak and the power supply rejection ratio is 65 db (the worst case value given on the LM385 data sheet), calculate the ripple in the op-amp output signal.

In section 1.4.5 we found that $R_f/R_i = 17.3$ volts/volt. As well, using equation 1.27 a PSRR of 65 db is equivalent to a PSRR ratio of:



(a) Step 1

(b) Step 2

(c) Step 3

(d) Step 4

Figure 1.23: Power Supply Rejection Ratio Equivalent Circuit

$$
\begin{aligned}
PSRR &= 10^{\left(\frac{65}{20}\right)} \\
&= 1778 \text{ volts/volt}
\end{aligned}
$$

Now use equation 1.29 to find v_{os}:

$$
\begin{aligned}
v_{ops} &= \frac{\Delta V_p s}{PSRR}\left(1 + \frac{R_f}{R_i}\right) \\
&= \frac{1.0}{1778}\,(1 + 17.3) \\
&= 10.3 \text{ millivolts peak-peak}
\end{aligned}
$$

Each 19.6 millivolt change in the output is equivalent to 0.5^{o}C, so 10.4 millivolts is equivalent to a ripple in the reading of about 0.25^{o}C.

A range of line voltage stability is +/-10%, so we should expect an unregulated 9 volt power supply to vary by

+/-0.9 volts for a total variation of 1.8 volts. Obviously, this will induce too large a change in the output voltage of the op-amp, so the op-amp power supply must be regulated.

Incidentally, the typical value of PSRR for the LM324 is 100db, a PSRR factor of 10^5. With this PSRR, the output voltage swing due to a power supply variation of 1 volt would be 0.18 mV, which is acceptable. A designer who lashed up the circuit and tested it in the lab might well happen to use an LM324 with a typical value of PSRR, thereby finding an acceptable output variation due to power supply fluctuations. One has to expect in a production run, however, that some LM324's will be found with a minimum value of PSRR and they will not work correctly.

There are two important lessons from this:

1. Always use the worst case specification in design. Never, never design with a typical value.

2. A working model of the circuit is a necessary but not sufficient condition for the circuit to work in quantity. A lab prototype is an exercise to test the design theory. If the prototype doesn't work, the theory is obviously faulty. If it does work, the circuit will work in production only if production units can function with a full range of component tolerances.

Finally, we note that the power supply rejection ratio of the 741 op-amp, a conventional design, has a power supply rejection ratio of 86 db(minimum). This is an order of magnitude better than the minimum PSRR of the LM324. Perhaps a lower PSRR is something one has to accept with single-supply op-amps.

1.4.8 The Effect of Bias Currents

The input stage of the LM324 is a bipolar PNP differential amplifier, as shown in figure 1.24.

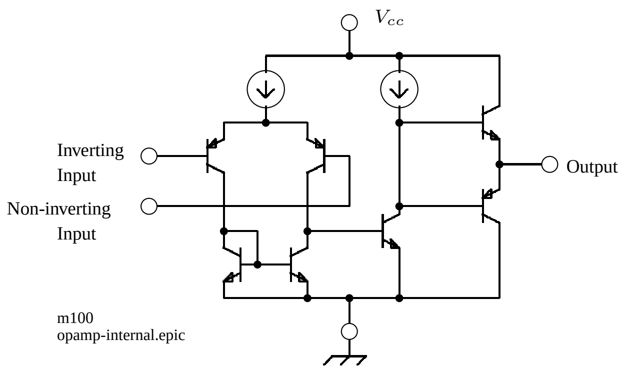


Figure 1.24: LM324 Internal Circuitry (Simplified)

As a result, a small amount of base current, the bias current will flow out of the input terminals of the op-amp. For operational amplifiers with FET inputs, the bias current is in the order of 100 picoamps, which is small enough to be neglected in most cases. In the case of the LM324, the bias current is 250nA (maximum), which is large enough to have an effect on the operation of the circuit. Fortunately, the effect of bias current can be minimized by reducing the size of the resistances through which the bias current flows. However, reducing resistances increases the power consumption of the circuit and adds to the heat dissapation, so resistors should be as large as possible.

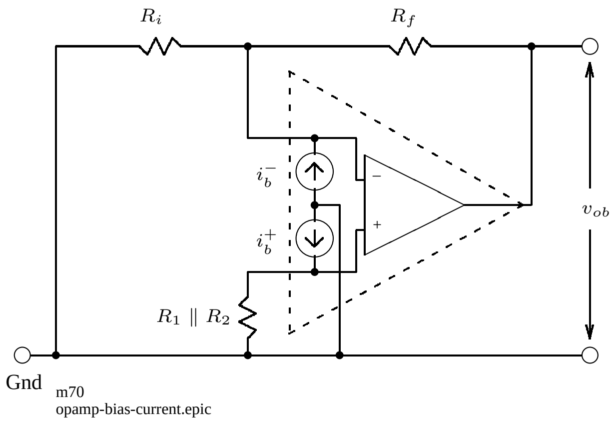


Figure 1.25: Operational Amplifier With Bias Current

A op-amp circuit model that incorporates the effect of bias current is shown in figure 1.25. As in figure 1.18, we have simplified and combined the external circuit to determine the effect of the bias currents on the output voltage.

The voltage at the non-inverting input terminal of the op-amp is

$$v^+ = i_b^+ (R_1 \parallel R_2) \tag{1.30}$$

The op amp drives the inverting input terminal to the same voltage. Then the current i_i through R_i is

$$\begin{aligned} i_i &= \frac{v^-}{R_i} && (1.31)\\ &= \frac{v^+}{R_i} && (1.32)\\ &= i_b^+ (R_1 \parallel R_2) && (1.33) \end{aligned}$$

If i_b^+ is equal to i_i, then $i_f = 0$ and $v_{ob} = v^+$. The practical implications are that

- The value of $R_1 \parallel R_2$ should be kept to a minimum to minimize the voltage generated by the bias current.
- The value of R_i should be made equal to $R_1 \parallel R_2$ to minimize the value of v_{ob}.

Putting numbers to this, let us choose the acceptable output due to the op-amp bias currents at equivalent to one quarter of a degree centigrate, or about 0.5 mV. For the LM324, the bias current is 250 nA maximum. Then $R_1 \parallel R_2$ should be

$$\begin{aligned} R_1 \parallel R_2 &= \frac{0.5 \times 10^{-3}}{250 \times 10^{-9}} && (1.34)\\ &= 2K\Omega && (1.35) \end{aligned}$$

In section 1.4.5, we determined that the divider ratio should be 0.148. Then

$$\frac{R_2}{R_1 + R_2} = 0.148 \tag{1.36}$$

Solving equations 1.35 and 1.36 for the resistor values, we obtain

$$\begin{aligned} R_1 &= 13.5K\Omega \\ R_2 &= 2.35K\Omega \end{aligned}$$

Since these exact values are not available and since the bias voltage should be adjustable, we use a potentiometer in the center to adjust the voltage exactly to 0.741 volts, as shown in the figure 1.26.

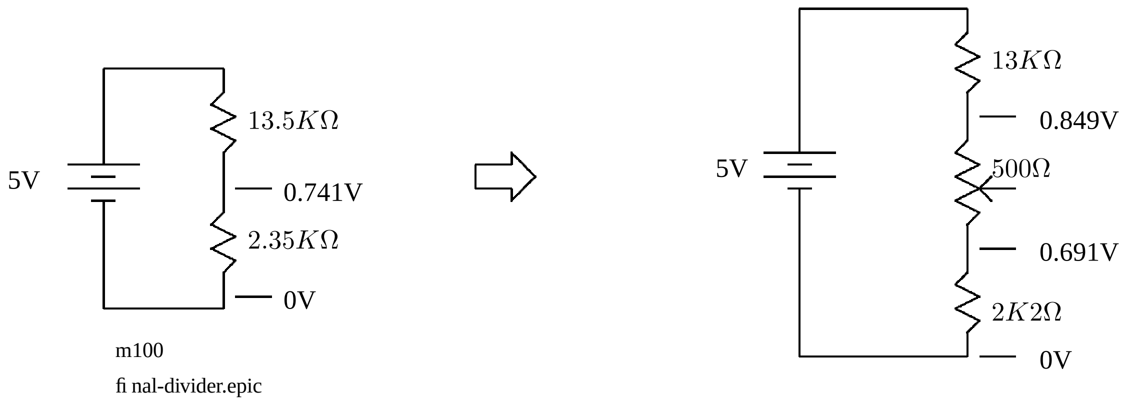


Figure 1.26: Bias Voltage Divider Design

1.4.9 The Effect of Offset Current

The offset current is the difference between the two input bias currents. In the case of the LM324, the bias currents can differ by as much as 50 nanoamps. In the previous analyis of the effect of bias current, we assumed that the two currents were equal. If they are not, we will show that the offset current flows through R_f and causes a change in the output voltage of $I_{os} \times R_f$ volts. As a result, the output voltage more sensitive to offset current when the amplifier gain is large.

The appropriate equivalent circuit is again figure 1.25. This time, however, the two bias currents i_b^+ and i_b^- are no longer equal.

To develop an expression for v_o, we call $R_1 \parallel R_2$ an equivalent resistor R_{eq}, and develop the following equations:

The bias current out of the non-inverting terminal generates a voltage given by

$$v^+ = i_b^+ R_{eq} \tag{1.37}$$

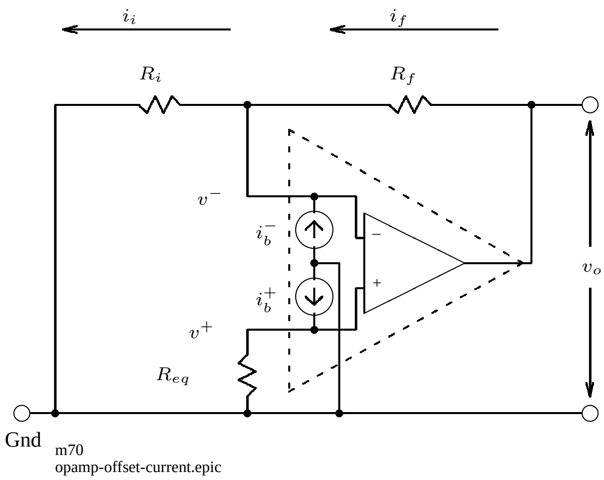


Figure 1.27: Offset Current Calculation

The op-amp forces the inverting and non-inverting inputs to be at the same potential, so

$$v^+ = v^- \tag{1.38}$$

The current through R_i is given by

$$i_i = \frac{v^+}{R_i} \tag{1.39}$$

At the virtual earth point, the currents are given by

$$i_f = i_i - i_b^- \tag{1.40}$$

The output voltage v_o due to the bias current is given by

$$v_o = v^+ + i_f R_f \tag{1.41}$$

Combining equations 1.37 through 1.41, to solve for v_o in terms of the resistances and bias currents, we have

$$v_o = i_b^+ R_{eq} + i_b^+ \frac{R_{eq} R_f}{R_i} - i_b^- R_f \tag{1.42}$$

If we make R_{eq} equal to R_i as suggested in section 1.4.8, then we can simplify equation 1.42 to

$$v_o = i_b^+ R_{eq} + (i_b^+ - i_b^-) R_f \tag{1.43}$$

Substituting the offset current i_{os} for $i_b^+ - i_b^-$, we have

$$v_o = i_b^+ R_{eq} + i_{os} R_f \tag{1.44}$$

This value for output voltage is the same result we obtained in section 1.4.8 for bias current, augmented by the term $i_{os}R_f$.

In our application, the bias current is 250 nanoamps maximum and the bias current 50 nanoamps maximum. R_{eq} and R_1 are each is 2KΩ and R_f is 34.96KΩ. Then the worst-case output voltage due to bias and offset currents will be

$$\begin{aligned} v_o &= i_b^+ R_{eq} + i_{os} R_f \\ &= 250 \times 10^{-9} \times 2000 + 50 \times 10^{-9} \times 34.96 \times 10^3 \\ &= 2.25 \times 10^{-3} \end{aligned}$$

This is at the output of the op-amp and input of the A-D converter, where each degree of temperature is a change of 38mV. Thus the voltage created by the bias and offset currents is a small percentage of one degree.

In applications where the operational amplifier must be extremely stable, the drift of offset current with temperature may be of concern. For the LM324, the offset current drift specification is 10 picoamps/^{o}C. In our case, an operating temperature range for the op-amp of 25^{o}C +/-10^{o}C would result in a change of 0.200 nanoamps. In our design, this is small enough to be ignored.

1.4.10 The Effect of Offset Voltage

The input stage of an operational amplifier consists of a differential pair of transistors. The base-emitter voltages of the two transistors are not perfectly matched. The difference between the two base-emitter voltages creates an effective input voltage, called the offset voltage. In this section, we examine the effect of the offset voltage, and the drift of offset voltage with temperature, on the accuracy of the circuit.

The equivalent circuit of an op-amp with offset voltage, in our simplified circuit for analysis, is shown in figure 1.28.

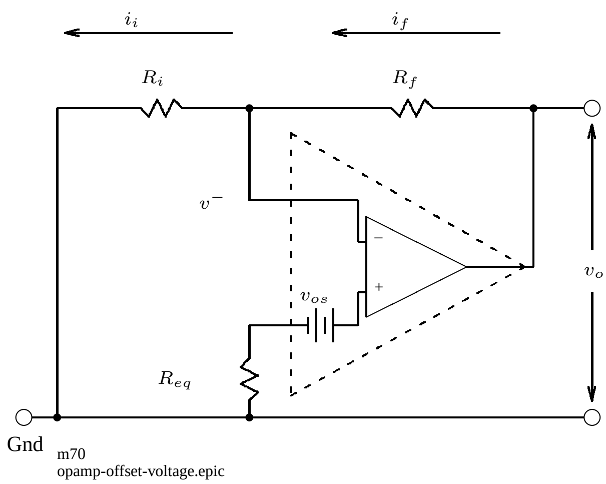


Figure 1.28: Operational Amplifier With Offset Voltage

The analysis of the circuit is quite similar to previous work:

Since there is no current into the non-inverting terminal of the op-amp, the voltage at the non-inverting input v^+ is equal to the offset voltage v_{os}.

$$v^+ = v_{os} \tag{1.45}$$

The op-amp negative feedback action drives the inverting input terminal to the same voltage as the non-inverting terminal.

$$v^+ = v^- \tag{1.46}$$

The current through R_i is

$$i_i = \frac{v^+}{R_i} \tag{1.47}$$

Since, in this analysis, there is no current into the inverting input terminal of the op-amp, the current through R_f is equal to R_f.

$$i_f = i_i \tag{1.48}$$

Finally, the output voltage v_o is equal to the voltage at the inverting input terminal plus the voltage across the feedback resistor.

$$v_o = v^- + i_f R_f \tag{1.49}$$

Putting equations 1.45 to 1.49 together, the output voltage is

$$v_o = v_{os}\left(1 + \frac{R_f}{R_i}\right) \tag{1.50}$$

In words, the output voltage is equal to the offset voltage times the non-inverting gain of the op-amp circuit. In our case, the non-inverting gain is 18.48 and the offset voltage of the LM324 is +/-9mV, so the output due to the offset will be

$$\begin{aligned} v_o &= v_{os}\left(1 + \frac{R_f}{R_i}\right) \\ &= +/-9 \times 10^{-3} \times 18.48 \\ &= +/-166mV \end{aligned}$$

At first glance, this is somewhat alarming, since it represents a temperature change of +/-9^{o}C. However, since V_{be} appears in series with the adjustable bias voltage source V_b, it is possible to compensate for the offset voltage by adjusting the bias in the opposite direction.

However, the drift in op-amp offset voltage with temperature is not so easily disposed of. The LM324 offset voltage drift is specified as 20μvolts per ^{o}C. Over a 20^{o}C operating temperature range for the op-amp, the offset voltage will then change by 400μvolts. Using equation 1.49, we determine that the total drift in voltage at the input to the A-D converter, will be 7.4mV, equivalent to about half a degree centigrade.

Sneaking a longing glance at the typical figure for offset voltage drift for the op-amp (7μvolts per ^{o}C, a third of the worst-case value), we decide that this is marginally acceptable.

1.4.11 Drift in Resistor Values

To this point, we have focussed on the drift characteristics of the operational amplifier. However, even the resistors in circuit do change their value with temperature. Carbon film resistors have a typical temperature coefficient of -250ppm (parts per million). Metal film resistors are much more stable, at around +/-20ppm.

In our circuit, the critical arrangements of resistors are R_1 and R_2, which establish the bias voltage, and R_i and R_f, which establish the voltage gain of the op-amp. Referring to equation 1.26, we can see that in both cases the key quantities are the ratio of two resistor values. If the two resistors change by the same percentage amount, there is no effect on their ratio, and the circuit is unaffected.

1.4.12 Estimate of Accuracy

1.5 Appendix: Operational Amplifier, Static Errors

In this section, we summarize the operational amplifier characteristics that may cause errors at zero and low frequencies. The definitions are from reference [27].

1.5.1 Input Offset Voltage

Definition

That voltage which must be applied between the input terminal resistance through two equal resistances to obtain zero output voltage.

Equivalent Circuit (figure 1.29)

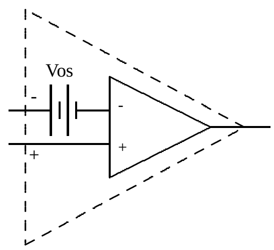


Figure 1.29: Operational Amplifier Offset Voltage

Notes

- Some operational amplifiers include nulling terminals which can be connected to an external pot to reduce the offset voltage to zero.
- Offset voltage appears at the output multiplied by the voltage gain of the amplifier, so it is especially significant for high-gain voltage amplifiers.

Application Circuit (figure 1.30)

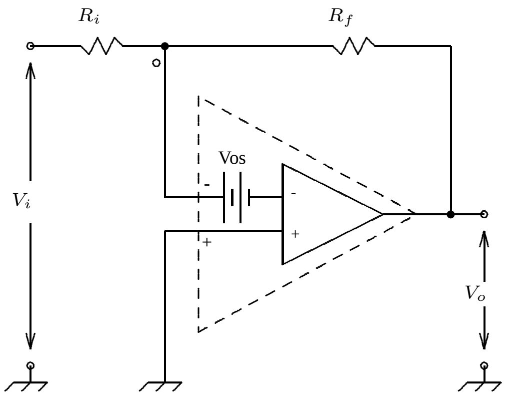


Figure 1.30: Operational Amplifier Offset Voltage Application

The output voltage for this circuit is

$$V_o = V_i \frac{R_f}{R_i} + V_{os}\left(1 + \frac{R_f}{R_i}\right) \tag{1.51}$$

The term $V_i \frac{R_f}{R_i}$ is desired: $V_{os}\left(1 + \frac{R_f}{R_i}\right)$ is an error term.

1.5.2 Input Bias Current

Definition

The average of the two input currents.

Equivalent Circuit (figure 1.31)

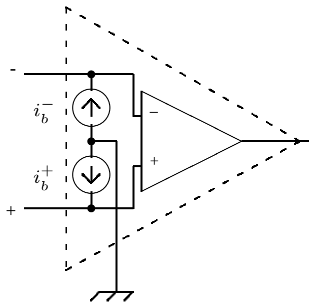
Notes

- Most BJT operational amplifiers have much higher bias currents than JFET or MOSFET operational amplifiers. Bias current creates an effective input offset voltage proportional to the resistances in the input leads.
- The effect of bias current may be cancelled by equalizing the resistances in the two input leads.

Application Circuit (figure 1.32)

The output voltage for this circuit is

$$V_o = V_i \frac{R_f}{R_i} + I_b (R_f \parallel R_i)\left(1 + \frac{R_f}{R_i}\right) \tag{1.52}$$



m70
opamp-ibias-equivalent.epic

Figure 1.31: Operational Amplifier Bias Current

The second term in the equation is the bias-current induced error term, proportional to bias current, magnitude of the input resistances, and the voltage gain.

1.5.3 Input Offset Current

Definition

The difference between the two input currents.

Equivalent Circuit (figure 1.33)

Notes

- Offset current may be minimized be selecting an opamp with small bias current. The usual problem is not the magnitude of offset current but its change with temperature, which appears in the output as voltage drift with temperature.

Application Circuit (figure 1.34)

The output voltage for this circuit is

$$V_o = V_i \frac{R_f}{R_i} + I_{os} \left(R_f \parallel R_i\right) \left(1 + \frac{R_f}{R_i}\right) \tag{1.53}$$

As in the case of bias current, the second term in the equation is the offset-current induced error term, proportional to offset current, magnitude of the input resistances, and the voltage gain.

1.5.4 Power Supply Rejection Ratio

Definition

The ratio of the change in input offset voltage to the change in power supply voltages producing it.

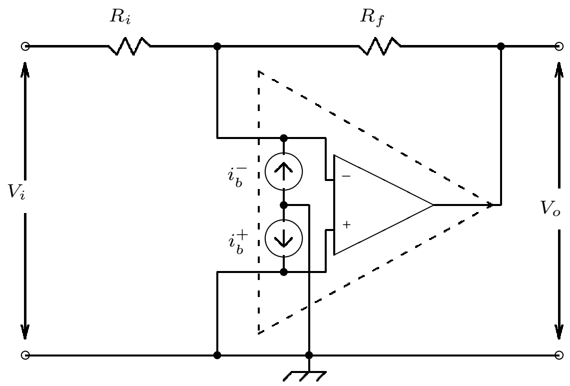


Figure 1.32: Operational Amplifier Bias Current Application

Equivalent Circuit (figure 1.35)

Notes

- Power supply rejection is an issue when the power supply shows a large variation or large ripple content. The change in power supply voltage may show up as drift in the output voltage of the op amp.
- Power supply rejection is usually expressed in decibels, so it must be converted to a voltage ratio before use.

Application Circuit (figure 1.36)

The output voltage for this circuit is

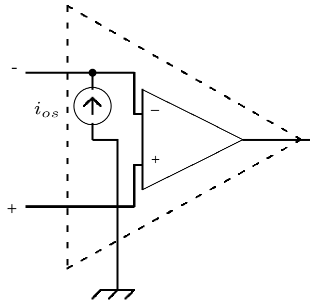
$$PSRR = 20 \log_{10} PSR \tag{1.54}$$

where PSR is the rejection ratio in volts/volt, PSRR is the ratio in decibels.

$$V_{ps} = \frac{V_s}{PSR} \tag{1.55}$$

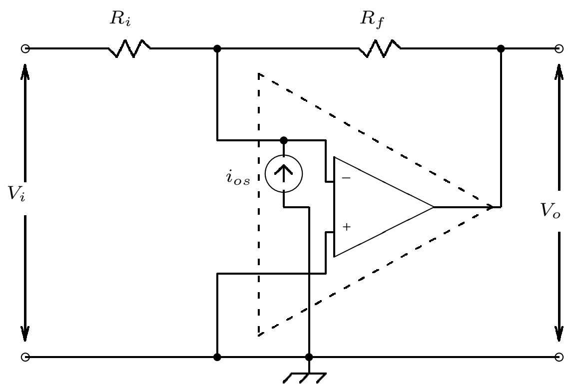
$$V_o = V_i \frac{R_f}{R_i} + V_{ps} \left(1 + \frac{R_f}{R_i} \right) \tag{1.56}$$

The second term in the equation is the power supply induced error term, proportional power supply ripple, inversely proportional to the power supply rejection, and proportional to the voltage gain.



m70
opamp-ios-equivalent.epic

Figure 1.33: Operational Amplifier Offset Current



m70
opamp-ios-application.epic

Figure 1.34: Operational Amplifier Offset Current Application

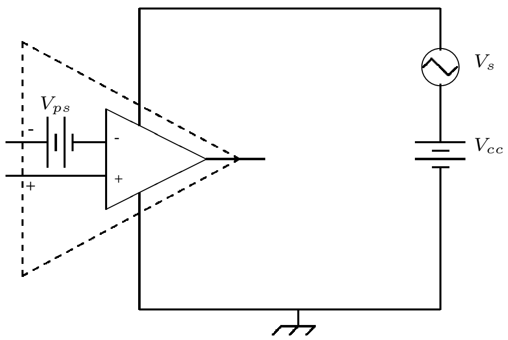


Figure 1.35: Op Amp Power Supply Rejection, Equivalent Circuit

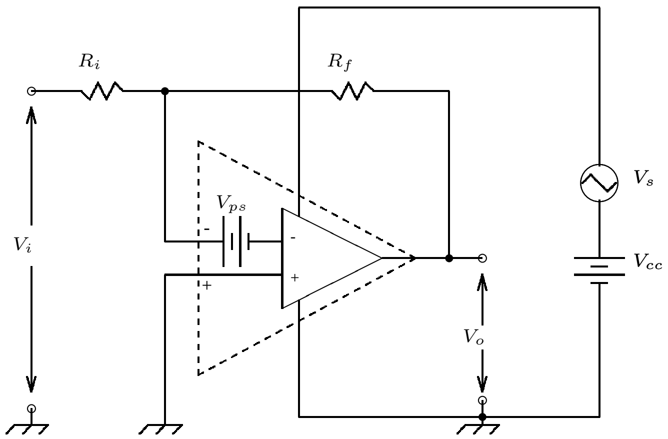


Figure 1.36: Op Amp Power Supply Rejection, Application Circuit

1.5.5 Software for the Diode Temperature Sensor

The temperature sensor interface is responsible for amplifying and level shifting the diode voltage so that it is a good fit to the range of the microprocessor analogue-digital converter. Once the diode signal is converted to a binary number, the microprocessor software is responsible for generating the appropriate display readout.

Diode voltage is essentially a linear function of temperature, so the relationship between A-D reading and display readout is also linear, and this simplifies the design of the temperature software.

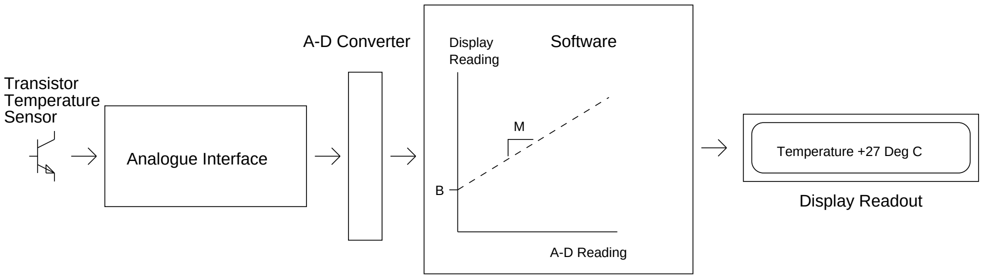


Figure 1.37: Diode Temperature Sensor Block Diagram

If we are only constructing one unit, we could generate a lookup table in which the index into the table is the A-D reading and the display value is the table entry. A-D readings can have 256 different values, so this fixes the length of the table. To save work, the table may be generated using a BASIC program on a P/C, and then edited into the assembly language source code.

If we are constructing a small number of these units, the analogue section would be adjusted, using potentiometers, so that each analogue interface has the same voltage gain and offset. Then the same lookup table could be used for each unit.

Calibration in Volume Production

Now consider that we are producing a large number of these units on an assembly line. Potentiometers in the analogue circuit are undesireable because the pot and the labour to adjust the pot are both expensive. If possible, we would like to use fixed resistors in the analogue interface section and adjust the software to suit. This is all the more appealing because the 68HC11 microprocessor has EEPROM (Electrically Eraseable Programmable Read Only Memory). Calibration constants can be written into EEPROM at the time the unit is manufactured, and they become essentially a permanent part of the program.

Picture the following automatic calibration setup:

- Thermometer units are delivered by conveyer line to the calibration station. At the station, a calibration computer plugs into the 68HC11 microprocessor. The computer touches the diode temperature sensor with a hot probe of known temperature and notes the HC11 A-D reading. It then repeats the process with a cold probe of known temperature.

- The calibration computer now knows two points on the function relating A-D reading and sensor temperature. The calibration function is linear, so the calibration computer can calculate the slope M and offset B constants of the calibration function, figure 1.37.

- Finally, the calibration computer writes the M and B constants into the EEPROM of the 68HC11 microprocessor. The thermometer unit is calibrated.

Transfer Function Calculation

The software to display temperature must solve equation 1.57.

$$T_d = M \times N_{AD} + B \tag{1.57}$$

where T_d is the displayed temperature reading, M and B are the calibration constants, and N_{AD} is the A-D reading.

This could be accomplished using integer arithmetic: the equation is not particularly complex and the range of numbers is limited so overflow and underflow are not likely to be a problem. However, in subsequent sections, we will make use of the temperature value to calculate dew point and humidex readings, and floating point calculations are much easier to use. To be consistent, then, it probably makes sense to use floating point routines to solve this equation.

1.5.6 Checking the Production Design

The fact that a prototype of this circuit works is, as mathematicians put it, a necessary but not sufficient condition that the design is correct. If the prototype fails, the design is flawed. If it works, the design may or may not be satisfactory.

To ensure that hundreds of this temperature measurement circuit can be produced and will work reliably, we must do a proper engineering design. This is the work that distinguishes a production design engineer from a hobbiest or scientist who needs one working circuit. The design must work in spite of the expected variations in component specifications.

Resistors, transistors, op-amps – all vary in their specifications and we may have gotten lucky in the prototype circuit, using components with values favorable to the design. In a production situation, we must assume that the components will vary through their worst case values.

There are many computer-based tools available to help with the analysis of electronic circuits: reference [97] has an overview. However, the computer spreadsheet is readily available on computers these days, and easy to use. Once the spreadsheet formulae are entered, the spreadsheet becomes an electronic model for the circuit behaviour and component values may easily be tweaked to determine their effect on the circuit, figure 1.38. As well, one can simulate circuit operation over the full specified temperature range (-40 to +60^{o}C) without access to a temperature testing chamber of this range.

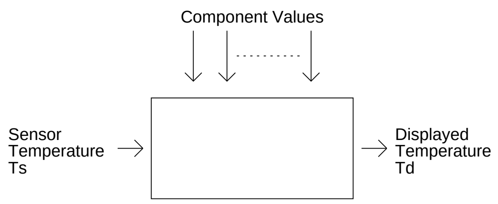


Figure 1.38: Checking Component Variation

In the case of this particular circuit, the software calibration process will take care of minor component variations. A more serious concern is that the amplifier may saturate or cutoff (the output may approach too closely either V_{cc} or ground), in which case it will cease to function correctly. If possible, we'd prefer to use 5% resistors because of their lower cost, compared to 1% resistors, so an important question is whether the circuit will function correctly with 5% resistors.

There are 4 resistors in the circuit. The amplifier may saturate or cutoff at either high or low temperature. Thus, even in the case of this simple circuit, ignoring possible variations in the sensor temperature coefficient and offsets in the operational amplifier, there are 8 possible situations to examine. Fortunately, the spreadsheet makes it simple.

1.5.7 Spreadsheet Model

A model for the diode temperature sensor circuit and software is shown in figure 1.39.

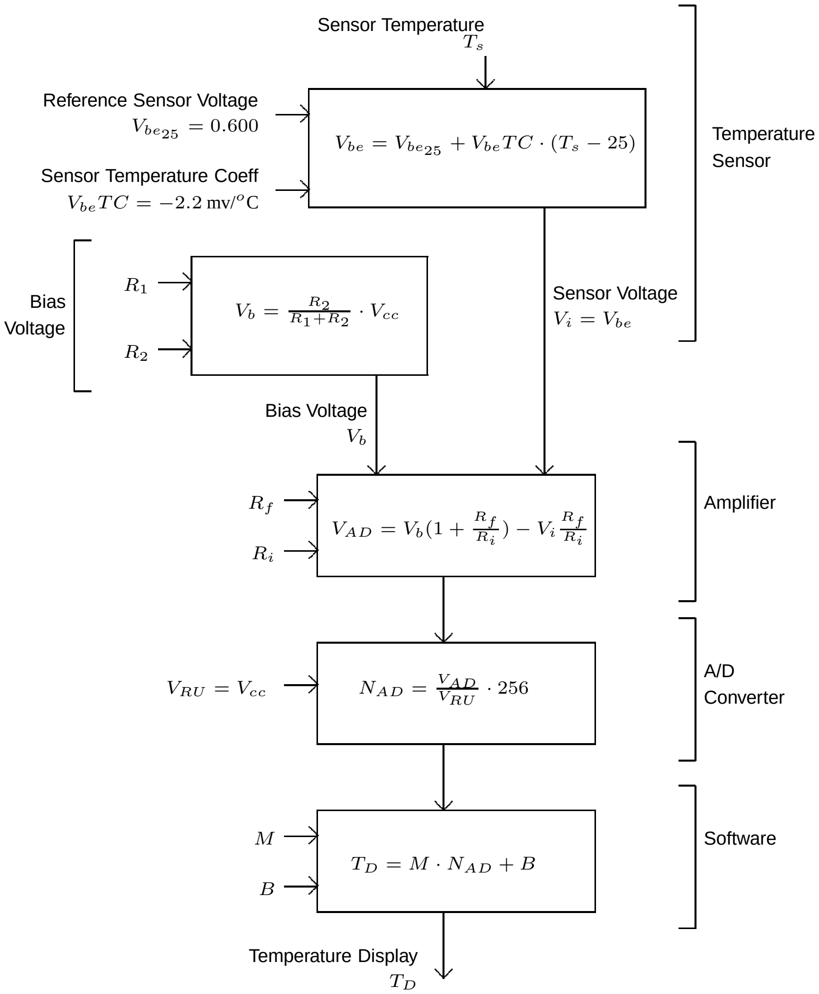


Figure 1.39: Diode Temperature Simulation Block Diagram

The first step in using the spreadsheet is to enter explanatory headings (the left column in table 1.1). Next, give the cells descriptive names, if possible. (Unfortunately, as in the case of the Microsoft Excel spreadsheet, the exact name may have to be some variant of what we have been using. For example, the name `R1` is illegal for some reason, and we have to use something like `R_1` instead. However, this is better than the generic name for a cell, like `B13`.)

Next, the enter the model formulae into the spreadsheet as shown in right column of table 1.1.

It may be useful to set the format specification for individual cells on the spreadsheet. For example, if the temperature shown on the LCD with one decimal point, the corresponding cell (last row, second column, table 1.1) should be set to the same number format.

Once the formulae are in place, the first tests should be used to determine that the model is producing credible results. (Our first attempt uncovered an arithmetic error in the original design. Whoops.)

Temperature Sensor Simulation Description	Constants and Formulae
Power Supply Voltage, volts	5
Temperature Sensor	
Sensor Temperature T_s, Degrees C	60
Sensor Reference Temperature T_{ref}, Degree	25
V_{be} at 25 Degrees, $V_{be_{25}}$, volts	0.595
Sensor Temperature Coef, $V_{be_{TC}}$, mv/Deg C	-2.265
Sensor Voltage V_s, Volts	$+V_{be_{25}} + (T_s - T_{ref}) \times (V_{be_{TC}}/1000)$
Bias Supply	
R1, Ohms	13000
R2, Ohms	2200
Bias Voltage V_b	$+V_{cc} \times (R2/(R1 + R_2))$
Amplifi er	
Input Resistor Ri, Ohms	6800
Feedback Resistor Rf, Ohms	120000
Amplifier Output Voltage V_o, Volts	$+V_b \times (1 + (Rf/Ri)) - V_s \times (Rf/Ri)$
A/D Conversion	
Output Count, N_{AD}	$INT((V_o/V_{cc}) \times 256)$
Software	
Function slope constant M	0.5
Function offset constant B	-52
Displayed Temperature T_d	$M \times N_{AD} + B$

Table 1.1: Spreadsheet Simulation Formulae and Data

When the spreadsheet is finally debugged, the model can be used to generate some useful results. A typical calculation run is shown in table 1.2.

This table has been modified so that the resistors R1,R2,Ri and Rf are at their extreme tolerance values ($\pm 1\%$) and the temperature is at its maximum specification, 60^{o}C. The software constants M and B have been adjusted so that the system is calibrated: the **Displayed Temperature** tracks the **Sensor Temperature**. Most importantly, the model shows that the amplifier output voltage is 4.705 volts under these conditions, within its allowable limit.

Results

Out of this simulation, we found that 1% resistors and an LMC660 op amp will work in the circuit. On the other hand, 5% resistors would not work: the amplifier saturated and cut off under high and low temperature extremes. We also learned that the LM358 bipolar single supply amplifier will not work correctly in the circuit: its output

Temperature Sensor Simulation Description	**Constants and Calculated Results**
Power Supply Voltage, volts	5
Temperature Sensor	
Sensor Temperature T_s, Degrees C	60
Sensor Reference Temperature T_{ref}, Degree	25
V_{be} at 25 Degrees, $V_{be_{25}}$, volts	0.595
Sensor Temperature Coef, $V_{be_{TC}}$, mv/Deg C	-2.265
Sensor Voltage V_s, Volts	0.516
Bias Supply	
R1, Ohms (-1% tolerance)	12870
R2, Ohms (+1% tolerance)	2222
Bias Voltage V_b	0.736
Amplifi er	
Input Resistor Ri, Ohms (-1%)	6732
Feedback Resistor Rf, Ohms (+1%)	121200
Amplifier Output Voltage V_o, Volts	4.705
A/D Conversion	
Output Count, N_{AD}	240
Software	
Function slope constant M	0.48
Function offset constant B	-55
Displayed Temperature T_d	60

Table 1.2: Spreadsheet Simulation Data and Results

voltage limits at +3.5 volts.

One final word. The spreadsheet simulation is convenient and a useful tool for exploring different scenarious. However, it is only as accurate as the original model. In this example, we have totally ignored the effect of the op-amp offset voltages and currents. If these effects are significant, they must be accounted for in the model.

Put another way, the spreadsheet or other simulation complements but is not a substitute for a working prototype, the necessary but not sufficient condition of a well-engineered circuit.

1.6 Integrated Circuit Temperature Sensors

The diode-connected transistor is a useful temperature sensor by itself but it may be enhanced with additional circuitry. The integrated circuit temperature sensor incorporates this additional circuitry on chip to

- Make the device less dependent on process variations
- Amplify the output signal to a more useable level
- Compensate for the non-linearity of the temperature-V_{be} characteristic

The result is a temperature sensor which is less likely to require calibration, requires less or simpler external circuitry, and is more accurate over its operating range. The cost of the integrated circuit temperature sensor is in the order of $3.50 vs a transistor temperature sensor at $0.35, approximately an order of magnitude. Thus, for all but the most cost-sensitive applications, the integrated circuit temperature sensor is an attractive choice for measuring ambient temperature.

1.6.1 National LM135 Precision Temperature Sensor

Throughout this section, we will use the National LM335AZ, a variant on the LM135 series of devices, as an example. The LM335AZ behaves as a temperature dependent zener diode in which the zener voltage is proportional to the absolute temperature at a rate of 10mV/^{o}K.

The performance specifications of the National LM335AZ are as follows:

Temperature Range	–40 to +100^{o}C				
Operating Current	I_r = 1.0 mA				
Parameter	Condition	Min	Typ	Max	Units
Uncalibrated Temperature Error	25^{o}C,	–	1	3	^{o}C
Uncalibrated Temperature Error	$T_{min} < T_c < T_{max}$	–	2	5	^{o}C
Calibrated Temperature Error	$T_{min} < T_c < T_{max}$	–	0.5	1	^{o}C
Non-linearity		–	0.3	1.5	%

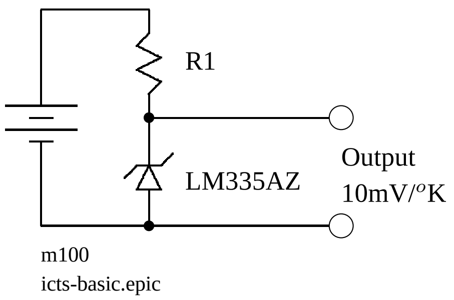
From the graph of Calibrated Error vs Temperature, it would appear that the error increases in a linear fashion over the temperature range. Thus, it would appear that an adjustment of scaling factor in the external circuit would improve the accuracy to that of the sensor non-linearity. As well, the accuracy of the sensor improves around normal ambient temperatures.

The basic application circuit for this device is shown in figure 1.40(a). A potentiometer may be added as shown in figure 1.40(b) to calibrate the device at 25^{o}C.

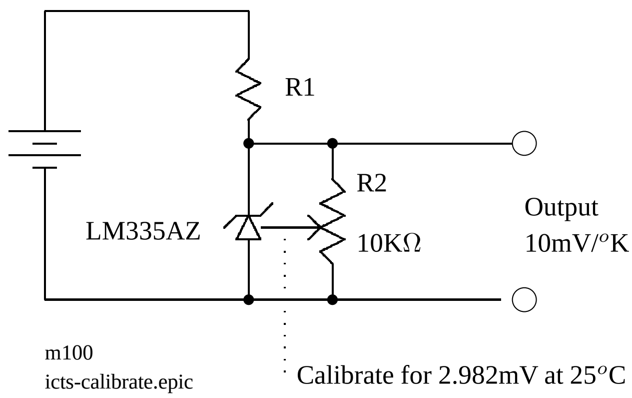
At 25^{o}C, the sensor will sustain a voltage of

$$\begin{aligned} V_{sensor} &= T_{abs} \times 10^{-3} \\ &= (273 + 25) \times 10^{-3} \\ &= 2.98 \text{ volts} \end{aligned}$$

To bias the device the recommended current of 1mA from a power supply of +5 volts, the series resistor should be



(a) Basic Temperature Sensor



(b) Sensor With 25^{o}C Calibration Pot

Figure 1.40: Integrated Circuit Temperature Sensor

$$\begin{aligned} R_1 &= \frac{V_{cc} - V_{sensor}}{I_{sensor}} \\ &= \frac{5 - 2.98}{1 \times 10^{-3}} \\ &= 2.02K\Omega \end{aligned}$$

1.6.2 Generating the Temperature-Sensitive Voltage

Consider the circuit of figure 1.41.

A single diode connected transistor Q1 is driven from a current source I_c. An equal current is fed to a triple of parallel diode-connected transistors, Q_2,Q_3 and Q_4.

Taking the logarithm of both sides of equation 1.11 and solving for the base-emitter voltage V_{be}, the voltage across Q_1 is given by

$$V_{be} = \frac{kT}{q} \ln\left(\frac{I_c}{I_s}\right) \tag{1.58}$$

If the transistors Q_2, Q_3 and Q_4 are identical, then the current I_c will split equally among them and each transistor will conduct one-third of the total. As a result, the base-emitter voltage will be reduced by some amount, as indicated in figure 1.42.

Then, using equation 1.58 again, the base-emitter voltage V'_{be} is

$$V_{be} = \frac{kT}{q} \ln\left(\frac{I'_c}{I_s}\right) \tag{1.59}$$

$$= \frac{kT}{q} \ln\left(\frac{I_c}{3} \times \frac{1}{I_s}\right) \tag{1.60}$$

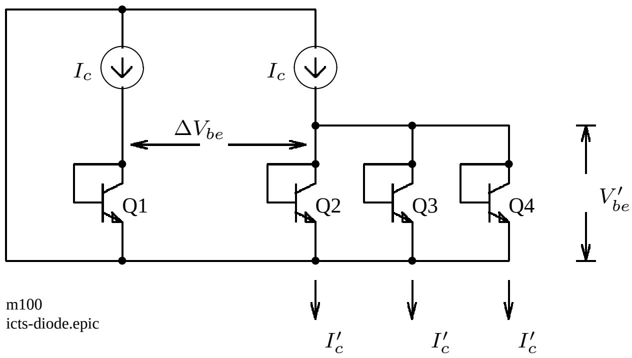


Figure 1.41: Temperature Coefficient Circuit

The voltage difference ΔV_{be} between the two base-emitter voltages is then

$$\begin{aligned}
\Delta V_{be} &= \frac{kT}{q}\ln\left(\frac{I_c}{I_s}\right) - \frac{kT}{q}\ln\left(\frac{I_c}{3I_s}\right) && (1.61)\\
&= \frac{kT}{q}\ln\left(\frac{I_c}{I_s} - \frac{I_c}{3I_s}\right) && (1.62)\\
&= \frac{kT}{q}\ln\left(\frac{I_c}{I_s} \times \frac{3I_s}{I_c}\right) && (1.63)\\
&= \frac{kT}{q}\ln 3 && (1.64)
\end{aligned}$$

We have the happy event that the collector current I_c and the saturation current I_s cancel out of the equation and the value of $\Delta V_b e$ is solely dependent on the absolute temperature.

In practice, ten transistors might be paralleled, in which case at 25^{o}C

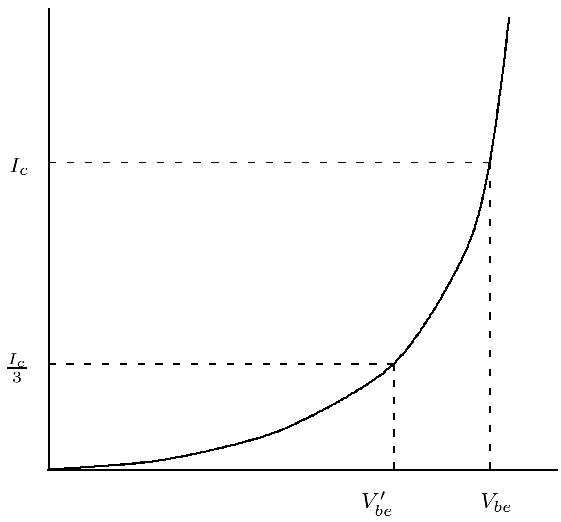
$$\begin{aligned}
\Delta V_{be} &= \frac{kT}{q}\ln 10 && (1.65)\\
&= \frac{1.38 \times 10^{-23} \times 298}{1.6 \times 10^{-19}} \times \ln 10 && (1.66)\\
&= 58.2\,\text{mV} && (1.67)
\end{aligned}$$

Expressed as a coefficient, this is

$$\begin{aligned}
\Delta V_{be} &= \frac{58.2 \times 10^{-3}\text{V}}{293^o\text{K}} && (1.68)\\
&\simeq 0.2\,\text{mV}/^o\text{C} && (1.69)
\end{aligned}$$

1.6.3 Amplifying the Temperature-Sensitive Voltage

The 0.2mV/^{o}K signal is much too small for most practical applications and must be substantially amplified before being used. The circuitry of the LM158 has two purposes: it multiplies the signal by a factor of 50 to 10mV/^{o}K

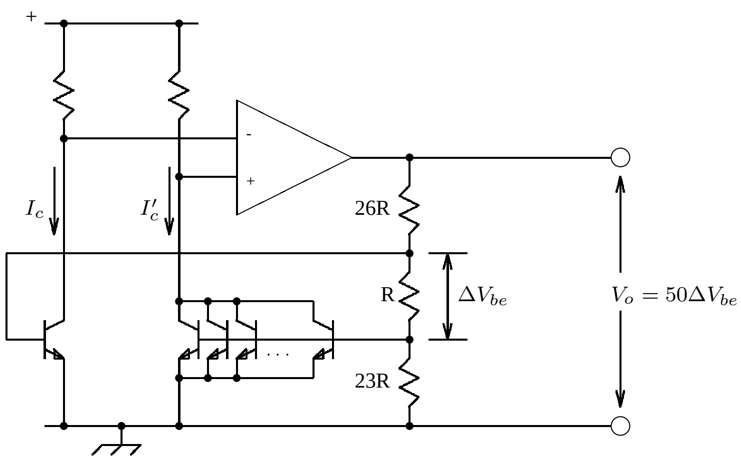


m80
vbe-curves.epic

Figure 1.42: The Effect of Varying Collector Current

and equalizes the currents in the two sets of transistors.

A block diagram of the basic circuit is shown in figure 1.43.



m80
icts-amplifi er.epic

Figure 1.43: Temperature Sensor Amplifier

The voltage ΔV_{be} is developed between the base of a single transistor on one side and a collection of 10 parallel transistors on the other side.

The collector resistors R_c translate the two collector currents into voltages, which are then compared in the operational amplifier. If they are not equal, the output of the op-amp increases or decreases the voltage across the R-49R resistor string. This change in base-emitter voltage of the transistors changes the collector currents in such a direction as to minimize the original difference in current.

When the system settles down, the voltage difference at the input terminals to the op amp must be close to zero, and the value of ΔV_{be} must be such as to make the two collector currents equal. Because of the divider action of the resistor string, the output voltage must be $50 \times \Delta V_{be}$.

1.6.4 Integrated Circuit Temperature Sensor Interface Circuit

The output of the integrated circuit voltage sensor is 10mV/^{o}C, which must be amplified to provide a sufficient input signal to the A-D converter. The requirements of this circuit are similar to that of 1.4.5 for amplifying the signal from a diode-connected transistor sensor.

Designing for the same temperature range (-40^{o}C to +60^{o}C) and redesigning that circuit to accomodate the LM335 temperature sensor, we have the result shown in figure 1.44.

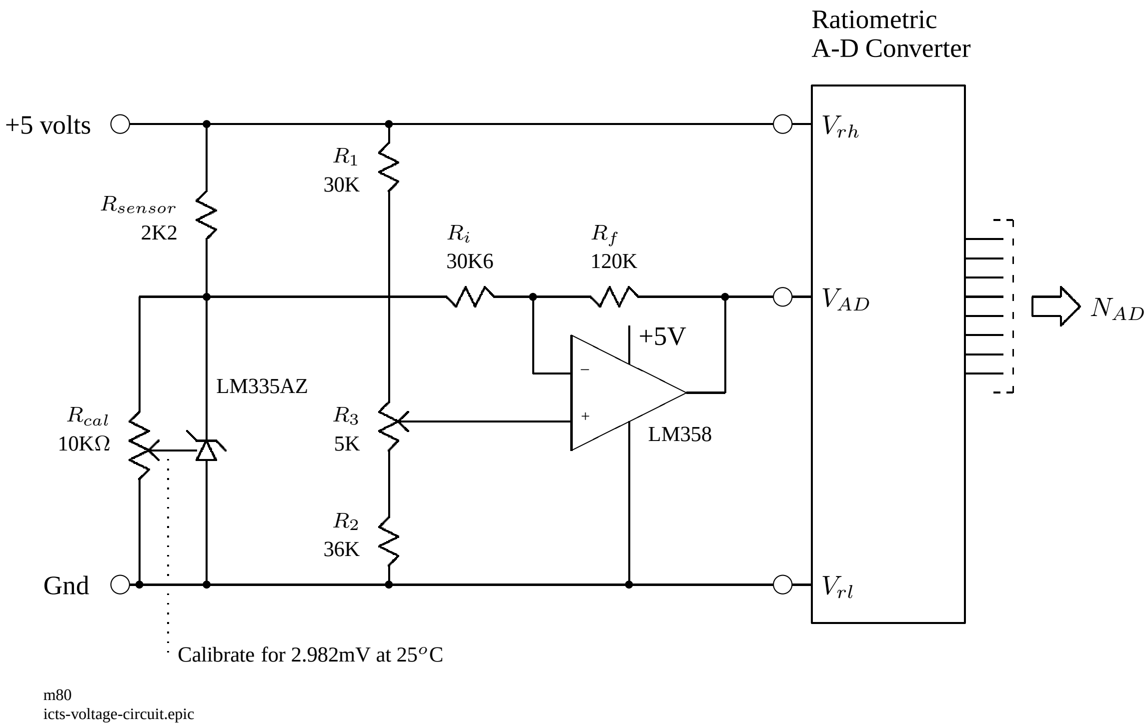


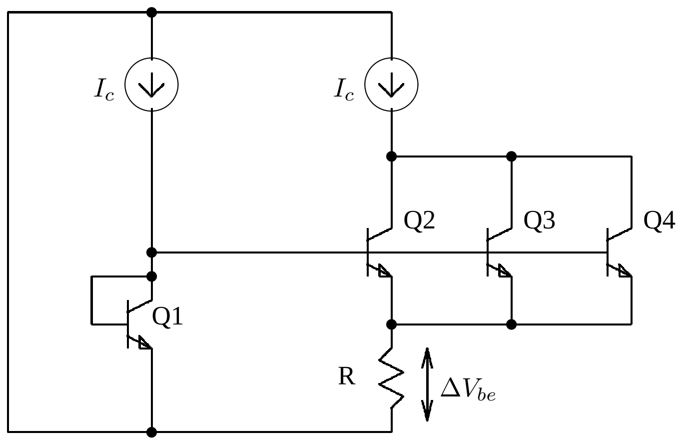
Figure 1.44: Temperature-Voltage Interface Circuit

The calculation of the components values of this circuit follows the development of section 1.4.5 and is left as an exercise.

1.7 Temperature-Current Transducer

When we design a high-resolution temperature measuring circuit, it will turn out to be useful to have a temperature sensor with output in the form of a current, rather than voltage, proportional to temperature.

In section 1.6.2, we showed that the difference between base-emitter voltages of two groups of transistors is proportional to absolute temperature when the two groups of transistors are operated at equal collector currents. The temperature-current transducer concept is a modification of this basic idea, as shown in figure 1.45 [30].



m100
current-temperature-concept.epic

Figure 1.45: Temperature-Current Transducer Concept

Transistor Q1 and transistor group Q2, Q3 and Q4 are operated at the same collector currents and so have different base-emitter voltages. The difference between these two voltages now appears across resistor R. Since this voltage difference is proportional to temperature, the current in R must also be proportional to temperature. Voila, temperature to current transducer.

A simple method of ensuring that the two collector currents are equal is shown in figure 1.46. The current mirror Q5, Q6 ensures that I_1 is equal to I_2.

To develop an equation for the output current, consider the voltage loop comprised of Q1, Q2 and R. From equation 1.64, the voltage V_R across the resistor R will be

$$\Delta V_{be} = \frac{kT}{q} \ln N \tag{1.70}$$

where N is the number of parallel connected transistors, 3 in this case.

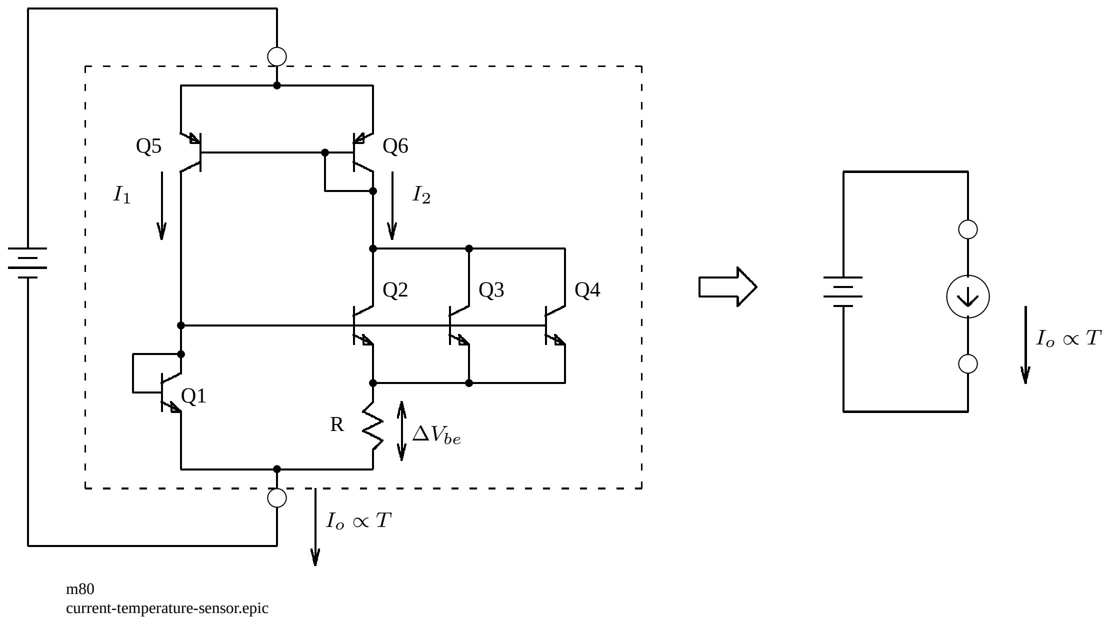


Figure 1.46: Temperature-Current Transducer

The voltage across resistor R is also given by

$$+v_R = I_2 R \tag{1.71}$$

Combine equations 1.70 and 1.71 and we have

$$I_2 = \frac{1}{R}\frac{kT}{q}\ln N \tag{1.72}$$

Since I_1 and I_2 are equal, then

$$I_o = 2 \times I_2 \tag{1.73}$$

$$= \frac{2}{R}\frac{kT}{q}\ln N \tag{1.74}$$

From equation 1.74, we can see that the output current I_o is proportional to absolute temperature T and that the current scale factor may be adjusted by adjusting resistor R.

1.7.1 The National LM134 Temperature Sensor

The LM134 [31] is primarily intended to be used as an adjustable current source. As shown in figure 1.47, once the voltage across the device exceeds about 1.2 volts, the LM134 conducts a current which is independent of the voltage.

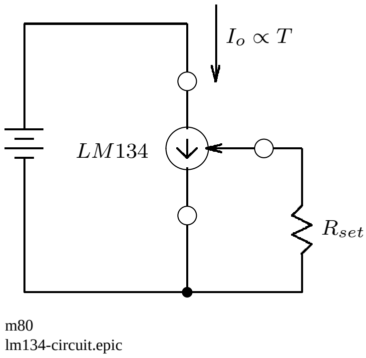


Figure 1.47: LM134 Adjustable Current Source

The value of current is given by

$$I_{set} = \frac{(227\mu V/^oK)(T)}{R_{set}} \tag{1.75}$$

where T is the temperature in ^{o}K.

A practical range of I_{set} is between 100 μA and 1mA. If R_{set} has a suitably low temperature coefficient (less than 20 ppm), the data sheet indicates that the temperature will be within 1% of its correct value.

The temperature dependent current may be converted to a voltage with a series resistance, as shown in figure 1.48.

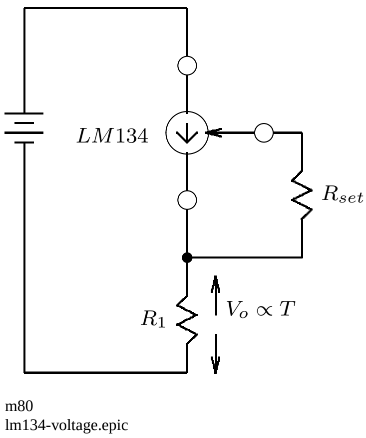


Figure 1.48: Conversion of Current to Voltage

1.8 A High Resolution Temperature Sensor

To this point, all the temperature sensors have had a resolution and accuracy in the order of half a degree centigrade. If it is feasible, a resolution of a tenth of a degree would be useful, and so we now examine a suitable design.

The design of the high resolution temperature sensor begins with the realization that the 8 bit A-D of the microprocessor is capable of a resolution of only 1 part in 256. Over a span of -30^{o}C to +50^{o}C, a range of 80^{o}C, this permits a maximum resolution of $80/256 \approx 0.3$ degrees, less than we need for 'high resolution'. However, the microprocessor also has a pulse accumulator and three 16 bit timer capture registers that can be used to measure a frequency or time interval. If the temperature can somehow be converted into a varying frequency, the microprocessor can resolve the temperature to much better than one part in 256.

The key to converting the temperature to a time interval is the voltage to frequency converter. It turns out to be convenient to use a temperature to current sensor (LM134, described in section 1.7.1) to generate an electrical current proportional to temperature. The V-F converter then converts this signal to a train of pulses, where the frequency is proportional to the input current. The schematic of an application circuit, redrawn from [34] is shown in figure 1.49.

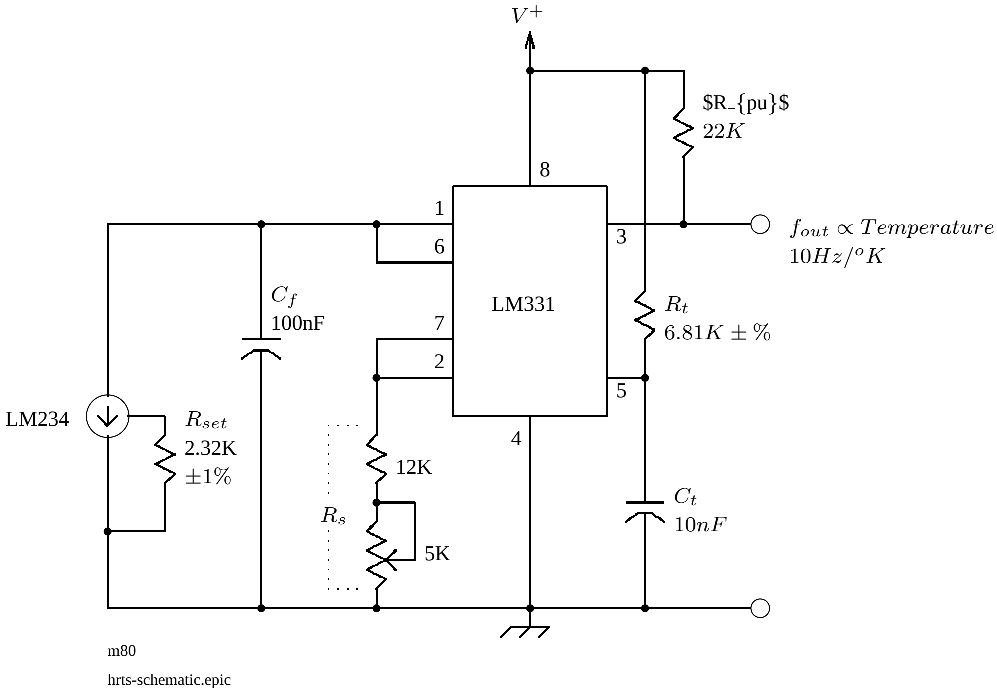


Figure 1.49: Schematic Diagram, High Resolution Temperature Sensor

The output of the circuit is 10Hz/^{o}K, so the output at various temperatures would be as shown in figure 1.50.

Temperature, ^{o}C	Temperature, ^{o}K	Frequency, Hz	Period, μSec
–30	243	2430	412
+25	298	2980	336
+50	323	3230	309

Figure 1.50: Temperature to Frequency Conversion Results

1.8.1 Analysis of the Temperature-Frequency Converter

The equivalent circuit diagram for the temperature-frequency converter of figure 1.49 is as shown in figure 1.51.

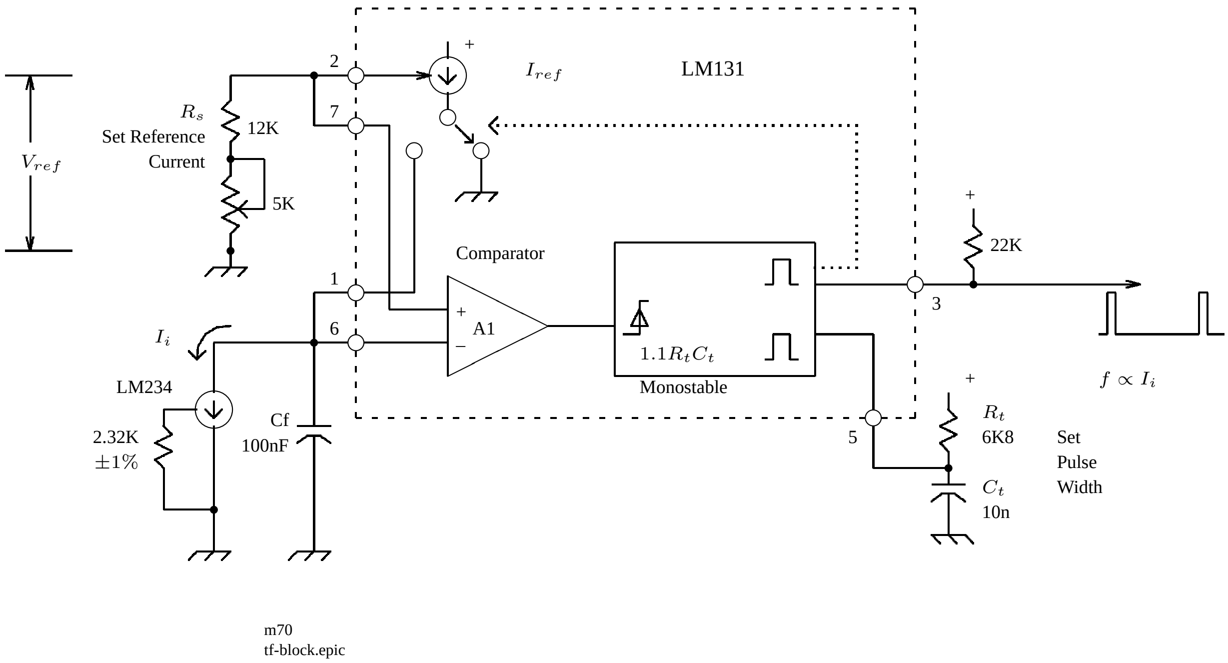


Figure 1.51: Internal Details, High Resolution Temperature Sensor

The reference current is summed directly in capacitor C_f, the filter capacitor. The temperature sensor discharges current out of the filter capacitor at a rate proportional to the absolute temperature, causing V_f to ramp downward.

The comparator compares the voltage on C_f with the reference voltage, 1.9 volts. When the voltage on C_f drops below V_{ref}, the comparator changes state, triggering the monostable. The monostable pulse appears at the output on pin 3 and causes the current switch to divert current into the filter capacitor. This causes the voltage on C_f to ramp upwards. Feedback action ensures that the charge removed by the temperature sensor is exactly balanced by the charge added during the monostable pulse interval.

The transfer function of the current-frequency converter may be shown to be:

$$\frac{f_o}{i_i} = \frac{R_s}{V_{ref}} \frac{1}{1.1 R_t C_t} \text{ Hz/amp} \tag{1.76}$$

The output current of the LM134 temperature sensor is

$$i_t = \frac{K_t T_k}{R_{set}} \tag{1.77}$$

where

K_t = 227 μvolts
T_k is the temperature in Kelvin
R_{set} is as shown in figure 1.51.

Substituting equation i_t from equation 1.77 for i_i in equation 1.76, we have the transfer function of the high resolution temperature sensor, in Hz/Kelvin.

$$\frac{f_o}{T_k} = \left(\frac{K_t}{R_{set}} \frac{R_s}{V_{ref}} \frac{1}{1.1 R_t C_t} \right) \text{ Hz/Kelvin} \tag{1.78}$$

As a check on the operation of the temperature-frequency converter, we will plug the known quantities, including a temperature of 25^{o}C and the corresponding output frequency ($10 \times (273 + 25) = 2980$ Hz), into equation 1.78. Solving for the value of the adjustable resistor R_s should yield a value between 12 and 17KΩ if the circuit can be adjusted properly.

$$2980 = \left(\frac{227 \times 10^{-6}}{2320} \frac{R_s}{1.9} \frac{1}{1.1 \times 6810 \times 10^10{-9}} \right) 293 \tag{1.79}$$

Solving for R_s, we obtain

$$R_s = 14.6 K\Omega$$

which is within the range of adjustment of R_s.

For this value of R_s, then, the transfer function of the temperature-frequency converter will be 10 Hz/Kelvin, as promised in the original application note.

Design Concept

Deriving the transfer function for this application circuit is a lot of work. Why bother when the value is given on the application note?

Two important reasons:

First, it sometimes happens that the application circuit simply won't work as advertized, or was designed for a particular set of circumstances that don't apply to your situation. By working through the analysis, you can check that the original assumptions of the designer were correct and will transfer to your design problem.

An excellent way to get burned is to use someone else's circuit without careful checking that it will work.

Second, the transfer function specifies the interdependencies of the various components in the circuit. In this particular example, the value of the capacitor C_f is not important to the value of the transfer function. On the other hand, the value of the timing capacitor C_t **does** show up in the transfer function, so it is critical that it be a stable capacitor near the value specified. A good circuit designer understands which components are important, and which can be taken for granted or substituted.

In the next section we'll have a look at how these frequencies would be measured by a microprocessor.

1.8.2 Frequency Measurement by Accumulating Pulses

The microprocessor contains a pulse accumulator, an 8 bit counter which may be configured to increment each time a pulse occurs. It may also be be read or written to by machine instructions, and may be set up to generate an interrupt each time it overflows from 255 back to 0. The pulse accumulator is ideally suited to measuring the frequency of a pulse waveform.

To use the pulse accumulator in a frequency counter application, the pulse train to be measured is counted in the pulse accumulator. At the end of a time interval the processor reads the accumulated total and divides it by the elapsed time to determine the frequency. If the elapsed time is one second, then the frequency is simply the accumulated count.

Because the pulse accumulator is only an 8 bit counter and the maximum count will exceed 255 for satisfactory resolution, the counter must extended by counting accumulator overflows with an interrupt service routine. The pulse accumulator interrupts will occur at 1/256 of the input frequency so the resultant processor computing load is not excessive.

The timebase for the frequency counter is another interrupt at the timebase interval, which might be 1 second[6]. This interrupt reads and displays the total count, and then clears the pulse accumulator to start another measurement cycle. Notice that the accuracy of the frequency measurement can be increased with a longer timebase at the expense of a slower update rate of the frequency reading.

In summary then:

- The temperature sensor and voltage-frequency converter of figure 1.49 are used to generate a pulse waveform of frequency 10 times the temperature in degrees Kelvin.

- In the processor, a 1 second real-time-interrupt (RTI) is used to establish the frequency counter timebase. At the beginning of a measurement cycle, the RTI service routine clears the 8 bit pulse accumulator and the byte used to count accumulator overflows.

- The pulse accumulator counts the incoming pulses. Every 256 pulses, the pulse accumulator generates an accumulator overflow interrupt. The service routine for this interrupt simply increments the overflow count byte.

- At the next 1 second real-time interrupt, the RTI service concatenates the current pulse accumulator count and the overflow count into a 16 bit count of pulses during the 1 second RTI interval.

- A display routine subtracts a count of 2730 and displays the result as the temperature in tenths of a degree centigrade.

[6] In practice, the real time interrupt would more likely be in the range of 1/60 second. A one-second interrupt would then be derived from the fast interrupt service routine. Every 60 interrupts, the RTI would determine the number of accumulated pulses. The real-time interrupt on the ubiquitous P/C has is at the bizarre value of 1/18.206 seconds, which complicates accurate timekeeping.

1.8.3 Measurement of Period

As an alternative to frequency measurement using the pulse accumulator, it is also possible for the microprocessor to use the input capture facility to measure the time between pulses (the period of the pulse train), and then relate this value back through the various transfer functions to determine the original sensor temperature.

In this case, however, the voltage to frequency converter circuit must be modified. Using the input capture method, each time a pulse occurs, the microprocessor timer hardware captures the corresponding 16 bit time value and generates an interrupt. The interrupt handler uses this time value to determine the elapsed time since the last interrupt, thereby obtaining the period of the pulse train. The free-running counter that serves as the master timer in this system has a maximum frequency of 1MHz, and so the input capture readings have a resolution of 1 μsecond.

The last column of figure 1.50 shows that the range of measured periods is from 412 μseconds to 309 μseconds. The change in count will be $412 - 309 = 103$, far less than the 800 count difference that is needed for a resolution of 0.1^o over 80^{o}C. Thus we can use the application circuit as a basis for our design, but the component values must be recooked.

A block diagram of the system, with the various transfer functions, is shown in figure 1.52.

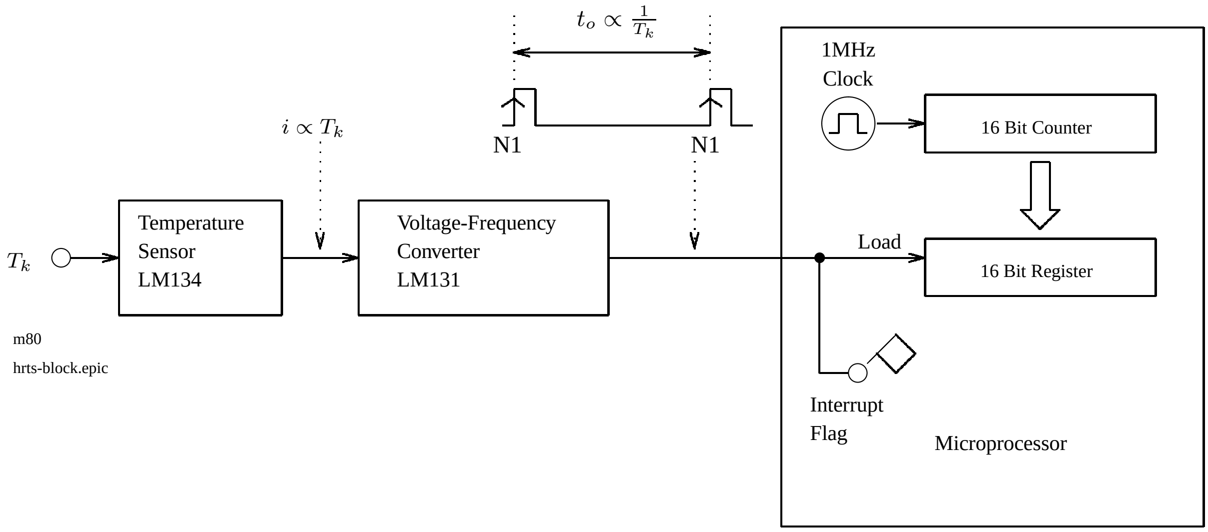


Figure 1.52: Block Diagram, High Resolution Temperature Sensor

The temperature transducer produces a current proportional to absolute temperature. The voltage-frequency converter generates a pulse train at frequency proportional to temperature or, what is saying the same thing, at a period inversely proportional to temperature. Each time a positive edge occurs in the pulse train, the microprocessor transfers the 16 bit count value to a 16 bit register and generates a processor interrupt. The processor determines the elapsed time between the two edges (the period of the pulse train) by subtracting two successive count values N1 and N2.

Our task is to determine a suitable value for the V-F components by relating the temperature to the microprocessor count value. We previously determined that the transfer function of temperature-frequency was given by equation 1.79. Then the equation for period is simply the inverse of this,

$$t_o = \left(\frac{R_{set}}{K_t} \frac{V_{ref}}{R_s} 1.1 R_t C_t \right) \frac{1}{T_k} \text{ seconds} \tag{1.80}$$

The value of the count N_c accumulated in this interval t_o is equal to the product of the counter master clock frequency f_c times the interval.

$$N_c = f_c t_o \tag{1.81}$$

Substituting for t_o in equation 1.80 and solving for the count value, we have

$$N_c = f_c \left(\frac{R_{set}}{K_t} \frac{V_{ref}}{R_s} 1.1 R_t C_t \right) \frac{1}{T_k} \tag{1.82}$$

This equation effectively consists of a constant term times the reciprocal of absolute temperature:

$$N_c = (\text{ a constant }) \frac{1}{T_k} \tag{1.83}$$

This tells us that the relationship between absolute temperature and measured interval will be in the form of a hyperbola, figure 1.53.

As a result, the number of counts per degree of temperature change is variable and depends on the temperature. This is not welcome news, for it means that the microprocessor will need to do a calculation (specifically, a division operation) to relate the measured count back to the value of the temperature.

The graph also indicates that we must ensure that the resolution is sufficient, that is, that the slope of the count-temperature curve is never less than 10 counts per degree. To do this, we must determine an expression for the slope of the count-temperature curve and relate that to the component values of the v-f converter.

The slope of the count-temperature curve is found by differentiating equation 1.80. (To minimize writing, all the constant terms have been bundled into one constant K_{vf}).

$$\frac{dN_c}{dT} = \frac{d}{dT} \left(K_{vf} \frac{1}{T_k} \right) \tag{1.84}$$

$$= K_{vf} \, \frac{-1}{T_k^2} \tag{1.85}$$

From figure 1.53, the minimum value of the slope occurs at the maximum temperature, 50^{o}C, or 323^{o}K. The minus sign can be thrown away, since it is the absolute value of the slope we are interested in. (Whether the count increases or decreases with increasing temperature is not important, since the microprocessor software can deal equally well with either case.) Putting T_k equal to 323, and dN/dT equal to 10, we can solve for the value of K_{vf}:

$$10 = K_{vf} \, \frac{-1}{323^2}$$

from which

$$K_{vf} = 1.043 \times 10^6$$

Now we can expand K_{vf} back into the full equation and solve for the unknown quantities:

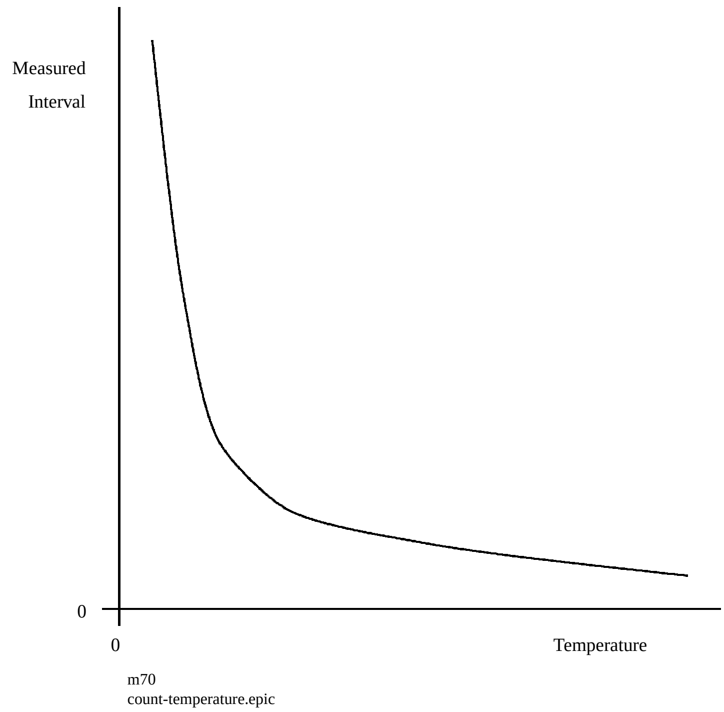


Figure 1.53: Count-Temperature Curve

$$f_c \frac{R_{set}}{K_T} \frac{V_{ref}}{R_s} 1.1 R_t C_t = 1.043 \times 10^6 \tag{1.86}$$

The values of the voltage-frequency converter reference voltage V_{ref} and the current sensor constant K_T are fixed and cannot be changed. A review of the data sheet and applications information for the v-f converter reveals that the reference current is never changed significantly from its value of 135 μamps, so the resistor R_s that sets the reference current should probably be left at 14.1KΩ. For similar reasons, the temperature sensor resistor R_{set} should not be changed significantly.

The microprocessor count frequency f_c can be set to 1MHz, 250KHz, 125KHz or 62.5KHz. The best resolution for a given pulse period will occur at the higher count frequency, or in other words, a higher count frequency will allow a shorter measurement period for the same resolution. A shorter period is desireable because a larger period requires larger timing resistor and capacitor, and components tend to become less accurate and stable at larger values. The best choice, then, is the 1MHz count frequency.

We are left with R_t and C_t, which determine the period of the voltage-frequency converter charge pulse T_p.

In summary, the known quantities are:

f_c	$1 \times 10^6 V$
R_{set}	2320 Ω
K_T	227 μvolts/^{o}K
V_{ref}	1.9 volts
R_s	14.1KΩ

Plugging these quantities into equation 1.86, we have

$$1 \times 10^6 \frac{2320}{227 \times 10^{-6}} \frac{1.9}{14100} 1.1 R_t C_t = 1.043 \times 10^6 \tag{1.87}$$

from which

$$R_t C_t = 757 \times 10^6 \tag{1.88}$$

The original value of timing capacitor C_t was 10nF. Choosing that value again, we have for the timing resistor R_t

$$R_t = \frac{757 \times 10^6}{10 \times 10{-9}} \tag{1.89}$$

$$= 7.57 K\Omega \tag{1.90}$$

Substituting all the known values in equation 1.80, we have

$$N_c = \left(1.147 \times 10^6\right) \frac{1}{T_k} \tag{1.91}$$

This is the key equation relating microprocessor count to absolute temperature. The microprocessor would solve for absolute temperature by dividing the measured count into the equation constant, using floating point arithmetic.

A graph of the equation over the range -30^{o}C to +50^{o}C is shown in figure 1.54.

A tangent at the temperature of 50^{o}C can be seen to have a value of approximately 10 counts per degree, which confirms the analysis.

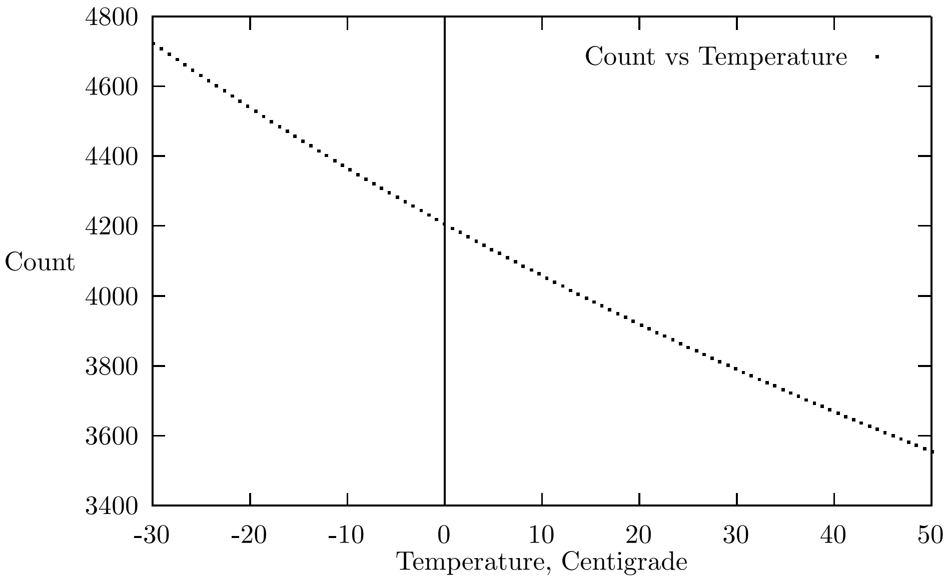


Figure 1.54: Count VS Temperature

1.9 Temperature Measurement Summary

In this chapter, we studied a variety of electronic ambient temperature devices: the thermistor, the diode-connected transistor and two integrated circuit temperature sensors: one with a voltage output and the other with a current output.

The final circuit of each, with a brief summary of its properties, is given below.

1.9.1 Thermistor

It is hard to imagine a simpler measurement circuit than that of the thermistor temperature sensor circuit shown for a 10KΩ thermistor in figure 1.55. The sensitivity of the thermistor to temperature makes it possible to connect the thermistor directly to the A-D converter without further amplification.

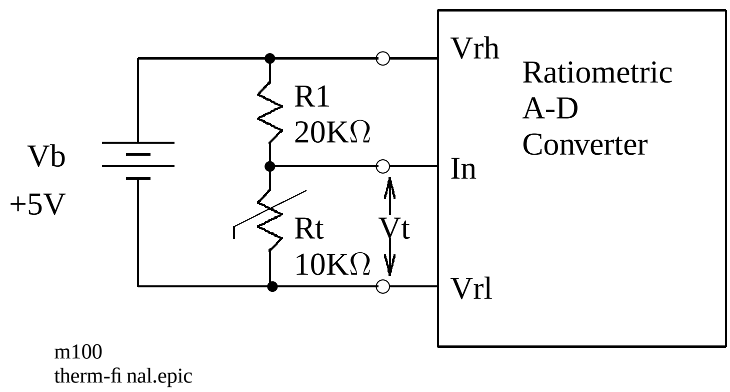


Figure 1.55: Final Thermistor Circuit

The non-linearity of the thermistor makes it necessary to use a lookup table or floating-point mathematical calculation to relate the A-D converter output back to the temperature of the thermistor.

A useful equation for the behaviour of the thermistor is equation 1.3.

$$R_T = R_{25}\, e^{\beta\left(\frac{1}{T} - \frac{1}{T_o}\right)} \tag{1.92}$$

This equation is easier to use but not accurate as accurate as the Steinhart-Hart equation, 1.4:

$$\frac{1}{T} = a + b\, \ln R_T + c\,(\ln R_T)^3 \tag{1.93}$$

The Steinhart-Hart equation is significantly more accurate than the standard thermistor equation at temperatures below zero ^{o}C.

For an inexpensive thermistor, the ultimate accuracy of the measuring circuit was only $\pm 3^o$C, not all that accurate.

In general, thermistors become progressively more inaccurate at low temperatures and are best used at temperatures above 0^{o}C. More accurate and linearized thermistors are available but at greater cost than other sensors of comparable accuracy.

1.9.2 Diode-Connected Transistor

The base-emitter voltage of a diode connected transistor changes at a a predictable and linear rate of about -2.2 millivolts/^{o}C. It must then be amplified to generate a useful swing for the A-D converter input. The final circuit is shown in figure 1.22.

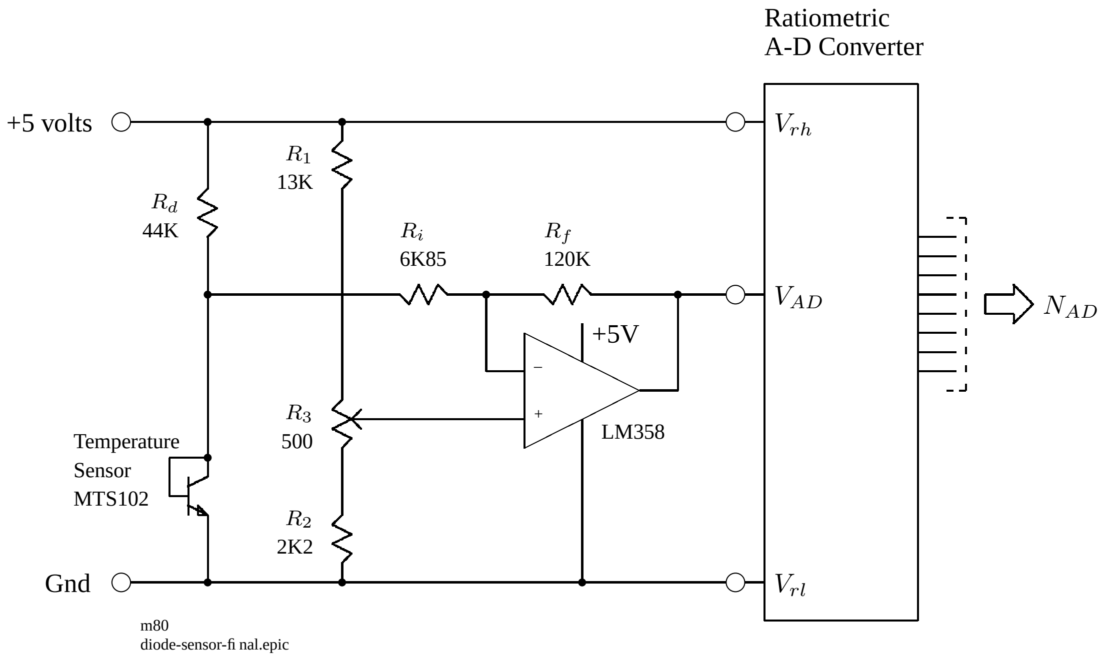


Figure 1.56: Temperature Sensor Circuit

Although any transistor can be used as a sensor, a transistor characterized specifically for temperature sensing (the MTS102) is available at low cost from Motorola Semiconductor.

Because of the large gain required of this circuit, the behaviour of the op-amp is a particular concern. In particular, the offset voltage, offset current, bias current and (if operated from an unregulated supply) the power supply rejection ratio must all be checked to ensure the amplifier drift is within acceptable limits.

For this circuit, the accuracy should be within 1 degree of the correct temperature over a range of -30 to +50^{o}C. The resolution, limited by the 8 bit A-D converter, is approximately 0.5^{o}C.

In general, the base-emitter voltage at some temperature may be predicted using equation 1.13 and a knowledge of the base-emitter voltage at a reference temperature. The slight non-linearity of the base-emitter voltage characteristic with temperature may be compensated by a squaring circuit, in which case it is possible to obtain an accuracy of $\pm 0.1^o$C.

1.9.3 Integrated Circuit Temperature-Voltage Sensor

For integrated transistors of different emitter areas operated at the same collector current, the difference between the base emitter voltages of the two transistors is proportional to temperature. This technique cancels out the leakage current induced non-linearity of a single diode-connected transistor. The integrated circuit can also incorporate an on-chip amplifier, reducing the requirements for the external amplifier.

The final circuit is similar to figure 1.56. With an sensor output signal of 10mV/^{o}K compared to the 2.2mV/^{o}/K of the diode-connected transistor, less amplifier gain is required to drive the A-D converter.

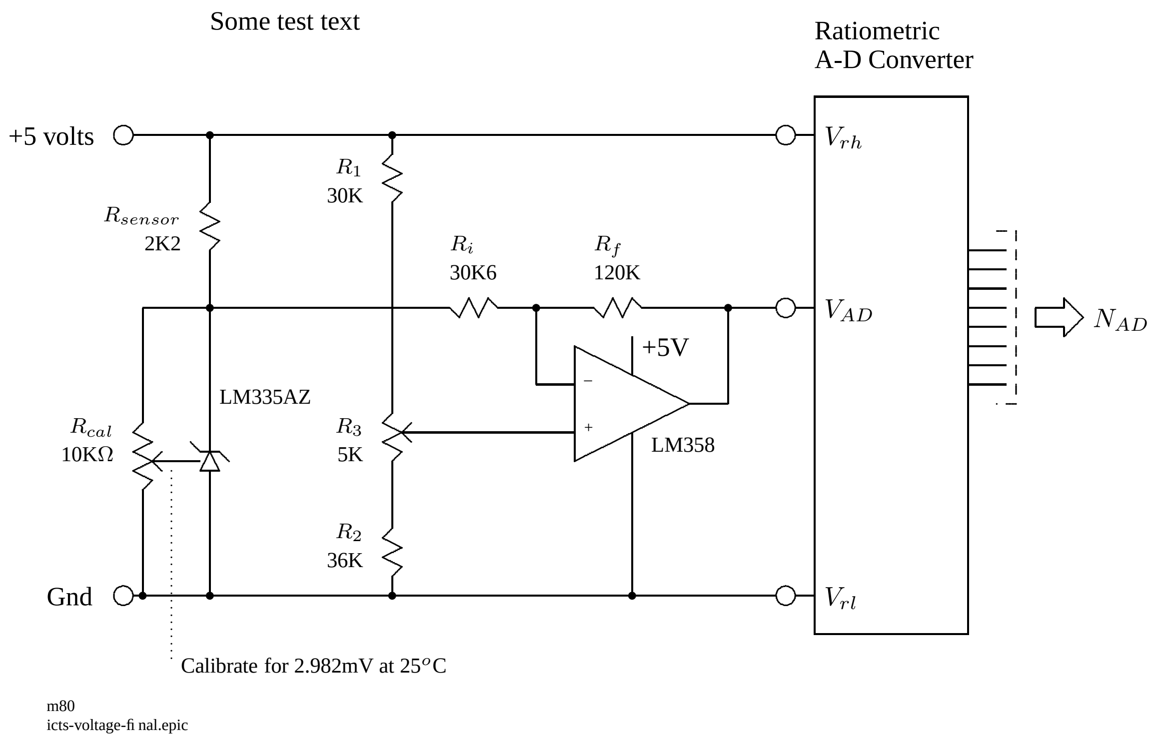


Figure 1.57: Temperature-Voltage Final Circuit

As a result, the amplifier contributes less error than it did in the diode-connected transistor sensor amplifier. The worst-case temperature error circuit is primarily dependent on the temperature sensor, which is specified to be about $\pm 0.5^o$C over the range -40 to +50^{o}C.

1.9.4 Integrated Circuit Temperature-Current Sensor

Using similar techniques to the Temperature-Voltage sensor, the Temperature-Current sensor generates a current proportional to temperature. The current may be converted into a voltage by a resistor. In section we showed how the current could be converted into a frequency with a voltage to frequency converter and then measured to 0.1% resolution using the microprocessor counter circuits.

The final high resolution temperature measuring circuit is shown in figure 1.58.

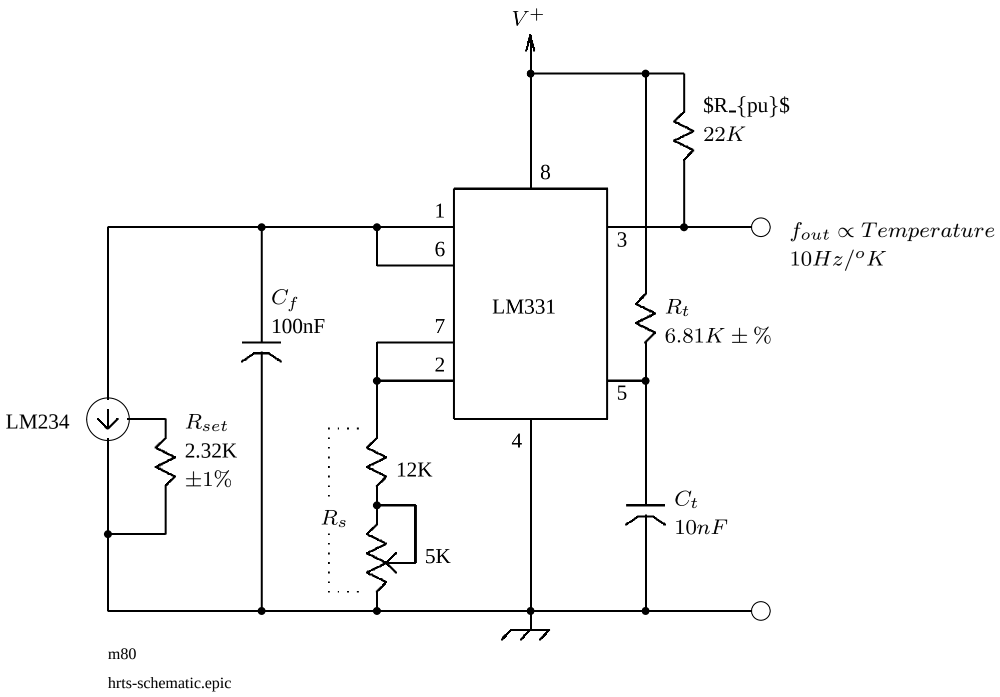


Figure 1.58: Schematic Diagram, High Resolution Temperature Sensor

Chapter 2

Humidity

It's not the heat, it's the humidity. — Everyone

2.1 Introduction

Water evaporates into the atmosphere, making it more or less humid. The degree of humidity is important for human comfort: if the air is too dry, typical of winter in Canada, the body becomes susceptible to infections, and sore throats and colds take up residence in the respiratory tract. As well, conditions of low humidity support the buildup of electrostatic charge, visible in the familiar winter zapping of electrostatic charge that occurs after dragging our feet across a synthetic carpet.

In the summer, if the air becomes too wet the human body has difficulty using evaporation to cool itself. Elevated temperatures and humidity are an unpleasant combination for any kind of human activity.

The amount of moisture that can exist in the atmosphere increases with temperature, and so an absolute measurement of water in the atmosphere is not very useful. A more helpful measure is the Relative Humidity, H_r, a measure of the amount that the air is saturated with moisture. For example, an H_r of 100% indicates that the air is saturated and no further evaporation can occur. If the temperature then increases, the air will be able to contain more water and it will no longer be saturated. If the temperature is lowered to the dew point, condensation or fog will occur as the evaporated water forms into droplets.

The Hair-hygrometer is commonly used to measure humidity. This device depends on the fact that human hair or hair from the tail of a horse is hydroscopic (absorbs moisture) and expands and contracts with changes in humidity. This small movement is then mechanically amplified with a lever mechanism to drive a dial mechanism or recording pen. The hair hygrometer is not particularly accurate. However, it does have a fast response time and is convenient to read. Most hygrometers seen in peoples homes are of this type.

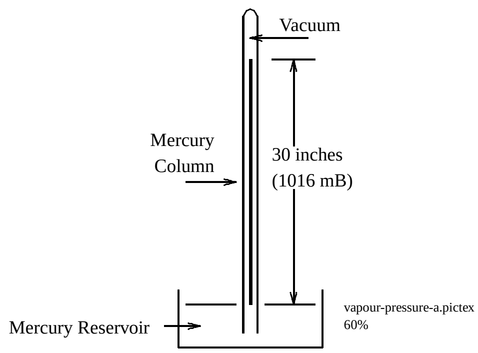
Readers may recall another charming domestic hygrometer: A model house is equipped with two figures, one with and one without an umbrella. Both figures were mounted on a bar that rotated horizontally, moving the figures in and out of doors in the house. As the humidity changed, one or the other figure would emerge from the model house.

2.2 Vapour Pressure

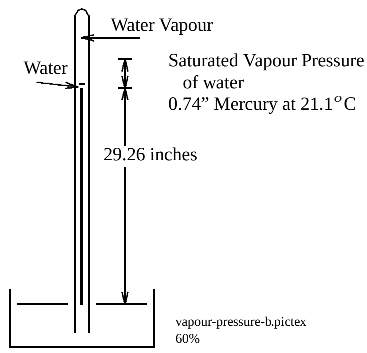
When liquid water is in contact with air, some water molecules have sufficient energy to leave the liquid and become water vapour, the process of evaporation. At the same time, molecules of water vapour are re-entering the liquid water by condensation. If the water vapour is saturated, the rate of evaporation is balanced by the rate of condensation and the two processes are in equilibrium.

The concept of vapour pressure is nicely illustrated by an experiment with an mercury barometer [39]. Water is introduced into the vacuum at the top of a mercury barometer column. As the water evaporates, the level of the mercury column falls. The magnitude of the decrease in column height is equal to the vapour pressure of the water vapour. In effect, the pressure of the water vapour is forcing the column downwards.

If more water is introduced, the column will fall further. However, at some point, liquid water forms and co-exists with the water vapour. The region above the liquid water is now saturated and the mercury column has fallen by an amount equal to the saturated vapour pressure e_{SAT} of the water, figure 2.1 .



(a) Before Introduction of Water



(b) After Introduction of Water

Figure 2.1: Vapour Pressure Demonstration

Further introduction of water has no effect on column height. However, if the temperature of the water is raised, the water molecules will have more energy and the saturation vapour will increase, more water will evaporate, and the column will fall again. Thus e_{SAT} is dependent on temperature.

As implied by the experiment, vapour pressure may be measured using the same units as atmospheric pressure: inches of mercury, or milliBars (mB).

It is important to note that the presence of air is not required and has no effect on the evaporation of liquid water into water vapour. The notion of water vapour dissolving into the air is attractive but inaccurate.

2.3 Relative Humidity

The relative humidity is simply the ratio of the current water vapour pressure to the saturation water vapour pressure. Put another way, relative humidity is the current water vapour pressure expressed as a fraction of the largest possible vapour pressure [40]. As an equation, this is:

$$H_r = 100 \times \frac{e_A}{e_{SAT}} \tag{2.1}$$

where

H_r	is the relative humidity in percent
e_A	is the vapour pressure of the water in the air in milliBars
e_{SAT}	is the maximum (ie, saturated) vapour pressure, in milliBars, at the current ambient temperature.

2.4 Saturation Vapour Pressure

The saturation vapour pressure e_{SAT} is the maximum vapour pressure of water that can exist at a given temperature. It is a fixed and known function of temperature, so if the relative humidity or water vapour pressure are known, the other variable, relative humidity or water vapour pressure, may be calculated.

A table of values of saturation vapour pressure vs temperature, adapted from reference [41, Appendix 7], is shown in figure 2.2.

Temperature, ^{o}C	e_{SAT}, millibars
-10	2.83
-5	4.55
0	6.12
+5	8.72
+10	12.27
+15	17.05
+20	23.43
+25	31.78
+30	42.56
+35	56.25
+40	73.43
+45	94.66
+50	120.61

Note: Air pressure is 1016 millibars.

Figure 2.2: Temperature and Saturation Vapour Pressure of Water

Notice that the saturation vapour pressure increases with temperature. For a constant relative humidity, with increasing temperature a larger quantity of water can be evaporated in the air.

Fitting a curve to the table of figure 2.2 yeilds equation 2.2, which can be used to interpolate between the temperatures of figure 2.2.

$$e_{SAT} = A_4 \cdot T^4 + A_3 \cdot T^3 + A_2 \cdot T^2 + A_1 \cdot T + A_0 \tag{2.2}$$

where

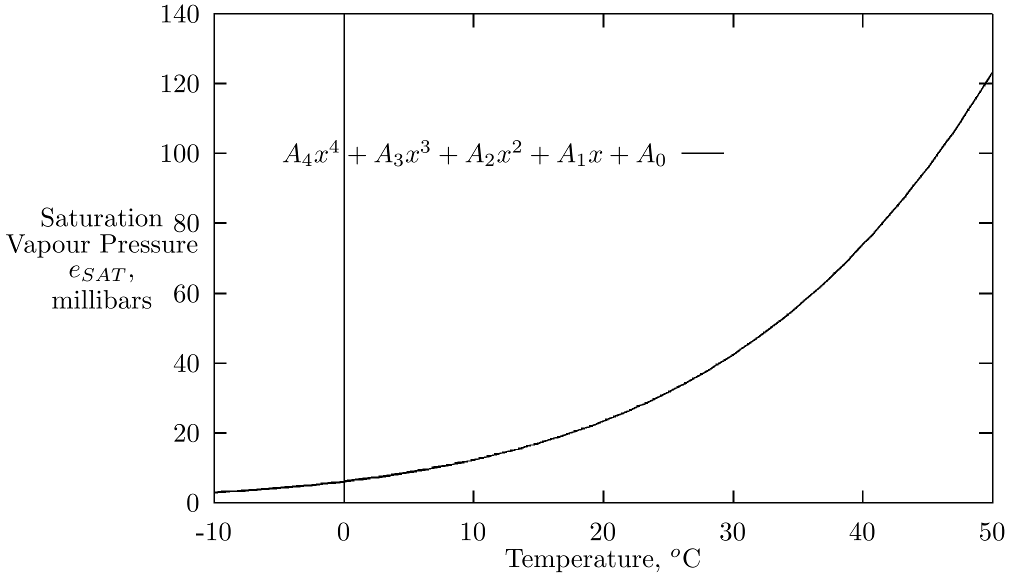


Figure 2.3: Vapour Pressure vs Temperature

e_{SAT}	is the saturated water vapour pressure, in millibars
T	is temperature in degrees C
A_4	= 5.33×10^{-6}
A_3	= 1.92×10^{-4}
A_2	= 0.0149
A_1	= 0.446
A_0	= 6.088

As an example, the saturated vapour pressure of water at a temperature of 21.1^oC is 25.02 millibars.

2.4.1 Vapour Pressure Applied: Humidex

As anyone who has lived though an August heat wave in Toronto can attest, the combination of high ambient temperature and high humidity is most uncomfortable. The body has difficulty cooling itself when humidity is high, so the apparent temperature is much higher than the actual temperature. The Humidex is an attempt to measure this effect.

The Humidex formula is given by:

$$H = T_c + (e_A - 10) \cdot \frac{5}{9} \tag{2.3}$$

where

H	is the humidex reading, given in apparent temperature (^{o}C)
T_c	is the temperature in ^{o}C
e_A	is the ambient vapour pressure

(This formula and many others relating to humidity are listed in the Humidity FAQ, reference [51].)
The Humidex is a quantity that a microprocessor can calculate from temperature and relative humidity.

Example

The current temperature is given as 22^{o}C and the relative humidity 85%. What is the Humidex reading?

Solution

First, substitute the current temperature T into equation 2.2 to determine the saturation water vapour pressure e_{SAT} in millibars.

$$\begin{aligned} e_{SAT} &= A_4 \cdot T^4 + A_3 \cdot T^3 + A_2 \cdot T^2 + A_1 \cdot T + A_0 \\ &= (5.33 \times 10^{-6} \times 22^4) + (1.92 \times 10^{-4} \times 22^3) + (0.0149 \times 22^2) + (0.446 \times 22) + 6.088 \\ &= 26.39 \text{ millibars} \end{aligned}$$

Then use the relative humidity formula, 2.1, to determine the ambient vapour pressure e_A.

$$H_r = 100 \times \frac{e_A}{e_{SAT}}$$

so

$$\begin{aligned} e_A &= \frac{H_r}{100} \times e_{SAT} \\ &= \frac{83}{100} 26.39 \\ &= 21.9 \text{ millibars} \end{aligned}$$

Finally, substitute for T_c and e_A in the humidex equation 2.3 to determine the humidex reading.

$$\begin{aligned} H &= T_c + (e_A - 10)5/9 \\ &= 22 + (21.9 - 10)5/9 \\ &= 28.6 \end{aligned}$$

The humidex reading[1], or apparent temperature, is 28.6^{o}C.

2.5 The Wet-Bulb, Dry-Bulb Hygrometer

A common instrument for measuring relative humidity is the wet-bulb, dry-bulb hygrometer, figure 2.4. This hygrometer is also known technically as a psychrometer [38].

[1]This formula appears to be consistent with the method used by Environment Canada, since their quotations of humidex correspond to the results obtained with this formula

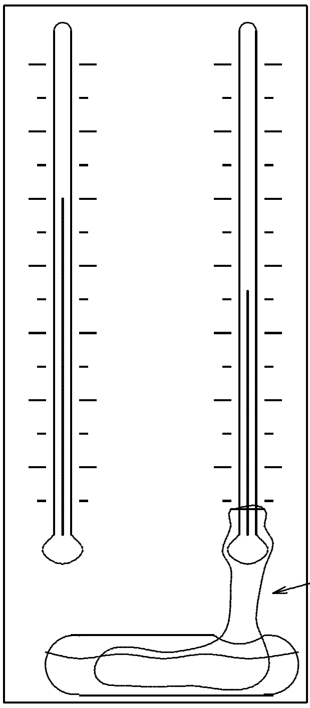


Figure 2.4: Wet-Dry Bulb Hygrometer

The psychrometer consists of two thermometers. One measures the air temperature directly. The other is cooled by a porus muslin wick. The wick is moistened and evaporates water at a rate dependent on the relative humidity, cooling the wet-bulb thermometer. The rate of cooling, and hence the difference between the two thermometer readings, is a measure of the relative humidity.

The rate of cooling of the wet bulb is very dependent on the movement of air past the wet bulb, and some psychrometers are equipped with fans, or handles so they can be swung through the air (the so-called sling psychrometer). Even so, the psychrometer is only about 5% accurate.

The speed of response of the psychrometer depends on the speed of response of the thermometers, some tens of seconds at least. For accurate readings of humidity, both thermometers must have the same time constant [38].

For a manual determination of humidity, the observer uses a chart or special slide rule to relate the ambient temperature and wet-dry temperature difference to relative humidity. For automatic calculation by microprocessor, the system is shown in figure 2.5.

When one temperature sensor is already in place to to measure ambient temperature, it is a simple matter to add a second temperature sensor, wick, water reservoir and fan. For a long-term installation, however, this arrangement has some disadvantages: the fan is subject to clogging with debris or mechanical breakdown, the reservoir must be refilled regularly, and mineral deposits will build up in the reservoir and wick unless distilled water is used.

On the other hand, the operation of a psychrometer can be analysed from basic physical princibles, which is reassuring where accuracy is required.

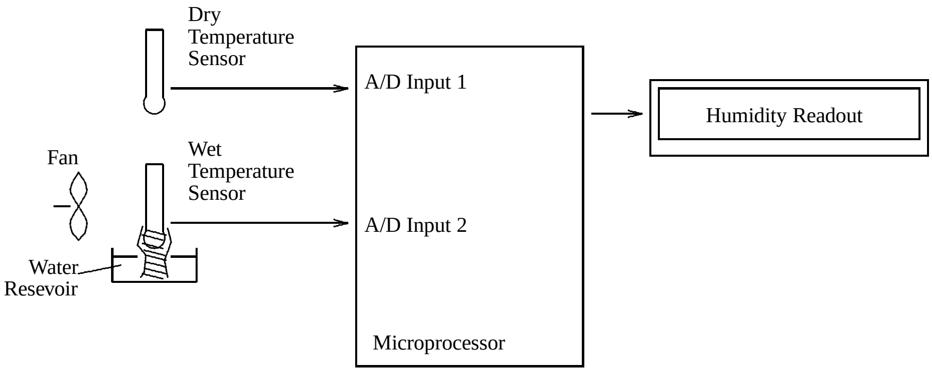


Figure 2.5: Automatic Measurement of Humidity

2.5.1 Computerizing Wet-Bulb, Dry-Bulb Measurements

To automate the measurement of relative humidity from a psychrometer, we need a mathematical method of determining humidity from from wet and dry bulb temperature measurements.

The vapour pressure in the air may be determined from the wet-bulb/dry-bulb hygrometer according to the following formula from reference [40]:

$$e_A = e_W - A \cdot P_A \cdot (T_A - T_W) \tag{2.4}$$

where

e_A	is the vapour pressure of the water in the air, millibars
e_W	is the saturated vapour pressure of water at the temperature of the wet bulb, T_W.
A	is a constant, 6.66×10^{-4} (or 5.94×10^{-4} if the wet bulb is covered with ice)
P_A	is the air pressure, in millibars, typically 1016 mB.
T_A	is the dry bulb (air) temperature, ^{o}C
T_W	is the wet bulb temperature, ^{o}C

In order for this formula to apply, the wet bulb must be in an air flow of at least 3.6 meters/sec, so the wet bulb must be fan cooled.

2.5.2 Wet-Dry Bulb Calculation: An Example

Determine the relative humidity if the dry bulb thermometer is reading 15.5^oC and the wet bulb 12.8^oC.

1. To find the saturated vapour pressure e_W, substitute the wet bulb temperature T_W, 12.8^{o}C, in equation 2.2:

$$\begin{aligned} e_W &= e_{SAT} \\ &= 5.333 \times 10^{-6} \times 12.8^4 \end{aligned}$$

$$
\begin{aligned}
&+ \quad 1.924 \times 10^{-4} \times 12.8^3 \\
&+ \quad 0.0149 \times 12.8^2 \\
&+ \quad 0.444 \times 12.8 \\
&+ \quad 6.088 \\
&= \quad 14.79 \text{ millibars}
\end{aligned}
$$

2. Now apply Wet-Dry Bulb formula 2.4 to determine the actual vapour pressure e_A of the water in the air:

$$
\begin{aligned}
e_A &= e_W - A \cdot P_A \cdot (T_A - T_W) \\
&= 14.77 - 6.66 \times 10^{-4} \times 1016 \times (15.5 - 12.8) \\
&= 12.94 \text{ millibars}
\end{aligned}
$$

3. Next, use equation 2.2 again, this time substituting the dry bulb temperature, 15.5^oC, to determine the saturation vapour pressure e_{SAT} at the dry bulb temperature.

$$e_{SAT} = 17.61 \text{ millibars}$$

4. Finally, use equation 2.1 to calculate the relative humidity:

$$
\begin{aligned}
H_r &= 100 \times \frac{e_A}{e_{SAT}} \\
&= 100 \times \frac{12.94}{17.61} \\
&= 73.5\%
\end{aligned}
$$

   Checking against a psychrometric table, we find that the table shows 73% relative humidity for these same wet-bulb/dry bulb temperatures.

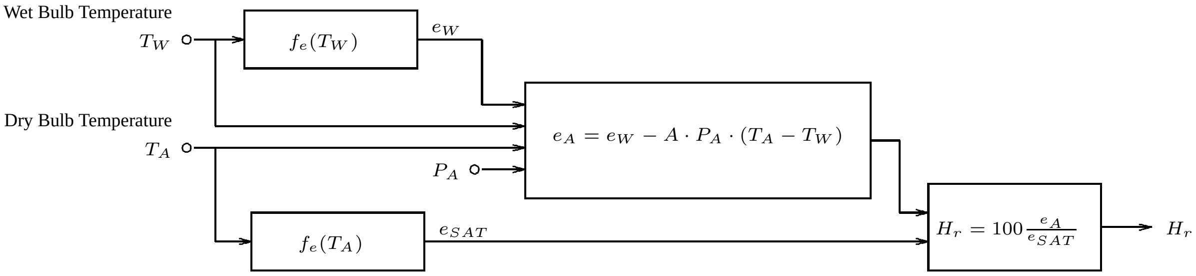
The data flow diagram for this calculation is shown in figure 2.6.

2.6 Dew Point

Relative humidity is a measure of the quantity of water vapour present compared to the maximum water vapour possible. The maximum quantity of water vapour that can exist in the air decreases with temperature (figure 2.3). Thus, if a sample of air is cooled without removing any water vapour, the relative humidity will increase.

As the air is cooled, at some temperature the relative humidity will reach 100% and moisture will begin to form as fog or dew. The temperature at which condensation appears is known as the dew point of the air sample.

For example, suppose a 20^{o}C sample of air contains moisture at a relative humidity of 63%. If the air is cooled to 12^{o}C, moisture will begin to form on the surfaces of the container. Then the dew point of that air sample is 12^{o}C. As a result, relative humidity at some temperature and dew point temperature are two different ways of quantifying the amount of water vapour in the air. One may be determined from the other.



$$f_e(T) = 3.36 \times 10^{-6}T^4 + 2.92 \times 10^{-4}T^3 + 0.139T^2 + 0.443T + 6.123$$

Figure 2.6: Psychrometric Calculation, Data Flow

2.6.1 Dew Point Calculation: An Example

Suppose the ambient air temperature is 25^{o}C and the relative humidity 74%. Calculate the dew point temperature.

Solution

Step 1 Use the current value of temperature to determine the saturation vapour pressure e_{SAT} from figure 2.2, figure 2.3 or equation 2.2.

From equation 2.2, the value of saturation vapour pressure at 24^{o}C is 31.78 millibars.

Step 2 Use the current value of relative humidity H_r and saturation vapour pressure e_{SAT} to determine the current value of water vapour pressure e_A from equation 2.1.

Substituting the value of e_{SAT} determined in step 1, and 74% for H_r in equation 2.1, we have:

$$74\% = 100\frac{e_A}{31.78}$$

Solving for e_A, we have

$$e_A = 23.5 \text{ millibars}$$

Step 3 Use this value of e_A and figure 2.2 or figure 2.3 to determine the temperature at which this vapour pressure is the saturation vapour pressure. This temperature is the dew point.

Referring to figure 2.3 again, the air will be saturated with moisture at this vapour pressure when the temperature cools to about 20^{o}C.

2.6.2 Dew Point Calculation by Computer

The computer solution for dew point temperature proceeds exactly as steps 1 and 2 of the previous example. For step 3, we need a mathematical relationship to relate e_{SAT} to the corresponding value of temperature T. We could find T by substituting for e_{SAT} in equation 2.2 and solving for T. However, it's easier to determine another polynomial to fit the inverse data, that is determine f_T where:

$$T = f_T(e_A)$$

Unfortunately, fitting a polynomial to the data is not very successful. Even a fourth order polynomial is in error by 10% or so. However, recognizing that e_{SAT} is approximately an exponential function of temperature, then a plot of temperature vs the natural log of e_{SAT} will yeild a graph with much less curvature and be easier to fit[2]. Trying this out, we find that a second order polynomial can predict temperature to within one or two percent, which is what is needed.

The result:

$$T = A_2 \cdot [\ln(e_{SAT}]^2 + A_1 \cdot [\ln(e_{SAT}] + A_0 \tag{2.5}$$

where

T	= temperature in ^{o}C
e_{SAT}	= saturation vapour pressure, millibars
A_2	= 1.073
A_1	= 9.642
A_0	= -21.06

For example, to find the temperature corresponding to a vapour pressure of 23.5 millibars:
First, find

$$\begin{aligned} \ln(e_{SAT}) &= \ln(23.5) \\ &= 3.157 \end{aligned}$$

Now, substitute this value in equation 2.5

$$\begin{aligned} T &= A_2 \cdot [\ln(e_{SAT}]^2 + A_1 \cdot [\ln(e_{SAT}] + A_0 \\ &= 1.073 \times 3.157^2 + 9.642 \times 3.157 - 21.06 \\ &= 20.07^oC \end{aligned}$$

which corresponds to our previous result for dew point.

[2]Notice the similarity of this approach to the Steinhart-Hart relationship, equation 1.4 on page 8

2.7 Electronic Measurement of Humidity

We described in section 2.5.1, page 85, how the Wet-Dry Bulb Psychrometer may be adapted to the electronic measurement of humidity [40]. This device requires significant care and feeding, and so is not a popular choice for unattended, long term operation.

Dew point temperature may be measured with the type of instrument described in reference [52]. Then relative humidity may be calculated from the ambient temperature and dew point temperature, as shown in section 2.6 on page 86.

For the direct measurement of humidity, both variable resistance and variable capacitance sensors have been used. Reference [43] describes a home-made sensor that uses the change in resistivity of ordinary table salt (sodium chloride) to measure humidity.

Several commercial sensors rely on the change in the dielectric constant of a capacitor to measure relative humidity. References [45], [44] and [50] show the use of a commercial sensor based on gold-plated plastic film.

2.8 A Capacitance-Humidity Sensor Circuit

In this section, we will spend some time describing a humidity measurement circuit based on the Philips Series 691 humidity sensor, which is inexpensive and readily available.

2.8.1 Sensor Specifications

The electrical specifications of the sensor, adapted the data sheet of reference [45], are shown in figure 2.7:

Electrical Specifi cations	
Operating Humidity Range	10 to 90% R.H.
Operating Temperature Range	0 to 85^o C
Capacitance at 25^{o}C, 43% R.H., 100KHz	122pF $\pm$15%
Frequency Range	1 KHz to 1MHz
Temperature Dependence	0.1% R.H./^{o}C
Response Time (to 90% of indicated R.H. in circulating air):	
Between 10 and 43% R.H.	3 minutes max
Between 43 and 90% R.H.	5 minutes max
Typical Hysterysis (10% R.H to 90% R.H and 90% R.H. to 10% R.H.	3%
Maximum voltage	15V
Storage Temperature Range	-25 to +85^{o}C
Current price	$10.10
Part number	`691-90001`

Figure 2.7: Vapour Pressure Demonstration

The change of capacitance with humidity is shown in figure 2.8.

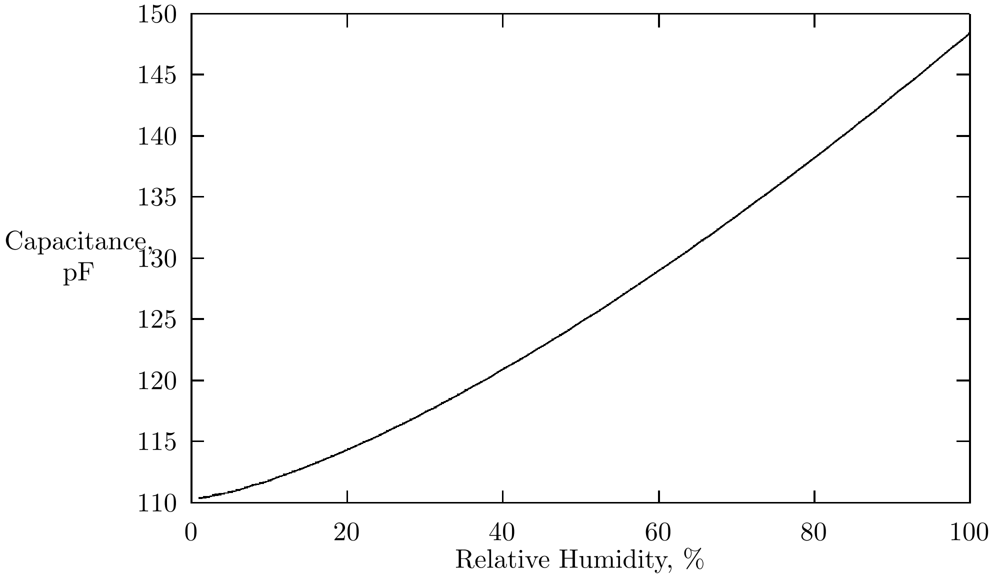


Figure 2.8: Capacitance vs Relative Humidity for Humidity Sensor

The equation of this function is

$$C_s = C_{12}\left(0.985 + 0.34\left(\frac{H_{rel}}{100}\right)^{1.4}\right) \tag{2.6}$$

where

C_s is the capacitance of the sensor C_{12} is the capacitance at a relative humidity of 12%
H_{rel} is the relative humidity in %

A number of important points may be determined from the data sheet information:

Operating Humidity Range The operation of the sensor is not specified below 10% or above 90%, so we're on our own in those regions. The curve probably continues in a smooth fashion into those areas, but we cannot complain to the vendor about accuracy.

Operating Temperature Range The sensor won't work below the freezing temperature of water (which is reasonable) but is not damaged until the temperature drops below –25^{o}C. Probably best to reserve it for indoor use.

Capacitance at 25^{o}C and 43% R.H. This value of capacitance is rather small, as capacitors go. Moreover, looking at figure 2.8, it doesn't change by much either: 110pF to 150pF. The measuring circuit will have to be adapted to these small values. Furthermore, the tolerance given for the base value is a rather large $\pm$15%. Calibration will be required.

Frequency Range A capacitive sensor measurement may be converted to frequency or period by using the capacitive sensor as the timing element in an oscillator. This specification indicates the allowable range of oscillator frequency.

Temperature Dependence Assuming that the sensor will be in an environment when the temperature change is 20^{o}C, the temperature coefficient of 0.1%/^{o}C represents a total total reading change of 2%. If possible, this change should be compensated for.

Response Time If you breath gently into a hair hygrometer, the meter will indicate an increase of humidity, with a time constant in the order of a few seconds. The capacitive sensor is much slower, with a time constant in the order of minutes.

Hysterisis When the sensor is subjected to a series of low humidity, high humidity and low humidity, the final reading may be expected to be in error by 3%. This is a rather large error, but it would be most important in applications such as the humidity sensor for a clothes dryer. In that case, the humidity would suddenly go from a low value to a high value when wet clothes were stuffed into the dryer. The humidity would then drop gradually as the clothes are dried.

For our application, such drastic changes in humidity should be rare[3], and hopefully the hysterisis effect will not be a major problem.

The overall impression, then, is of a sensor with a small change in capacitance and relatively long time constant. It should be compensated for temperature and must be calibrated for humidity, and the expected accuracy should be in the region of $\pm$2%.

2.8.2 Oscillator Measuring Circuits

There are two fundamental methods of converting the variation in sensor capacitance into an electrical signal. The capacitor may be used as one of the frequency-determining elements in an oscillator, the method discussed in this section. In the next section, we discuss an alternative technique.

In section 1.8, we showed how the current output temperature sensor could be converted into a variable frequency or period waveform. The microprocessor could then relate this frequency or period back to the temperature of the sensor.

The same technique is to be used here, with the difference that the variation of the capacitor is used to determine the frequency of oscillation. The question is then what type of oscillator is most suitable. There are an enormous number of possible oscillator circuits, so it we must develop some overriding princibles to narrow down the field. To begin with, the timing circuits of oscillators depend either on inductor-capacitor circuits networks or resistor-capacitor networks.

Figure 2.9 shows a comparison of these two oscillator types.

Because large value inductors are inconvenient to make, , LC oscillators tend to be radio frequency devices (ie, frequency of oscillation greater than 1MHz). The frequency of operation is dependent on the square root of the capacitance, so a doubling of capacitance changes the frequency by a factor of 1.41. On the other hand, RC oscillators tend to operate at frequencies below 1MHz and the frequency is a function of the capacitance directly.

Comparing these two, the RC oscillator appears to be more suitable. The humidity sensor is specified to operated at frequencies below 1Mz. and the extra sensitivity will be useful.

[3]However, the humidity in a household rises quite drastically during cooking hour. This is most noticeable in the winter months when the humidity is low to begin with.

Inductor-Capacitor Oscillators	Resistor Capacitor Oscillators
$f_{osc} > 1$MHz	$f_{osc} <$1MHz
$f_{osc} \propto \frac{1}{\sqrt{LC}}$	$f_{osc} \propto \frac{1}{RC}$

Figure 2.9: LC vs RC Oscillators

Now we have to choose a suitable RC oscillator. Again, because there are many possible RC oscillator designs, let us attempt to narrow down the field with some general observations.

First off, we must decide between measuring the frequency or the period of the oscillator waveform. The frequency is proportional to $1/C$ and period is proportional to C: the choice of period simplifies the calculations. Furthermore, referring to the resources of our microprocessor, we have three input capture ports suited to period measurement, and only one pulse accumulator, suited to frequency measurement. It would make better sense to use the input capture port if we can. Period measurement would seem to be the correct choice.

Next, we need to get an estimate of a suitable period for the oscillator waveform. The reasoning goes something like this:

- The transducer capacitance varies from 150pF to 110pF, for a total variation of 40pF and a minimum capacitance of 110pF. Put another way, the capacitance ranges from its minimum value C to 1.36C as the humidity varies from 0% to 100%.

- The period of the oscillator waveform is proportional RC, so it will vary the same way as the capacitance, from a minimum value of T to a maximum of 1.36T.

- The part that we are interested in is the varying part of T. For a resolution of 1% R.H., the varying part of T must be long enough to contain 100 discrete values. Now the microprocessor timing circuits have a resolution of 1μsecond (at best), so 0.36T must correspond to 100μseconds. Then the non-varying part of the period, 1.0T, must be 277μseconds in duration. All of this is shown in figure 2.10.

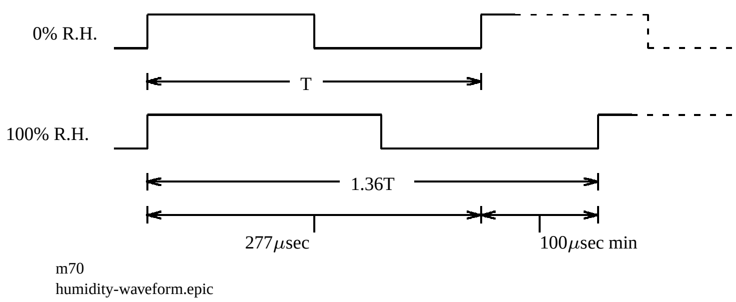


Figure 2.10: Humidity Sensor Oscillator Waveform

- The period of an RC oscillator is given by something like $T = KRC$, where K depends on the circuit, but might be 1.4 or 2.2. For simplicity, we'll assume that K is 1: this will get us into the computational ballpark. Then if RC=277μseconds and C=110pF, the resistance is 2MΩ.

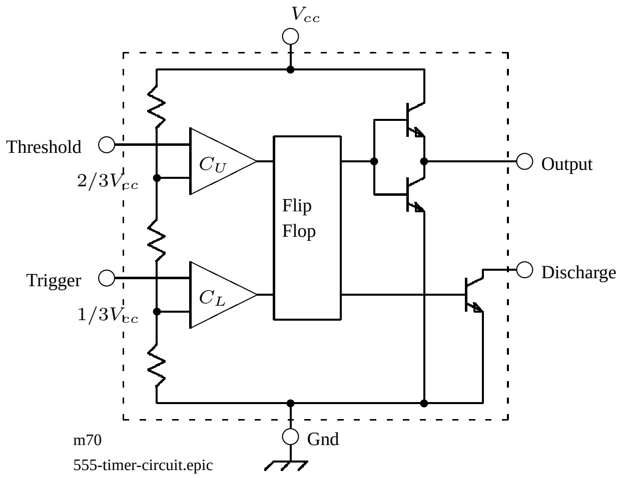
- Now the final step: with the timing resistance in the order of 2MΩ and an operating voltage of 5 volts, we can expect the timing current to be in the order of 2.5μamps. This is a very small value of timing current. It is not all that different from the leakage current of timing devices or the bias current of conventional operational amplifiers. The bias and leakage currents are liable to interfere with the timing current sufficiently to affect timing accuracy.

The timer design example given in section 2.8.3 confirms this effect.

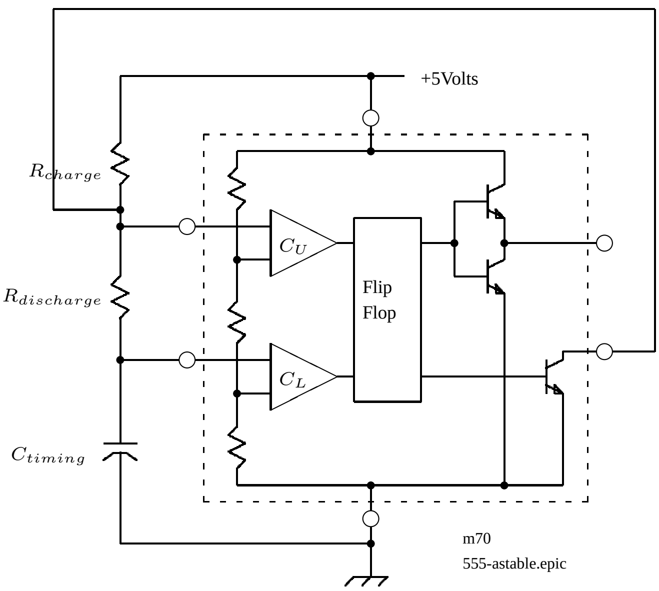
2.8.3 555 Timer Design Example

An obvious candidate for any astable oscillator is the 555 Timer. It is simple to apply, readily available, and inexpensive. As we shall show, however, the small oscillator currents make the 555 Timer a poor choice for this particular application.

A simplified diagram of the internals of the timer is shown in figure 2.11(a). The timer consists of a flip flop which is actuated from two comparators. The thresholds of the two comparators are established at $2/3V_{cc}$ and $1/3V_{cc}$ by a string of three resistors. The inputs to the two comparators are known as threshold and trigger.



(a) Timer Integrated Circuit



(b) Timer Astable

Figure 2.11: Timer Circuit and Astable Configuration

There are two outputs from the timer: a class B totem pole output and an open collector discharge output.

Now consider the astable oscillator circuit of figure 2.11(b). The timing capacitor C_T first charges towards V_{cc} through resistors R_{charge} and $R_{discharge}$. When the capacitor voltage reaches the upper comparator threshold, the comparator causes the flip-flop to change state, activating the discharge transistor. The timing capacitor then discharges toward ground through resistor $R_{discharge}$. When the capacitor voltage reaches the lower comparator threshold, this comparator causes the flip-flop to change back into the original state, deactivating the discharge transistor. The cycle then repeats.

The voltage across the timing capacitor assumes a sawtooth-like shape[4], oscillating between $1/3V_{cc}$ and $2/3V_{cc}$.

With a little work, one can show that the period of the cycle is given by

$$T = 0.693(R_c + 2R_d)C_T \tag{2.7}$$

The exact proportions of R_c affect the duty cycle of the output square wave, which is not important. We're only interested in the period, so we might simply make $2R_d = R_C$. Then

$$T = 1.38R_cC_T \tag{2.8}$$

Now let us calculate a suitable value for R_c, using equation 2.8 and the known values of 110pF for C_T and 277μsec for T.

$$R_c = \frac{T}{1.38C_T} \tag{2.9}$$
$$= 1.8M\Omega \tag{2.10}$$

We set $2R_d = R_c$ so $R_d = R_c/2 = 912K\Omega$.

To this point, we have assumed that no current flows into the threshold or trigger comparator inputs. Let us compare the charge and discharge currents with the leakage currents of these inputs.

During charging, the voltage across the charging capacitor varies from $1/3V_{cc}$ to $2/3V_{cc}$. The currents are then as tabulated below:

V_c	I_c
$1/3V_{cc}$	1.22μamps
$2/3V_{cc}$	0.615μamps

The input current for the ordinary (BJT version) 555 timer trigger input, is given as 0.9μamp maximum. This exceeds the charging current of the capacitor as it approaches its highest voltage, so the astable might not function.

On the other hand, there is a CMOS version of the 555 for which the trigger current is given as 10pA (10×10^{-12}amps). However, though significantly superior in a number of respects to the BJT version, the CMOS version is less commonly available. Murphy's law being what it is, it is a dead certainty that someone will attempt to substitute a BJT 555 timer in the circuit. Insidiously, some BJT 555's will probably work, leading to a lot of head scratching over the ones that don't function.

[4]The sides of the sawtooth are actually somewhat curved because the charge and discharge functions are exponential.

2.8.4 Engineering Proverb: Do it the easy way

With a few assumptions this back of the envellope analysis established that the oscillator circuit could be problematic because of the small timing currents. This is not to say that it cannot be done: there are low current timers and op-amps available. However, it is a suggestion that it might be worthwhile to attempt to outflank the problem rather than tackling it head-on.

In general, this analysis illustrates that the most obvious solution is not always the best. A good engineer can step back from a solution and become a critic, finding the flaws in the proposed design. A good engineer can also come up with several different design approaches. Hopefully, one of the designs poses less risk than the others, and becomes the final choice.

Engineering can be an adventure. If the wrong choice is made at the outset, and the wrong design chosen, it can take a long time and many resources to back out and start over again. Moreover, it may take some substantial exploration to identify which solution is the best choice.

At the preliminary design stage, identify several possible solutions and try to choose the one with the least risk of failure.

2.8.5 CMOS Oscillator and Divider

The next design springs from the realization that

- the RC oscillator needs to operate at high frequencies to minimize the oscillatior resistance, and therefore the problem of leakage currents.
- we could lengthen the effective period by dividing down the frequency (and hence multiplying up the period) in a cascade of flipflop dividers.

As is the often the case, such inspiration is precipitated by the availability of a suitable part. In this case the suitable part is a CMOS integrated circuit, type CD4060 14-Stage Ripple-Carry Binary Counter/Divider and Oscillator.

The 4060 consists of a CMOS oscillator section followed by a 14 stage binary divider chain. A high level on the reset input stops the oscillator and forces all the flip-flops into the zero state. All of this is available for a parts cost of about one dollar.

The oscillator period is given by

$$T \approx 2.3 R_T C_T \tag{2.11}$$

The frequency, of course, is the inverse of this:

$$f_{osc} \approx \frac{1}{2.3 R_T C_T} \tag{2.12}$$

In our case the oscillator capacitance C_T is the humidity sensor. The value of resistor R_S is not critical. It helps stabilize the oscillator frequency and is set to a value between $2R_T$ and $10R_T$. The data sheet indicates that a maximum value for these resistors is 1MΩ, so we might set R_S at 120KΩ and R_T at 56KΩ. Then substituting 110pF and 150pF for the minimum and maximum values of C_T in equation 2.11, we obtain maximum and minimum oscillator frequencies of 70.6KHz and 51.8KHz. These are well within the maximum frequency capability of the oscillator, 100KHz.

The minimum and maximum frequencies correspond to periods of 19.3μsec and 14.2μsec. We wish the difference between these two periods, 5.1μsec, to be multiplied up to at least 100μsec to obtain a resolution of 1 part in 100 by our microprocessor timer circuits. The minimum binary multiplication factor is therefore 32, or

2^5. To be on the safe side, we might choose to use the Q7 divider output, which gives a period multiplication of $2^7 = 128$. Then as the humidity varies from 0 to 100%, the period of the divider output waveform will vary from 2470μsec to 1818μsec, for a difference of 652μsec. The resolution will be 1 part in 652, or about 0.15%, much more than required.

Notice that we have to take care that the waveform period does not exceed $2^{16} - 1$ μseconds, since this is the maximum measuring period of the 16 bit microprocessor input capture registers.

Oscillator Tolerance

Now we come to an interesting conundrum. According to [46] (page 522),

> 'Our experience is that the internal oscillator…(of the 4060)…has poor frequency tolerance and (in some HC versions) may malfunction.'

Prompted by this warning, we examine the data sheet for the RCA version of the 4060 (the CD4060B) and find a minimum and maximum specification for oscillator frequency stability under certain test conditions. Good.

The data sheets for the Motorola version of the 4060 (the MC1460B) show the oscillator stability under 'Typical RC Oscillator Characteristics'. Oscillator performance is not guaranteed. As Bob Pease says in [47], 'Typical could mean that they once saw a device with those characteristics'. Not so good.

The data sheet for the National version, the CD4060BCN, does not show any specifications for oscillator performance. Uh-oh.

Obviously, there is a certain amount of risk associate with CMOS oscillators. This is also fueled by the knowledge that the logic threshold in CMOS, which we tend to visualize as around $1/2V_{cc}$ may and does vary from $1/3V_{cc}$ to $2/3V_{cc}$ between devices.

However, reference [48] indicates that the period of oscillation of a CMOS oscillator will vary as tabulated below:

Threshold, % of V_{DD}	Oscillator Period
33	2.25RC
50	2.14RC
66	2.22RC

This variation is in the order of 5%, much less than the humidity sensor tolerance. The system must be calibrated to allow for the 15% tolerance on the humidity sensor, so this tolerance on the oscillator frequency is quite acceptable.

Oscillator Temperature Stability

The stability of the oscillator with changes in temperature is much more of an issue than the original frequency tolerance. The tolerance can be calibrated out, but a variation with temperature presents more of a difficulty.

The Motorola data sheet shows a graph of oscillator frequency vs temperature in which the oscillator frequency changes by about 1% over a 25^{o}C range of temperature, equivalent to 400ppm/^{o}C. Reference [48] shows that oscillators of this type have a variation of 700ppm/^{o}C and about 1% per volt of supply change. All of these values are quite acceptable.

This is not an iron-clad case for oscillator stability and it certainly wouldn't satisfy a flinty-eyed mil-spec inspector, but it is enough reassurance to encourage us to try this design.

Sensor Temperature Stability

The humidity sensor responds to temperature at a rate of 0.1%/^{o}C. If the expected range of ambient temperature is such this represents an unacceptable error, it is necessary to temperature compensate the sensor. This circuit is not simple to compensate in hardware. However, if the microprocessor is aware of the temperature reading, it may compensate by adding a software correction to the reading of humidity.

2.8.6 Humidity Oscillator Circuit

The final humidity measurement circuit based on a variable frequency oscillator is shown in figure 2.12.

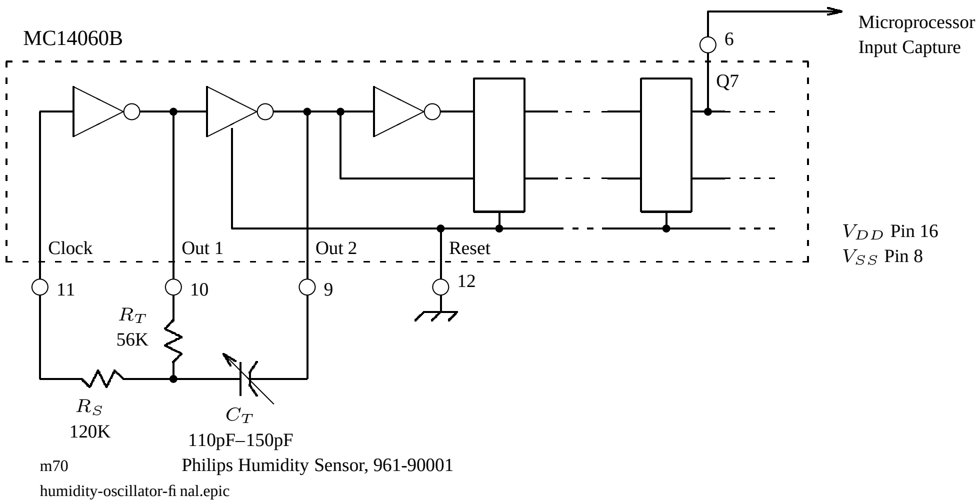


Figure 2.12: Humidity Measurement With Variable Frequency Oscillator

2.9 Calibrating the Hygrometer

When certain salts are dissolved in water to the saturation point, and the resultant solution placed in a closed container, the solution will maintain a predictable humidity in the atmosphere of the container[38, 45]. The humidity sensor may be lowered into the atmosphere of a large jar containing the saturated salt solution in the bottom (figure 2.13) to expose it to a known humidity.

The table of figure 2.14 shows suitable salts for the calibration, together with the calibration humidity at various temperatures.

If two different salt solutions are available, the humidity sensor may then be calibrated at two points.

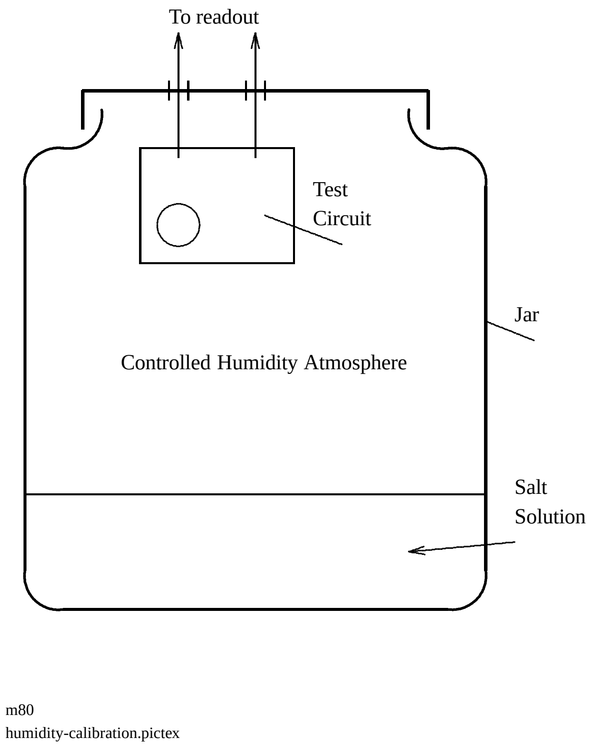


Figure 2.13: Hygrometer Calibration

Formulation	Name	15^{o}C	20^{o}C	25^{o}C
KNO_3	potassium nitrate	95	94	93
$NaCl$	sodium chloride (table salt)	75	75	75
$Mg(NO_3)_2.6H_2O$	magnesium nitrate	53	52	52
K_2CO_3	potassium carbonate	44	44	43
$MgCl_2.6H_2O$	magnesium chloride	33	33	33
$LiCl$	lithium chloride	13	12	11

Figure 2.14: Salts for Humidity Calibration

2.10 Humidity Measurement Software

In this section, we discuss the software for measuring the period of the humidity signal.

A general strategy for the measurement and display of humidity is as follows:

- As described in section 2.8.2 previously, the humidity sensor generates pulses at periods between 1818 and 2470 μsec, corresponding to relative humidity between 0% and 100%.
- The pulses are connected to Input Capture #1 of the 68HC11 microprocessor. This input is configured to generate an interrupt on each positive edge. With each interrupt, the 68HC11 16 bit free running timer, which is counting at a rate of 1 musec per count, is transferred to the input capture #1 latch. This may be regarded as timestamping the occurence of the edge.
- The interrupt service routine notes the time T1 of the first edge and the time T2 of the second edge and then finds $T2 - T1$ to determine the pulse period in μseconds.
- The display manager routine converts the pulse period into a humidity value and writes it to the display.

2.10.1 Humidity Task State Machine

The humidity task must cycle through a series of actions in an organized fashion. Notice that the pulse measurement task is called whenever a positive edge occurs in the input waveform, or hundreds of times per second. It's up to the measurement task (interrupt service routine) to perform the correct action (which might be to do nothing) each time it is called. A state machine organization organizes this nicely, with the following states (see figure 2.15):

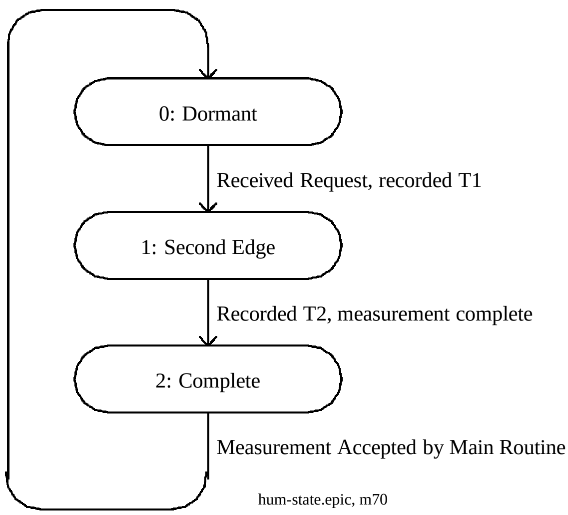


Figure 2.15: Humidity Task State Diagram

Dormant The humidity task has completed the last measurement cycle and handed over results to the display manager. If the Request Flag is cleared, there is no request to take a measurement, then the task simply exits.

However, if the Request Flag is set, the task has received a request for a new measurement. It therefore notes time $T1$, clears the request flag (since it is now honouring the request), changes to the Second Edge state and exits.

Second Edge When the processor enters the humidity task and the state is Second Edge, it records the time as $T2$. Then it calculates the pulse period $T = T2 - T1$, raises the Ready Flag to signal the display manager that a new humidity period measurement is ready and changes to the Complete state.

Complete When the display manager reads the value of the time interval $T = T2 - T1$, it clears the Ready Flag to indicate that it has accepted the reading. The next time an interrupt occurs, the humidity task is called into state 3, Complete. It then detects that the Ready Flag is cleared and so changes the state back to the Dormant state.

The humidity task and the main routine are essentially cooperating tasks that communicate with each other by the setting and clearing of software flags.

It's not immediately obvious that a state machine is necessary for organizing the interrupt service routine. However, imagine that the system is to be expanded to measure three pulse width input signals. Each interrupt service routine must then have some way of keeping track of the state of the measurement. A state machine is the right way to accomplish this.

The flow chart of the humidity pulse measurement task (interrupt service routine) is shown in figure 2.16.

2.10.2 Humidity Task Pseudo-Code

The humidity interrupt service routine is structured as a state dispatcher, that selects various state handlers, and the four state handler routines themselves.

```
H_STATE:   Byte  {Current state of humidity task}
H_REQUEST: Byte {Measurement request flag:      }
                {SET by display manager, CLEARED by humidity task}
H_READY:   Byte {Measurement ready flag}
                {SET by humidity task, CLEARED by display manager}
H_TIME1:   Word {First Edge Timestamp, microseconds}
H_TIME2:   Word {Second Edge Timestamp, microseconds}
H_PERIOD:  Word {Pulse width, microseconds}

Begin
Case H_STATE Of {Dispatch to the appropriate state handler routine}
     0: H_DORMANT     {Waiting for measurement request}
     1: H_SECOND_EDGE {Waiting for second edge}
     2: H_COMPLETE    {Completed measurement, waiting for display manager
                      to acknowledge}

{State 0 Handler Routine:}
 H_DORMANT: If H_REQUEST is SET
```

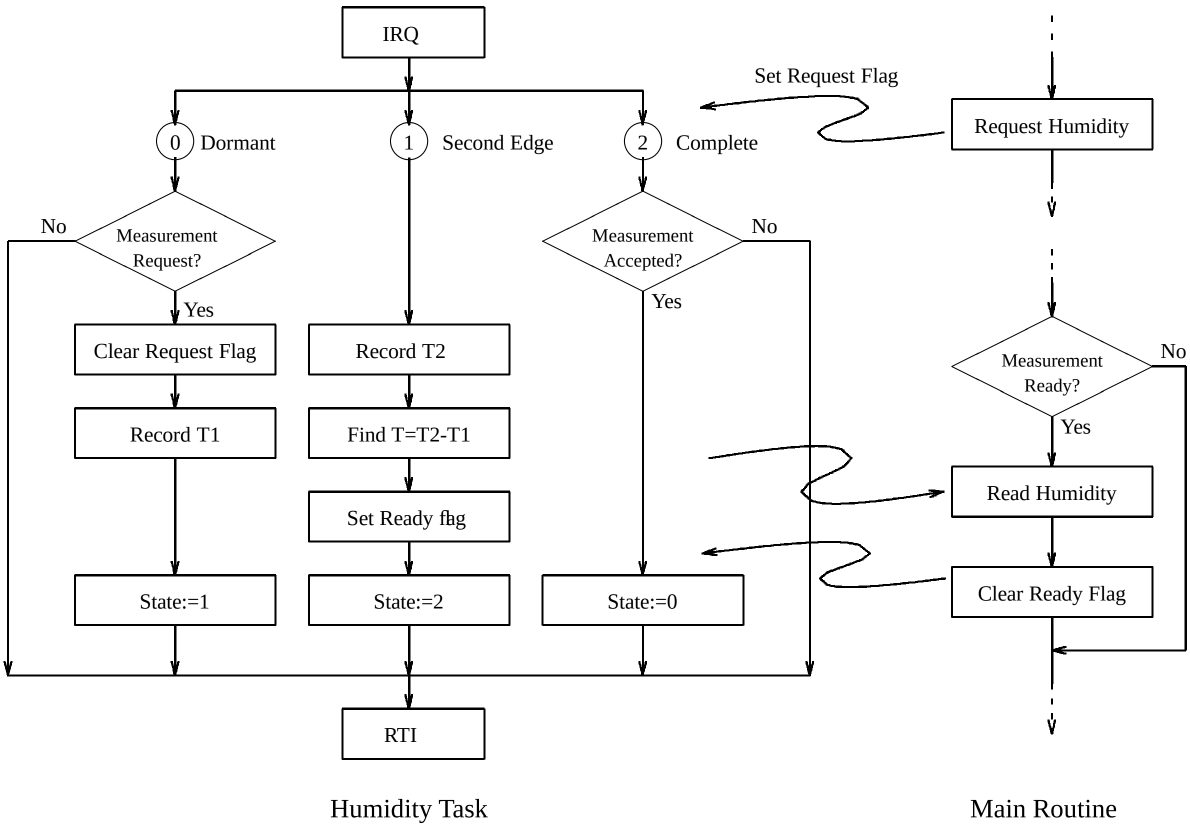


Figure 2.16: Humidity Task Flow Chart

```
                    Then
                     H_REQUEST := 0          {Acknowledge request}
                       H_TIME1   := TIMESTAMP1 {Save timestamp}
                       H_STATE := 1     {Move to next state}
                       Exit
                 Else
                       Exit

        {State 1 Handler Routine:}
        H_SECOND_EDGE:  H_TIME2 := FREE_RUNNING_COUNTER {Save Timestamp}
                        H_PERIOD:= H_TIME2-H_TIME1      {Calculate Period}
    H_STATE  := 2  {Move to next state}
                        Exit

        {State 2 Handler Routine}
```

```
    H_COMPLETE: If H_READY = CLEARED {If reading has been taken}
                  Then { by main routine       }
                       H_STATE := 0 { then move to state 0    }
                       Exit
                  Else
                       Exit

      Exit: Return from ISR
      End
```

2.10.3 Humidity Display Routine

The display manager

- requests a humidity reading by setting the `H_REQUEST` flag.
- watches the `H_READY` flag until it becomes set, indicating that new humidity data is available
- retrieves the new humidity value from the humidity measurement task
- maps the humidity frequency count to a value of relative humidity
- displays the humidity value on the LCD

Notice that the main routine may treat the humidity task as an independent machine. When it is requested to make a measurement, it will eventually complete a measurement and signal that it is ready. No interventions are required beyond setting the Request Flag (ie, making a request), copying the time value to a safe location, and clearing the Ready Flag (ie, acknowledging that a new value has been accepted).

In pseudocode:

```
   H_BUFFER: Word    {Storage for humidity frequency count}

Begin
   H_REQUEST := SET  {Request a new humidity reading}

   If H_READY = SET
      Then
          H_BUFFER := H_COUNT {Get the new reading}
  H_READY  := CLEAR   {Signal that the reading has been received}
          CALL H_CONVERT      {Convert frequency to humidity}
          CALL DISPLAY        {Display the new reading}
End
```

2.10.4 Converting Humidity Sensor Frequency to Relative Humidity

To relate measured period to humidity, we begin by solving for RH in equation 5.1:

$$H_r = 100\left(\frac{1}{0.34}\frac{C_s}{C_{12}} - 2.89\right)^{0.714} \tag{2.13}$$

where H_r is the relative humidity in percent
C_s is the sensor capacitance in farads
C_{12} is the sensor capacitance at 12% RH, 113 pF

The period of oscillation is given by equation 2.11. Solving for the timing capacitance C_T, we have

$$C_T = \frac{T}{2.3R_T} \tag{2.14}$$

where C_T is the timing capacitance in farads
R_T is the timing resistance in ohms
T is the period of oscillation in seconds

Substituting C_T for C_S in 2.13 above and we have

$$RH = 100\left(\frac{T}{0.782R_TC_{12}} - 2.89\right)^{0.714} \tag{2.15}$$

The measured period at the input to the microprocessor, which we will call T_m, is increased by the digital divider chain to 7 octaves above the oscillator period T. Then $T_m = 2^7T$. Substituting $T_m/2^7$ for T, 56×10^3 for R_T and 113×10^{-12} for C_{12}, we have the final relationship between relative humidity and period:

$$H_r = 100\left(\frac{T}{633 \times 10^{-6}} - 2.89\right)^{0.714} \tag{2.16}$$

2.10.5 Calibration Software for the Humidity Sensor

The reference capacitance C_{12} may vary by as much as $\pm15\%$ (section 2.8.1), and the oscillator frequency by an additional $\pm2.5\%$. Thus the sensor reading may vary by a total of $\pm17.5\%$ from unit to unit. We would like to minimize the hardware cost by compensating for this variation in the humidity software.

The tolerance may be incorporated into the humidity-frequency equation as a tolerance factor K_t which varies from by 1.0 ± 0.175.

$$RH = 100\left(\frac{1}{K_t\ 633 \times 10^{-6}f} - 2.89\right)^{0.714} \tag{2.17}$$

To cancel out the effect of the tolerance constant K_t, we multiply the measured frequency by an adjustment constant K_a, where $K_a = 1/K_t$.

$$RH = 100\left(\frac{1}{K_a\ K_t\ 633 \times 10^{-6}f} - 2.89\right)^{0.714} \tag{2.18}$$

To ensure that the adjustment has sufficient range to cancel out the tolerance, we shall assume K_a varies from 0.8 to 1.2.

To calibrate the system, put the humidity sensor in an atmosphere of known humidity and adjust the constant K_a until the display is showing the currect reading. Then install the adjustment constant K_a in the processor EEPROM.

When the processor is running through its floating point calculation of humidity, it will read the EEPROM-installed value of K_a and use it to calculate a calibrated humidity display.

2.11 Project Exploration: Resistance Humidity Sensor

Reference [43] describes a very simple hygrometer based on the use of a hygroscopic salt, lithium chloride. This salt absorbs moisture from the air and changes resistance in proportion to the moisture content.

A block diagram of the system and a sketch of the sensor arragement are shown in figure 2.17.

The resistance of the sensor is measured using an AC current to prevent the salt from disassociating. In the original circuit, the oscillator and difference amplifier are each one NPN transistor and the low pass filter simply a moving-coil current meter. The power supply is a 4.2 volt battery, unregulated.

Giannelli describes the construction of the sensor as follows:

> A $1/2$ inch wide strip of fiberglass cloth (about the same weight as a handkerchief) is mounted as shown (in figure 2.17(b)). After clamping both ends in the brass plates, the weave threads are removed from the cloth to improve response time. The completed sensor is dipped into a solution of lithium chloride (a mound of salt about the size of a dime dissolved in a tablespoon of water). After soaking the sensor, shake off the excess solution and allow the sensor to dry.

The Project

Redesign the circuit for operational amplifiers in the oscillator, difference amplifier, rectifier and low-pass filter sections. Some suggestions on the redesign:

Oscillator A Phase Shift, Wien Bridge or Twin Tee oscillator circuit would work satisfactorily in this circuit. Purity of the output waveform is not terribly important, but amplitude stability could be an issue. If the power supply is regulated, the amplitude of the oscillator can be limited simply by clipping in the operational amplifier.

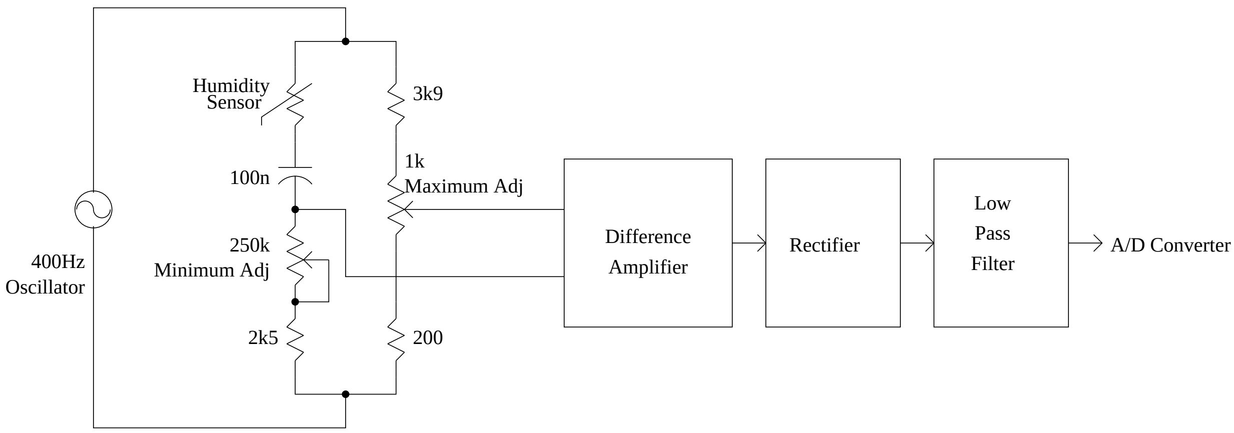
Difference Amplifier A single, four resistor difference amplifier is probably sufficient for this function. The input to the difference amplifier may be AC coupled, since the bridge error signal will be an AC voltage. The resistances should be chosen so that the diff amp does not significantly load the bridge circuit.

Rectifier A half-wave or full-wave precision op-amp rectifier, or a synchronous rectifier, could be used. A full wave rectifier simplifies the design of the low pass filter.

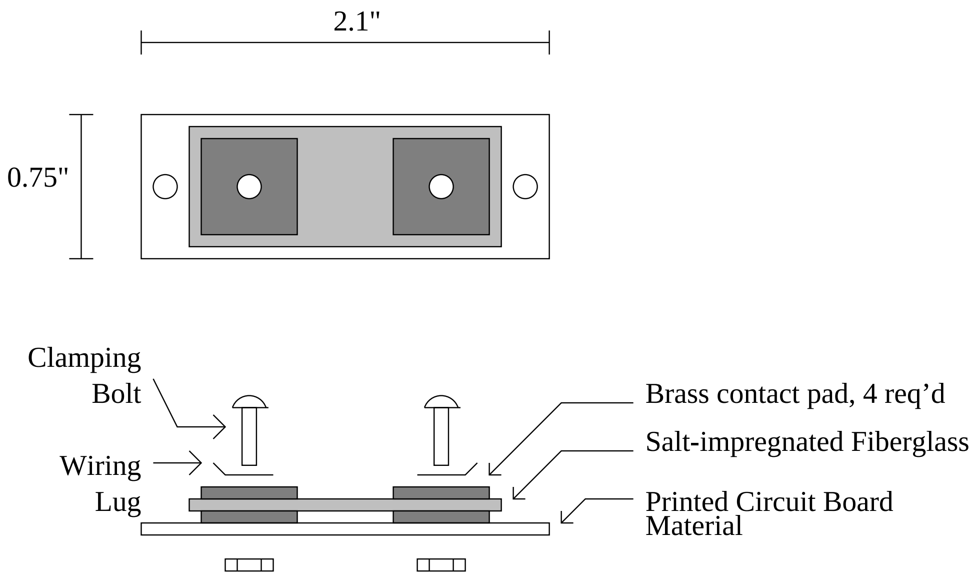
Low Pass Filter A simple RC low pass filter on an operational amplifier active filter (such as a single-pole Sallen-Key low pass) could be used here. Better filtering is obtained with the op-amp filter, and since op-amps are probably available, this would be a better choice. You should ensure that the 400Hz ripple is less than one step on the A-D converter.

Power Supply Ideally, the unit will operate from a single 5 volt regulated supply, stolen from them microprocessor logic. A negative voltage at currents in the order of one or two milliamps is available, or the existing 5 volt power supply could be split to provide a ± 2.5 volt supply.

Several useful building blocks are shown in the circuit diagram for a syncho based wind direction interface, figure 4.15 on page 159.



(a) Block Diagram



(b) Resistance-Humidity Sensor

Figure 2.17: Variable Resistance Humidity Sensor

2.12 Project Exploration: Monostable Interface

Application notes for the Philips capacitance humidity sensor [10] show the simple circuit of figure 2.18.

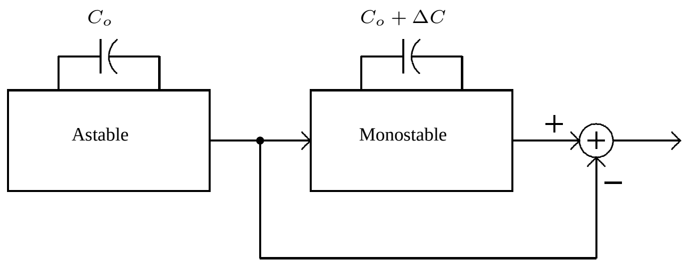


Figure 2.18: Humidity Sensor using Monostables

The astable oscillator is designed so that the period of the output waveform is proportional to a fixed capacitance C_o, chosen to be equal to the base capacitance of the humidity sensor. The monostable is synchronized to the astable so that it produces pulses at the same time. The pulse width of the monostable output is proportional to the capacitance of the humidity sensor, $C_o + \Delta C$.

The two pulses are subtracted so that the final output pulse width is proportional to the difference between the fixed pulse width and the variable pulse width. The width of this final difference pulse is then proportional to ΔC, which is in turn proportional to the relative humidity.

The pulse width could be measured directly by the timer circuitry of a microprocessor. Alternatively, since the pulse is of fixed and known amplitude, it may be averaged in a low pass filter to obtain a voltage proportional to the width, and hence the humidity. This voltage could then be measured in the microprocessor A-D converter.

2.12.1 Analysis of the Astable Multivibrator

The basic circuit of the astable multivibrator is shown in figure 2.19.

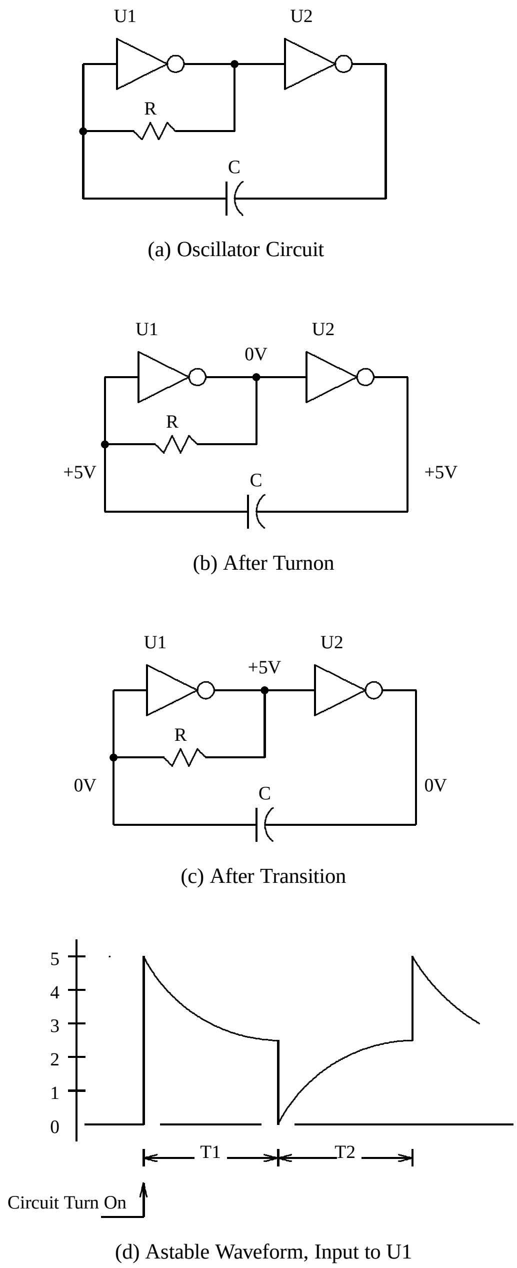


Figure 2.19: Astable Multivibrator Operation

After Turn On

To understand the operation of the astable, consider that it has just been turned on, and that the supply voltage is +5 volts. The output from U2 could be 0 or +5 volts: let's assume it's +5 volts. The capacitor C is uncharged, as yet, so the voltage across it is zero. Voltage cannot change instantaneously across a capacitor, so when the right end of the capacitor is lifted up to +5 volts, the left end must go to +5 volts as well. The complete situation, just after the power is applied, is as shown in figure 2.19(b).

During T1

These are CMOS gates, so no significant current flows into their inputs. However, the resistor R has a voltage across it, so current flows through it, gradually discharging the capacitor. The left end of the capacitor now moves in an exponential discharge toward ground.

Transition to T2

Eventually, at the end of a time which we'll call T1, the voltage at the input to U1 reaches the threshold voltage of the gate, about +2.5 volts or so. At that point, the state of U1 changes: its input is seen as below threshold (logic 0), so its output goes up to logic 1, +5 volts. This forces the output of U2 immediately down to zero volts. The astable has changed state, and the voltages are as shown in figure 2.19(c).

During T2

Now the whole process runs in reverse. Current flows from the output of U1, through the resistor, changing the capacitor. The input to gate U1 moves in an exponentially charging curve back up to the voltage threshold. When it reaches the threshold, the whole cycle begins anew.

Assuming that the threshold voltage is at half the supply voltage, it may be shown that the intervals T1 and T2 are equal, and the total period T of the waveform is approximately

$$T = 0.693RC \tag{2.19}$$

where R and C are the values of the timing resistance (ohms) and capacitance (farads).

2.12.2 Humidity Astable Schematic

The complete schematic of the Humidity Astable circuit is shown in figure 2.20.

Two CMOS quad 2-input NOR gate packages are used, type CD4001 or equivalent. U1A, U1C, and U1D are wired as inverters. U1C and U1D are the free-running oscillator, which produces a square wave at about 10KHz. U1A and U1B form a monostable, the pulse width of which is proportional to the humidity sensor capacitance. Gate U2A performs the subtraction operation, so that its output width is proportional to relative humidity.

The original reference suggests that C1 be composed of two 47pF and one 22pF capacitors, all with large positive temperature coefficients of capacitance. This will help to compensate for the temperature coefficient of the humidity sensor.

The peak value of the output pulse is equal to the +5 volt supply voltage, and the duty cycle approaches 25% at 100% relative humidity. The average value of the output pulse would then approach 1.33 volts, and an amplifier stage would be required to drive the A-D converter to full scale.

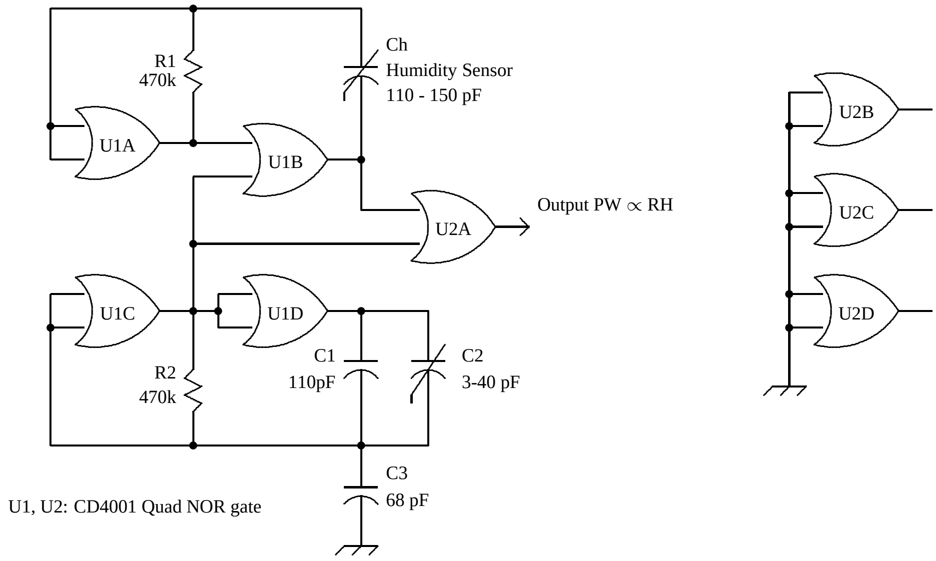


Figure 2.20: Humidity Sensor using Monostables, Schematic

Chapter 3

Wind Speed

> Who has seen the wind?
> Neither I nor you:
> But when the leaves hang trembling,
> The wind is passing through.
>
> Who has seen the wind?
> Neither you nor I:
> But when the leaves bow down their heads,
> The wind is passing by.
>
> Christina Rossetti
> [4]

3.1 Introduction

It is helpful to relate wind velocity to its effects using the Beaufort Scale [2] shown in figure 3.1.

Based on this scale, the wind speed measuring device, or anemometer, should have a range of zero to 120Km/hr or so in normal use. If the anemometer is to be capable of measuring record wind speeds, then the rather more demanding maximum speed requirements are as shown in figure 3.2, from reference [3].

The highest wind speeds on the face of the earth occur in tornadoes, which contain wind velocities in excess of 480Km/hr. Tornado wind velocities are measured by doppler radar or estimated from the extent of the destruction. Since tornadoes destroy buildings, it is probably unrealistic to expect our anemometer design to operate under tornado conditions .

Over the years, a great many devices have been used to sense wind velocity. We shall look at two measuring instruments, the cup-anemometer and cooling effect sensor.

3.2 Cup Anemometer

The cup anemometer usually consists of three conical or spherical cups mounted on a rotating shaft. The speed of rotation is proportional to wind speed. The sight of such an anemometer, whirling in the breeze, is synonymous with weather measurements.

Beaufort Number	Name	Kilometers per hour	Effect on Land
0	Calm	Less than 1	Calm: smoke rises vertically
1	Light Air	1–5	Weather vanes inactive: smoke drifts with air
2	Light Breeze	6–11	Weather vanes active: wind felt on face: leaves rustle
3	Gentle Breeze	12–19	Leaves and small twigs move: light flags extend
4	Moderate Breeze	20–28	Small branches sway: dust and loose paper blow about
5	Fresh Breeze	29–38	Small trees sway: waves break on inland waters
6	Strong Breeze	39–49	Large branches sway: umbrellas difficult to use
7	Moderate Gale	50–61	Whole trees sway: difficult to walk against wind
8	Fresh Gale	62–74	Twigs broken off trees: walking against wind very difficult
9	Strong Gale	75–88	Slight damage to buildings: shingles blown off roof
10	Whole Gale	89–102	Trees uprooted: considerable damage to buildings
11	Storm	103–117	Widespread damage: very rare occurance
12–17	Hurricane	more than 117	Violent destruction

Figure 3.1: Beaufort Wind Scale

Canadian Record:	201.1 Km/hr, Cape Hopes Advance (Quaqtaq) Quebec, 18 Nov 1931
World Record:	371 Km/hr, Mt. Washington New Hampshire, 12 April 1934

Figure 3.2: Highest Wind Speed for 1 Hour

An example design, constructed from found parts by Jim Koch, is shown in figure 3.3.

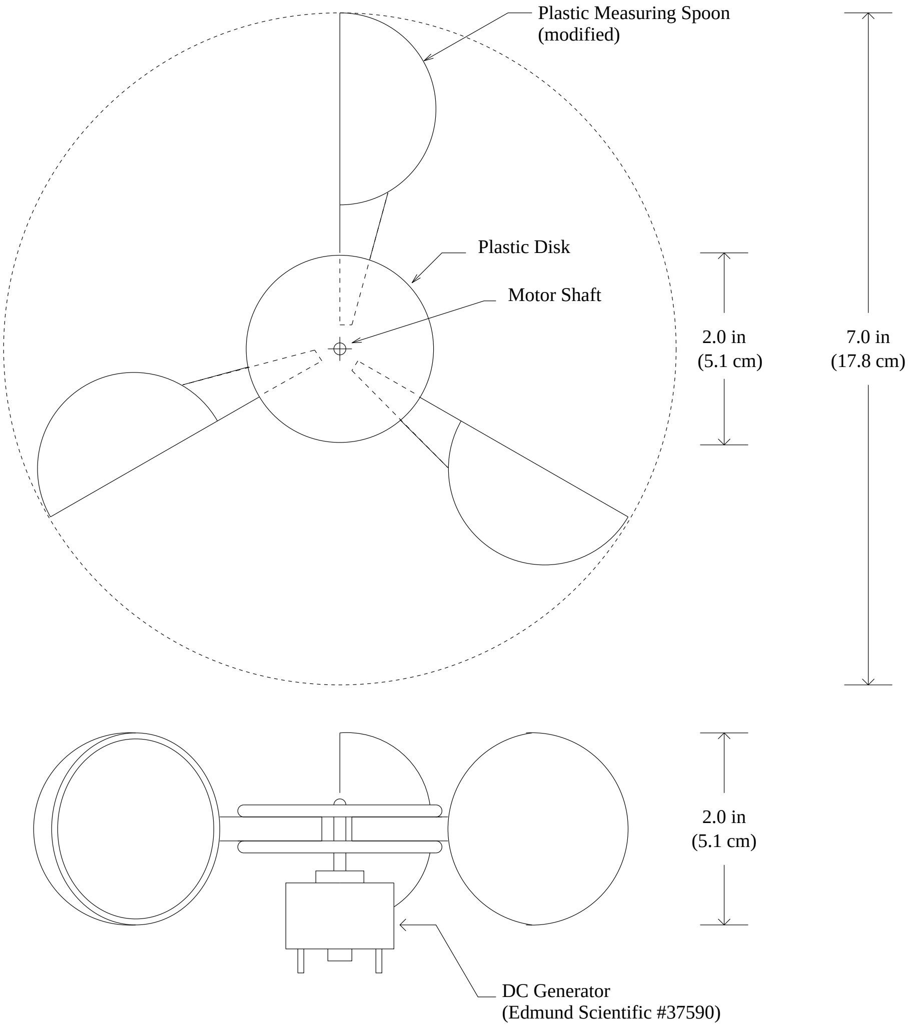


Figure 3.3: Rotating Cup Anemometer

Although almost any design of cup anemometer will work after a fashion, wind tunnel tests on cup anemometers have shown that an optimum design will include the following features[55, 54]:

- The rotating part of the anemometer should be as light as possible to start at low wind velocities and respond quickly to wind gusts. (In contrast to this, the British Casella anemometer type W/1254/2 the rotating part weighs in at over 1Kg, somewhat excessive in these days of lightweight plastics. On the other hand, in the tradition of British Robustness, it could probably survive a major hurricane.)

- The anemometer should have three cups. Two cups won't start rotation reliably and more than three simply add to the mass of the rotating part.

- The cups should ideally have a conical shape but spherical is also acceptable. The edges of the cups should preferably be beaded (rounded) to reduce turbulence.

- For the most linear response of rotation speed with wind velocity, the cup diameter should be approximately half the total wheel radius. In other words, the entire spoke should be cup, and the stalk length should be close to zero. In the design shown in figure 3.3, the ratio of cup diameter to wheel diameter is $2/7 = 0.285$, below optimum, so we would expect a non-linear function of rotational speed versus wind velocity.

A lightweight anemometer with these design features may be expected to have a threshold wind speed of 0.8Km/hr (0.5mph) and respond to a step in wind speed with a first order time constant of 0.4 seconds.

Cup Anemometer Output Signal

In meterological applications, it has been customary to gear down the rotation of the anemometer and count the number of turns over an interval of time. Some anemometers are equipped with a cam-operated switch which closes briefly each time the anemometer completes a certain number of revolutions. Others include an odometer-type turns counter[1].

Counting the number of turns of the anemometer disc is the mechanical equivalent of integration, in which case the readout is a measure of the average distance moved by the air in the time interval. This is useful to meterologists, who are interested in the movement of weather systems. However, if one is interested in the instantaneous velocity, as we are, some other readout method must be used.

Anemometer Distance Factor

If the circumference and output scale factor of the cup anemometer are known, then the distance the air has moved in a give period of time may be calculated.

The key coefficient is the anemometerDistance Factor, which is the ratio of the circumference of the anemometer wheel to the distance moved in one revolution of the anemometer. The distance factor may be thought of as the slippage of the anemometer cups through the air which is required to generate torque in the anemometer mechanism. For an ideal anemometer, the distance factor would be 1.0.

For example, suppose an anemometer has a diameter of 14 cm and rotates at 13 RPM for every Km/r of wind speed. What is its distance factor?

[1] Negretti and Zambra, manufacturers of one such instrument, suggested that the anemometer could be mounted on a mast and the turns counter read with binoculars. This arrangement might be rather inconvenient during a force 10 gale, the very situation in which one is most interested in wind velocity.

The circumference is

$$\begin{aligned} S &= \pi D \\ &= \pi \times 14 \text{ cm} \\ &= 43.9 \text{ cm} \\ &= 0.439 \text{ metres} \end{aligned}$$

The velocity constant for the anemometer is 13RPM per Km/hr. Then, in one minute, a point on the circumference will move

$$\begin{aligned} S' &= 0.439 \times 13 \\ &= 5.7 \text{ metres} \end{aligned}$$

In one minute, at 1 Km/hr, the air moves

$$\begin{aligned} S'' &= 1000 \text{ metres}/60 minutes \\ &= 16.67 \text{ metres} \end{aligned}$$

The distance factor F_d is the ratio of S" to S', or

$$\begin{aligned} F_d &= frac16.675.7 \\ &\approx 3 \text{ metres/metre} \end{aligned}$$

Thus the wind moves three metres for every metre of movement of the anemometer perifery.

Tachogenerator Output

A DC generator (or DC motor driven as a generator) generates a terminal voltage proportional to its rotational speed, so it is useful in generating an electrical voltage proportional to wind speed. The electrical circuit is simplicity in itself: the tachogenerator (tachometer-generator) output is connected to a voltmeter and the dial reading of the voltmeter redrawn to read wind speed.

This works best with a moving-coil movement meter rather than a digital voltmeter: the inertia of the meter movement averages out the ripple and commutator noise of the generator.

When the generator voltage is to be read by an A-D converter as part of a digital voltmeter or microprocessor, an RC low pass filter will be required between the generator and A-D converter. The cutoff frequency must be chosen so that the is effective at the lowest ripple frequency whithout materially affecting the average A-D reading. A low frequency cutoff corresponds to a large time constant in the step response, and so an effective low pass filter may slow the transient response unacceptably.

For example, if the step response of the anemometer to a wind gust is to have a time constant of 0.4 seconds, then the lowpass filter should have a time constant a small fraction of this, or less than 40 milliseconds. This may time constant not be effective in blocking noise from the generator.

Pulse Frequency Output

A tachogenerator has the virtue of simplicity if the output of the anemometer is to be displayed on a moving coil meter. However, tachogenerators have brushes that become progressively more noisy and eventually wear out. The tachogenerator signal voltage also requires analogue-digital conversion if the ultimate destination is a the digital domain.

When a microprocessor or other digital system is to process the tachometer output, it is convenient to convert the rotation of the anemometer into a variable frequency pulse train. The anemometer is directly coupled to an optical encoder, a transparent disc with alternating opaque and transparent areas. The disk is positioned between an LED and a phototransistor. As it rotates, the opaque stripes on the optical encoder interrupt the light beam, causing a pulse waveform to be generated at the output of the phototransistor (figure 3.4).

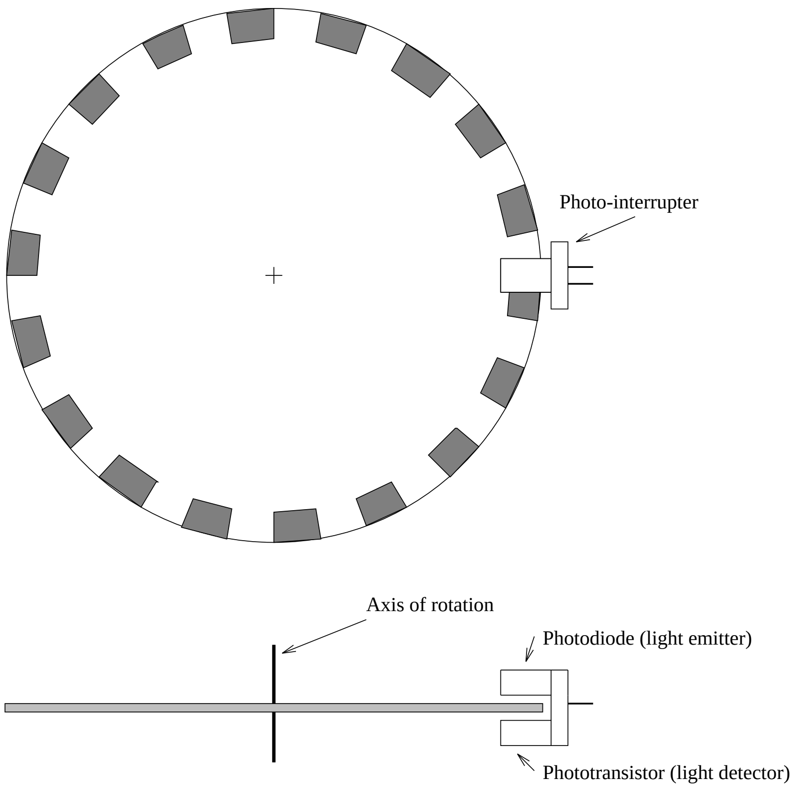


Figure 3.4: Optical Encoder Disc

The output pulse frequency is directly proportional to the rotational speed of the anemometer and the number of stripes on the disk.

From this point, the usual methods of pulse frequency measurement by microprocessor may be applied to determine the pulse frequency. For example, the anemometer pulses may be used to generate a processor interrupt. The interrupt service routine increments an anemometer count variable and, once per second, the main routine reads the number of accumulated pulses to determine the current anemometer frequency and wind speed.

Accumulating the total count of anemometer pulses over a longer period of time may be used to determine the distance of movement of the air mass.

There is nothing particularly special about the value of the number of stripes on the encoder wheel, and the number may be reduced for simpler construction. It is not a good idea to base the measurement of wind speed on the period of the anemometer pulse train, since the anemometer will completely stop in calm air and the pulse period will go to infinity.

3.3 Anemometer using the Cooling Effect

We know from our earliest experience that hot food can be cooled off by blowing on it, that is, heat is removed more quickly from a substance when the air in contact with the substance is in motion. This is the cooling effect, and it is some function of the air speed. A measurement of the cooling effect is then an indirect measurement of wind speed.

There are two methods of measuring the cooling effect:

Constant Power Input A temperature sensor is heated with constant power input and the temperature measured. If the air speed increases, the temperature of the sensor decreases.

Constant Temperature A temperature sensor is held at a fixed temperature above ambient by a feedback system. As wind speed increases, the heat loss from the temperature sensor increases. As a result, the power input to the sensor must increase to hold the sensor at the setpoint temperature. A measure of the power input is thereby an indicator of the wind speed.

A cooling effect sensor has no moving parts and may therefore be expected to be more reliable than the rotating-cup anemometer. On the other hand, it turns out that the cooling effect is a non-linear function of wind speed. The non-linearity makes the cooling effect sensor very sensitive to air movement at low air velocities and less sensitive as the wind speed increases.

As well, the thermal inertia of the temperature sensor tends to slow the response of a cooling effect sensor, so the sensor must have low thermal mass.

3.4 Thermal Circuits, A Review

Before launching into the details of the cooling effect sensor, we review of thermal equivalent circuit concepts. Using this approach, we can model thermal behaviour with electric circuits familiar to electrical engineers.

Fourier's Law of heat transfer states that the flow of heat through a heat conducting substance, due to thermal conduction, is proportional to the temperature gradient across the conductor.

This may be stated as

$$P = k\Delta T$$

where

P	is the power (rate of flow of energy) in watts
Δ T	is the temperature gradient (temperature difference) between the ends of the conductor, degrees C
k	is the constant of proportionality, known as the thermal conductivity, watts per degree C

For our purposes, it is convenient to rewrite this equation as

$$P = \frac{1}{\theta}\Delta T \tag{3.1}$$

where θ is the inverse of k, and is called the thermal resistance, degrees C per watt. This form of the equation resembles Ohm's law, where voltage is similar to temperature gradient, power is similar to electrical current, and thermal resistance is similar to electrical resistance. For an electrical engineer, this helps to provide an intuitive understanding of thermal behaviour. For example, if the thermal resistance is increased for a given power disappation, the temperature gradient will increase.

Like voltage, the temperature gradient is measured between two points. In many cases, the reference temperature (similar to voltage ground) is the ambient temperature, T_A.

Example

A certain power transistor is disappating 15 watts and has a thermal resistance to ambient of 3.2^{o}C/watt. Calculate the temperature rise above ambient of the junction.

Solution

The equivalent circuit is shown in figure 3.5.

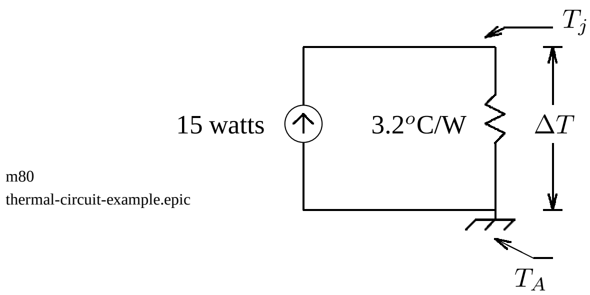


Figure 3.5: Thermal Circuit Example

Rewriting equation 3.1 to solve for the temperature gradient, we have

$$\begin{aligned}\Delta T &= P \times \theta \\ &= 15 \times 3.2 \\ &= 48^o\text{C above ambient}\end{aligned}$$

For example, if the ambient temperature is 40^{o}C, then the transistor junction is at $40 + 48 = 88^o$C.

3.5 Thermal Model of the Cooling Effect

In 1914, Louis Vessot King of McGill University showed that the heat loss from a heated wire in moving air is of the form

$$P = (A + B\sqrt{U})\Delta T \text{ watts} \tag{3.2}$$

where

A and $B\sqrt{u}$ are dissapation constants, watts/°C
U is the air velocity (we use U rather than V to avoid confusion with Voltage
ΔT is the temperature above ambient
This suggests that the thermal equivalent circuit consists of two parallel thermal resistances as shown in figure 3.6, where $\theta_S = 1/A$ and $\theta_D = 1/B$.

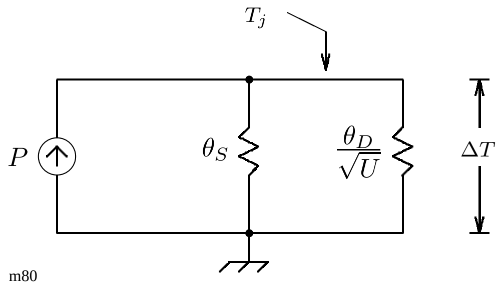


Figure 3.6: Thermal Equivalent Circuit, Cooling Effect

One of these resistances is the previously identified thermal resistance θ that has been renamed θ_S, to indicate that this is the static thermal resistance, independent of wind velocity. The other thermal resistance is a constant θ_D (for dynamic), divided by the square root of wind velocity U. (The quantity $\theta_D/\sqrt{U}$ has the units degrees-C per watt, so θ_S has different units than θ_D).

If the wind velocity U is zero, the dynamic branch resistance goes to infinity and the current through that branch is zero. As the wind velocity increases, the dynamic branch resistance decreases and provides more of a path for the thermal power, decreasing the temperature differential.

Notice that the wind factor is the square root of the wind velocity. The effect is to make a small wind velocity very noticeable, with decreasing effectiveness at higher velocities.

At very small wind velocity, the dynamic effect predominates, and the thermal resistance may be approximated by

$$\theta_T \approx \frac{\theta_D}{\sqrt{U}} \tag{3.3}$$

(3.4)

The explanation is given in an appendix, page 139.

3.6 A Constant Power Anemometer

The conceptual circuit diagram of a constant power anemometer is shown in figure 3.7.

In this circuit, the feedback action via the operation amplifier is such as to force the two transistors to have equal collector-emitter voltages and equal collector currents, thereby forcing them to dissapate the same power.

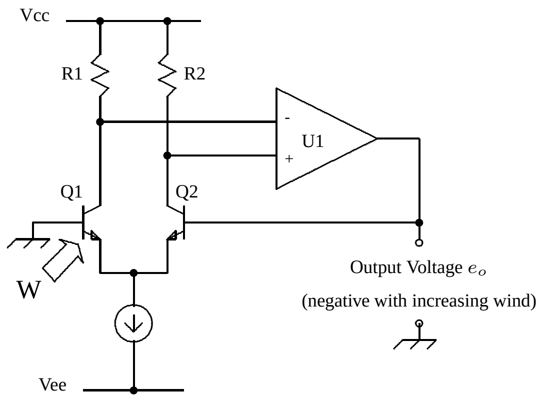


Figure 3.7: Constant Power Anemometer Concept

In calm air, the thermal resistances of the two transistors are equal, forcing their junction temperatures to be equal as well.

Now suppose that Q1 is cooled by moving air. Then its base-emitter voltage will tend to increase, causing it to conduct less base current, causing it to conduct less collector current. A lower collector current I_{c1} results in a higher voltage at the inverting input to the op-amp. The op-amp responds by lowering the base voltage of Q2, lowering the collector current of Q2 until it is again equal to the collector current in Q1.

Putting it another way: the negative feedback action of an op amp forces the inverting and non-inverting terminals to equal voltages. In this case, the negative feedback action forces the two collector resistors R1 and R2 to have the same voltage across them and consequently the same current through them. This ensures that the collector currents and voltages of the two transistors are forced to be equal, and therefore that they dissapate equal power. The feedback signal e_o is a measure of the cooling effect on Q1, and therefore the wind velocity.

A very simple circuit that demonstrates this concept is shown in figure 3.8.

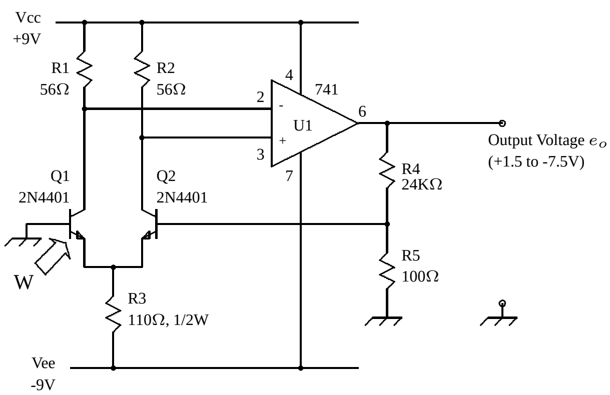


Figure 3.8: Constant Power Anemometer, Basic Circuit

In the original circuit, the output voltage of the op-amp is tied directly to the base of a transistor, and so the output voltage will change only a few millivolts as transistor Q1 is cooled by moving air. In the practical circuit of figure 3.8, a voltage attenuator between the output of the op-amp and the base of Q2 to forces the output signal e_o to change by larger amount to generate the same feedback signal. If the gain of the attenuator is $1/K$, then the change in output is multiplied by the factor K. For figure 3.8,

$$
\begin{aligned}
K &= \frac{R5}{R4 + R5} \\
&\approx \frac{100}{24000} \\
\text{so} & \\
1/K &= 240
\end{aligned}
$$

The change in the base-emitter voltage, which is proportional to the cooling effect, is multiplied by a factor of 240 as the output voltage.

As shown, the circuit of figure 3.8 has several limitations:

- The power dissapated in R3 is rather wasteful. A transistor constant current sink would be more efficient and provide better common mode rejection in the bargain.
- The circuit requires two power supplies. A single supply would be more convenient.
- The output voltage runs from 0 volts to -7 volts. For input to a microprocessor A-D converter, a range between 0 and +5 volts is preferred.

The circuit of figure 3.9 incorporates these refinements. The availability of a low cost, quad operational amplifier, the LM324 makes the design much easier. The cost of an op-amp, which used to be in the tens of dollars, is now below $0.20 each, which means they can be used with wild abandon in the circuit design.

The Complete Circuit

Amplifier U1A and Q3 establish the voltage of VR1 across R3, thereby setting the constant current to 75mA, as in the basic circuit. In this case, however, the voltage across Q3 and R3 is about 3.4 volts, so the power loss in the constant current sink is about a third of the previous circuit in the 110Ω resistor. The collector impedance of Q3 is also much higher than 110Ω, which will improve the common mode rejection ratio of the circuit.

An 'artifical ground' is created at the junction of R6 and R7, allowing a single supply to be used. The ground node of the previous circuit is then moved to the output of op-amp U1D, which buffers the voltage divider to supply a lower source impedance for this 'ground' to the ensuing circuit. For this to work correctly, of course, the currents into and out of the new ground node must not exceed the capabilities of the operational amplifier, a few milliamps.

The voltage across R5 is translated to a ground referenced signal by the differential amplifier U1C and its associated resistors. Notice that the feedback gain of this circuit is reduced to 120, to keep the output signal less than 5 volts.

Design Review

Now that we have a design for the circuit, it is important to check that static (DC) requirements are met. For example, we must ensure that the transistors are not cutoff or saturated, and that the DC conditions into the op-amps allow them to work correctly.

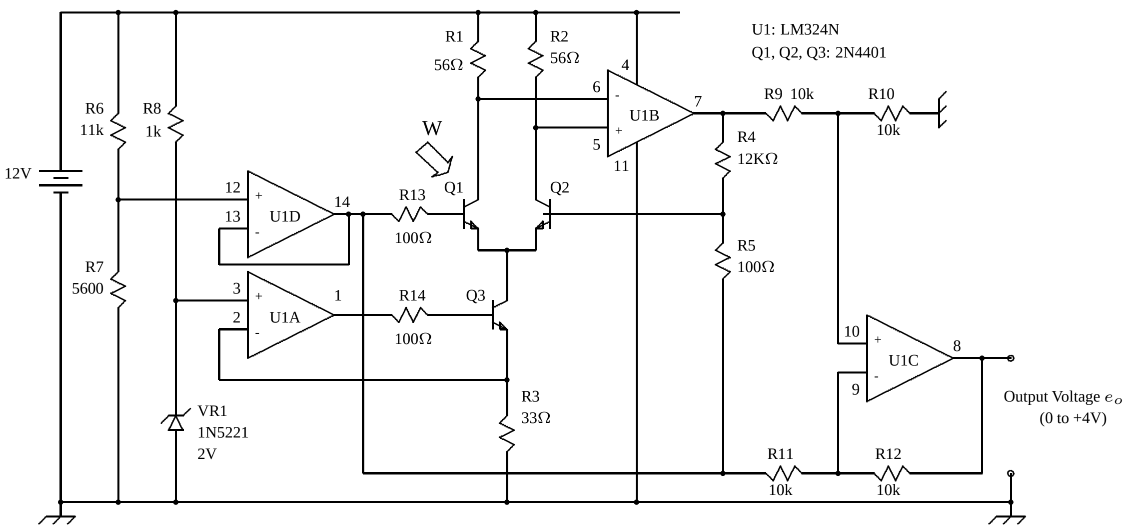


Figure 3.9: Constant Power Anemometer Circuit, Final Version

Starting with the information provided in the text, you should be able to determine the voltages and currents in the circuit of figure 3.9. As it says on the memo paper of a colleague: ”I've got a good idea: you do this instead of me.” Work through the following list of questions about the circuit.

1. Assuming that the current out of the R6-R7 voltage divider tap is negligible compared to the divider current (an assumption that we'll confirm in a moment) what is the voltage at the base of transistor Q1?

2. What is the voltage across R3? What is the tail current of the differential amplifier Q1-Q2?

3. The voltage across transistor Q3 must be sufficient to ensure that it is not saturated. Calculate V_{ce3}.

4. A reasonable minimum value of β for the 2N4401 is 50. Estimate the base current of Q1. Does this confirm our original assumption about the voltage divider?

5. Assuming that there is no cooling effect in Q1, calculate the collector voltage for Q1 and Q2.

6. Determine the power dissapation in Q1 and Q2.

7. The thermal resistance of the 2N4401 is given on the data sheet as 357 ^{o}C/watt. Calculate the junction temperature, degrees centigrade above ambient, of Q1 and Q2. The data sheet maximum is 135^{o}C. Is this likely to be exceeded in warm ambient conditions?

8. The output differential amplifier should not load the source that is driving it. By eyeballing the circuit, without calculations, make an argument that this is in fact the case.

9. Resistors R9, R10, R11 and R12 are not critical in absolute value, but should match each other as closely as possible. Why? What would be the effect if they are not matched?

10. If the op amp common mode input voltage (the average voltage at the two inputs) exceeds its allowable range, the op amp will cease to work correctly. The common mode input voltage range of the LM324 op amp is given on its data sheet as 0 to $V_{cc} - 1.5$ volts. Does this circuit meet that requirement? What does this requirement tell you about the range of values for resistors R1,R2?
11. The output voltage range of the LM324 is 0 to $V_{cc} - 2.0$. Is this a problem for any of the op amps in this circuit?
12. Estimate the total power consumption of the circuit, in milliwatts.
13. The LM324 op amp was chosen for this circuit because its output voltage can go to zero volts. The output of the 741 op-amp, in contrast, is limited to moving with 1.5 volts of either supply rail. In which of the three op-amp positions of figure 3.9 would a 741 not operate correctly?
14. An interesting way to view this circuit: Transistors Q1 and Q2 form the differential input to an operational amplifier, configured as an inverting amplifier, in which R4 is the feedback resistor and R5 the input resistor. The offset voltage of this new op-amp is enhanced by differentially heating the input transistors (actually, heating them both and cooling one of them). The offset voltage is amplified, as always, by the gain of the op-amp circuit.Redraw the circuit, including as many components as possible in this new operational amplifier.

3.7 Constant Temperature Anemometer

The constant power anemometer relies on the changing temperature of the wind sensor. Heating and cooling the sensor takes time, and the time constant of the previous circuit, with the TO92 (plastic) case 2N4401 transistor as a sensor, is about 18 seconds. This is far short of the ideal, which is about 0.4 seconds for a lightweit whirling cup anemometer. As a consequence, the constant temperature sensor will not follow rapidly varying winds, such as gusts.

As an attempt to speed up the reaction time of the sensor, let us turn to a constant temperature scheme. Here, the sensor is maintained at a constant temperature above ambient, and the power required to do this is some function of the cooling effect and the wind speed. Because the temperature sensor does not have to change temperature, thermal inertia should be less and the instrument should respond more quickly.

As in the case of the constant temperature circuit, we will use the base-emitter voltage to signal the junction temperature of the device. The basic scheme is shown in figure 3.10.

Both transistors are operated at the same emitter current, enforced by the current sources I_E. Transistor Q2 is heated to a reference temperature above ambient by the fixed collector supply V_{ce2}. The operational amplifier then compares the emitter voltages of the two transistors and drives the the collector of Q1 to a potential where the two emitter voltages equalize. For example, if Q1 is cooler than Q2, its base emitter voltage will be larger. The inverting terminal of U1 will then be more negative than the non-inverting terminal, and the op-amp output voltage will rise until the two inputs are equal again.

The collector voltage of Q1 is now a measure of wind speed. As wind cools Q1, its collector voltage rises to pump more power into Q1 to maintain its junction at the same temperature as Q2. At zero wind speed, the two collector voltages are equal and then, as the wind speed rises, the voltage between the two collectors will increase.

A voltmeter between the collectors of Q1 and Q2 could be used directly to read out wind speed. However, it is more useful to convert the floating differential signal at the collectors of Q1 and Q2 to a single-ended, ground referenced signal which can be fed into an A-D converter of a microprocessor. As in the constant-power anemometer, this may be accomplished with a simple differential amplifier.

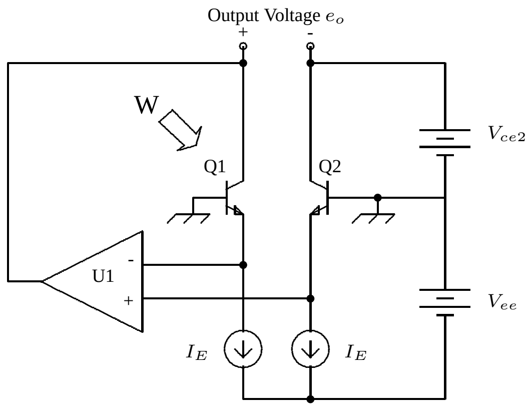


Figure 3.10: Constant Temperature Anemometer Concept

A practical circuit is shown in figure 3.11.

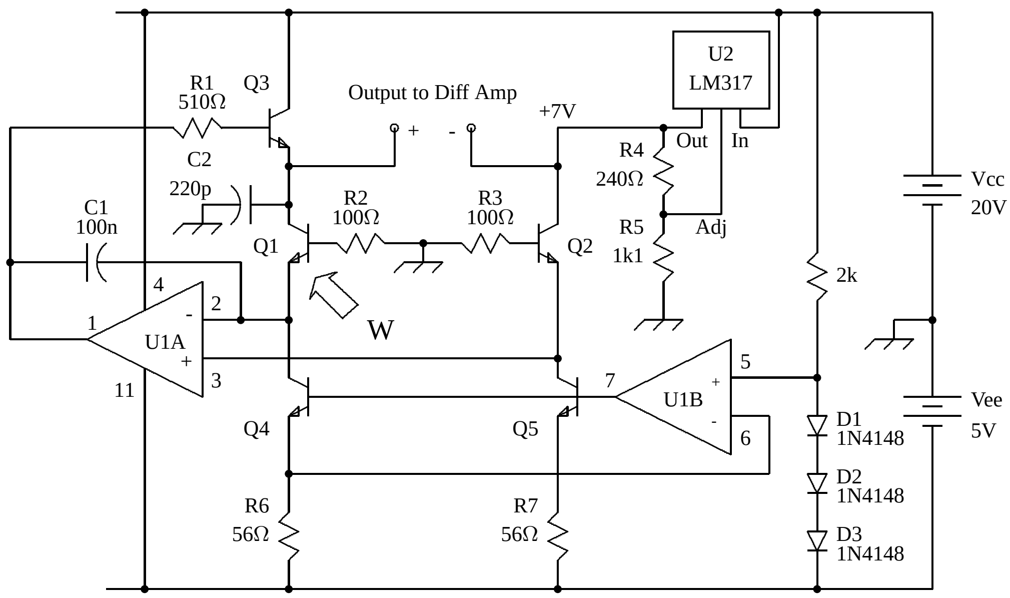


Figure 3.11: Constant Temperature Anemometer Circuit

In the practical circuit, the output current of the op-amp U1A is not sufficient to heat transistor Q1 so its output current is augmented by the emitter follower Q3.

All feedback loops have the potential to oscillate, and this one is especially enthusiastic: a high frequency waveform persisted at the output of U1A until C1, C2, R1, R2 and R3 were added to the circuit. Emitter followers have a strong tendency to radio frequency oscillation, and a common fix is a small resistor in the base lead, which is the function of R1, R2 and R3 in the circuit.

The constant voltage for the collector of Q2 is supplied by a three-terminal regulator IC, U2. An op amp voltage regulator could have been used, but the LM317 has much better regulation than a simple op-amp circuit, and can dissapate more power to boot.

Op-amp U1B and transistors Q5 and Q5 form the constant current sinks for Q1 and Q2 respectively. For their currents to be equal, Q4 and Q5 must be similar and at the same temperature.

Preliminary tests indicated that the differential output voltage could go as high as 7 volts, With a fixed voltage across Q2 of 7 volts and a 2 volt margin at the output of op-amp U1A, this requires that Vcc be at least 18 volts, so 20 volts is specified here.

3.8 Comparing Thermal Anemometers: A Morality Tale

Once upon a time, I was commissioned to develop a microwave link between a balloon borne video camera and a ground station, a distance of some 300 metres or so. Lashing up a couple of gunnplexer transmitter-receivers [68, 69] in the lab was immediately successful, and in an hour or so, I could transmit video pictures across the lab. Extrapolating from this, I thought that the communications link would be completely straightforward.

Over a longer range, however, the signal-noise ratio was terrible. After much experimentation and head scratching, I finally did a proper signal budget calculation and discovered that I needed a low noise preamp circuit and a high gain parabolic antenna to make the system work. The parabolic antenna had lots of gain but had to be aimed precisely at the balloon and so, in the end, I had to abandon the microwave system for a diffent approach altogether.

If I had done a systematic engineering analysis at the start of the project, I would have saved weeks of work. It was an important lesson.

Now let us consider the circuts we described in sections 3.6 and 3.7. Both seem to be credible designs. If breadboarded in the lab, both circuits seem to work correctly: the output voltage changes with wind speed.

However, it turns out that one of these circuits is much better than the other, and this is revealed by a careful analysis of the workings of the circuits.

This highlights why it is critical that the performance of the circuit can be defined mathematically. A systematic analysis will reveal critical factors that may not be evident from a visual inspection or even the operation of a prototype. In effect, the engineer must be creative at first, and then analytical to properly critique the design. This can be a challenge. Psychologically, we don't want to believe that our beautiful circuits have problems. However, if we don't find them, nature and Murphy's law will.

On to the analysis:

Analysis of the Constant Power Anemometer

The basic circuit of the Constant Power Anemometer is repeated here for convenience:

As we've remarked before, the collector current of a junction transistor is given by

$$I_c = I_s e^{\frac{V_{be}}{V_T}} \tag{3.5}$$

where

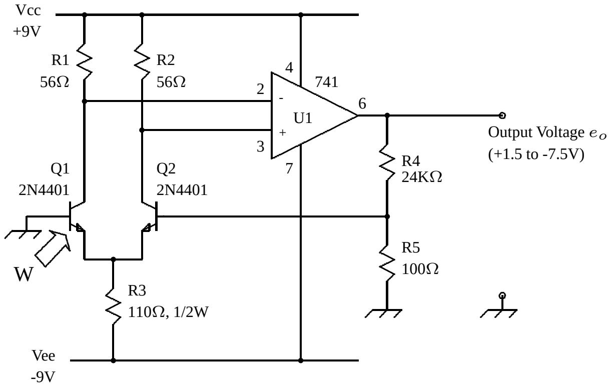


Figure 3.12: Constant Power Anemometer, Basic Circuit

I_s	is the saturation current
V_{be}	is the base-emitter voltage
V_T	is the thermal voltage, a temperature-dependent voltage about 26 millivolts at room temperature

For figure 3.12 we have

$$
\begin{aligned}
I_{c1} &= I_{s1} e^{\frac{V_{be1}}{V_{T1}}} \\
I_{c2} &= I_{s2} e^{\frac{V_{be2}}{V_{T2}}}
\end{aligned}
$$

Feedback action in this circuit keeps the two collector currents equal so

$$
I_{s1} e^{\frac{V_{be1}}{V_{T1}}} = I_{s2} e^{\frac{V_{be2}}{V_{T2}}}
$$

For matched transistors, the saturation currents are equal and cancel. Taking the natural log of both sides, we have

$$
\frac{V_{be1}}{V_{be2}} = \frac{V_{T1}}{V_{T2}}
$$

Now,

$$
V_T = \frac{kT}{q_e}
$$

where
k is Boltzmann's constant
T is the absolute temperature
q_e is the charge on the electron
Substituting for V_T in 3.6, we have

$$\frac{V_{be1}}{V_{be2}} = \frac{\frac{\eta kT1}{q_e}}{\frac{\eta kT2}{q_e}} = \frac{T1}{T2}$$

In words, the ratio of base-emitter voltages is the same as the ratio of the junction temperatures of the two transistors.

By KVL around the base-emitter loop of Q1 and Q2:

$$-V_{be1} + V_{be2} - \frac{e_o}{K_g} = 0$$

Substitute for V_{be1} from 3.6 and

$$-V_{be2}\frac{T2}{T1} + V_{be1} - \frac{e_o}{K_g} = 0$$

Solving for the output voltage e_o we have

$$e_o = K_g V_{be2}\left(\frac{T2 - T1}{T2}\right)$$

Not good. Recall that transistor T1 is cooled by moving air, transistor T2 is the 'still air' reference transistor, which will be at ambient temperature.

Ideally, the output should be independent of the ambient temperature, but the equation indicates that the output voltage is dependent on the base-emitter voltage V_{be2} of the reference transistor and the junction temperature of the reference transistor. Unless these factors can be made to cancel out, the temperature of the reference transistor Q2 will have to be stabilized, which adds to the circuit complexity and power consumption.

Let's see if we can do better with the constant temperature anemometer.

Analysis of the Constant Temperature Anemometer

The circuit concept for the constant temperature anemometer is shown again in figure 3.13 for convenience.

The emitter currents I_E are forced to be equal by the two constant current sinks. If β is large, then the collector currents I_{c1} and I_{c2} will likewise be equal.

By equations 3.5 and 3.6 above, we again have:

$$\frac{V_{be1}}{V_{be2}} = \frac{T1}{T2} \tag{3.6}$$

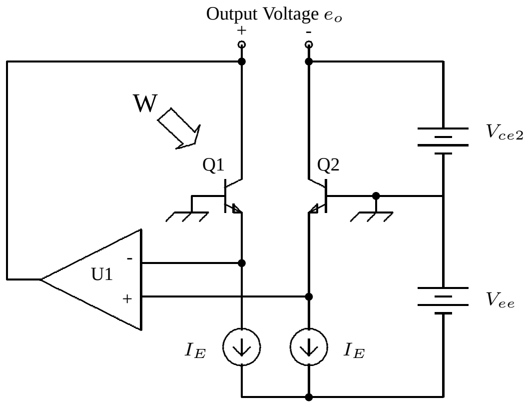


Figure 3.13: Constant Temperature Anemometer Concept

In this circuit, feedback action forces the base-emitter voltage of the two transistors to be equal. Since the base-emitter voltage is a known function of temperature, the two transistors must be at the same temperature.

In both cases, the thermal dissapation equations hold:

$$T_1 = T_A + \theta_1 P_{D1} \tag{3.7}$$
$$T_2 = T_A + \theta_2 P_{D2} \tag{3.8}$$

where

T_1,T_2	are the junction temperatures of the two transistors
T_A	is the ambient temperature
θ_1, θ_2	are the thermal resistances of the two transistors to ambient
P_{D1}, P_{D2}	are the power dissapations of the two transistors

The temperatures can be in degrees Celcius or in Kelvins, as long as they are all in the same units. We'll choose Kelvins, since it is temperature in Kelvins that is used in the junction temperature equation for the BJT, and this is consistent with equation 3.5.

Since these two temperatures are equal,

$$T_A + \theta_1 P_{D1} = T_A + \theta_2 P_{D2} \tag{3.9}$$

Ambient temperature T_A cancels out, which is a hopeful sign. We are interested in the thermal resistance θ_1 of the cooled transistor, which we can relate to wind speed using King's equation, so solve for θ_1:

$$\theta_1 = \frac{P_{D2}}{P_{D1}}\,\theta_2 \tag{3.10}$$

The power in each transistor is equal to its collector-emitter voltage times its collector current, so

$$P_{D1} = V_{ce1} I_{c1} \tag{3.11}$$

$$P_{D2} = V_{ce2} I_{c2} \tag{3.12}$$

Substitute in 3.10 for P_{D1} and P_{D2}:

$$\theta_1 = \frac{V_{ce1} I_{c1}}{V_{ce2} I_{c2}} \theta_2 \tag{3.13}$$

The collector currents cancel because they are equal, and we have that

$$\theta_1 = \frac{V_{ce1}}{V_{ce2}} \theta_2 \tag{3.14}$$

Referring now to the thermal model of the cooling effect, figure 3.6 on page 119, the thermal resistance of the reference transistor is simply θ_s, the static thermal resistance. For our circuit,

$$\theta_2 = \theta_s \tag{3.15}$$

The total thermal resistance of the wind-cooled transistor is

$$\theta_1 = \theta_s \parallel \frac{\theta_D}{\sqrt{U}} \tag{3.16}$$

where U is the wind speed and the other terms are constants.

Applying the equation for parallel resistances:

$$\frac{1}{\theta_1} = \frac{1}{\theta_s} + \frac{\sqrt{U}}{\theta_D} \tag{3.17}$$

Substitute for θ_1 from 3.17 into 3.14 and solve for the wind velocity:

$$U = \left(\frac{\theta_D}{\theta_S} \frac{e_o}{V_{ce2}} \right)^2 \tag{3.18}$$

This is very nice. The wind velocity depends on some constants and the square of the output voltage. No temperature stabilization, beyond that inherently provided by the circuit, is required. It's very clear the supply voltage V_{ce2} must be stabilized, but that the output is independent of other circuit parameters.

Clearly, of the constant power and constant temperature designs, constant temperature is the one to pursue.

3.9 The Hot Wire Anemometer

In previous circuits to detect wind speed using the cooling effect, the BJT base-emitter voltage was used as the temperature sensing element. The BJT has the advantage that its base-emitter voltage is a predictable, linear function of temperature, about -2.2mV/^{o}C, as discussed in section 1.4.2 on page 23.

However, transistor packages create considerable thermal mass, and this mass slows the response of the anemometer. Ideally, an anemometer should respond to wind gusts with a time constant of about 0.4 seconds while a test of a transistor with TO-92 case showed a time constant of 18 seconds.

In the constant temperature anemometer circuit, the time constant of the same transistor varied from 3 to 6 seconds, still an order of magnitude larger than desireable.

An alternative cooling effect sensor with lower thermal mass is a hot wire such as the filament of an incandescent lamp. The resistance of a wire conductor increases with temperature and this may be used to sense the cooling effect. The time constant of a 6 volt, 100mA lamp was measured in the circuit of figure 3.14 at about 150 milliseconds, much less than any transistor package.

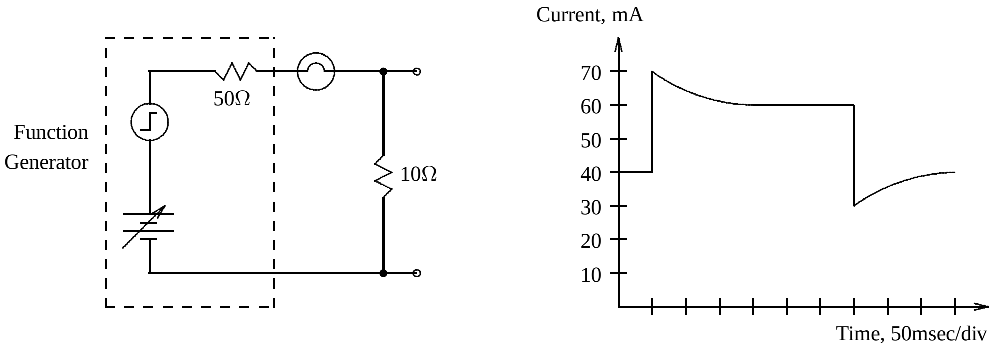


Figure 3.14: Measuring Lamp Time Constant

A suitable circuit for the operation of the Hot Wire anemometer is shown in figure 3.15.

Transistor Q1 provides power to the resistor bridge, one branch of which is the hot wire. The operational amplifier senses any voltage between its input terminals and drives transistor Q1 appropriately until the voltage difference disappears, and the bridge is balanced. For equal values of R_1 and R_2, the bridge will balance when $R_w = R_3$.

Step by step:
If wind cools the resistive sensor, its resistance will decrease. This causes the voltage at the non-inverting terminal of the op-amp to decrease and the output of the op-amp to increase. The increasing voltage at the output of the op amp increases the output voltage at the emitter of Q_1, increasing the current flow through the elements of the bridge, which increases the heat in the resistance wire R_w, causing its resistance to increase again. The net effect is that $R_w = R_3$, even when wind is cooling the resistance wire. However, the voltage across the bridge will increase when the resistance wire is cooled, and this may be used as a signal to measure the cooling effect.

Because the circuit maintains the resistance wire at a constant resistance, it also maintains it at a constant temperature, so this is a type of constant temperature anemometer.

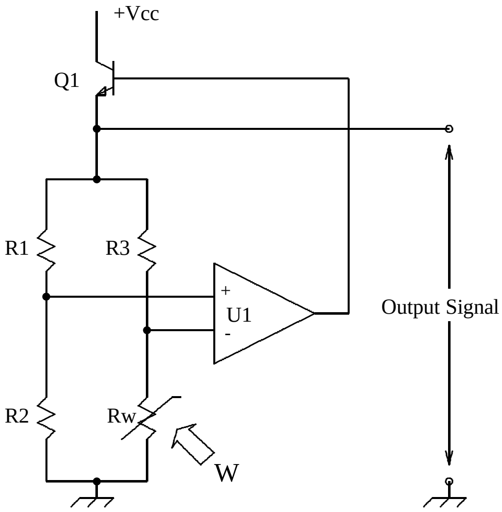


Figure 3.15: Hot Wire Anemometer Circuit

Detailed Analysis

The temperature T_w of the wire is typically maintained just below incandescence, about 1000^{o}C, much greater than the ambient temperature. Then the power dissapation P_w is given by

$$P_w = \frac{T_w - T_A}{\theta} \tag{3.19}$$

$$\approx \frac{T_w}{\theta} \tag{3.20}$$

where

P_w	is the power dissapation in the wire, watts
T_w	is the temperature of the wire, degrees C
T_A	is the ambient temperature, degrees C
θ	is the thermal resistance from the wire to ambient, ^{o}C/watt

When the air is moving, the thermal resistance is given by

$$\frac{1}{\theta} = \frac{\sqrt{U}}{\theta_D} \tag{3.21}$$

where

U	is the wind velocity
θ_D	is the dynamic thermal resistance constant

Substituting for θ from 3.21 into 3.20 we have

$$P_w = \frac{T_w}{\theta_D}\sqrt{U} \tag{3.22}$$

Now that we have the wire power dissapation related to wind speed, we need to relate the wire power dissapation in the output voltage. In a normal resistor, the current is directly proportional to voltage. The power dissapated in the resistor is then given by the familiar

$$P = \frac{V^2}{R} \tag{3.23}$$

In an incandescent lamp, the resistance of the filament increases as the applied voltage increases, so the current does not increase directly with the voltage. However, we'll ignore this effect for the moment and deal with it later.

Assuming that equation 3.23 is correct, then, the voltage V_w across R_w is

$$V_w = \sqrt{\frac{P_w}{R_W}} \tag{3.24}$$

If R_3 is chosen to have the same resistance as R_w, the output voltage e_o is simply twice V_w. Substitute for P_w from 3.22 and we have

$$e_o = 2\sqrt{\frac{P_w}{R_w}} \tag{3.25}$$

$$= 2\sqrt{\frac{T_w}{\theta_D}\frac{\sqrt{U}}{R_w}} \tag{3.26}$$

$$= 2\left(\frac{T_w}{\theta_D R_w}\right)^{1/2} U^{1/4} \tag{3.27}$$

The bad news here is that the output voltage e_o is proportional to the fourth root of the wind speed, which makes it a highly non-linear function, as shown in figure 3.16.

In figure 3.16, the values of k_1 and k_2 were chosen that the wind speed and output voltage each vary from 0 to 100 units, so that it is easy to compare the shape of the fourth root and square root functions. The fourth-root function changes value dramatically at low wind speeds and then only very slightly at high wind speed. When the output signal e_o is applied to the input of an A-D converter, each resolveable increment in wind speed should cause at least one increment in the A-D converter reading. For an 8 bit converter, because the fourth-root function changes so slowly at high wind speed, the ability to detect small changes in wind speed becomes very poor.

The square root function changes quickly at low wind speeds, but there is still sufficient slope to the function at higher wind speeds that the resolution is satisfactory.

To deal with the fourth root function, the designer could increase the resolution of the A-D converter or place a squarer circuit between the wind sensor and A-D converter. Such non-linear function circuits are useful in many applications, so we shall spend the next section describing this one.

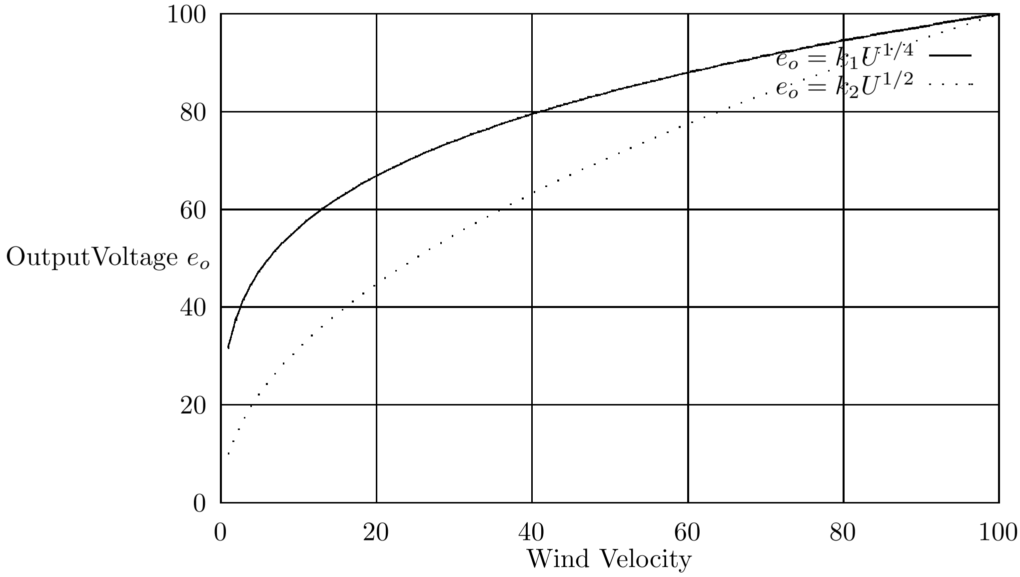


Figure 3.16: Cooling Functions

3.10 Analog Squaring Circuit

The conceptual squaring circuit is shown in figure 3.17. First we will show that the output current I_3 is proportional to the square of the input current I_1, and then how these currents are converted from and to voltages.

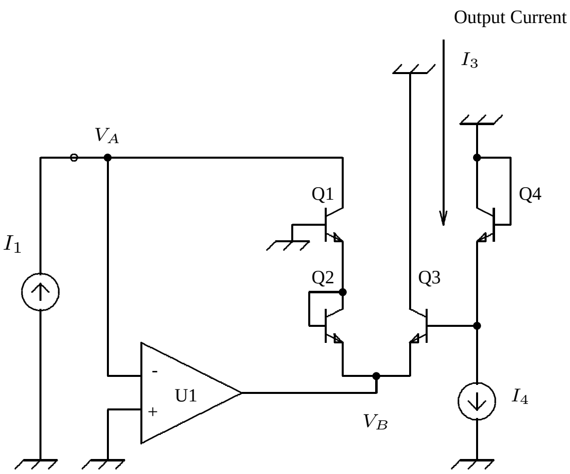


Figure 3.17: Square Function Circuit Concept

The operational amplifier establishes that voltage V_A is a virtual earth, and stays at zero volts. It also makes V_B sufficiently negative that all the input current I_1 flows through transistors Q1 and Q2.

The squarer circuit relies on the exponential relationship between base-emitter voltage and collector current:

$$I_c = I_s e^{\frac{V_{be}}{V_T}}$$

This relationship holds over a very wide range of collector currents: at least 1000:1, so the dynamic range of a non-linear function circuit using this principle will be similar.

Taking the logarithm of both sides and re-arranging:

$$V_{be} = V_T \ln \frac{I_c}{I_s} \tag{3.28}$$

Adding logarithms is equivalent to multiplication. Because the collector current of a transistor is exponentially related to its base-emitter voltage, the sum of base emitter voltages creates a product of collector currents, an effect that Gilbert [64] calls the translinear principle.

Summing base-emitter voltages around the loop of transistors, starting with Q1, we have

$$-V_{be1} - V_{be2} + V_{be3} + V_{be4} = 0$$

Substituting for the V_{be} terms from 3.28 we have

$$-V_T \ln \frac{I_1}{I_s} - V_T \ln \frac{I_2}{I_s} + V_T \ln \frac{I_3}{I_s} + V_T \ln \frac{I_3}{I_s} = 0$$

All the transistors are assumed to be matched and so all the saturation currents I_s are equal. If the transistors are all at the same temperature, then the V_T terms are also equal. Transistors Q1 and Q2 are in series, so $I_1 = I_2$. Applying these simplifications, we have

$$-2 \ln \frac{I_1}{I_s} + \ln \frac{I_3}{I_s} + \ln \frac{I_3}{I_s} = 0$$

Taking the antilog of both sides,

$$\frac{I_3 I_4}{I_1^2} = 1$$

or

$$I_3 = \frac{I_1^2}{I_4}$$

In this circuit, I_4 is fixed, so it simply acts as a scaling factor, and the output current I_3 is proportional to the square of the input current I_1.

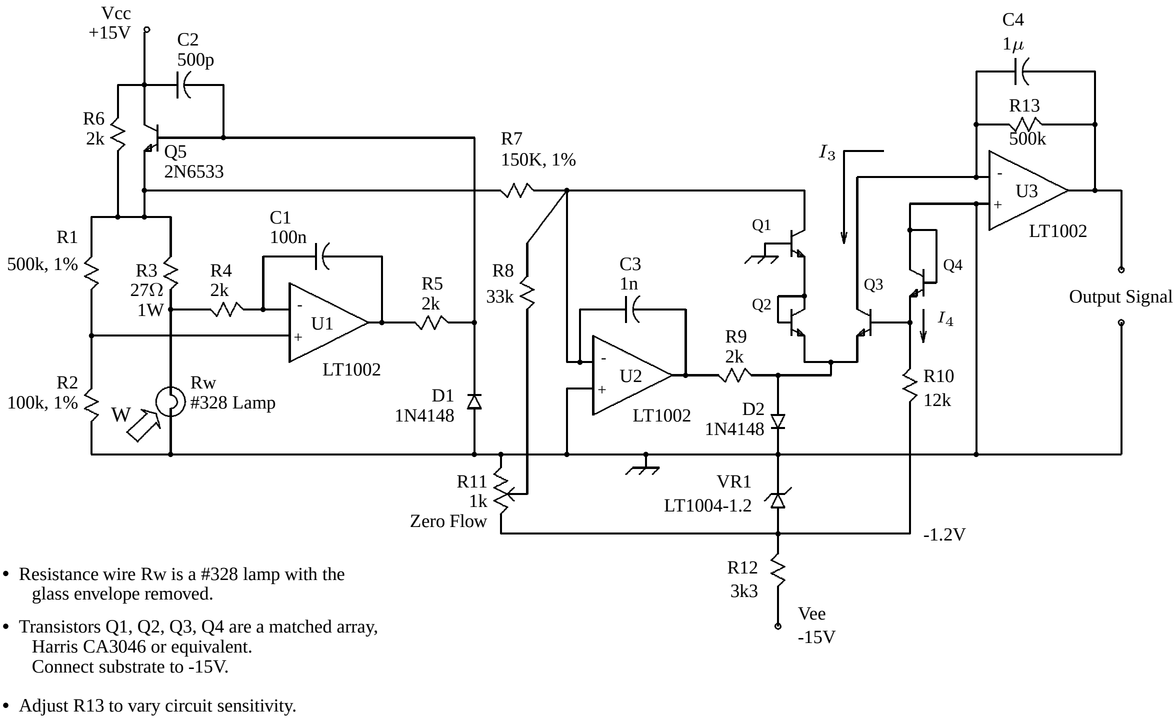


Figure 3.18: Hot Wire Anemometer with Squarer

3.11 A Hot Wire Anemometer Circuit

A complete hot wire anemometer circuit, from reference [59], is shown in figure 3.18.

There are a number of refinements in this circuit:

- The resistance wire is a #328 lamp, rated at 6 volts and 200mA. The glass envellope is removed and the lamp is operated below incandesence.
- The outputs of op-amps U1 and U2 are both limited by 2kΩ resistors and clamped with diodes. The clamping circuit ensures that an op-amp cannot produce the wrong polarity of voltage and damage the circuitry it is connected to.
- Feedback capacitances around the op-amps helps to prevent spurious oscillation in the circuit.
- In the circuit analysis, the resistors in each arm of the bridge were assumed to be equal. In this circuit, they are in the ratio of 5:1, which has the effect of increasing the voltage across the bridge and, more importantly, any change in the voltage across the bridge.
- Resistor R6 supplies 'start-up' current to drive the feedback loop into action.

- The output voltage from the anemometer sensor, at the emitter of Q6, is converted to a current by resistor R7. The squarer circuit generates a current I_3 proportional to the square of this current. Finally, op-amp U3 acts as a current-voltage converter to create an output voltage proportional to I_3.
- Voltage regulator VR1 creates a very stable -1.2 volt supply which is used for the Zero Flow pot and, via resistor R10, to generate the reference current I_4.
- Resistor R13 sets the overall circuit gain, output voltage per unit air speed, so it may need adjustment to set a suitable range for the microprocessor A-D converter.
- Transistors Q1 through Q4 are a matched transistor array such as the Harris CA3046 or equivalent. Notice that the substrate of the array must be connected to -15V to properly bias these transistors.

The Effect of Changing Wire Resistance

We previously noted that the characteristic of an incandescent lamp is not linear, but tends toward constant current as the voltage is increased. We should check to determine what effect this has on the performance of the hot-wire anemometer.

The voltage-current characteristic for a 6 volt, 100mA lamp is shown in figure 3.19.

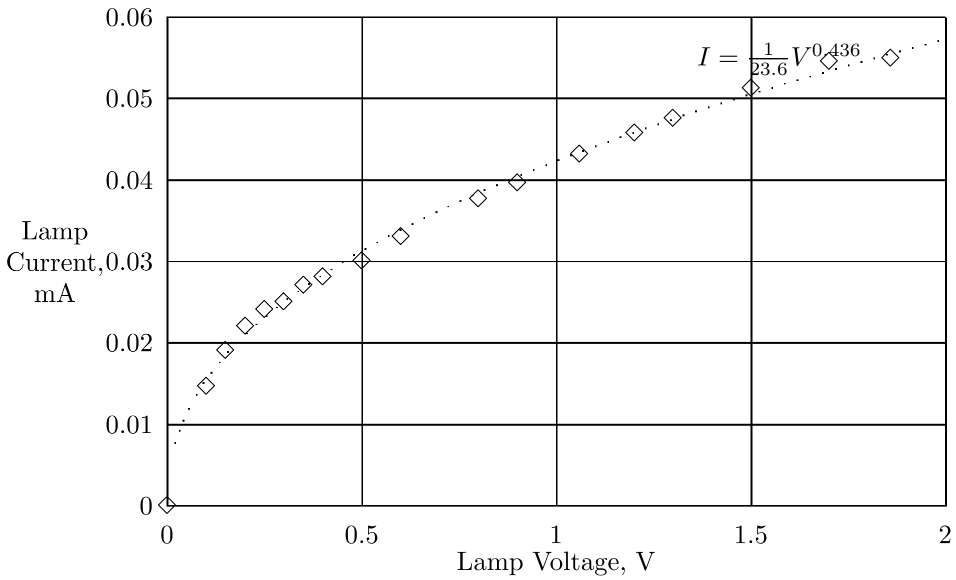


Figure 3.19: Lamp Voltage-Current Characteristic

Fitting an equation to the data, we have that

$$I = \frac{1}{23.6}V^{0.436}$$

The power in the lamp is the product of lamp voltage and current, so

$$
\begin{aligned}
P &= V \times I \\
&= V \times \frac{1}{23.6} V^{0.436} \\
&= \frac{1}{23.6} V^{1.436}
\end{aligned}
$$

Comparing this with $P = V^2/R$, it's clear that the lamp power does not increase as rapidly with voltage as the power in a fixed resistor. In the hot wire anemometer, this is an advantage because it tends to straighten out the transfer function, output voltage/wind speed, of the device.

The output function is now proportional to $U^{\frac{1}{3.46}}$ rather than $U^{\frac{1}{4}}$. After squaring, the output function is proportional to $U^{\frac{1}{1.46}}$: not quite linear, but easily linearized in software.

3.12 Heated Thermistor Anemometer

A cooling effect anemometer may be constructed using a thermistor as the temperature sensing element, in a circuit that is very similar to the hot wire anemometer. A thermistor uses simpler construction than a hot wire and is more rugged, so it may be more suitable for the outdoor environment.

On the other hand, a thermistor must be operated at a much lower temperature than a hot wire and so the effect of ambient temperature is important and must be subtracted from the output signal.

As well, a thermistor has a much longer time constant than a hot wire. The typical time constant for a small thermistor is in the order of 1 second or so. As well, the time constant is dependent in a complex manner on the magnitude of the change in wind velocity and the bias current in the thermistor [61, 62], which may have an effect on the calibration of the instrument.

A simple cooling effect anemometer circuit using a thermistor is shown in figure 3.20.

In contrast to the hot wire, whose resistance increases with temperature, thermistor resistance decreases with temperature. As a result, the input terminals of the op amp in the thermistor anemometer are opposite to the polarity of the hot wire circuit.

The value of R2 and the position of the potentiometer wiper are chosen to heat the thermistor to about 100^{o}C above the ambient temperature. The appropriate level of thermistor current depends on the thermistor resistance and its thermal resistance to ambient. (For a small value of R2, the circuit became a thermal oscillator with a period of 8 seconds, so the thermistor must be operated as a 'constant current' device to be stable.)

The output signal from the thermistor anemometer of reference [60] is shown in figure 3.21.

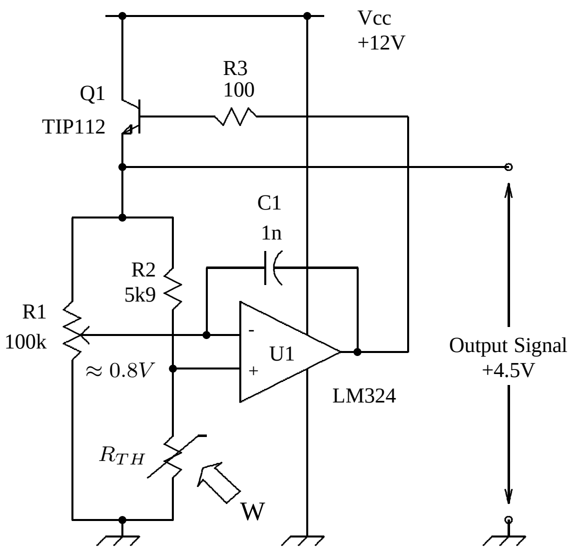
The curves fit to a function of the form

$$e_o = ku^{1/8.3}$$

which is approximately an 8th root function. Obviously, this will present even more severe problems in dynamic range than the hot wire anemometer. One can admire the persistence of Okamoto and his co-workers in making this work, but other approaches may be simpler.

An Engineering Proverb

Just because someone else made it work doesn't mean you can. Check other peoples designs very carefully.



- Thermistor: Stantel P23, R_{TH} 1k6Ω at T_A at 23^{o}C
- Output voltage change ≈ 2V in gusts

Figure 3.20: Thermistor Anemometer

The Okamoto heated thermistor design is a good example. An analysis warns us that the circuit will be very difficult to make work, and that it might be better to choose a different approach.

Another example of a difficult-to-replicate approach is shown in reference [67]. In this design, wind speed is measured by measuring the stress in a beam that is deformed by wind drag. Unfortunately, most materials exhibit a certain amount of hysterisis, so that the material tends to stay in a deformed state once it has been stressed. This is not noticeable when the stress is always in the same direction, but is quite evident when the direction of the stress is reversed.

3.13 Conclusions

- The traditional method of measuring wind speed is the rotating cup anemometer. Anemometers of this type are not particularly accurate, especially at low wind speed. With relatively large mechanical inertia, they do not follow wind gusts accurately.
- An alternative technique is the cooling effect anemometer, in which the thermal resistance of a heated sensor changes with the square root of wind speed.
- A number of possible configurations are possible. The constant power anemometer using a BJT sensor requires compensation for changes in ambient temperature, which is not particularly convenient.

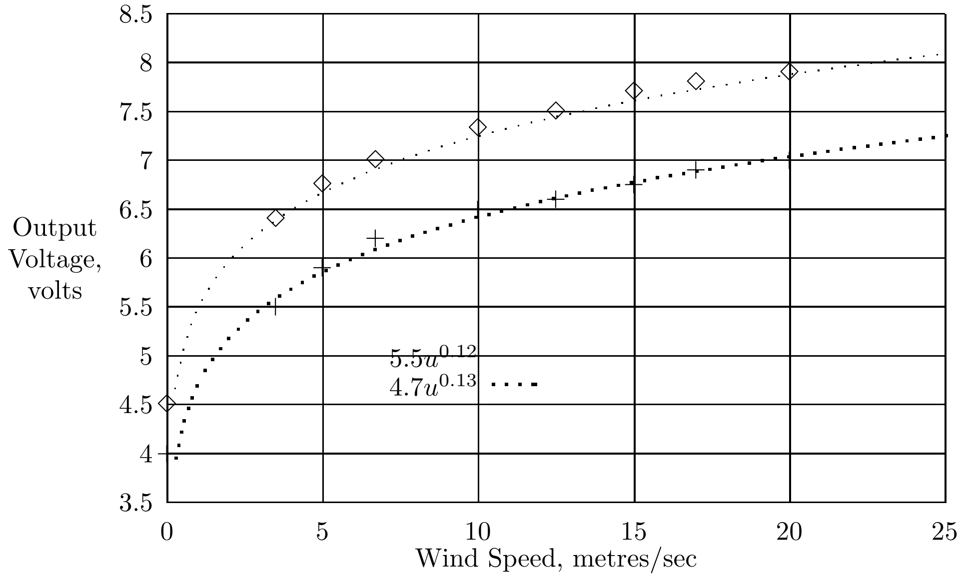


Figure 3.21: Thermistor Anemometer, Output Signal

- The constant temperature anemometer using a BJT automatically compensates for ambient temperature. Its output is proportional to the square root of wind speed. Transient response is relatively slow, similar to a rotating cup anemometer.

- The hot wire anemometer has very fast transient response and ignores changes in ambient temperature. With a squaring circuit, its response to wind speed lies somewhere between linear and square root functions.

- The heated thermistor anemometer is dependent on ambient temperature and highly nonlinear, proportional to the eighth root of the wind speed. Some sort of complex linearization circuit would be required with this approach.

3.14 Appendix: Simplifying King's Law

The form of King's law governing the heat loss from a filament in a current of air is given in his original paper [58] as

$$H \quad = \quad \kappa T + 2\sqrt{(\pi \kappa s \sigma a)} U^{1/2} T$$

where

H	is the heat loss, in watts, per unit length of the filament
κ	is the thermal conductivity of the fluid, 5.66×10^{-5}
T	is the temperature of the filament, in degrees C
s	is the specific heat of the fluid per unit mass, 0.171 calorie
σ	is the fluid density, 0.00193
a	is the diameter of the filament, in centimetres. Measuring a miniature lamp, we find a diameter of about 7.6×10^{-3}cm.
U	is the fluid velocity in cm/sec.

If the dynamic term is to be at least ten times the static term, then

$$2\sqrt{(\pi\kappa s\sigma a)}U^{1/2}T \quad \geq \quad 10\kappa T$$

Substituting numeric values in equation 3.29, we have

$$3.96 \times 10^{-4}\sqrt{U} \quad \geq \quad 5.66 \times 10^{-4}$$

and

$$\begin{aligned} U &\geq \text{ 2 cm/sec} \\ &\geq \text{ 0.072 km/hr} \end{aligned}$$

The dynamic thermal resistance, due to the movement of air, will predominate if the air velocity exceeds 0.072 km/hr, so for all practical purposes this will always be true.

3.15 Appendix: A Wind Tunnel for Calibration

A variable speed wind tunnel is useful in testing wind speed sensors. An arragement that is useful for cooling effect sensors is shown in figure 3.22 on page 141.

Speed control of the vacuum cleaner is obtained by supplying it from a variable transformer. In our case, we used a vacuum cleaner with a nameplate motor rating of 11 amps, and it generated wind speeds up to 89 miles/hour in the 2” diameter tunnel[2]

The wind speed sensor to be tested, such as a transistor or hot wire, is mounted on its circuit board, on the end of few centimetres of stiff wire. The circuit board is then placed below the tunnel, and the sensor offered up through the tunnel access hole. Should the sensor come loose during a test, it may be recovered from the canister of the vacuum cleaner.

The manometer shows the difference, in inches of water, between the air pressure at the static port and the dynamic port. To test that the manometer is operating correctly, operate the wind tunnel at a moderate speed and

[2] Alas, after several hours of running at low speed, the vacuum cleaner motor began to smoke badly and then expired. It turns out that the manufacturer relies on a high-speed fbw of air across the motor to cool it. When cooled with a blast of air, the wire used to wind the motor can be of a thinner guage, thereby reducing the manufacturing cost of the motor. However, driven at low speeds, the motor overheats. If you build a wind tunnel like this, you’ll need a fan such a variable speed squirrel cage impeller that is designed to run at low as well as high speeds. In this application, squirrel cage fan is probably superior to an axial fan because it can develop more back-pressure. Of course, if you have a budget, you can buy a small wind-tunnel.

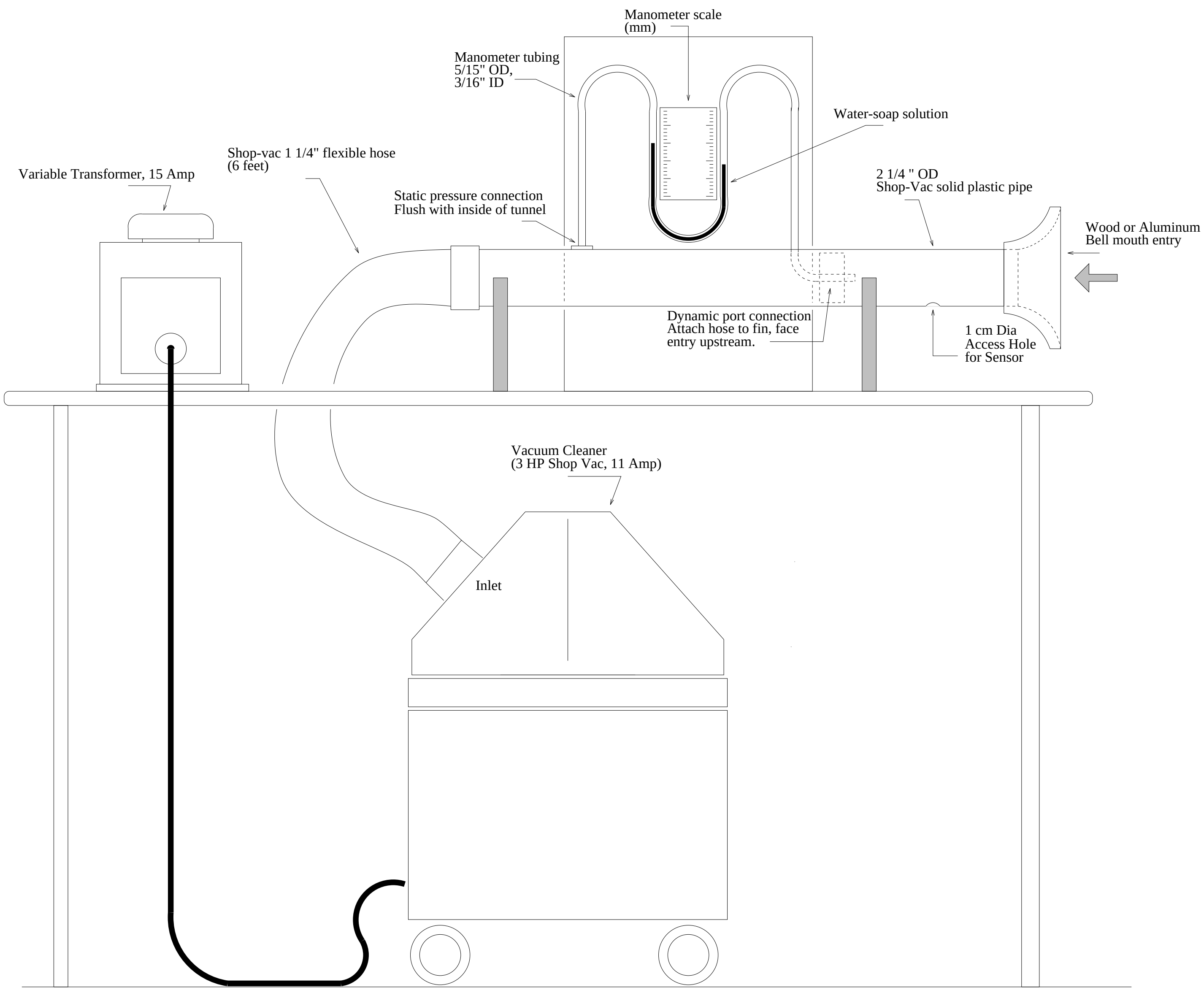


Figure 3.22: Wind Tunnel

then block the opening where wind enters the tunnel. After some oscillations, the manometer should show zero height difference between the two columns. The pressure in the tunnel is well below ambient under this condition, but it should be equal at the dynamic and static ports, and the manometer should reflect that.

Under normal operating conditions, the height difference h in the manometer is proportional to air speed, as follows [6], page 124:

$$P = \rho_w g h \tag{3.29}$$

where

P	is the pressure in dynes
ρ_w	is the density of water, 1 gm/cm^3
g	is the gravitational constant, 981 cm/sec^2
h	is the height difference between the two manometer water columns, cm

This is balanced by the pressure of the air velocity impinging on the dynamic tube opening:

$$P = \frac{1}{2}\rho_a V^2 \tag{3.30}$$

where

ρ_a	is the density of air, 1.225×10^{-3} gm/cm^3
V	is the velocity of the air in cm/sec.

Equating these two equations and solving for the air velocity, we have

$$V = \sqrt{\frac{2rho_w gh}{\rho_a}} \tag{3.31}$$

$$= \sqrt{\frac{2 \times 1 \times 980 \times h}{1.225 \times 10^{-3}}} \tag{3.32}$$

$$= 1264\sqrt{h} \text{ V in cm/sec, h in cm} \tag{3.33}$$

$$= 12.64\sqrt{h} \text{ V in m/sec, h in cm} \tag{3.34}$$

In case you don't have your conversion tables handy, 1 metre/sec is equal to 3.6 kilometers per hour or 2.237 miles per hour.

As an example, we found a manometer height difference of 9.4 cm at the maximum speed of the wind tunnel. This corresponds to:

$$V = 12.64\sqrt{h} \tag{3.35}$$

$$= 12.64\sqrt{9.4} \tag{3.36}$$

$$= 38.8\text{m}etres/sec \tag{3.37}$$

$$= 140\text{k}ilometers/hour \tag{3.38}$$

$$= 86.7\text{m}iles/hour \tag{3.39}$$

Chapter 4

Wind Direction

When the wind is in the east,
Tis neither good for man nor beast;
When the wind is in the north,
The skilful fisher goes not forth;
When the wind is in the south,
It blows the bait in the fishes mouth;
When the wind is in the west,
Then it is at the very best

Anonymous, quoted in J.O.Halliwell,
Popular Rymes, 1849

4.1 Introduction

In the latitudes between 30^o and 60^o, warm air from the tropics meets cold air from the artic. The intermingling of these air currents causes very changeable weather, in contrast to the stable weather of tropical regions.

Changing weather manifests itself as a depression, a low pressure air system. As a result of the rotation of the earth, the Coriolis effect causes winds in the Northern hemisphere to circulate around the depression in an anticlockwise direction. Thus the rule,

if you stand with your back to the wind, your left arm points in the direction of the depression.

The direction of the wind is of interest not only to meteorologists in forecasting the weather, but to aircraft and boats.

4.2 Accuracy and Resolution

A traditional division of the compass is shown in figure 4.1. The 360 degrees of the compass are divided into 16 segments, each labelled as shown in the diagram.

For a readout on the LCD of our microprocessor, we could display the appropriate segment name (**NNE**, for example) and/or the wind heading in degrees. The resolution required of the direction sensing device is rather low: one part in 16.

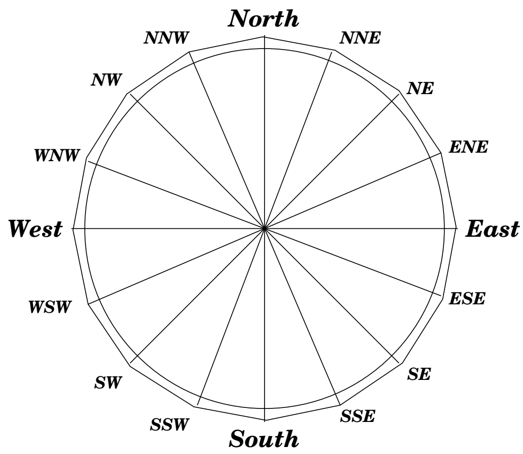


Figure 4.1: Compass Rose

The wind direction system must include a windvane, some sort of paddle that is free to rotate with the wind direction. In nautical environments, the main requirement of the windvane is that it not be carried off during a gale nor seize up from corrosion.

The position of the windvane is converted into an electrical signal by a rotary transducer of some description. The electrical signal is then processed and displayed on the computer.

4.3 Shaft Encoder

In one possible approach, the 16 points of the compass rose are directly encoded into 4 bits by an encoder of the sort shown in figure 4.2 (reference [7]).

As the encoder shaft rotates, it moves a pattern of code tracks between the light emitting diodes and the phototransistors. At any given rotation angle, the some phototransistors are illuminated and others dark, according to by a 4 bit code. The code creates a pattern of high and low signals on the four output lines, which the external circuit can interpret as an angle of rotation.

A four-bit binary code is the obvious one to use, but not the best choice. Consider that the disk tracks are binary coded and the position is such that the disk is at the edge of a transition from 0111 to 1000. It is unlikely that all four phototransistors will switch at once, and false states such as 0110 are likely to occur between 0111 and 1000. This difficulty can be avoided by using the Gray or reflected code shown in figure 4.3.

The name reflected code derives from the algorithm to generate the code sequence:

1. Write 0 followed by 1 in the first column.

2. Reflect this about a horizontal line, so that the next entries in the first column are 1 and 0.

3. Put zeros in front of the first half of this group, and 1's in front of the second half.

4. Reflect the table so far, and again put 0's in front of the first half and 1's in front of the second half.

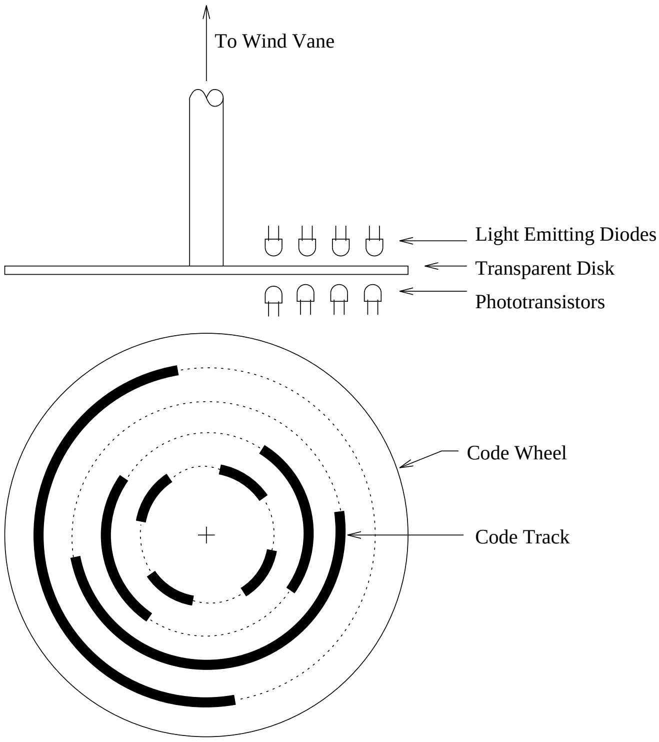


Figure 4.2: Shaft Encoder

5. Continue until the table has 16 entries.

The result is a code such that only one bit changes at any transition. In other words, the detected code will belong to one group or the other, but not some erroneous value between the two.

4.4 Synchro Resolver

The resolver is a electro-mechanical component widely used in commercial and military avionics equipment for the measurement of shaft angle. It looks much like a small motor, but is actually more closely related to transformers.

The resolver consists of three coils, a rotor attached to the shaft of the resolver and two orthoganal stators attached to the frame. The terminals of each coil are brought to the outside world. An AC voltage supply, typically 118volts RMS at 400Hz, is connected to the rotor winding. As the shaft rotates, the coupling between the rotor and the two stators changes, changing the magnitude of the output voltage of the stator windings. The voltage induced in a stator winding is at a maximum when the rotor and stator windings are aligned, and zero when they are at right angles to each other, figure 4.4.

The equations for the two stator voltages are:

$$e_{s1} = e_r K_r \sin\theta \tag{4.1}$$

0000
0001
0011
0010
0110
0111
0101
0100
1100
1101
1111
1110
1010
1011
1001
1000

Figure 4.3: Gray Code

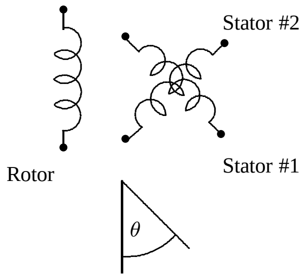


Figure 4.4: Resolver

$$e_{s2} = e_r K_r \cos\theta \tag{4.2}$$

where

e_r	is the rotor AC supply voltage
K_r	is a constant for the resolver, typically 0.3
e_{s1}, e_{s2}	are the voltages induced in the two stator windings
θ	is the angle between the rotor and stator S1

The magnitude of the AC stator voltage follows the sine function through a full 360^o of shaft rotation, figure 4.5.

For the first 180^o of rotation, the sine function and stator voltage are positive, so the AC voltage induced in the stator is in phase with the rotor voltage. For the second 180^o of rotation, the sine function and stator voltage are negative, implying that the AC voltage induced in the stator is inverted with respect to the phase of the rotor AC voltage.

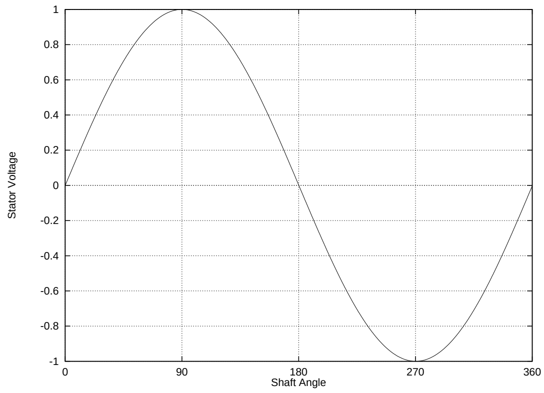


Figure 4.5: Resolver SINE Output

For any given voltage on this waveform (except for those at a maximum or minimum point) there are two possible angles. To unambiguously determine the shaft angle, we need the cosine output of the resolver as well, figure 4.6.

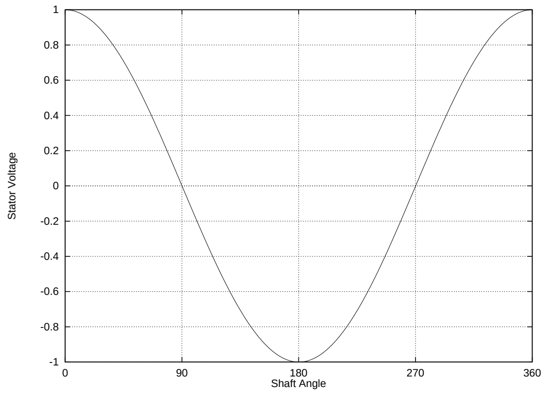


Figure 4.6: Resolver COSINE Output

In this system, the microprocessor reads both the Sine and Cosine voltages. The shaft angle is then

$$\tan\theta = \frac{e_r K_r \sin\theta}{e_r K_r \cos\theta}$$

so

$$\theta = \tan^{-1}\frac{\sin\theta}{\cos\theta} \tag{4.3}$$

(Some care must be taken in calculating θ at the angles 90^o and 270^o, where the value of the cosine approaches zero and the tangent function approaches infinity.)

Figure 4.7 shows what the stator voltage waveform would look like if the rotor were to be rotated so that one rotation corresponded to 20 cycles of the AC excitation voltage.[1]

A careful examination of that figure shows that the stator voltages in the 0-180^o segment and 180-360^o segment equal in magnitude but out of phase. A suitable electronic circuit can decode the magnitude and phase to determine the angle of the rotor.

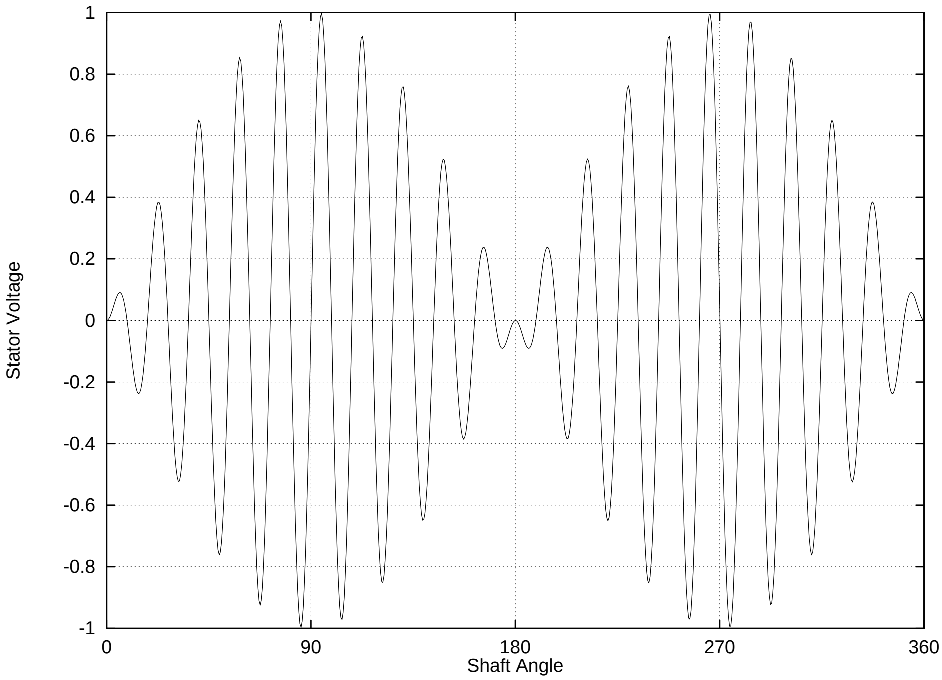


Figure 4.7: Resolver Stator Voltage Waveform vs Shaft Angle

The resolver is an attractive device to use for shaft angle measurement. Little torque is required to rotate the shaft and the output function is an accurate measurement of the shaft angle.

For a low-cost consumer grade weather station, the resolver is a rather expensive device to use for wind direction measurement. As an avionics component, the unit cost is in the order of $50 (and up) each. However, resolvers and their near-relatives, the synchro-transmitter are commonly available on the surplus market for a very modest price. For one or two custom-built units, it may be a practical approach.

Now we need a method of converting the resolver signals into something that can be digested by the microprocessor. A suitable interface circuit is described in the next section.

[1] To obtain the waveform shown in fi gure 4.7, the resolver shaft would have to be rotated at exactly the correct frequency, which is all but impossible to do in practice.

4.5 Resolver Interface

4.5.1 An Aside on Circuit Design Methodology

It is customary in academic texts to begin an explanation with a general description of the matter to be studied, and then (if space permits and the reader is fortunate) develop examples or show the principles realized in hardware. The general description may be difficult to comprehend at first reading, but is eventually useful in extrapolating to new applications and different implementations of the general princible.

Unfortunately, this misrepresents the process of electronic circuit design: a mixture of brainstorming, theoretical analysis, exploration of the subject literature, reading of manufacturer's catalogues, and circuit experiments, all in no particular order. Many useful insights come from the contemplation of circuit design failures, a subject that is almost never discussed in the literature.

Nontheless, once the thing works, we organize the results in an attactive sequence of theory, circuit and results, as if we planned it that way in the first place.

The circuit at hand is a good example. The building blocks came from a variety of sources, a circuit lash-up was made to work after a certain amount of tinkering, and only then did the author develop a conceptual framework around the final circuit. The moral? Do not feel that your process of arriving at a circuit design must proceed in a linear fashion from theory to working hardware. Life is not that neat.

4.5.2 Resolver Interface as Communications System

Some insight into the configuration of the resolver interface can be obtained by regarding the resolver as a modulator in a communications system, in which the carrier is the 400Hz excitation of the resolver and the information signal is the rotation of the shaft. The corresponding block diagram (for one channel) is shown in figure 4.8.

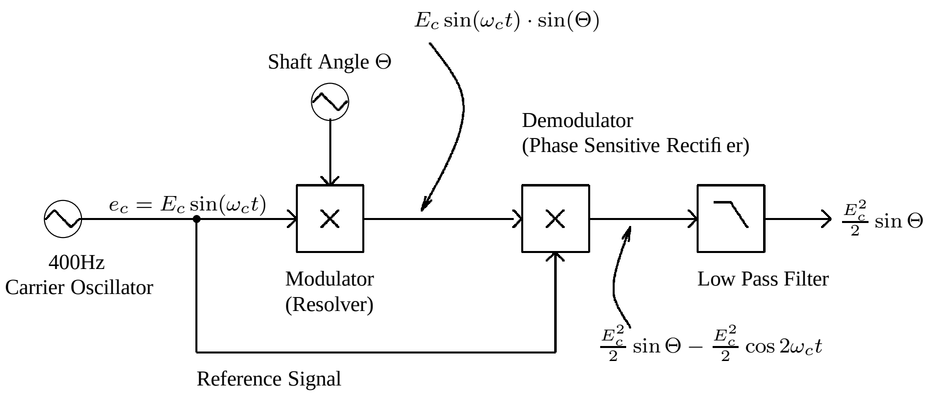


Figure 4.8: Resolver Interface Block Diagram

The amplitude modulated waveform, output from the resolver, is shown in figure 4.7. It is given by the equation

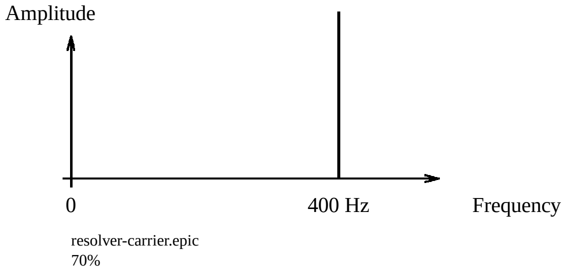
$$E_c \sin(\omega_c t) \cdot \sin(\Theta) \tag{4.4}$$

where

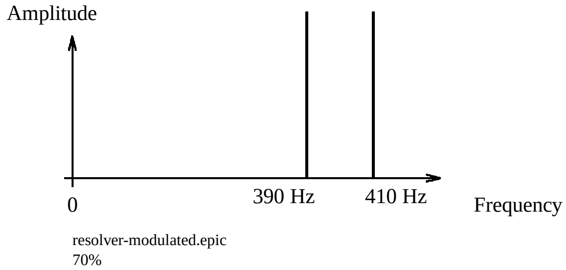
E_c is the peak value of the carrier voltage
ω_c is the carrier frequency in radians per second
Θ is the shaft angle of the resolver

In spectral terms, the original 400 Hz carrier is shown in 4.9(a). With a 10 Hz modulation signal (resolver shaft rotation of 600 RPM[2]), the amplitude modulated carrier is as shown in figure 4.9(b). In communications terms, the amplitude modulated carrier is known as a Double Sideband Suppressed Carrier (DSB) signal.

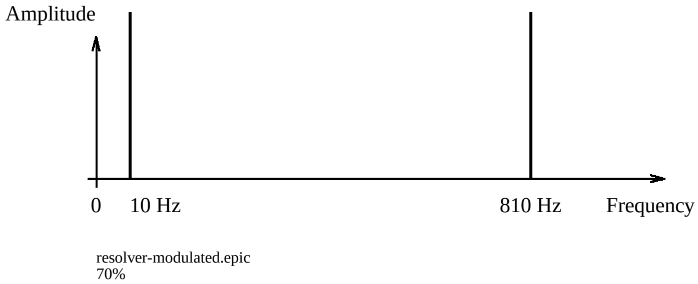
[2] In practice, of course, the resolver shaft is connected to a wind vane and does not normally rotate at this rate. As a result, the modulation frequency is very nearly zero Hz.



(a) 400 Hz Carrier Spectrum



(b) DSB Modulated Carrier



(c) Demodulated Signal

Figure 4.9: Resolver Spectum Diagrams

To extract the angle information from this waveform, it is multiplied in the demodulator by the carrier frequency,

$$e_c = E_c \sin(\omega_c t)$$

The result is a signal containing two frequencies. The first frequency is at twice the carrier and a second, consisting only of the modulating signal

$$\frac{E_c^2}{2}\sin(\Theta)$$

is at or near zero frequency. The corresponding spectrum is shown in figure 4.9(c).

$$\frac{E_c^2}{2}\sin\Theta - \frac{E_c^2}{2}\cos 2\omega_c t \tag{4.5}$$

The signal of interest is the modulation signal, which may be separated from the twice-carrier-frequency signal by a low pass filter.

The output signal is then the modulation signal

$$\frac{E_c^2}{2}\sin\Theta \tag{4.6}$$

The corner frequency of the low pass filter must be set low enough that the twice-carrier signal is sufficiently attenuated. On the other hand, the modulation signal becomes a low frequency sine wave (moves away from zero frequency) when the shaft is rotated, and this should not be attenuated by the low pass filter.

4.5.3 Resolver Interface Building Blocks

The resolver interface consists of the following functional blocks:

Carrier Oscillator

A source of 400Hz sine wave alternating voltage is required to excite the resolver rotor winding. A common sine wave oscillator, the Wein Bridge oscillator, is shown in figure 4.10, reference [74].

The resistor/capacitor bridge network R1 C1 R2 C2 determines the frequency of oscillation ω_o. If R1=R2=R and C1=C2=C, then it may be shown that

$$\omega_o = 1/RC \tag{4.7}$$

For oscillation to occur, the loop gain around the amplifier and its feedback network must have a magnitude of unity and a phase shift of zero degrees, the so-called Barkenhousen criterion. At frequency ω_o, it turns out that the phase shift through the bridge network is 0^o and the gain 1/3 volts/volt, so the amplifier must have a gain of 3 volts/volt to generate unity loop gain overall. If the amplifier gain is slightly larger than 3, the amplitude of the oscillation will grow until the amplifier clips the waveform. In effect, this lowers the gain until it is exactly 3, and the amplitude of the waveform stabilizes. The clipping action of the amplifier results in a slightly distorted sine wave. This would not be acceptable in an audio application, and some more elaborate feedback stabilization system would be required. In this application, some distortion is acceptable and a reasonable tradeoff for the simplicity of the circuit.

Resistors R3 and R4 define the closed loop forward gain of the op-amp circuit and are proportioned to provide a gain of slightly more than 3 to ensure that the oscillator will start reliably.

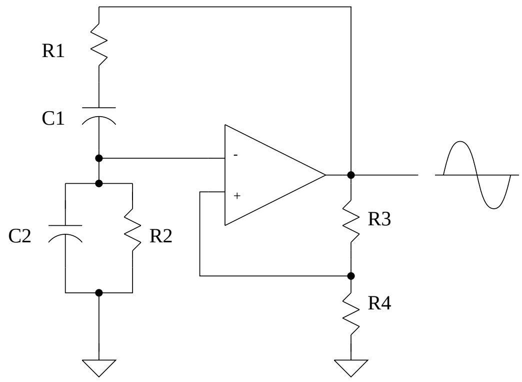


Figure 4.10: Wein Bridge Oscillator

Demodulator

An analogue multiplier could be used as the demodulator for the resolver output. However, an analogue multiplier is a relatively complex device and it is worthwhile to look for a simplification. A useful approach is to convert the reference waveform to a square wave of value ± 1, at the same frequency as the reference. Multiplication is then equivalent to switching the polarity of a signal, and the phase sensitive rectifier of figure 4.11 may be used.

In the frequency domain, the square wave reference signal may be regarded as a Fourier series containing frequencies at the fundamental, and with decreasing amplitudes, at odd harmonics of the fundamental. Multiplication in the demodulator by harmonics of the fundamental will generate product frequencies at the sum and difference of the input signal and the various harmonics of the reference. However, these product frequencies do not occur at zero frequency so the low pass filter will remove them and the low pass filter correctly extracts the zero-frequency resolver angle signal. In other words, a square wave reference signal and a polarity switcher will correctly demodulate the resolver signal.

Referring to figure 4.11, the operation of the phase sensitive rectifier is follows: Consider initially the part of the cycle for which the reference signal causes the switch S to be open.

- Because no current i_{ref} flows into the non-inverting terminal of the op-amp, the voltage at that terminal is equal to the input voltage.
- By negative feedback action, the voltage at the inverting terminal must be equal to the voltage at the non-inverting terminal.
- Then the input current i_i must be zero, the feedback current i_f must be zero, there are no voltage drops in any of the resistors, and the output voltage is equal to the input voltage.
- In other words, the voltage gain through this interval is $+1$.

Now consider second half of the cycle, for which the reference signal causes the switch S to be closed.

- The non-inverting terminal is connected to ground, so its voltage is zero.
- By negative feedback action, the voltage at the inverting terminal must be equal to the voltage at the non-inverting terminal, so it also is zero.

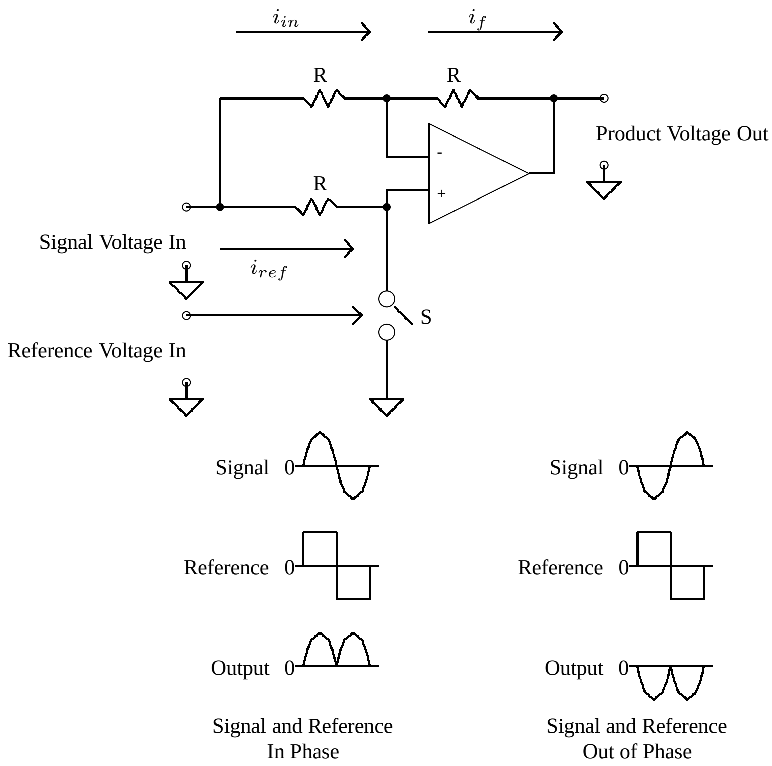


Figure 4.11: Phase Sensitive Rectifier

- With the inverting terminal at zero volts and equal input and feedback resistors, the amplifier is configured as a inverting amplifier of gain -1.

In other words, the phase sensitive rectifier has a gain of plus or minus 1, depending on whether the reference switch is open or closed. The waveforms of figure 4.11 show the output signal when the input waveform is in phase with the reference. If the input and reference are out of phase, then the output voltage will be inverted.

The effect of phase sensitive rectification on the signal from the resolver may be seen by comparing figure 4.7 on page 148 with the waveform at the output of the phase-sensitive rectifier, figure 4.12 on page 155.

Passing this waveform through a low-pass filter removes the high frequency components, leaving the envellope of the waveform, a signal proportional to the sine of the resolver shaft angle. In the frequency domain, for a fixed shaft angle, the PSR output, a full wave rectified signal, is given by the Fourier series (ref [5]):

$$e(\omega) = E_c \left(\frac{2}{\pi} - \frac{4}{\pi} \left(\frac{\cos 2\omega t}{1 \cdot 3} + \frac{\cos 4\omega t}{3 \cdot 3} + \frac{\cos 6\omega t}{5 \cdot 7} \cdots \right) \right) \tag{4.8}$$

The average (DC) term in the series, the voltage passed by the low pass filter, is

$$e_{dc} = E_c \frac{2}{\pi}$$

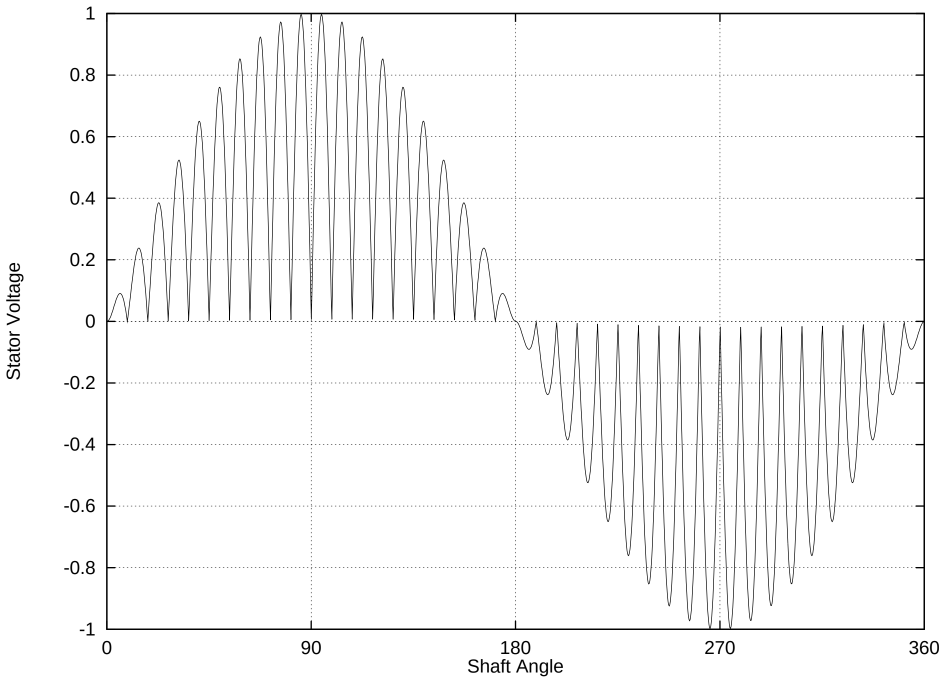


Figure 4.12: Phase-Sensitive Rectifier Output

The remaining AC terms contribute to the ripple at the output of the low pass filter.

The switch in figure 4.11 must conduct current in both directions, so a single junction transistor or FET will not work properly. The 4066 Quad Bilateral Switch is a better choice. It conducts in both directions and may be controlled by a single 5 volt signal. The **on** resistance of the switch is given as 1050Ω maximum, so the other resistors in the phase sensitive rectifier should be much larger than this.

Low Pass Filter Design

The low pass filter extracts the average value of the full wave signal, and supplies some gain to make up for losses in the system.

The circuit shown in figure 4.13 is suitable. The transfer function of the circuit shown in figure 4.13 is

$$\frac{e_o(j\omega)}{e_i(j\omega)} = -\frac{R_f}{R_i}\left(\frac{1}{1 + j\omega R_f C_f}\right) \tag{4.9}$$

This describes a voltage gain function for which the low frequency gain is R_f/R_i volts/volt and the gain decreases at 20 db/decade above the cutoff frequency ω_o, where

$$\omega_o = \frac{1}{R_f C_f} \text{ radians/sec} \tag{4.10}$$

The amplitude response of the filter is shown in figure 4.14.

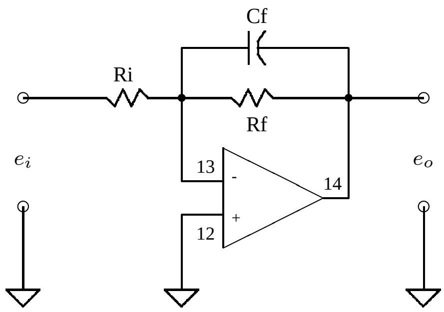


Figure 4.13: Low Pass Filter

Determining the Required Gain

The gain in the final stage is required to set the peak positive and negative values of the voltage at the input to the A-D converter, as the angle Θ moves between $+90^o$ and -90^o. The shaft angle modulates the carrier amplitude, but the output swing is less than the amplitude of the carrier wave for two reasons:

- the transfer gain of the resolver is K_r, less than 1 (equation 4.2 on page 4.2). In the prototype, K_r was found to be 0.3 volts/volt
- the positional information is found by averaging the modulated output waveform, and the average of a full-wave rectified waveform is $2/\pi$ of the peak value, equation 4.8.

Putting these two together, the peak output swing is

$$\Delta V_o = K_r \times \frac{2}{\pi} \times E_c \times K_f \tag{4.11}$$

where

K_r	is a constant for the resolver, typically 0.3
ΔV_o	is the output voltage swing of the amplifier-filter stage, volts
E_c	is the peak value of the carrier voltage, volts
K_f	is the voltage gain of the amplifier stage, volts/volt

We might choose a voltage swing into the A-D converter ΔV_o of 4 volts p-p, 2 volts peak. The carrier voltage swings more or less between the supply rails at 0 and +5 volts, so E_c is 2.5 volts. Solving for the required gain K_f in equation 4.11, we have

$$\begin{aligned} K_f &= \frac{2,0}{0.3 \times \frac{2}{\pi} \times 2.5} \\ &= 4.19 \text{ volts/volt} \end{aligned}$$

Arbitrarily choosing R_i as 100kΩ, the feedback resistor must then be at least $4.19 \times 100k\Omega = 419k\Omega$.

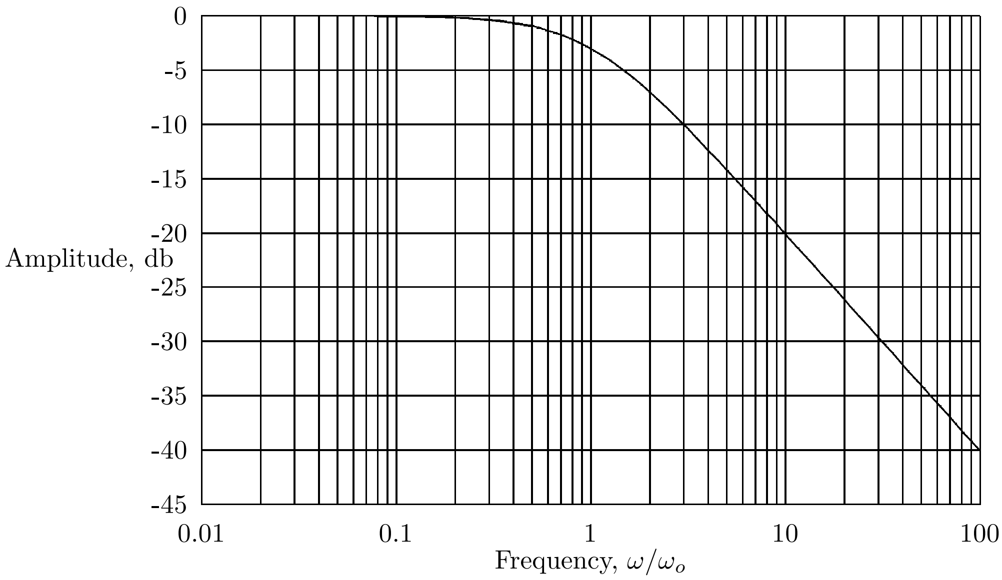


Figure 4.14: Lowpass Filter Response

Designing the Low Pass Filter

Now we need to calculate the value of the low pass filter feedback capacitor C_f. The position of the filter cutoff frequency ω_o should be set low enough that the 800Hz demodulation signal does not appear in the signal to the A-D converter.

From the Fourier series for a full-wave rectified sine wave, equation 4.8, the most significant AC term is

$$e_{2f} = E_p \left(-\frac{4}{\pi} \cdot \frac{\cos 2\omega t}{1 \cdot 3} \right) \tag{4.12}$$

This describes a signal at a frequency of twice the carrier, 800Hz, and amplitude E_p volts peak. In this case, using the same reasoning used to develop equation 4.11, we can determine that $E_p = 2.5 \times 0.3 = 0.75$ volts peak. Substituting in equation 4.12 for E_p and the equation of the AC component is

$$e_{2f} = 0.424 \cos 2\omega t$$

The low pass filter should reduce the peak-peak amplitude of this sine wave to well below the amplitude of one A-D converter step, which is

$$5 \text{ volts}/256 \text{ steps} = 0.0195 \text{ volts/step}$$

We might choose an acceptable peak-peak ripple amplitude of one half this, about 9.7 millivolts. To determine the corresponding value of ω_o, we can begin by manipulating equation 4.9. For frequencies much higher than ω_o (ie, on the -20 db/decade section of the amplitude function in figure 4.14), equation 4.9 becomes

$$\frac{e_o(j\omega)}{e_i(j\omega)} \approx -\frac{R_f}{R_i}\left(\frac{1}{j\omega/\omega_o}\right) \tag{4.13}$$

The magnitude is given by

$$\left|\frac{e_o}{e_i}\right| = \frac{R_f}{R_i}\left(\frac{1}{\omega/\omega_o}\right) \tag{4.14}$$

Substitute the following values:

e_o	= 9.7 millivolts
e_i	= 0.424 volts
ω	=800 Hz (5×10^3 radians/sec)
R_f	=430kΩ
R_i	=100kΩ

Solving for ω_o, we find that

$$\omega_o = 11.34 \text{ radians/second, (1.8Hz)}$$

Now substitute for ω_o and R_f in equation 4.10 and solve for C_f, from which we find that

$$C_f = 205 \text{ nanofarads}$$

Using a value of 220nF in the circuit, the ripple into the A-D converter measured approximately 8mV p-p.

Notice that the angle signal $\sin\Theta$ will be attenuated by the low pass filter if the shaft rotation speed is faster than 1.8 Hz. If this is a problem, one would choose a 2nd order filter which has a sharper rolloff and can therefore accomodate a higher cutoff frequency.

Power Supply Splitter

In previous circuits, we have taken care to fit the design to the available power supply, 0 and +5 volts. In this case, we take a different approach, and generate a bipolar supply, $\pm$2.5 volts.

This may be accomplished by splitting the power supply at its halfway point, using a voltage divider. The centre point then becomes a pseudo-ground: effectively a local ground for the devices in this circuit.

If the current from the positive supply is equal to the current to the negative supply, the current in the pseudo-ground will be zero. However, that is unlikely to be the case, and a buffer amplifier is required to supply the difference current to ensure that the pseudo-ground stays halfway between the positive and negative supply rails.

An operational amplifier could be used to buffer the voltage divider. However, the maximum output current for an op-amp is in the order of some tens of milliamps, which may not be sufficient. In this circuit, an audio power amplifier, the National LM386, has been used.

The LM386 will work from power supplies as low as 4 volts, automatically biases its output at (approximately) half the supply voltage, and can source or sink in the order of 250 milliamps, all for a cost of under a dollar. It must, however, be decoupled with the capacitors shown in figure 4.15 to prevent spurious oscillations.

Resolver Interface Circuit Diagram

The complete circuit diagram of the resolver-based wind direction interface is shown in figure 4.15. The Cosine channel is simply a copy of the Sine channel.

One minor detail that is not discussed elsewhere: the output of each amplifier-lowpass filter stage is connected to its A-D converter input via resistor R10, 3kΩ. The input of the A-D converter is very high resistance, so there is effectively no voltage drop across R10. However, the LMC660 is easily destabilized by load capacitance, and the A-D converter presents a capacitive load to the op-amp. Resistor R10 separates the output of the op-amp from the A-D input capacitance and prevents the op-amp from oscillating. Of course, the op-amps should all be decoupled near their power supply pins, using a capacitor in the range of 10nF or so.

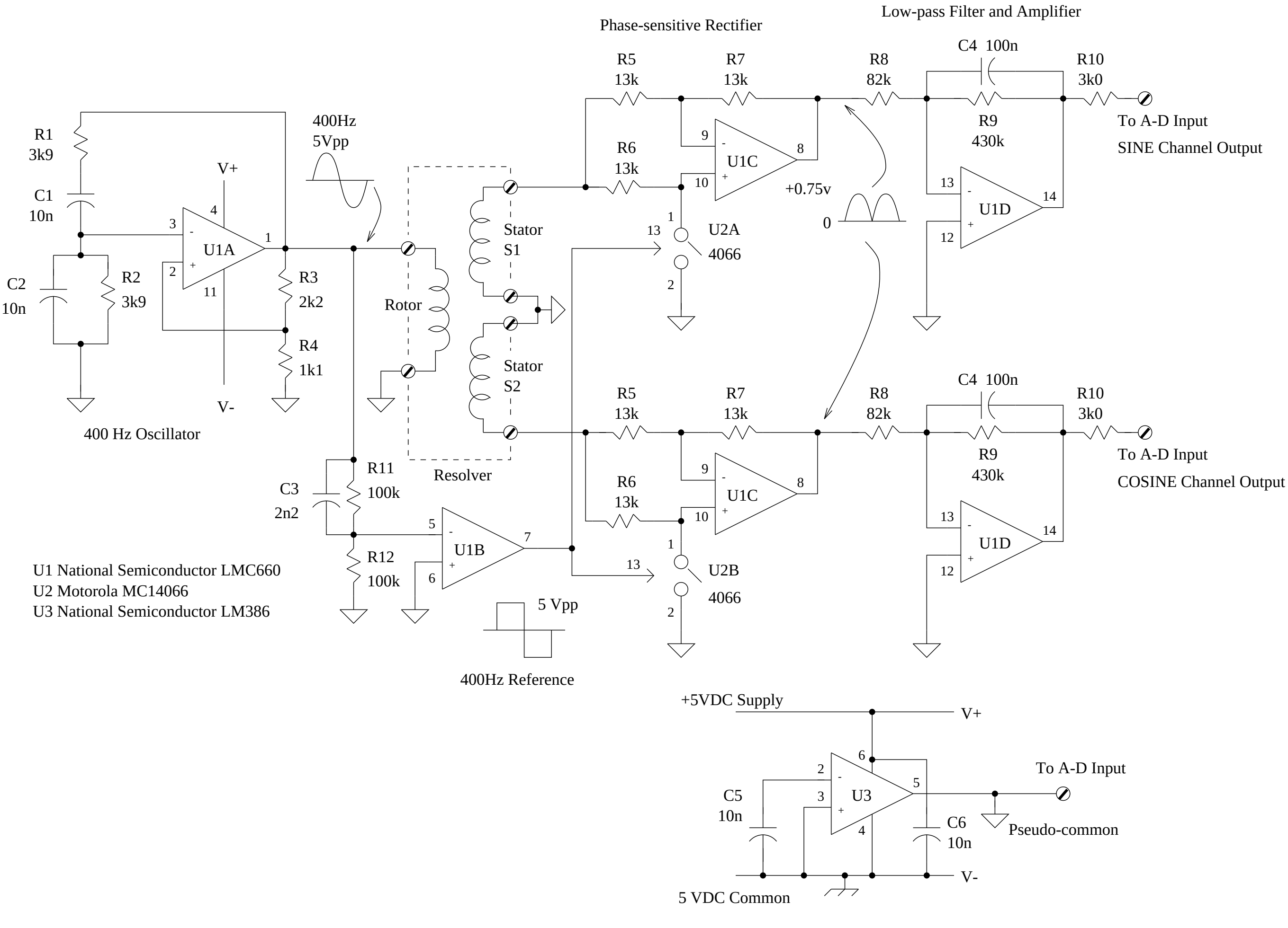


Figure 4.15: Resolver Interface Schematic

Software

The microprocessor reads the two A-D channels and then subtracts the pseudo-ground voltage to determine the values of the sine and cosine functions. It then applies equation 4.3 to determine the resolver shaft angle θ. Notice that the arctan function will report an error if the angle is sufficiently close to 90 or 270 degrees that the floating point values overflow.

Calibration Procedure

Providing the angle function is symmetrical about the zero point, it is possible to calibrate the system at the angles 90^o and 270^o, points where the angle function is maximum. The calibration device, a human or robot, rotates the wind vane until the A-D voltage is at its positive maximum. This is the 90^o angle. The human or robot then rotates the wind vane to obtain a minimum input voltage. This is the 270^o angle. The computer can now determine intermediate voltages and their corresponding angles.

With most wind vane devices, the unit must be carefully aligned so that North on the case of the transducer is actually pointing to geographic north. With a microprocessor controller, it is possible to have the software add a correction angle β to the reading from the wind vane. The wind vane is caused to point to some known heading and the correction angle incremented or decremented until the true heading is obtained. This would be particularly convenient where the wind vane is inaccessible for adjustment, at the top of a mast, for example.

Chapter 5

Air Pressure

> There is a sumptuous variety about the New England weather that compels the stranger's admiration - and regret. The weather is always doing something there; always attending strictly to business; always getting up new designs and trying them on the people to see how they will go. But it gets through more business in spring than in any other season. In the spring I have counted one hundred and thirty-six different kinds of weather inside of twenty-four hours. Mark Twain, 1876

5.1 Introduction

As the air surrounding the earth is heated by the engine of the sun and cooled by radiation into space, air density differences from place to place result in the air movements we sense as winds. These winds bring us different types of weather, so measuring the air pressure is a very important technique in the prediction of weather.

For example, a sudden drop in air pressure often signals the onset of stormy weather; high pressure signals continuing fine weather.

5.2 Measuring Air Pressure

The classical method of measuring air pressure is the mercury barometer, a column of liquid that is supported by atmospheric pressure, figure 5.1.

A closed tube is filled with mercury and then inverted into a reservoir or cistern of the liquid. The liquid column will fall, forming a vacuum above its top surface, until the weight of the column is balanced by the atmospheric pressure. Other liquids can be used, but mercury is attractive because its high density results in a relatively compact instrument. For precise measurements, the observer must carefully determine the height of the column above the level in the reservoir, and compensate for the temperature of the barometer.

In the resevoir, the downward pressure P_m of the mercury column is balanced by the air pressure P_A.

$$P_m = P_A \tag{5.1}$$

The pressure of the mercury column is

$$P_m = \frac{f_m}{a_m} \tag{5.2}$$

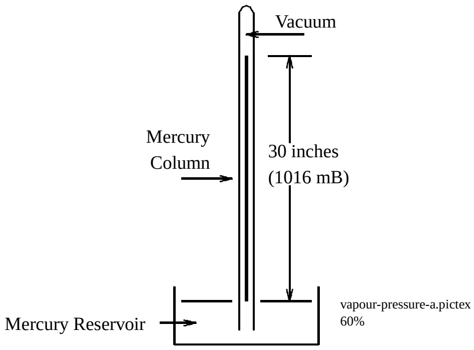


Figure 5.1: Mercury Column Barometer

where

f_m	is the force exerted by the column
a_m	is the cross-sectional area of the column

The column force is

$$f_m = m_m \cdot g \tag{5.3}$$

where

m_m	is the mass of the mercury column
g	is the gravitational constant, 980 cm/sec^2 in the cgs system

The mass of the column is

$$m_m = d_m a_m h_m \tag{5.4}$$

where

d_m	is the density of the mercury, 13.6 g/cm^3 in the cgs system
a_m	is the cross-sectional area of the mercury column
h_m	is the height of the column

Collapsing these equations, we obtain a useful expression for air pressure in terms of the column height and density.

$$P_A = \frac{d_m a_m h_m g}{a_m} \tag{5.5}$$

$$= d_m h_m g \tag{5.6}$$

For example, the so-called standard pressure of physics and chemistry causes a mercury column height of 76 cm. This is an atmospheric pressure of

$$\begin{aligned} P_A &= 13.6 \times 76 \times 980 \\ &= 1013 \times 10^6 \text{dynes/cm}^2 \end{aligned}$$

Pressure Units

A variety of units of pressure have evolved over time. The bar is defined as 10^6 dynes/cm^2 so standard pressure is 1.013 bars. Weather forcasts commonly quote air pressure in millibars, so standard air pressure is 1013 millibars.

In the metric system of measurement, the standard unit of pressure is the pascal, one newton/metre^2. Air pressure is conveniently described in kilopascals, or kPa. Standard pressure becomes 101.3 kPa.

In the English system, the corresponding units are inches of mercury for atmospheric pressure and pounds per square inch for pressure guages.

The values of standard pressure in some common units of pressure are summarized in figure 5.2.

1.0	Atmosphere	ATM
1013	millibars	mB
101.3	kilopascals	kPa
76.0	centimetres mercury	cm.Hg
160.2	centimetres water	cm.H_2O
14.69	pounds per square inch	PSI
29.92	inches mercury	in.Hg
406.8	inches water	in.H_2O

Figure 5.2: Pressure Units

The Aneroid Barometer

With careful attention to detail, a mercury column barometer can be very accurate. However, for household use where accuracy is less critical, the aneroid barometer is more practical.

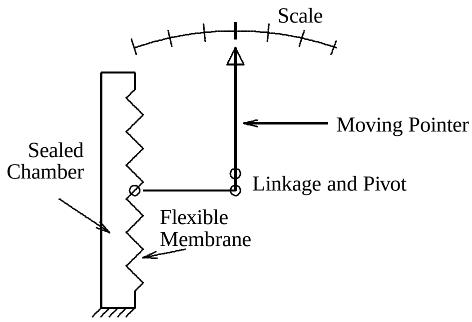


Figure 5.3: Aneroid Barometer

As shown schematically in figure 5.3, the sensing element of the barometer is a sealed chamber is equipped with a flexible membrane. As the atmospheric pressure changes, the gas in the chamber increases or decreases in volume. The resultant slight movement of the membrane is mechanically amplified and causes a pointer needle to move.

A typical aneroid barometer scale (unrolled) is shown in figure 5.4.

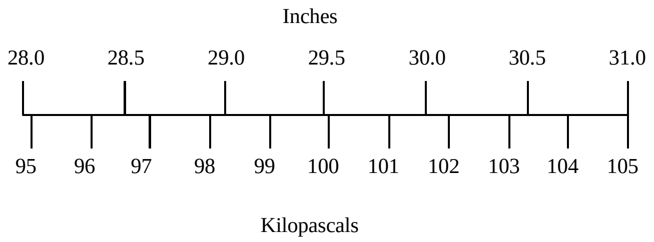


Figure 5.4: Barometer Scale

We will use this in section 5.4.3 to determine the requirements for an electronic barometer.

It is also interesting to consider the absolute extremes of measured air pressure, to see if our instrument can cope with them. According to the Canadian Encyclopedia [1], the air pressure extremes for Canada and the world are:

Measurement	**Canadian Record**	**World Record**
Maximum	106.7 kPa, Mayo, Yukon Territory, 1 Jan 1984	108.38 kPa, Agata, Siberia, 31 Dec 1969
Minimum	94.02 kPa, St Anthony, Nfld, 20 Jan 1977	87.64 kPa, Eye of Pacific Ocean typhoon June, 19 Nov 1975

Table 5.1: Air Pressure Records

Notice that the record high pressures tend to occur in artic regions, where the air mass is cold and therefore dense. The record low occurred in the eye of a hurricane.

Domestic barometers cannot normally cope with these extremes of air pressures, but we should design for them if the cost penalty is not severe.

5.3 Air Pressure and Altimetry

Air pressure decreases with height, an effect that is used by aircraft altimeters. If the barometer is sensitive enough, a change of altitude by a known amount (an elevator ride, for example) may be used to calibrate the barometer.

First, we need to know the air density, which is given by:

$$\rho = \frac{p}{R \cdot T} \tag{5.7}$$

where

ρ = density of air, grams/cm^3
P = air pressure, dynes/cm^2 (or Kilopascals $\times$ 1000)
R = gas constant for air, 2.87×10^6 cm^2/sec^2 ^{o}C
T = air temperature, ^{o}K

Then the change in air pressure is

$$\Delta P = g \cdot \rho \cdot \Delta H \tag{5.8}$$

where

ΔP = change in air pressure, millibars
g = gravitational constant, 981 cm/sec^2
ΔH = change in height, cm

Example

Find the change in air pressure over a change in height of 30 metres if the air temperature is 20^{o}C and the pressure 1013 kPa.
Solution
From 5.7 above,

$$\begin{aligned} \rho &= \frac{1013 \times 1000}{2.87 \times 10^6 \times (273 + 20)} \\ &= 1.2 \times 10^{-3} \text{ gms/cm}^3 \end{aligned}$$

Then, substituting for ρ in equation 5.8,

$$\begin{aligned} \Delta P &= 981 \times (1.2 \times 10^{-3}) \times (30 \times 1000) \\ &= 3532 \text{ dynes/cm}^2 \\ &= 3.53 \text{ millibars} \end{aligned}$$

5.4 Electronic Measurement of Air Pressure

Electronic pressure sensors are used in great numbers in automobile engine control systems. As a result, suitable air pressure sensors have become available at very reasonable cost.

The design shown here is based on the Motorola MPX100AP sensor, a sensor for absolute pressures between 0 and 1000 millibars[1](figure 5.6). The pressure sensor is essentially a miniature aneroid barometer. The membrane is a thin silicon diaphram into which has been diffused a network of four resistors in a bridge configuration. The resistors function as sensitive strain guages, changing resistance as atmospheric pressure deforms the diaphram, figure 5.5.

The resistance of a conductor is given by

$$R = \rho \frac{l}{A} \tag{5.9}$$

where

[1]We require a maximum pressure measurement of 1050 millibars, which exceeds the maximum rating of the sensor by some 5%. As we will see in the design notes, using a sensor rated for higher pressure would halve the sensor electrical output signal and require double the voltage gain from the sensor amplifi er. Even the 5% overload is well below the maximum rating of the MPX100AP (2000 millibars), so we are in no danger of damaging the sensor. Hopefully, its output remains linear in the 5% overload region.

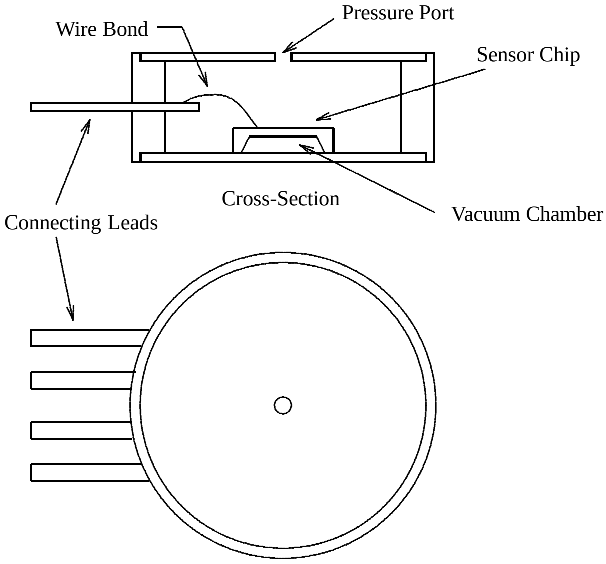


Figure 5.5: Pressure Sensor

ρ	is the resistivity of the conductor
l	is the length
A	is the cross-sectional area

When a resistor is stretched, its length increases and cross-sectional area decreases, both increasing the resistance. In most conductors, this effect is very slight. In the pressure sensors, the resistors are constructed of semiconductor material that shows large changes with small deformations.

When configured as a strain guage bridge, the resistors are located so that diagonally opposite resistors in the bridge change resistance in the same direction, either $R(1+\Delta)$ or $R(1-\Delta)$.

The differential output voltage is then simply $V_{24} = V_{cc} \times \Delta$.

Example

For the MPX100AP sensor, $R = 500\Omega$ and $V_{cc} = 3V$.
If the maximum differential output V_{24}, at full pressure, is 0.06 volts, determine the corresponding values of the bridge resistors at full pressure.

Solution:
From

$$V_{24} = V_{cc} \cdot \Delta$$

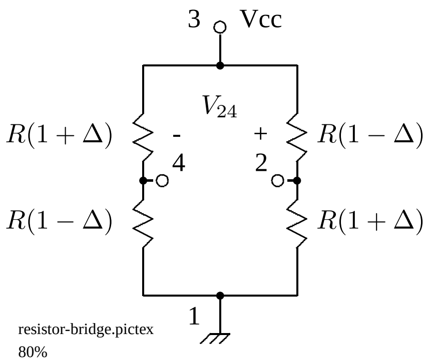


Figure 5.6: Resistor Bridge

so

$$\begin{aligned} \Delta &= \frac{V_{24}}{V_{cc}} \\ &= \frac{0.06}{3} \\ &= 0.02 \end{aligned}$$

Then

$$\begin{aligned} R(1+\Delta) &= 500 \cdot (1+0.02) \\ &= 510\Omega \end{aligned}$$

and

$$\begin{aligned} R(1-\Delta) &= 500 \cdot (1-0.02) \\ &= 490\Omega \end{aligned}$$

5.4.1 Sensor Specifications

The key specifications for the MPX100AP pressure sensor are as follows:

Notes:

Pressure Range We will exceed the maximum pressure slightly to 105kPa. This is still well below the burst pressure.

Supply Voltage If the three volt supply is obtained by dropping 2 volts across resistors in series with the sensor, it turns out that the temperature drift of the sensor is substantially reduced (reference [80]).

	Minimum	Typical	Maximum	Unit
Burst Pressure	-	-	200	kPa
Pressure Range	0	-	100	kPa
Supply Voltage	-	3.0	6.0	Volts
Supply Current	-	6.0	-	mA
Full Scale Span	45	60	90	mV
Offset	0	20	35	mV
Sensitivity	-	0.6	-	mv/kPa

Table 5.2: Pressure Sensor Specifications

Full Scale Span From these figures, we can determine that the sensor gain varies from 0.45 to 0.9mv/kPa.

Offset This voltage is caused by mismatch of the bridge resistors and appears as a fixed voltage at the output of the sensor.

Supply Current This figure enables us to determine that the bridge resistors are nominally 500Ω.

Sensitivity K_T This is a somewhat redundant statement of the transducer gain, which we determined from the full scale span specification

5.4.2 Barometer Design Issues

There are a number of design challenges which need to be addressed in this system:

Power Supply The available power supply is +5 volts. Either the interface circuit operational amplifier must operate from this or a converter must be available to generate the usual positive and negative voltages for the op amp.

The latter approach requires a DC-DC converter and some method of ensuring that the output of the op amp does not exceed the 0-5 volt range of the HC11 A/D converter input.

Ever mindful of cost, we've chosen the single supply approach. For the DC-DC converter approach, see [87].

Amplifier Output Swing The output voltage of a single supply bipolar op amp such as the National LM324 or Motorola MC34074 is very limited: 0.5 to 3.5 volts when operated from a +5 volt power supply. Some CMOS op amps, such as the National LMC660CN, will produce a larger output swing. The data sheet for the LMC660CN shows 0.2V to 4.7V for a +5 volt supply and load greater than 2KΩ, so this is a suitable amplifier for the pressure sensor interface.

Temperature Drift A back-of-the-envellope calculation shows that the sensor amplifier will require a voltage gain in the order of 300V/V or so. Any drift in offset voltage, bias, power supply or resistance values has the potential for being amplified by this large gain to appear as drift in the output voltage. As well, the sensor itself is sensitive to temperature.

All of these temperature effects must be checked to ensure that the circuit functions as a barometer rather than a thermometer. References [78] and [79] mention the problem of temperature drift, a warning that it must be taken seriously.

As well as designing for low temperature drift, we should choose the minimum voltage gain that meets the requirements, thereby reducing the effect of resistor and voltage drifts.

Calibration The Pressure Sensor Specifications shown in Table 5.2 on page 168 show that the sensor gain can vary over a 2:1 range, so some sort of calibration procedure will be required. The usual approach is to provide two potentiometers: one for gain and the other for offset. Potentiometers are inherently undesireable in a production design. The part cost is higher than a fixed resistor and a pot requires human intervention for adjustment. It is preferable, if at all possible, that adjustments be done in software.

Subtraction of Bias and Offsets Referring to the Barometer Scale shown in figure 5.4 on page 164, the interesting part of the air pressure signal is a 10kPa variation sitting on top of a 100kPa constant pressure. The constant pressure must be subtracted at some point. As well, the output of the pressure sensor bridge is a differential signal sitting on a half-supply common mode signal. The common mode signal must be ignored, so the sensor amplifier must be differential and have a satisfactory common mode rejection ratio.

5.4.3 Barometer Resolution and Dynamic Range

An early and critical decision is the resolution of the barometer, in units of A/D counts per kilopascal of pressure. We'd like a large resolution in order to detect small changes in air pressure. However, higher resolution requires higher voltage gain from the interface and consequent greater sensitivity to a variety of nasty drift signals. Our philosophy should therefore be to make the resolution no higher than necessary.

The face of an aneroid barometer is typically divided into 60 divisions [77] and weather broadcasts are typically given to the nearest tenth of a kilopascal. This would imply 100 steps over the 10kPa variation in air pressure. Therefore, we might fix on 1 part in 100 as a suitable target for resolution.

A suitable dynamic range, referring to figure 5.4 on page 164, might be 95 to 105 kPa. This does not cope with the extremes of pressure shown in table 5.2 on page 168, but will do for routine operation.

5.4.4 Transfer Function

It is useful to characterize the fixed component of air pressure, 100kPa, as P_{ref}, which creates a fixed component of voltage V_{ref} at the input to the microcomputer A-D converter. The variation in air pressure is $\pm\Delta P_a$ around P_{ref}, creating a variation in A-D voltage of $\pm\Delta V_{AD}$ around V_{ref}.

The value of ΔV_{AD} is the product of the resolution, previously fixed at 100 steps, and the voltage per step, 19.5 mv/step, for a 5 volt, 8 bit A-D converter.

Then

$$\begin{aligned}\Delta V_{AD} &= 100 \times 19.5 \times 10^{-3}/2 \\ &= 1.95/2 \\ &= 0.975 \text{ volts}\end{aligned}$$

We'll round this off to ± 1.0 volts.

Now we can fix V_{ref}. It must be large enough that the amplifier doesn't exceed its maximum or minimum output voltages. A good choice is 2.5 volts, halfway between 0 and 5 volts.

With this information, we can draw the transfer function, shown in figure 5.7.

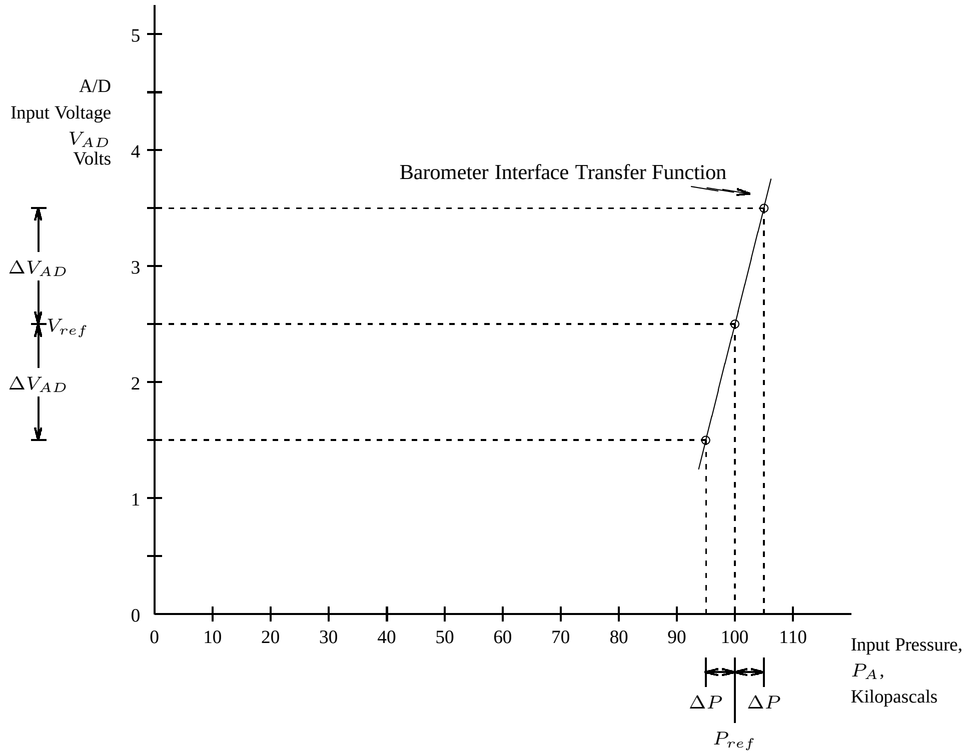


Figure 5.7: Barometer Interface Transfer Function

Substituting two point coordinates in the straight line equation $y = mx + b$, we can solve for m and b, determining that the transfer function is

$$V_{AD} = 0.2P_A - 17.5 \tag{5.10}$$

where

V_{AD}	is the input voltage to the A-D converter
P_A	is the air pressure in kilopascals

Translating the interface transfer function into a block diagram, we have figure 5.8.

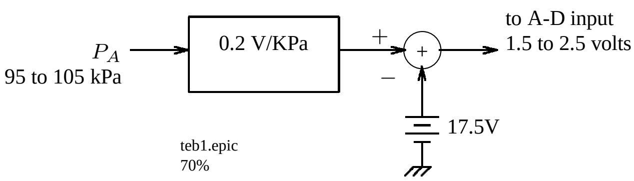


Figure 5.8: Barometer Interface Block Diagram

The typical sensor gain K_T (table 5.2 on page 168) is 0.6×10^{-3}, so the amplifier gain must be $0.2/0.6 \times 10^{-3} = 333$ volts/volt as shown in figure 5.9.

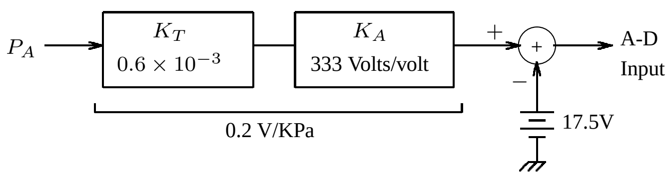


Figure 5.9: Interface Block Diagram, Adding Sensor

There are two practical problems with figure 5.9.

- The 100kPa pressure P_{ref} tries to generate a 20 volt signal at the output of the amplifier K_A. This will saturate the amplifier, since it is operated from a +5 volt supply.
- The 17.5 volt offset is difficult to generate in a +5 volt system.

The solution to both problems is to divide the gain K_A into two roughly equal stages, K_{A1} and K_{A2}, as shown in figure 5.10.

In this case, the offset voltage V_{OS} is +1.0 volts, easily generated from +5 volts. (Henceforth, for clarity, we shall rename K_{A2} to K_{OS}, the offset gain).

5.4.5 The Barometer Circuit

The final circuit is shown in figure 5.11.

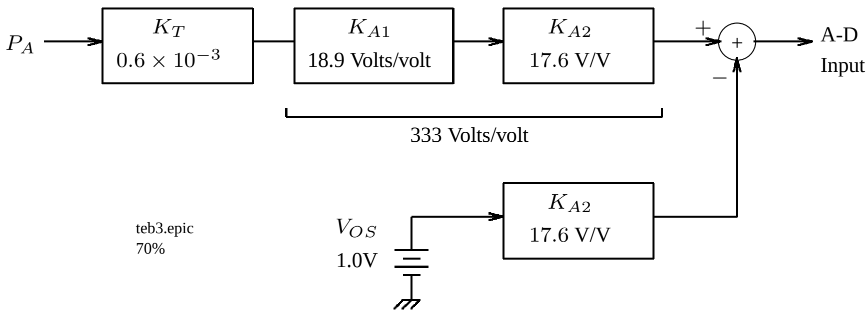
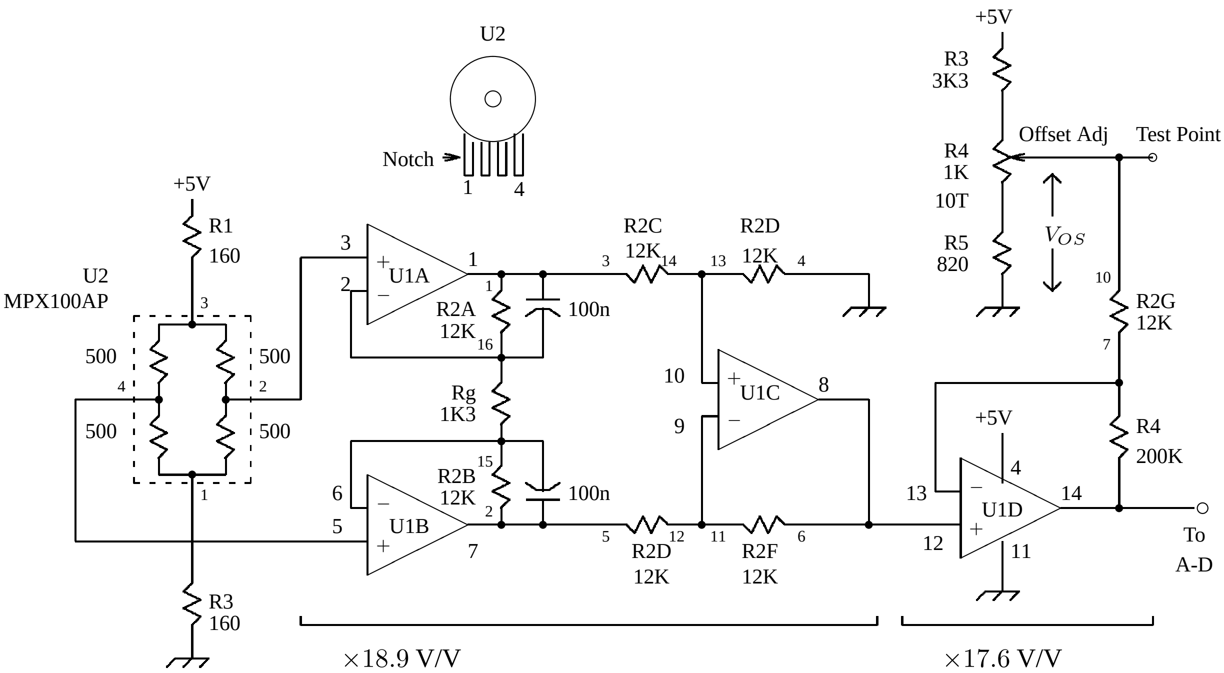


Figure 5.10: Interface Block Diagram, Splitting the Gain



U1	National Semiconductor LMC660CN
U2	Motorola MPX100AP, absolute pressure, hose port
U3	Bourns 4116R-001-123 Resistor Array
	All 12K resistors are part of U3. Small numbers are DIP package pin numbers.
	Discrete resistors must be low temperature coefficient, Philips MRS 25F series or equivalent.
R4	Cermet 10 turn pot, Bourns 3299-1-102 or equivalent.
	Change Rg to alter gain.
R1,R3,Rg	160Ω, 1/4 watt, 1%

Figure 5.11: Barometer Interface Schematic

From the data sheet for the sensor, the resistors in the sensor are 500Ω each and 3 volts should appear across the pressure sensor. Then resistors R_1 and R_2 are 160Ω each.

The instrumentation amplifier, U1A, U1B and U1C provides a high impedance input for the pressure sensor signal. The voltage gain is given by

$$K_{A1} = 1 + \frac{2R_2}{R_g} \tag{5.11}$$

and is set to 18.9 volts/volt. Somewhat arbitrarily, we have chosen R2 as 12KΩ, which makes R_g

$$\begin{aligned} R_g &= \frac{2R_2}{K_{A1} - 1} \\ &= \frac{2 \times 12000}{18.9 - 1} \\ &= 1341\Omega \end{aligned}$$

The nearest standard 5% value is 1300Ω.

If the gain needs to be adjusted, R_g should be changed. To simplify calibration, it should be a fixed resistor, not variable.

In addition to providing voltage gain, this stage removes the 2.5 volt common mode sensor voltage.

The second stage, U1D, subtracts the offset and provides a final gain of 17.6 volts/volt. It could have been implemented with a differential amplifier of gain $\times 17.6$, as shown in figure 5.12.

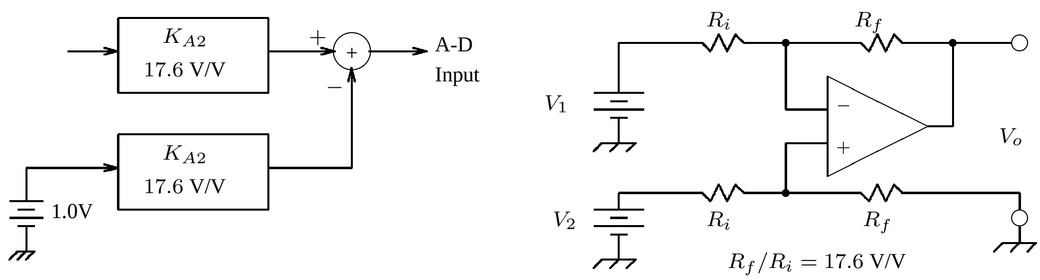


Figure 5.12: Second Amplifier Stage, Differential

However, if we're willing to tweak the voltage at the non-inverting input, we can simplify the circuit as shown in figure 5.13.

By superposition, the output voltage is

$$kv_o = -\frac{R_f}{R_i}v_1 + (\frac{R_f}{R_i} + 1)v_2 \tag{5.12}$$

where $R_f = 200K$ and $R_i = 12K$ to obtain a non-inverting gain of 17.6V/V and an inverting gain of 16.6V/V for this circuit. Then V_1, the offset voltage, should be set to $18/16.6 = 1.084$ volts.

The final block diagram is shown in figure 5.14.

For good common mode rejection, the resistors of the differential stage U1C are from a resistor array. All other resistors must be low temperature film, ± 50ppm temperature coefficient. The offset pot should be cermet for low drift.

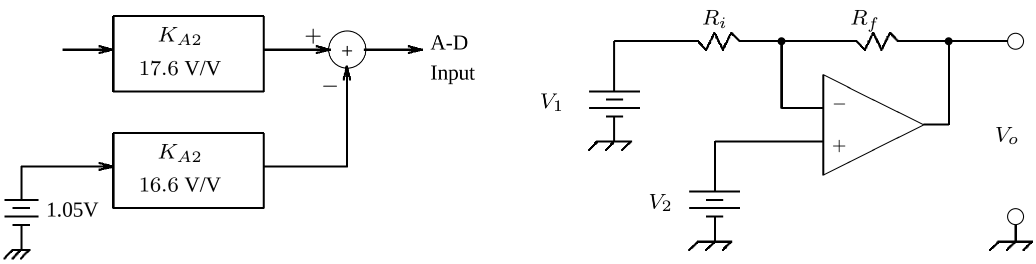


Figure 5.13: Second Amplifier Stage, Simplified

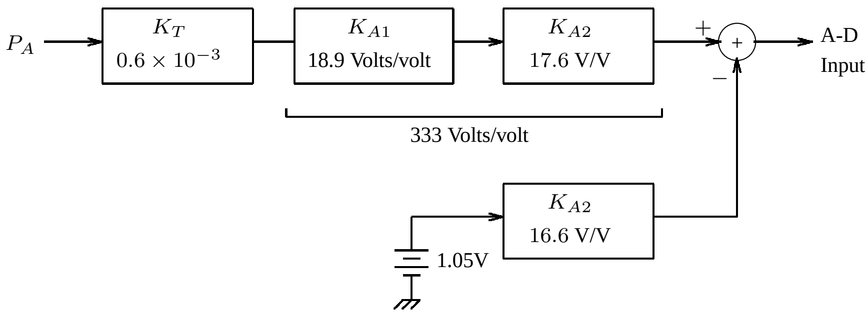


Figure 5.14: Complete Block Diagram

5.4.6 Barometric Software

In section 5.4.4, we established the transfer function of the system, relating the atmospheric pressure P_A to the input voltage of the A-D converter, V_{AD}. Equation 5.10 was given as:

$$V_{AD} = 0.2P_A - 17.5 \text{ volts/kPa} \tag{5.13}$$

We may rewrite this as

$$V_{AD} = K_{TF}P_A - V_{OS}K_{OS} \text{ volts/kPa} \tag{5.14}$$

where

K_{TF}	is the gain of the transfer function in volts/kilopascal
K_{OS}	is the offset gain, in volts/volt
V_{OS}	is the offset voltage, in volts

The transfer function gain K_{TF} is the product of two components: the sensor (transducer) gain K_T, and the amplifier gain K_A. Then substituting K_TK_A for K_{TF} in equation 5.13,

$$V_{AD} = K_TK_AP_A - V_{OS}K_{OS} \text{ volts/kPa} \tag{5.15}$$

We have one final constant to consider. Ultimately, we would like to relate the A-D reading, N_{AD}, to air pressure. We do this with the relationship

$$V_{AD} = N_{AD}K_S \tag{5.16}$$

where K_S is the step size of the A/D, in volts. For an 8 bit A-D converter operated from a 5 volt supply, the value of K_S is $5/256 = 19.5 \times 10^{-3}$ volts.

Substituting $N_{AD}K_S$ for V_{AD} in equation 5.17 we have

$$N_{AD}K_s = K_T K_A P_A - V_{OS}K_{OS} \text{ volts/kPa} \tag{5.17}$$

Solving for atmospheric pressure P_A we obtain the equation that the software must use to find atmospheric pressure:

$$P_A = \frac{N_{AD}K_s + V_{OS}K_{OS}}{K_T K_A} \tag{5.18}$$

where

P_A	is the air pressure, in kilopascals, to be calculated by the computer
N_{AD}	is the A-D reading
K_S	is the step size of the A-D converter, in volts. In this system, it is 19.5×10^{-3} volts
V_{OS}	is the offset voltage, in volts This value will depend on the offset value that is adjusted into the hardware to set the 100 kPa output to 2.5 volts, and is set at calibration.
K_{OS}	is the offset gain, set at 16.6 volts/volt.
K_T	is the gain of the pressure transducer, nominally 0.6×10^-3 volts/kilopascal, set precisely at calibration.
K_A	is the gain of the amplifiers in the interface, about 333 volts/volt for this system.

The transducer gain K_T and the offset voltage V_{OS} must be determined in order that equation 5.18 contain sufficient information that the computer program can relate A-D reading N_{AD} to air pressure P_A.

5.5 Barometer Calibration

In this section, we develop methods of calibrating the electronic barometer. We shall look at two manual methods of calibration, and then an automatic method that eliminates the offset potentiometer and the need for human intervention in adjusting it.

5.5.1 Approximate Method

Calculating Sensor Gain K_T

In this method, set the potentiometer R4 so that the voltage into the A-D converter is within its operating range. Then measure the offset voltage V_{OS} at the test point shown on figure 5.11, and the current reading of the A-D converter N_{AD}. (You can obtain this from the microprocessor or from the voltage V_{AD} into the A-D).

The the current value of air pressure P_A may be obtained from a weather broadcast. The values of amplifier gain K_A and offset gain K_{OS} are known, since they are determined by fixed resistor ratios.

This is sufficient information that equation 5.18 may be used to solve for the unknown variable, transducer gain K_T.

Setting Offset Voltage V_{OS}

Once K_T is known, the offset voltage V_{OS} may be adjusted to its correct value. To do this, again use equation 5.18. This time substitute the newly-found value for transducer gain K_T together with the known values for step size K_S, amplifier gain K_A and offset gain K_{OS}.

We also know that an air pressure of 100kPa corresponds to an A/D input count of 125 (halfway between 0 and 255).

5.5.2 Accurate Method

The calibration method of section 5.5.1 is only approximate because it assumes amplifier and offset gain values, based on nominal resistor values. A more accurate method of determining transducer gain is to apply a known change in pressure ΔP_A to the sensor and observe the corresponding change in A-D input voltage, ΔV_{AD}. The ratio of these two is the slope of the transfer characteristic:

$$\frac{\Delta V_{AD}}{\Delta P_A} = K_T K_A \tag{5.19}$$

A suitable apparatus for generating a known change in pressure is shown in figure 5.15. The liquid is water, laced with red food colouring to make it visible. The tubing is flexible plastic hose available from the local hardware store. The hose is filled with water so that it forms a U shape.

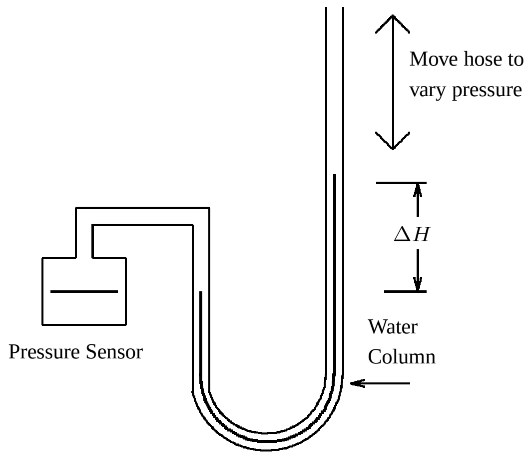


Figure 5.15: Water Manometer

The right side of the manometer is raised or lowered to create a height differential of ΔH. The resulting pressure may be determined from figure 5.2 on page 163. For example, a pressure differential of 2 kilopascals may be created by a height differential of

$$\begin{aligned}\Delta H &= \frac{5}{101.3} \times 160.2 \\ &= 7.9 \text{ cm}\end{aligned}$$

Once the transducer-amplifier gain $K_T K_A$ is determined, the current air pressure P_A and corresponding A-D reading N_{AD} may be used in equation 5.18 to solve for the offset term $V_{OS}K_{OS}$. Finally, the offset voltage V_{OS} may be set as in the approximate procedure.

Once the values of the various terms in equation 5.18 are known, they may be entered into the computer equation that displays the current air pressure.

5.5.3 Automatic Calibration

If the current air pressure is known and the pressure interface is constructed with fixed resistors, then it should be possible for the microprocessor to read the A-D converter and determine the calibration constants automatically. This need only be done once: the constants are written into EEPROM and are not changed unless the system is re-calibrated.

Unfortunately, there is a problem. The large variation in sensor gain coupled with the high gain of the amplifier section will cause the amplifier to saturate or cutoff unless the offset is adjusted correctly. In section 5.5.1 the operator did this manually.

If the microprocessor can be provided with the means to adjust the offset so that the amplifier is operating in its linear range, the microprocessor can determine the calibration constants for its computer program. This may be accomplished by a D-A converter, controlled by the microprocessor, that generates the offset voltage V_{OS}. It turns out that modest resolution is acceptable. As a result, the D-A converter circuit is quite simple and may be driven by a microprocessor parallel port.

The D-A Converter

The schematic of a suitable type of D-A converter, a voltage switching converter, [85] is shown in figure 5.16.

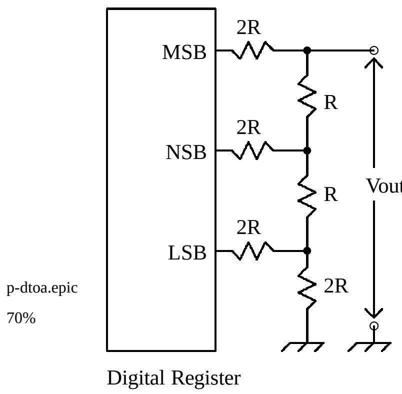


Figure 5.16: Digital to Analog Converter

If the MSB and NSB are both at 0 volts and the LSB is at +5 volts, we may determine the effect on V_{out} by repeatedly applying Thevenin's theorem to the divider circuit. Then $V_{out} = 0.625$ volts. Similarly, the NSB contributes 1.24 volts and the MSB 2.5 volts. If all the digital bits are set to logic 1 (+5 volts), then according to the Superposition Theorem, $V_{out} = 0.625 + 1.25 + 2.5 = 4.375$ volts. In other words, this D-A has a resolution of 0.625 volts and a range of 0 to 4.375 volts.

In general,

$$V_{out} = \frac{V_{logic}}{2}\left(N_2 + \frac{N_1}{2} + \frac{N_0}{4}\right) \tag{5.20}$$

where

V_{out}	is the output voltage of the D-A converter
V_{logic}	is the logic level into the D-A converter, 5 volts in this case
N_2, N_1 and N_0	are the MSB, NSB and LSB respectively of the binary number input to the D-A converter

Now we need to determine how many bits are required in the D-A converter for the pressure amplifier circuit.

D-A Converter Resolution

We can determine the required resolution of the D-A converter according to the following reasoning:

- The D-A will generate a signal V_{OS} that replaces the offset pot, which had a range of 0.75 volts to 1.65 volts.This is amplified by the offset gain K_{OS} (16.6 volts/volt) to shift the amplifier output signal V_{AD} up or down. The total range of shift is then $(1.65 - 0.75) \times 16.6 = 18.3$ volts. The step size of the D-A must be a small fraction of this 18.3 volts.
- The output swing of the amplifier worst case occurs for a maximum sensor gain K_T of 0.9mv/kPa. This is amplified by the forward gain of the amplifier K_A, 333. The output voltage swing for a full-scale change in air pressure ΔP_A of 10KPa is then

$$\begin{aligned} \Delta V_{AD} &= \Delta P_A \times K_T \times K_A \\ &= 10 \times 0.9 \times 10^{-3} \times 333 \\ &= 2.97\text{volts} \\ &\approx \pm 1.5\text{volts} \end{aligned}$$

- We would like to locate the output of the amplifier so that the signal never swings below 0.2 volts or above 4.7 volts. This allows a guard band of 0.75 volts above and below the output swing. We might somewhat arbitrarily choose to be able to place the offset signal with a resolution of half this, 0.375 volts.
- The required resolution of the D-A is in then the order of one part in $18.3/0.375 = 48.8$. The next larger binary number is $2^6 = 64$, so we require a 6 bit D-A converter.

This resolution is low enough that discrete 1% resistors (20KΩ and 10KΩ for example) may be used for the R-2R resistor ladder.

CMOS Output Specifi cations

The usual approach to D-A design is to have the logic signals switch an accurate, stable reference voltage. However, the accuracy required of this D-A converter (1 part in 64, 1.5%) may be low enough that a CMOS latch may be used to drive the ladder directly. This would greatly simplify the circuit design.

The output logic swing from CMOS logic (unlike the TTL family) is very nearly equal to the power supply levels: 0 and +5 volts in this case. The data dook for Texas Instruments 74HC logic ([86]) shows a typical output logic swing to within 1 millivolt of the supply levels. For the worst case, the output swing is still to within 10 millivolts of the supply levels. This suggests that the CMOS latch will drive the R-2R ladder with sufficient accuracy.

Furthermore, the output resistance of the logic, worst case, is given in the data book as 50Ω. If we use 10kΩ and 20kΩ as resistors in the R-2R ladder network, then the driving resistance of the logic device will be much lower than the resistance of the ladder network, so resistive loading will not be a problem.

Based on these specs, a CMOS latch such as the 74HC273 can drive the 6 bit ladder network directly.

D-A Amplifi er

Analysing the output of a 6 bit D-A converter as we did in section 5.5.3 , using equation 5.20 on page 179 (modified for 6 bit input) we can determine that the output of the 6 bit D-A converter ranges from 0 to 4.92 volts in steps of 0.078 volts. The barometer interface circuit requires that the offset voltage V_{OS} vary between 0.75 volts to 1.65 volts, so amplification (actually, attenuation) and level shifting are required. The amplifier that provides this level shifting and attenuation also serves to buffer the internal resistance of the D-A from its load.

A non-inverting amplifier won't work, since the required gain is less than unity. The gain of an inverting amplifier may be set to anything from zero up[2], and the sign inversion introduced by the inverting amplifier may be taken care of in the software.

The transfer function of the amplifier circuit is shown in figure 5.17, from which the slope **m** and offset **b** may be determined.

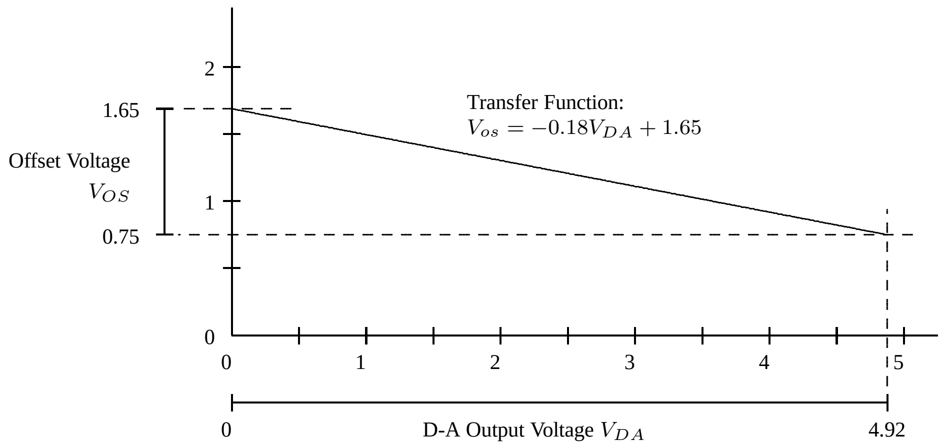


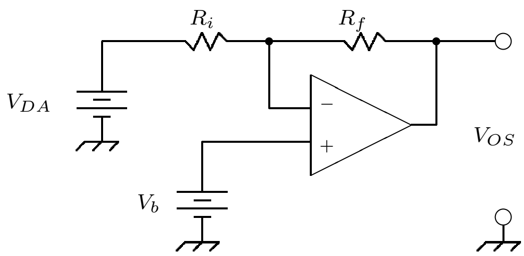
Figure 5.17: D-A Amplifier Transfer Function

The circuit of the amplifier is shown in figure 5.18.

By comparison between the equation of the transfer function the equations for the amplifier, we have that

$$\frac{R_f}{R_i} = 0.18$$

[2]Well, actually, up to the open loop gain of the op-amp, to be precise.



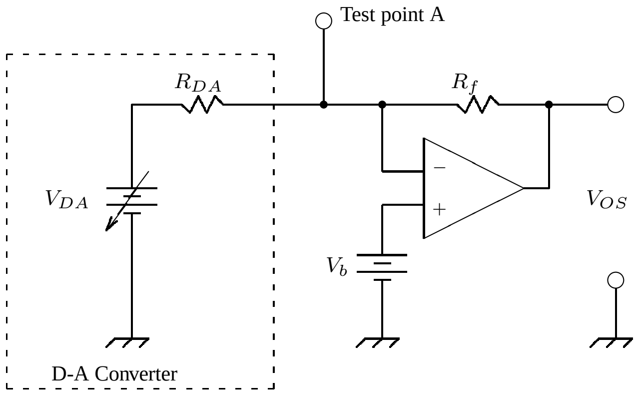
$$V_{OS} = -V_{DA}\frac{R_f}{R_i} + V_b(1 + \frac{R_f}{R_i})$$

Figure 5.18: D-A Amplifier

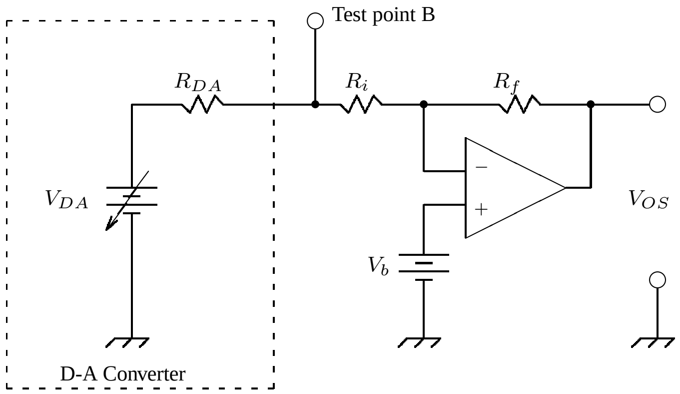
and

$$\begin{aligned} V_b &= 1.65/\left(1 + \frac{R_f}{R_i}\right) \\ &= 1.39 \text{ volts} \end{aligned}$$

Looking back into the D-A converter output, the load sees an internal D-A resistance R_{DA} of **R** ohms. This internal resistance could be used as R_i, (figure 5.19A).



(a) D-A, Direct Connection to Amplifi er



(b) D-A, With R_i

Figure 5.19: D-A Amplifier Connection

However, in this arrangement there is no voltage signal representing the D-A output voltage by itself: the test point **A** is a virtual earth. For troubleshooting purposes, it is better to provide an extra resistance and test point **B** (figure 5.19B) where the output of the D-A can be measured. The voltage at **B** will be half the open circuit D-A voltage V_{DA}, but can be used to indicate that the D-A is operating correctly.

Automatic Calibration: Schematic

The complete schematic of the system, including a D-A converter for automatic calibration, is shown in figure 5.20.

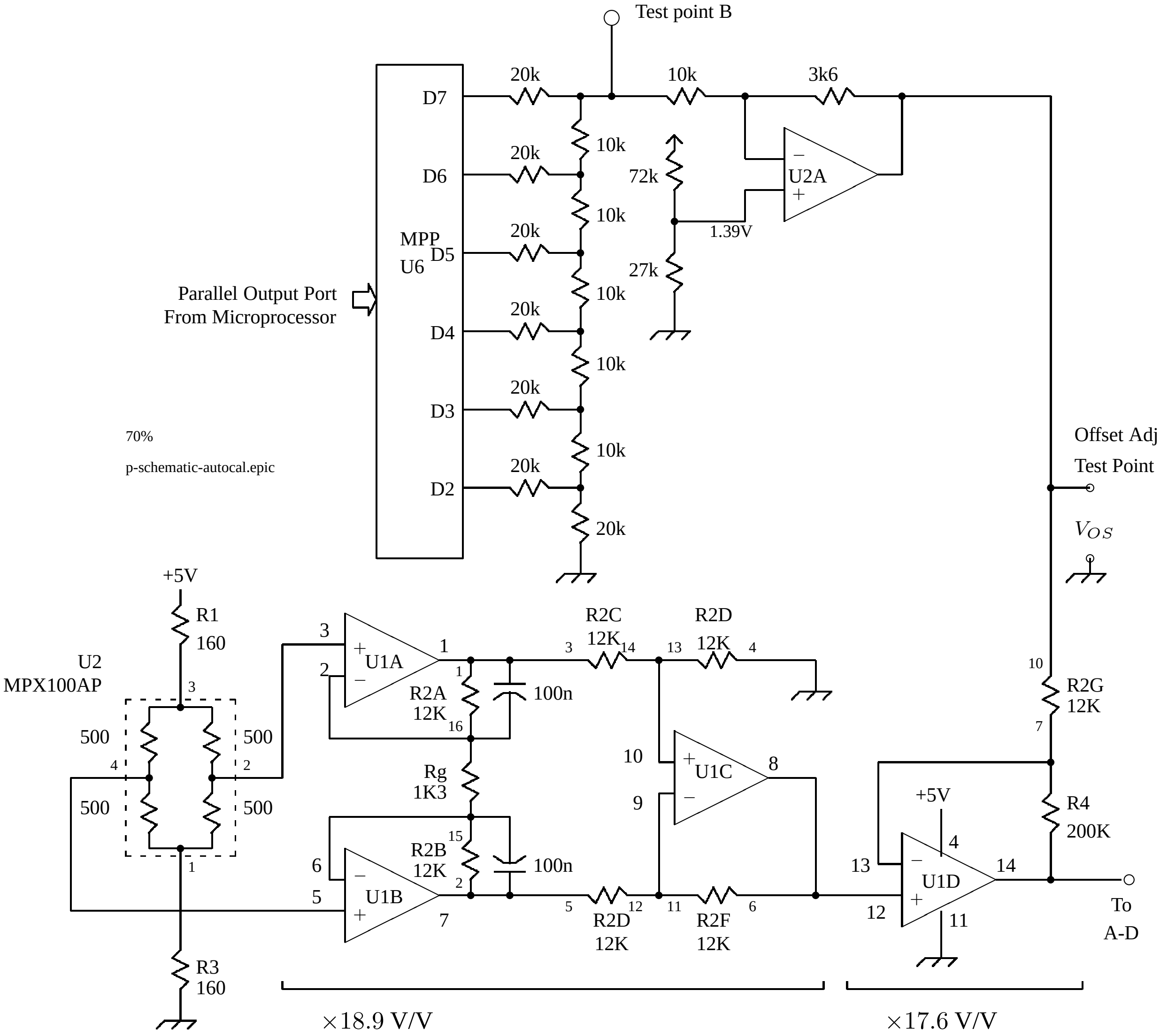


Figure 5.20: Barometric Pressure Interface, Automatic Calibration

Automatic Calibration: Algorithm

The microprocessor essentially mimics the manual calibration process of section 5.5.1 on page 176. The only information it needs from the external world is the current value of atmospheric air pressure P_A.

1. When the microprocessor first reads the A-D input voltage V_{AD}, it will probably be near +5 volts or 0 volts. The processor then monitors that voltage while increasing the count into the D-A converter, which causes the offset voltage to ramp between its maximum and minimum values. At some point, the A-D input voltage should move to a value near the centre of the A-D input range. At this point, the micro freezes the value in the D-A register. Because it knows the constants relating D-A count and offset voltage, the micro now knows the value of the offset voltage V_{OS} that moves the interface into its linear region.

2. The micro uses the current value of air pressure P_A with the known values of amplifier gain K_A, offset gain K_{OS} and the offset voltage V_{OS} that it set in the previous step, in equation 5.18 (page 176) to calculate the sensor gain K_T.

3. The micro now adjusts the D-A output so that the offset voltage V_{OS} is at such a value that an air pressure P_A of 100kPa would cause an A-D input voltage V_{AD} of 2.5 volts.

4. The micro stores the current values of the interface constants K_A, K_{OS}, V_{OS} and K_T in semi-permanent EEPROM memory. The interface is now calibrated and A-D readings can be used to calculate and display the current air pressure.

An assembly line production would use this process. An external control computer would download a calibration program and the current air pressure into the microprocessor. The calibration would take place without human intervention.

The calibration program should also have the capability for detecting that calibration did not occur properly. Then a failed production unit can be shunted into a reject bin for rework.

5.6 Reliability of the Design

Now that we have a circuit design, we must ensure that the circuit will work reliably, allowing for component tolerances and the effect of temperature induced drift of the components.

For example, the sensor constant can vary over a range of 2:1. Resistors have a tolerance of $\pm$5%. The operational amplifiers have offset voltages which can vary by $\pm$7mV. Can we be sure that the circuit will work when components of the tolerance extremes are used in the circuit?

The sensor has a temperature coefficient of -0.16% per degree C, the resistors change by 250ppm (parts per million) per degree C, and the amplifier offset voltages may change by as much as $\pm 30\mu$volts per degree C. What effect will these drifts have on the operation of the barometer, bearing in mind that the circuit is not supposed to act as a thermometer?

We can and should build and test one or more prototypes. However, the correct functioning of a prototype is a necessary but not sufficient condition to determine a reliable design. The fact that a prototype works merely means that at least one version of the circuit will function. It's no guarantee that all circuits will function.

To ensure the reliable operation of the circuit, the correct strategy is to perform an engineering analysis, checking circuit operation by calculation and simulation. This will provide the necessary confidence to build the circuit in quantity, and be assured that it will function under all specified conditions. Where possible, to ensure that some massive blunder has not occurred, the calculations should be checked against the prototype. If the calculations and simulation accurately predict the behaviour of the prototype, then we can have some confidence in the predictions.

5.6.1 Circuit Tolerance Analysis

The tolerances of the circuit components raise two concerns:

- Will the circuit function, or will some voltage or current run into saturation or cutoff?
- Can the circuit calibration procedure compensate for circuit tolerances, or do we lose measurement accuracy under some conditions?

This is potentially an unweildy problem, because of the combinatorial explosion of tolerance variables. The parameters and equations of a spreadsheet model for the manual offset adjustment version of the interface are shown in figure 5.21. A typical spreadsheet printout is shown in figure 5.22.

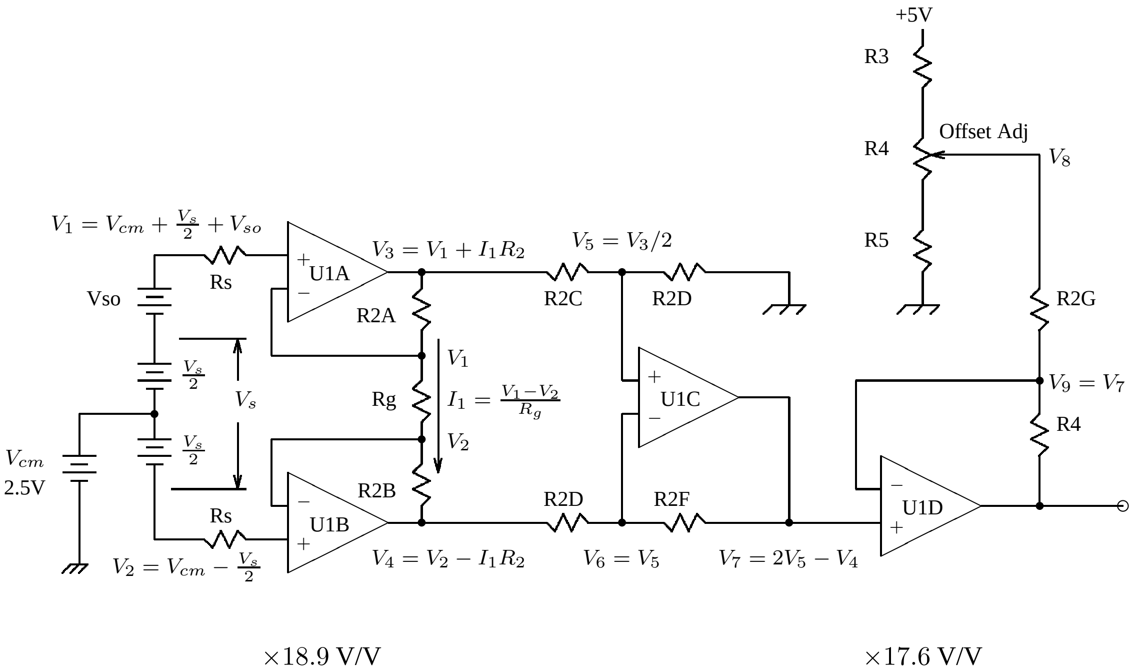


Figure 5.21: Circuit Equations

The tinkering with spreadsheet model turned up the following results:

- The sensor offset V_{so} does not cause the amplifier to saturate but does have a dramatic effect on the output voltage V_{AD}. (Notice that sensor offset V_{so} is a property of the sensor: do not confuse it with the offset voltage V_{OS} of the interface.)

Sensor Constant	K_T	(data)	0.6×10^{-3}	V/kPa
Air Pressure	P_A	(data)	100	kPa
Sensor Offset	V_{so}	(data)	10×10^{-3}	V
Gain Resistor	R_g	(data)	1300	Ω
Resistor	R_2	(data)	12000	Ω
Sensor Output	V_s	$K_T P_A$	0.06	V
	V_1	$V_{cm} + \frac{V_s}{2} + V_{so}$	2.53	V
	V_2	$V_{cm} - \frac{V_s}{2}$	2.47	V
	I_1	$(V_1 - V_2)/R_g$	48	μAmp
	V_3	$V_1 + R_2 I_1$	3.084	V
	V_4	$V_1 - R_2 I_1$	1.916	V
	V_5	$V_3/2$	1.542	V
	V_6	V_5	1.524	V
	V_7	$2V_5 - V_4$	1.167	V
K_{os}	V_8	(data)	1.68	V
	V_9	V_7	1.167	V
Resistor	R_4	(data)	200000	Ω
Output Voltage	V_{AD}	$V_7(1 + \frac{R_4}{R_2}) - V_8 \frac{R_4}{R_2}$	2.00	V
A/D Reading	N_{AD}	$\frac{V_{AD}}{5} 256$	102	counts

Figure 5.22: Amplifier Spreadsheet Model and Results

- The offset voltage V_{OS} may be adjusted to compensate for the effect of sensor offset voltage, but a larger range of offset is required than that originally anticipated. The output voltage V_{AD} is very sensitive to the setting of V_{OS}, so the pot R_4 should be a multi-turn unit.
- A combination of large offset and high sensor gain K_T cause V_3, the output of U1A, to exceed 3.5 volts. This is the maximum output of the LM324 operational amplifier, and so an LMC660 is required.
- For low sensor gain, the change in A-D reading over the full range of air pressures (95 to 105 kPa) is over 60 counts, so the resolution is satisfactory even for low sensor gain.

These results could have been predicted from an analysis of the circuit, but the spreadsheet model makes it easy to explore the effect of a variety of options and combinations of parameters.

A circuit simulation program such as SPICE could also be used to analyse the circuit, and is a better choice where an accurate op-amp model is required. The spreadsheet model assumes ideal op amps. On the other hand, spreadsheet programs are readily available and easy to use.

5.6.2 Temperature Drift

In every engineering project, there is at least one killer problem which determines success or failure. It's important to identify the killer problem as early as possible. In this system, the killer problem is temperature drift.

There are three evident sources of temperature drift:

Pressure Sensor Drift An analysis of the pressure sensor temperature drift [80] in the circuit of figure 5.11 shows two competing effects: the gain of the sensor decreases with temperature, but this is partially compensated by in increase in bridge resistance. The net result is a coefficient of -0.16% per degree C.

The effect on the output is a change of A-D reading of about 1.6 counts/^{o}C. Over a $\pm 10^o$C temperature range, this is an error of 16 counts out of a total of 100, or a 16% error: not very acceptable. Fortunately, since it is a predictable effect, it may be compensated for by measuring the ambient temperature and modifying the sensor constant.

Offset voltage drift For the LMC660, the typical figure for offset voltage drift is given as 1.3μvolts per degree C. The spreadsheet model (or an algebraic analysis) turn up the result that this causes a drift of about 0.5 mv/^{o}C, much less than one count of the A-D converter. This can therefore be neglected.

Resistor temperature coeffi cients Most of the resistors (R2A through R2H) are on the same package and so can be expected to track in temperature. As well, the two amplifier gains are the result of ratios of resistance (equations 5.11 and 5.12), so the temperature coefficients may be expected to cancel. Simulation of resistor drift with the spreadsheet confirms this: the effects of resistor drift are small enough to be neglected.

Chapter 6

Software Design

The design and construction of reliable control software must be approached with the same respect and care as the design of hardware systems. It is usual to underestimate the difficulty of building a computer program. Programs often contain defects and bugs that are poorly understood and difficult to eliminate.

In this part of the book, we provide some general approaches and specific techniques that help organize the design of software and make the process of development more predictable and systematic.

The ideas of this section are applicable to any microprocessor, but the assembly language examples are specific to the Motorola 68HC11 microprocessor. For information on programming the 68HC11, consult [98].

6.0.3 The Complexity Monster

Programming in machine language is difficult. Unlike Pascal and even C, the structure of a machine language program is unconstrained and unstructured. Thus, any discipline or structure that is imposed on a machine language program originates in the design skills of the programmer: it is not encouraged by the features of the language.

This may be construed as an argument for using a structured language such as C for program development and indeed, C has many attractions. However, as the integrators of software and hardware, electrical engineers must be able to work at the machine language level to develop and test hardware and to build the structures that will support a high level language. Thus, although it is possible to minimize the use of C in program development, it is neither possible nor desirable to avoid assembly language entirely.[1]

To understand the complexity of assembly language, consider the diagram of figure 6.1.

Each block in this diagram represents a machine instruction. As the machine executes these instructions, branch instructions and jump instructions can transfer execution to other instructions.

The first program contains two instructions, which can transfer control only to each other. The second program contains three instructions, and there are 6 possible transfer paths. With three instructions, there are 12 paths, and so on. In general, for n instructions, there are $n \times (n-1)$ possible transfer paths.

In practice, only some of the instructions in a computer program transfer control, so this analysis overstates the case. However, it is true that the number of transfer paths increases very rapidly with the number of instructions.

Now, a moderate sized assembly language control program might contain two or three thousand machine instructions. For 3000 lines of code, there are some 9 million possible transfer paths. This enormous number of

[1] At Ryerson University, after years of teaching assembly language, we we ran two semesters of microprocessor programming based on the C progamming language. The results were unsatisfactory. Students had no understanding of the underlying hardware, and there was too much 'magic' in the program development process. We then reverted back to assembly language.

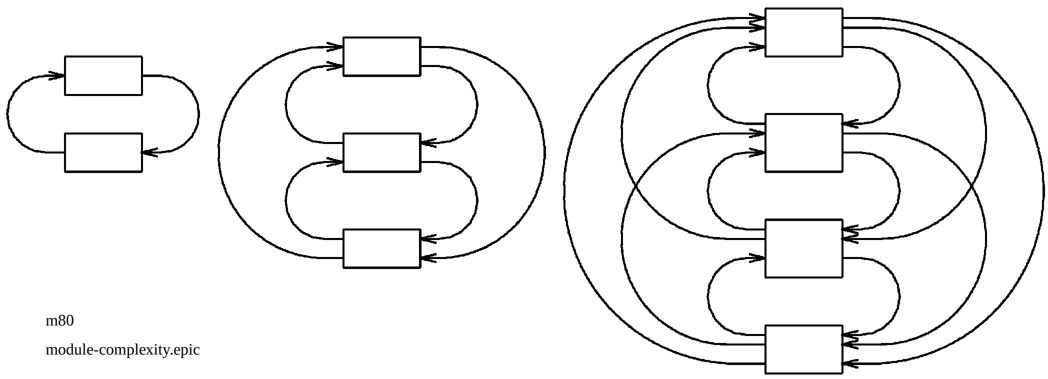


Figure 6.1: Program Complexity

possible control paths will completely overwhelm the ability of human intelligence to understand it.

The Complexity Monster is particularly insidious because it is not evident for small programs. Whatever their structure, small programs can eventually be debugged. However, as unstructured programs become larger, they suddenly overwhelm the designer and become impossible to troubleshoot and make reliable.

How do we manage this complexity? In overview, the solution is to clump the program into groups of instructions, which are then known as software modules. Then:

- Each module has a clearly defined function, rather like a giant machine instruction, and this makes it easier for the human mind to understand the larger structure.
- Control transfer takes place between modules rather than between individual instructions, thereby minimizing the number of possible transfer paths.
- The modules themselves are arranged in a heirarchy, in which a master routine calls the subsidiary modules.
- Each module may be individually tested as a small program, which makes its complexity manageable. Inside modules, the transfer of control is structured into a few carefully chosen patterns.

With this approach, bugs are minimized at the outset and a systematic procedure can applied to pinpoint errors and eradicate them entirely. Furthermore, extending the program becomes more routine and less of an exercise in brinkmanship. (Will this last feature break the program?)

However, structure does not arrive by accident and it cannot be applied after the fact: it has to be designed in from the beginning. In the next sections, we discuss how this process of software design may be accomplished.

6.0.4 Organization of a Control Program

Control programs are part of a larger system, accept sensor inputs, and generate the appropriate display and output control signals. Often, a control program operates under time constraints: it must update a control output within a certain time interval of the input signals. If the system is deterministic, that is, if its response time is predictable, then it is known as a real-time system.

The general organization of a control program is shown in figure 6.2.

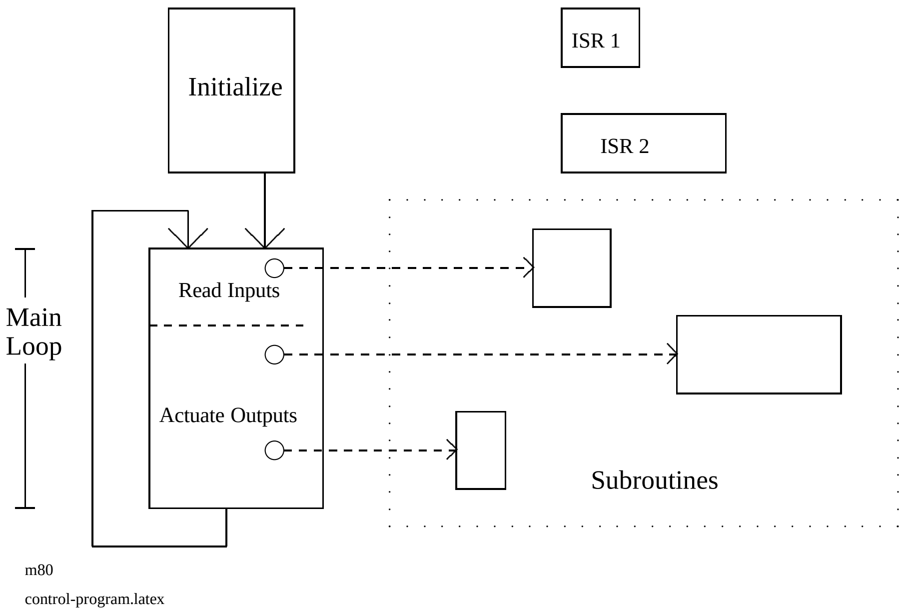


Figure 6.2: Control Program Structure

The program begins with an initialization section, in which the hardware is configured and variables initialized. Then it enters the main loop. The main loop gathers input information and then actuates outputs. In its simplest form, the main loop is a sequence of calls to subroutines. Usually there are conditional branches that determine which of the subroutines is actually called.

The main routine is the focus of control for the system, and it determines what processing takes place. It should be kept as simple and short as possible.

The actual processing is delegated to a number of subroutines, which may in turn call lower level routines.

The main routine may be executed frequently enough that external events can be detected reliably by routines called from the main loop. If an external event is not reliably detected from the main loop, it will be necessary to use an interrupt service routine, ISR to do this. The function of interrupt service routines is maintain a record of the occurrence of external events, not to do serious processing functions. The ISR simply communicates messages about events to the other routines in the program.

This simple model of a control program brings us to a number of useful design concepts.

Loop Timing It's quite easy to monitor the speed of this system. Simply cause an output signal to be generated with each trip around the loop (flip the polarity of an output line, for example) and monitor this signal with an oscilloscope. This is a useful indicator of the average speed of execution and, what is often more important, of deviations caused by increases in computing load. On a more fundamental level, the lack of a pulse indicates that the patient has expired: the software has crashed.

Decoupling Inputs and Outputs The naive way of organizing a control program is to read one input, actuate the corresponding output, read the next input, actuate the next output, and so on.

However, it is much better to separate the operations of gathering information and actuating outputs . This requires that the information gathering routines write input information into memory registers as flags or data values. The output routines then read these flags and values when actuating outputs. The input routines may be regarded as accumulating and saving messages, which are then acted upon by the output routines.

This decoupling of input and output processing into two distinct phases has several advantages.

- Output actions may become conditional on different combinations of input signals simply by various boolean operations on flag variables, something that is difficult or impossible to do if the input and output routines are intertwined.
- Debugging is simplified. By breakpointing the main loop at the end of the information gathering phase, it's possible to examine the complete set of system input signals. By modifying the input variables and running the output routines, its possible to check that the output routines are acting correctly. Various routines can be disabled and enabled by patching the subroutine call in the main loop. (To disable a subroutine call, use the monitor program to change the `JSR WHATEVER` instruction to three `NOP` instructions.)
- The input flags are available as operator display status indicators. These have considerable debugging value and ultimately, may be useful in impressing and entertaining clients.

Waiting Loops It is undesireable to have the processor tied up in a waiting loop, whether waiting for some input signal to become available or in time sequencing output variables. A corollary of this is that delay loops should be avoided. If a particular routine cannot proceed, then its state (the values of its important variables) should be stored, the main loop resumed, and the routine checked again on the next pass around the loop.

Prelude to Multitasking In a sense, the main loop is a kind of task dispatcher and the subroutines are system tasks[2]. The main loop schedules the system, calling system tasks as they are needed. No task has the right to tie up the system while waiting for an input signal to arrive or delay loop to complete. Tasks communicate by sending messages to each other, in this case via shared variables. A message is nothing more than a flag bit that is set by one task routine and read by another task routine.

It is important to keep this model of organization in mind as you develop your own system programs.

6.0.5 Example Control Program: CAPCON

To attach some reality to these concepts, we will now take a look at a project completed by the author, the Solar Telescope Capsule Controller for the McLaughlin Planetarum in Toronto.

A block diagram the system is shown in figure 6.3.

[2] We shall use the term task extensively. In this context, you may interpret the term to mean one or more subroutines that work together to obtain a particular result.

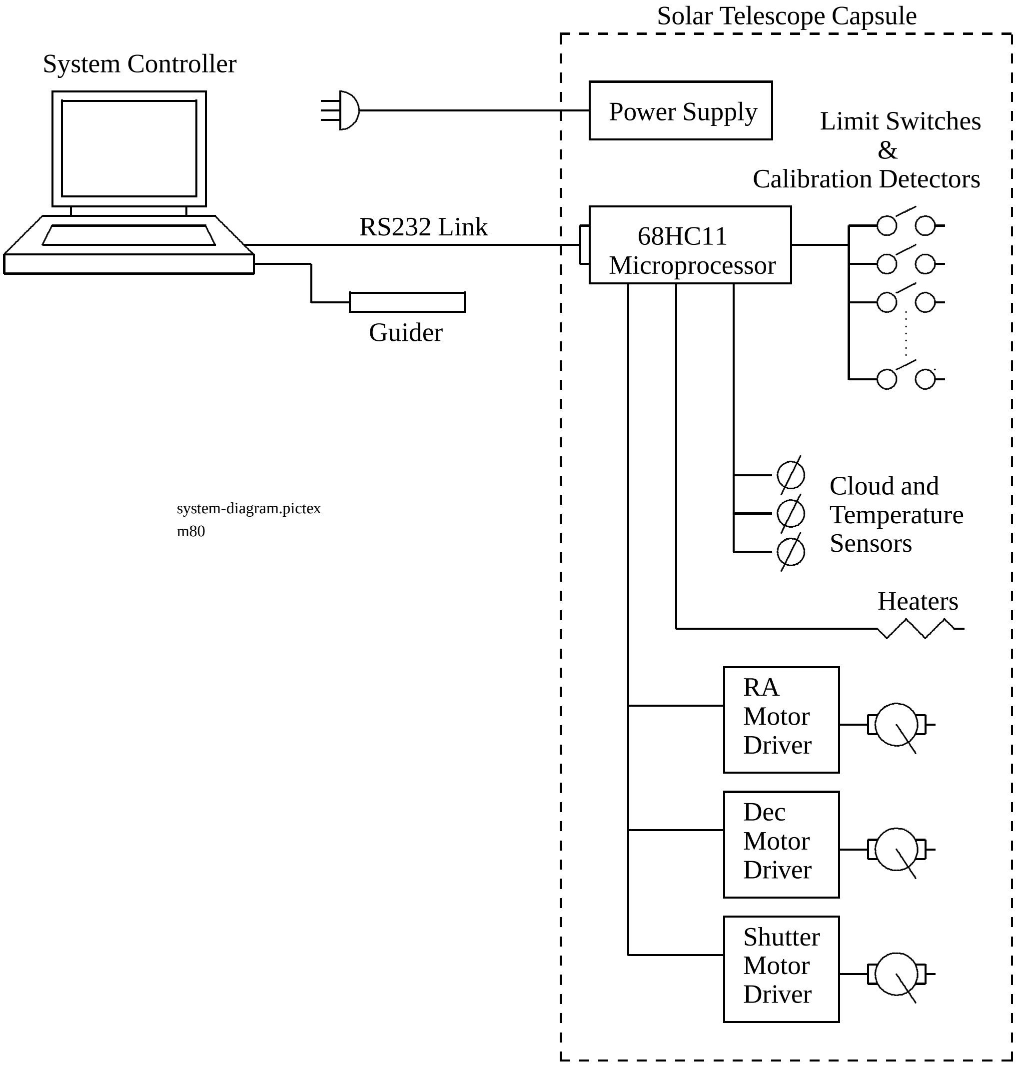


Figure 6.3: Solar Telescope System Diagram

The solar telescope consists of a moveable primary mirror on the roof of the McLaughlin Planetarium. The primary mirror is moved by two stepping motors to reflect the sun onto one of several fixed mirrors, which then deliver an image of the sun to a viewing screen in the planetarium gallery.

The two telescope stepping motors (Right Ascension and Declination) are operated by the micro via stepping motor translator drives located in the capsule electronics package. Each motor drive accepts step and direction logic signals to actuate the motors.

The control system is operated by a **System Controller**, a 486-33 IBM compatible P/C, and a **Capsule Controller**, a 68HC11 microprocessor. The P/C is located in the building, adjacent to the exhibit display. The micro is located in the telescope capsule, on the roof of the building. These two computers communicate with each other via an RS232 bidirectional serial link.

The P/C controller has sufficient computing power to perform the astronomical calculations of sun position while the telescope is tracking[3]. The only cables going into the capsule are 117VAC power and RS232 serial communications, which minimizes the number of control wires connecting the system controller to the capsule.

Under normal operation, the controller P/C issues commands to the capsule processor and displays the resulting status messages. However, for debugging purposes, the capsule may be tested independently of the P/C controller by plugging it into a source of 117VAC and sending it serial commands from a P/C terminal emulator.

The 68HC11 capsule controller responds to a repertoire of 35 commands in the following categories:

Capsule Motion Requests for single steps, move to limits, move to the calibration points, reset position counters, inquiries of capsule position, start slew at controlled rate.

Capsule Heat Requests for temperature readings and commands to enable and disable capsule heat.

Cloud and Shutter Requests to open or close the shutter and to send the current cloud reading.

Switch Status Request for the status of all capsule switches and calibration sensors.

Housekeeping Verify capsule internal program, check processor memory, terminate to processor monitor, report supply voltage.

[3] As an interesting sidelight on the development of computing technology, the original telescope system used an 8080 based processor and required 30 minutes to calculate a table of sun positions for the day. This new system calculates the sun position 30 times per second.

6.0.6 Example: CAPCON Main Loop

.

The main loop of the capsule controller program is shown in the listing below. (To head off any criticism about the missing preamble to this section, it has been removed for brevity.)

You should be able to identify the following features:

Initialization The system uses three interrupts. The SCI (Serial Communication Interface) generates an interrupt each time a message character arrives. The slew command requires that the stepping motor receive step pulses at a constant rate, and a timer interrupt is used for this. If the system tries to execute an illegal operation code, this causes an interrupt which generates an error message and transfers control to the Buffalo monitor. (Hopefully, this will not happen during normal operation.) Each of these interrupt sources requires various setup operations, which are called from this section. Finally, the `CLI` instruction enables all maskable interrupts.

The Main Loop The first thing to notice about the Main Loop is the predominance of subroutine call instructions. In fact, the main loop is primarily subroutine calls with a few branch instructions to enable and disable certain tasks.

Consider the instruction sequence

```
NO_STATUS_REQ   LDAA AUTO_MESSAGE    If auto status messages is enabled
                ANDA NEW_SWITCH_STAT  and switch status has changed
                BEQ NO_OUT_MESSAGES
                JSR SEND_STATUS      Then send the status message
                LDAA #0               and clear the 'changed switch' flag
                STAA NEW_SWITCH_STAT

NO_OUT_MESSAGES (etc)
```

The precise function of this group of instructions is not important, but it does serve to illustrate the concept of `AND`ing two flags together to determine if a task should be executed.

Now consider the sequence

```
NO_NS_POS       LDAA REQ_IN_TEMP     If any of the temperature
                ORAA REQ_OUT_TEMP
                ORAA REQ_CLOUD_LEVEL    or cloud data requests
                ORAA REQ_VOLTAGE        or 28 volt reading
                BEQ NO_AD_REQ
                JSR READ_AD           then read the A/D converter

NO_AD_REQ       (etc)
```

The various request flags (`REQ_IN_TEMP` and so on) are set or cleared by communication commands, which are an input to the system, and are established in the first part of the Main Loop.

The logic of these instructions ensures that the A-D converter is read if there is a request for internal temperature **OR** a request for external temperature **OR** a request for cloud level **OR** a request for motor voltage reading. This kind of logic is simple when the various flags are available and there is a single command for calling a particular task.

The last instruction in the Main Loop is `JMP MAIN`, which closes the loop.

```
**********************************************************************
*                     Capcon Initialize
*
*          This is where the capcon program starts up
*
CAPCON_INIT     EQU *            Capcon entry point
                LDS #STACK       Initialize the stack pointer, compatable
                                    with Buffalo
*
                LDAA #$93        Set up the 'option' register per Buffalo
                STAA OPTION      adpu, dly, irqe, cop
*
                LDAA TMSK2       Set the timer prescaler to x4 rate
                ORAA #%00000001   for the 'stepping' signal
                STAA TMSK2       Must happen in first 64 cycles after reset
*
                JSR INIT_FLAGS   Initialize control switch flags
                JSR INIT_SPI     Initialize the SPI parallel ports
                JSR INIT_SCI     Initialize SCI system
                JSR INIT_BAD_OP  Initialize illegal opcode interrupt vector
                JSR INIT_SLEW    Initialize slew interrupt vectors
                CLI              Enable maskable interrupts
*       *********** Main Loop ************
MAIN_LOOP       JSR UPDATE_INPUTS   Read the input data
                JSR RA_TASK         Update right ascension position
                JSR DEC_TASK        Update declination position
                JSR DOOR_TASK       Update (shutter) door position
                JSR UPDATE_SPI_OUT  Update SPI port output data
                JSR ACTUATE_SPI     Send SPI data and get SPI input data
*
                LDAA MSG_COMPLETE   Check whether there is an incoming
                                     message
                BEQ NO_IN_MESSAGES
                JSR DISPATCHER      If so, handle the message
*
NO_IN_MESSAGES  LDAA REQ_STATUS_MSG  Check for request for status message
                BEQ NO_STATUS_REQ
                JSR SEND_STATUS      If so, output a status message
*
NO_STATUS_REQ   LDAA AUTO_MESSAGE    If auto status messages is enabled
                ANDA NEW_SWITCH_STAT  and switch status has changed
                BEQ NO_OUT_MESSAGES
                JSR SEND_STATUS      Then send the status message
                LDAA #0               and clear the 'changed switch' flag
                STAA NEW_SWITCH_STAT
*
NO_OUT_MESSAGES LDAA REQ_EW_TRANS   Check for position count request, e/w
                BEQ NO_EW_POS
                JSR SEND_EW_POS
*
NO_EW_POS       LDAA REQ_NS_TRANS   Check for position count request, n/s
```

```
                BEQ NO_NS_POS
                JSR SEND_NS_POS
*
NO_NS_POS       LDAA REQ_IN_TEMP     If any of the temperature
                ORAA REQ_OUT_TEMP
                ORAA REQ_CLOUD_LEVEL    or cloud data requests
                ORAA REQ_VOLTAGE        or 28 volt reading
                BEQ NO_AD_REQ
                JSR READ_AD              then read the A/D converter
*
NO_AD_REQ       LDAA REQ_IN_TEMP    Check for request for inside temperature
                BEQ NO_IN_TEMP
                JSR SEND_I_TEMP     If so, send it.
*
NO_IN_TEMP      LDAA REQ_OUT_TEMP   Check for request for outside
                                     temperature
                BEQ NO_OUT_TEMP
                JSR SEND_O_TEMP     If so, send it.
*
NO_OUT_TEMP     LDAA REQ_CLOUD_LEVEL  Check for request for cloud level
                BEQ NO_CLOUD_REQ
                JSR SEND_CLOUD        If so, send it.
*
NO_CLOUD_REQ    LDAA BREAK_FLAG     Check for break to monitor
                BEQ NO_BREAK
                JMP BUFFALO
*
NO_BREAK        LDAA REQ_EPROM_CHECK  Check for request to send checksum
                BEQ NO_EPROM_CHECK
                JSR CHECK_EPROM
*
NO_EPROM_CHECK  LDAA REQ_RAM_CHECK    Check for request to check RAM
                BEQ NO_RAM_CHECK
                JSR CHECK_RAM
*
NO_RAM_CHECK    LDAA REQ_VOLTAGE      Check for request to send voltage
                BEQ NO_VOLTAGE_CHK
                JSR SEND_VOLTAGE

NO_VOLTAGE_CHK  JMP MAIN_LOOP        and do it all again
```

6.1 Module Design

The program Main Loop calls a number of subsidiary subroutines. One of the first challenges of the software design, then, is partitioning the project into these various modules.

Each module has the following characteristics:

Functionality The function of a module should be describable in a few words. If it takes several sentences to describe the function of the module, it's probably too complex and should be further decomposed.

Communication The module should act on the machine state in a clearly defined way. That is, we should be quite clear what does and does not get changed by a particular module. Most modules change the contents of the machine registers: the accumulator, index registers, stack pointer and so on. If you want to maintain the machine state, then you must push it onto the stack before calling the subroutine and then pop it back off the stack at the end of the subroutine. If the module uses a buffer area of RAM for some calculations, the information in the buffer area is destroyed by the module, and this is relevant.

The more things the module changes, the more difficult it will be to understand what the function of the module if (when) it misbehaves. To minimize complexity, minimize what each module changes.

Entry and Exit Points Each module should have one entry point and one exit point. It may call other subroutines, but the exit from this module should occur at the same point, the physical end of the module, regardless of the processing that takes place in the module. A practical consequence of this is that a subroutine should have one and only one `RTS` instruction.

Re-useability If a software module is properly designed, it should be re-useable from project to project. For this to work, the interfaces to the module (what data goes in, what data comes out) must be carefully defined. This is especially true for high level languages, which can be recompiled on a new target machine to generate new assembly language code. However, even assembly language modules may be ported (adapted) to a new machine if they are well structured and documented. For example, much of the reference material for the 6502 microprocessor, a predecessor to the 68HC11, can be easily adapted to the 68HC11 if it is correctly structured.

6.1.1 Module Example

The CAPCON command set includes a command to report the inside temperature of the capsule. In the CAPCON software, this is accomplished by a heirarchy of modules:

`SEND_INSIDE_TEMP` is the highest level module, called from the main loop when a temperature report is requested. It retrieves temperature sensor input voltage reading from the A-D buffer area, generates the binary representation of the temperature, calls some routines to format the binary number suitably and then finally calls `SEND_S_HTEMP` to transmit the ascii temperature message.

`SEND_S_HTEMP` in turn calls `DO_MESSAGE` to send the ascii string out the serial port.

`DO_MESSAGE` calls `OUTSTRG`, a general purpose routine which issues a string out the serial port. It then calls `OUTCRLF`, which issues a carriage return and line feed character.

Both `OUTSTRG` and `OUTCRLF` use `OUTCHAR`, a routine that sends one character out the serial port.

This example uses a heirarchy of routines, ranging from the high level and special purpose routine `SEND_INSIDE_TEMP` down to the general purpose routine `OUTCHAR`.

Notice the functionality of each of the routines. Each one may be described in a short sentence:

- `SEND_INSIDE_TEMP`: Send message string containing inside temperature value.
- `SEND_S_HTEMP`: Send message from the HTEMP buffer
- `DO_MESSAGE`: Send message out the serial port, followed by carriage return and line feed.
- `OUTSTRG`: Send null terminated ascii string.
- `OUTCRLF`: Send carriage-return, line-feed characters.
- `OUTCHAR`: Send ascii character.

6.1.2 Module Description

Once the function module function is described in one-sentence, the design of the module header can proceed. The header is extremely important. It provides three vital pieces of information:

What does this module do? This the one sentence description of the module. Additional material may explain the function in the context of the larger program.

How do I use it? This section provides enough information to re-use the routine for some other application. In order to do this, it precisely defines the parameters that are used and produced by the module. In technical terms, this section defines the module interface. With a knowledge of the interface, it should be possible to use the routine as a black box, without understanding of the internal workings of the module.

How does it work? This information is the key guide to debugging and modifying the internal operation of the module. It may give an algorithm or reference for the method, or may set out the logic of the module in pseudo-code. Comments in the body of the code are helpful but are no replacement for an overview of the design in the header.

Once the heading is written, then it is a matter of translating the interface and internal description into working code. Writing the header is part of the module design process, not something that is done after the code is written.

6.1.3 Module Heading Example

The 68HC11 processor generates an interrupt when it attempts to execute an illegal instruction operation code. This is usually the result of an erroneous attempt to execute data as instructions.

The routines given below cause a bad op code interrupt to print out the location and terminate to the Buffalo monitor.

```
*******************************************************************
*          Initialize Bad Op Code Vector
* This routine sets up the vector to point at the 'bad op code'
*  interrupt handler routine.
*
INIT_BAD_OP LDAA #$7E             Revector the bad opcode interrupt
            STAA BAD_OP_VECTOR    using the pseudo-vector location
            LDD #BAD_OP_ISR
            STD BAD_OP_VECTOR+1
            RTS
*
*******************************************************************
*          Bad Op Code Handler
* This routine handles an interrupt caused by the attempted
*  execution of an illegal op code.
* It generates an error message indicating the error location
*  and then terminates to the monitor.
* Determining the location of the bad opcode:
*  When the interrupt occurs, the processor puts the machine
*   state on the stack as follows (starting at the higher
*   address and working down):
*    8     PCL
*    7     PCH
*    6     IYL
*    5     IYH
*    4     IXL
*    3     IXH
*    2     ACCA
*    1     ACCB
*    0     CCR    <- .X points here after TSX instruction
*                 <- Stack Pointer
* To display the location of the illegal op code, we must transfer
*  the stack to the X register using the TSX instruction. This
*  puts the stack pointer value plus one in the X index register.
* Now load the accumulator with an offset indexed load, where
*  the offset corresponds to the offset of the PCH on the stack.
* Then use OUTLHLF, OUTRHLF to print it.
* Repeat for PCL, and we're done.
*
BAD_OP_ISR   LDX #EM_BAD_OPCODE   Print error message
             JSR OUTSTRG           without crlf
             TSX                  Find bottom of stack frame
             LDAA 7,X             Get PCH from stack
             JSR OUTLHLF          Output MSD
             LDAA 7,X             Get PCH again
             JSR OUTRHLF          and output LSD
             LDAA 8,X             Get PCL from stack
             JSR OUTLHLF          Output MSD
```

```
             LDAA 8,X                Get PCL again
             JSR OUTRHLF             and output LSD
             JMP BUFFALO             Break to the monitor.
```

Notice there are more lines of comment in the header than there are of code. This is as it should be: the most important comments give the big picture. For different commenting styles, it is illustrative to look through the source code for the Buffalo monitor and MATH11 floating point package[4] As you read these listings, ask yourself what additional information would have been helpful in the module documentation.

6.1.4 Module Internal Design

A brief word on code comments, beginning with the author's favorite comment:

```
            LDA BLAH           Load blah into the accumulator
```

Now, anyone reading assembly language code is likely to know that `LDA` means `Load the Accumulator`, so the comment is completely useless. What the reader needs to know is not what the machine does but rather what is your intent with this instruction. Put another way, how does this particular instruction contribute to the logic of the module?

Now contrast this set of comments[5]

```
            LDA FUBAR      Load FUBAR into the accumulator
            CMP NURT       Compare with NURT
            BNE ZING       If they are not equal, branch to ZING
NON_ZING    etc                  etc
```

with these:

```
            LDA FUBAR      If the ZURTBLOT flag
            CMP NURT       is not equal to NURT then
            BNE ZING       do the ZING stuff
NON_ZING      etc          else do the NON-ZING stuff
```

The two sets of code are identical, but the second set of comments, while somewhat surreal, convey a much better idea of the intent of the module. Notice as well that the second set mimic the IF-THEN-ELSE decision structure of Pascal or C. Again, this increases readability.

Other useful decision structures are REPEAT-UNTIL, DO-WHILE, and CASE.

6.1.5 Decisions, Decisions

Decision structures (conditional branches) are particularly error prone in assembly language. Fortunately, the 68HC11 reference manual [98] includes a handy summary of branch instructions in Appendix A: Instruction Set Details. The table makes it much easier to use the branch instructions correctly.

The table is reproduced (without the Op Codes) in figure 6.4.

[4] Both listings are available on the web page of Jim Koch at Ryerson University.

[5] For the benefi t of readers for whom English is not their fi rst language, I should explain that FUBAR, NURT, ZING, NON-ZING, and ZURTBLOT are nonsense words that are used to represent machine symbolic addresses. FUBAR does, however, have another application: it stands for Fouled Up Beyond All Repair, as in The equipment is completely FUBAR. Various other **F** words have been known to stand in for Fouled.

Test	Boolean	Mnemonic	Complementary Branch		Comment
$r > m$	$Z + (N \oplus V) = 0$	BGT	$r \leq m$	BLE	Signed
$r \geq m$	$N \oplus V = 0$	BGE	$r < m$	BLT	Signed
$r = m$	$Z = 1$	BEQ	$r \neq m$	BNE	Signed
$r \leq m$	$Z + (N \oplus V) = 0$	BLE	$r > m$	BGT	Signed
$r < m$	$N \oplus V = 1$	BLT	$r \geq m$	BGE	Signed
$r > m$	$C + Z = 0$	BHI	$r \leq m$	BLS	Unsigned
$r \geq m$	$C = 0$	BHS/BCC	$r < m$	BLO/BCS	Unsigned
$r = m$	$Z = 1$	BEQ	$r \neq m$	BNE	Unsigned
$r \leq m$	$C + Z = 1$	BLS	$r > m$	BHI	Unsigned
$r < m$	$C = 1$	BLO/BCS	$r \geq m$	BHS/BCC	Unsigned
Carry	$C = 1$	BCS	No Carry	BCC	Simple
Negative	$N = 1$	BMI	Positive	BPL	Simple
Overflow	$V = 1$	BVS	No Overflow	BVC	Simple
r=0	$Z = 1$	BEQ	$r \neq 0$	BNE	Simple
Always	-	BRA	Never	BRN	Unconditional

Figure 6.4: 68HC11 Branch Instructions

Comparisons are always between a machine register, such as accumulator A, and some memory location. In this table, r represents the machine register and m a memory location.

For example, in the seqence

```
                LDA FUBAR
                CMP NURT
                BNE ZING
NON_ZING          etc
```

the comparison is between the value in the accumulator and the value in the memory register ZING.

Four different types of conditional branches are shown in this table: unconditional, simple, unsigned and signed.

Unconditional Branches The `BRA` instruction causes an unconditional branch to the offset operand, +/-127 bytes away. It is useful as a two byte, relocatable replacement for the `JMP` instruction. The reference manual attempts to make a case for the `BRN` instruction, but it's difficult to imagine where it would actually be used.

Simple Branches These branches are determined directly by the various bits of the condition code register: `Carry`, `Zero`, `oVerflow` and `Negative`. Each machine instruction in the reference manual contains information on how it affects these bits in the condition code register.

Unsigned Branches These branches are based on byte comparisons where the bytes are considered to be unsigned numbers. Thus, for example, if r contains $1111\ 1111$ and m contains $0000\ 0000$, then r is considered larger than m.

Signed Branches These branches consider the bytes to be 2's complement numbers. In this case, a number is negative if its most significant bit is set, and this will be reflected in the condition code `N` bit. In this case, if

r contains $1000\ 0000$ and m contains $1111\ 1111$, then r is considered smaller than m because $1111\ 1111$ is interpreted as -1 in decimal.

Branch Code Example

Generate the assembly language code for a decision between ALPHA and BETA. We wish one block of code to be executed if ALPHA is larger that BETA. A different block of code is executed if ALPHA is less than or equal to BETA. Both numbers are in 2's complement format.

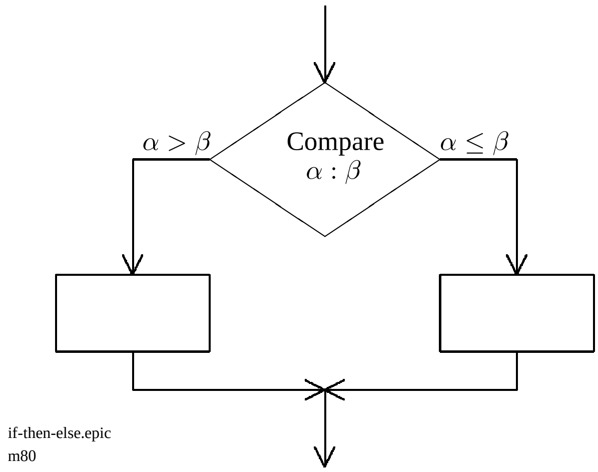


Figure 6.5: IF-THEN-ELSE Flowchart

Solution: We should use the first line of the chart, because we are comparing signed numbers. Then the code is

```
                    LDAA ALPHA              If alpha is greater
                    CMPA BETA                than beta, then
                    BGT ALPHA_IS_LARGER      do the 'alpha_is_larger'code

ALPHA_IS_LEQ_BETA   (etc)                    else 'alpha_is_less_than_or_equal'.
                    (some code here}
                    BRA CONTINUE_HERE       Skip the alpha_is_larger code

ALPHA_IS_LARGER     (etc)
                    (more code here)

CONTINUE_HERE       (both branches continue here)
```

If we goofed and wanted to reverse the sense of the decision, we would replace BGT with its complement from the table, BLE. If ALPHA and BETA were both unsigned variables, then the appropriate branch instruction would be BHI.

Miscellaneous Hints on Branching

It is possible to do a conditional branch after any instruction, and the branch will take place based on the contents of the condition code register. For example, after an `ADDA` operation, the `C` bit will indicate whether there was a carry and the `Z` bit will indicate if the result is zero. If the operands were 2's complement numbers, the `N` bit will indicate whether the result was negative and `V` will indicate whether an overflow occurred.

However, if the ultimate in speed is not required, it is safest to do an explicit compare operation before each branch instruction. For example, the Carry bit is not affected by an `INCA` instruction, so a conditional branch on the carry flag cannot be used to branch after `INCA`. Likewise, `INX` (an increment on the 16 bit X index register) only affects the Zero flag.

In other words, it is best to use a compare instruction to unambiguously set up the condition code register in preparation for a conditional branch. For example, to check for a zero result in accumulator A after some operation, `CMPA #0` is guaranteed to work. Similarily, CPX #$FFFF will detect when the X index register has been decremented from $0000 to $FFFF

6.2 Module Debugging

With the design constraint that a module has one entry and one exit point, it is easy to place breakpoints in the module for debugging purposes. One breakpoint goes at the beginning of the module, to check variables at module entry. A second breakpoint goes at the exit point of the module, to check variables at module exit. If the processor enters the module, it **must** do so at the entry point and leave at the exit point. Between those two points, the module changes the machine state, hopefully in some predictable way.

In a sense, a comparison of the input and output states provides us with a measurement of the transfer function of the module. A transfer function is the ratio of the input and output signals, so it is essential to verify the content of the input signals. In this context, you must know the state of the machine and memory registers when the processor enters the module. Determine this by examining memory and register contents at the entry breakpoint.

Now suppose the module contains an error. Move the exit breakpoint backward through the module until the code between the entry and exit breakpoints is known to work correctly. In difficult cases, this may be as little code as one instruction. Then move the exit breakpoint forward, carefully verifying operation with each change.

It helps to be suspicious. Verify all your assumptions. Cause a bug to occur again and again, checking that the symptoms are predictable. Write down the symptoms.

If all else fails:

Simply stare at the code until little drops of blood appear on your forehead,
then it will come to you.

6.2.1 Data Structures

A data structure is a way of organizing data. Arrays, linked lists and lookup tables are all examples of data structures.

Procedures act on data structures, and it is often possible to tradeoff one for the other. A large procedure, possibly with embedded data items, may be made into a smaller procedure and an associated data structure.

6.2.2 Data Structure Example: Temperature Conversion

Now we reconsider the example of a thermistor temperature measuring circuit first mentioned in section 1.2.3. To recap, the ambient temperature is some function of the microprocessor A-D reading. In section 1.2.3 I suggested that we use the A-D reading as an index into a lookup table, and read out the value of temperature corresponding to that A-D reading.

Now let us look at that solution more critically. As engineers, it is important to be able to propose several solutions to every problem, and then to identify the most effective one.

Here are some possible solutions:

Single Entry Lookup Table Under this scheme, the lookup table consists of 256 entries, each one corresponding to an A-D reading. Each table entry is a byte value representing temperature. Unfortunately, this limits the temperature value byte to a range of 0-255. As well, we have to be able to represent negative temperatures, so the range of readings is really something like +/-127. This is sufficient for temperature readings to the nearest degree, but insufficient for temperature readings to a tenth of a degree. However, temperature lookup is fast and simple. The table requires 256 bytes of storage.

Double Entry Lookup Table This is similar to the previous scheme, except that each temperature value is contained in 2 bytes, for a total range of +/-32768. Obviously, this will provide sufficient resolution for tenth of a degree resolution. The storage requirement of 512 bytes is starting to become significant. Again, however, temperature lookup is fast and simple. The two bytes of each reading could be stored in adjacent table entries in a 512 byte table. Alternatively, there could be two, 256 byte tables, one for the high byte and one for the low byte of each temperature value. (Notice that double byte entries still allow you to display only 256 distinct temperature readings, but each reading can have a greater display precision. One can argue rather effectively that this type of display is misleading: it implies greater accuracy than the A-D converter provides.)

Floating Point Calculation If we have the floating point package present in any case, it might be just as easy to process the A-D reading through a series of floating point calculations to generate the temperature reading. (For a description of the floating point package and its application, see Reference [100]). This approach has several attractions:

- The storage requirement will undoubtable be less than a double entry table, and maybe even less than a single entry table.
- The floating point routines essentially eliminate the necessity of scaling the A-D readings to fit the table.
- If the temperature value is to be used in further calculations, having it in floating point format could be helpful.
- The floating point package has a routine (`FLTASC`) which converts floating point numbers to a string of ascii characters, making temperature easy to display on an LCD display. (You might want to convert from exponential to fixed point format for display, however.)

On the other hand, floating point calculations will be much slower than table lookup. This is not a consideration if a human is viewing the readout, but it might be important if the output is part of a high speed control system.

Remember, you have to use discretion in interpreting the results of the floating point calculation. The results will valid to 0.5%, limited by the original A-D reading.

Before we commit to one or other method, we should think ahead a bit. Suppose this device is going to be shipped to a country where temperature measurement is still practiced in Fahrenheit. To which of these schemes would it be easier to add a Fahrenheit readout? Using table lookup would require a second table, doubling the table storage space. Or, the table output value could be converted using fixed point numerical calculations. Using floating point would require an additional formula calculation. Now suppose the thermistor is changed, requiring recalibration of the system. Could we somehow use an existing lookup table, or would it be easier to modify a floating point formula constant?

Notice the space-time tradeoff here: lookup table methods are require lots of space and little time. Floating point calculations require little space and lots of time. The tradeoff of space and time occurs again and again in software design. There's no one approach that gives the 'right' answer every time: it depends entirely on the application. However, it is important for the designer to be aware of the different possible methods and the strengths and weaknesses of each.

6.3 The Real-Time Interrupt

Microprocessor control programs are commonly required to perform certain tasks at a known interval. For example, the program may maintain an internal clock that must be reasonably accurate and updated 60 times per second. Or there may be peripheral devices that need to be serviced at regular intervals. For example, it may be necessary to scan a group of switches 30 times per second to determine if any one is closed.

The real time interrupt, RTI is a useful device for handling these types of tasks. The microprocessor timer is set up to interrupt the processor at a regular rate, say 60 times per second. With each interrupt, the processor stacks the machine state and then branches through an interrupt vector to the timer interrupt service routine. This ISR calls the various subroutines that need to be serviced on a regular basis. When it has completed the ISR tasks, the processor restores the machine state and returns to the point of the original interruption. The tasks called by the real time interrupt are often referred to as background tasks, compared to the main application program, the foreground task.

The magnitude of the background task has a direct impact on the speed of the foreground task. For example, if the real time interrupt frequency is 60 Hz (period 16.6 millisec) and the ISR takes a total of 5 milliseconds to complete, then the processor is spending $5/16.6 \times 100 = 30\%$ of its time servicing the background task. This then reduces the throughput of the foreground task to 70% of what it would be if it did nothing but process the foreground task. The background tasks must be kept as short as possible and the interval between real time interrupts as long as possible.

For example, it would not be wise to include floating point calculations in the background task: they are too time intensive. Nor would it be wise to set the real time interrupt interval to 1 millisecond: Unless the background routine were very short, the processor would spend all its time executing background routine.

The background task should service its peripheral and leave the gathered data for the foreground routine to process. For example, a switch scanner routine would leave the result of the switch scan in some memory registers, but it would not make any decisions based on the detected switch positions.

6.3.1 Accurate Time Delays

Many tasks require accurate time delays. To provide a task with a delay timer

1. Allocate some memory bytes to contain the time delay count.

2. Allocate one byte as a timer enable flag to control whether the timer is counting or not.

3. Include a small timer service routine as part of the real time interrupt. The routine checks the timer enable flag and, if the timer is enabled, increments it by one count.

For example, a two byte counter and 60 Hz real time interrupt would be able to time up to $2^{16} \times 16.6 \times 10^{-3} = 1087$ seconds.

To clear the timer, the foreground task should first disable the timer and then clear all the timer count bytes to zero.

The foreground task can now use this timer as a resource. It can enable and disable the count, and can periodically read the timer to determine elapsed time. These background timers do not add much load to the real time interrupt, so they can be added as needed.

There is, however, the possibility of a Very Nasty Bug that you should be aware of.

Suppose the timing count is contained in two bytes, which we will call `TIMERLOW` and `TIMERHI`. Further, suppose the two bytes contain the count $FFFF. The foreground routine now decides to do a read of the timing count. It begins by reading the low byte, which is $FF.

However, and this is the nasty bit, before the foreground task can read the high byte, a real time interrupt occurs. The real time interrupt increments the counter, which rolls over to $0000. The real time interrupt service routine then completes and returns control to the foreground task.

The foreground task now reads the high byte, now $00. Putting the low and high byte readings together, the foreground task determines a timer delay count of $00FF, which is completely erroneous.

This problem is particularly nasty because it only occurs when the real time interrupt happens to occur exactly between foreground task reads of the two bytes. As a result, the bug appears to occur at random intervals and is not reproducible.

The solution is simply to disable the real time interrupt while reading a multiple byte counter[6]. This will cause a brief jitter in the timing interval, but no counts will be lost. Of course, the real time interrupt should be disabled for the shortest possible time.

Other forms of this bug can afflict the servicing of peripheral devices. It is best to have either the background task or the foreground task service some peripheral, but not both. For example, the background task should read various switches and the foreground task should read data provided by the background task. The foreground task should not read the switches directly.

6.3.2 Flip A Bit: Real Time Interrupt on the 68HC11

This section shows an example of a Real Time Interrupt (RTI) initialization and service routine for the 68HC11 microprocessor.

Points to notice:

- This program sets up the real time interrupt so that the state of the output PA6 is changed with each interrupt. When the interrupt is running correctly, the PA6 output should show a 61 Hz square wave, even when the main loop has stopped. For example, even if the main loop terminates back to the monitor, the output waveform should continue.
- The `INITIALIZE` section of the program sets the `OPTION` register and the stack pointer exactly the same as the Buffalo monitor would do. These instructions are not necessary if the program is being started from the monitor program. However, once the program is debugged, you may then point the reset vector at this program so that it, rather than the monitor, starts after reset. In that case, it may be a good idea to mimic the startup of the Buffalo monitor.

[6] Technically, reading the two byte counter is known as a critical section, which should not be interrupted.

- Certain clock rates require adjusting the 68HC11 timer prescaler. The HC11 hardware requires that any changes to the prescaler must be done within 64 cycles of reset. Thus, for example, you cannot reset to the monitor, and then start this program, and expect the prescaler value to be changed by the instructions in this program. If you **must** change the prescaler, you have two choices: (a) incorporate the prescaler instructions is part of the monitor startup routine (you'll have to re-assemble and re-burn the monitor program) or (b) do without the monitor and have this program start directly out of reset. Obviously, if you can live with the prescaler default value, this simplifies life greatly.
- Notice the `CLI` instruction at the end of the initialization section. The interrupt mask bit in the status register is automatically set during reset, in order to disable all maskable interrupts. The `CLI` instruction clears the interrupt mask bit in the status register to enable all maskable interrupts. Each interrupt source must also be individually enabled to activate it.
- As shown in this example, the main loop is simply one `NOP` instruction. In practice, a variety of tasks would be installed here.

```
*******************************************************************
*                 Real Time Interrupt Demonstration Program
*
* Flips bit PA6 as a background task, driven by real-time interrupt.
*
* On the use of CLI and SEI:
*  The machine comes out of reset with the I bit in the CCR set, so CLI
*   must be executed during initialization to enable interrupts.
*  An interrupt sequence automatically disables interrupts and RTI
*   automatically re-enables them. Thus SEI is not normally required.
* Interrupts and Buffalo Monitor:
* The RTI will run behind the Buffalo monitor.
*  However, startup must be via a monitor 'CALL' instruction. A 'GO'
*   followed by an SWI causes the interrupts to be disabled.
*  A reset disables the interrupts, of course.

* Definitions used in this example:

STACK           EQU $0047   Stack, to be compatable with BUFFALO monitor
RTI_JMP_VECTOR  EQU $00EB   RTI vector in interrupt jump table
TMSK2           EQU $1024   TOI,RTII,-,-,0,0,PR1,PR2    Timer Mask Register2
TFLG1           EQU $1023   OC1F,OC2F,OC3F,OC4F,OC5F,IC1F,IC2F,IC3F
TFLG2           EQU $1025   TOF,RTIF,PAOVF,PA1F,0,0,0,0 Timer Flag Register2
PACTL           EQU $1026   DDRA7,-,-,-,0,0,RTR1,RTR2   Pulse Accumulator Reg
PORTA           EQU $1000   Data register for port A
OPTION          EQU $1039   Option register (see page 3-9)

******************************************************************
INITIALIZE  EQU *            Program starts here
            LDS #STACK       Initialize the stack pointer, compatable with
*                                Buffalo
*
            LDAA #$93        Set up the 'option' register per Buffalo
            STAA OPTION      adpu, dly, irqe, cop
*
            LDAA #%00000011  Set free running timer prescaler to divide by 16
            STAA TMSK2       (must be done within 64 cycles of reset)
*
            NOP              More initialization here, if required
*
            CLI              Enable maskable interrupts
            JSR ENABLE_RTI   Enable the real time interrupt
*

MAIN_LOOP   NOP              Foreground task goes here
            JMP MAIN_LOOP    Breakpoint here to return to monitor.

******************************************************************
*              Real Time Interrupt Routines
* Enable_RTI:
*  1. Set RTI rate to generate 16.38 msec real time interrupt.
*     (See page 10-12 of HC11 manual)
*     With 4MHz crystal, must set RTR0 and clear RTR1
*  2. Install JMP IRQ_SERVICE instruction in interrupt vector table
*  3. Enable RTI interrupt bit.
*
```

```
ENABLE_RTI  LDAA PACTL          Initialize RTI rate
            ORAA #%00000001     Set RTR0
            ANDA #%11111101     Clear RTR1
            STAA PACTL
*
            LDAA #$7E           Install 'JMP' op code
            STAA RTI_JMP_VECTOR
            LDD #RTI_SERVICE    Install IRQ service routine address
            STD RTI_JMP_VECTOR+1
*
            LDAA #%01000000     Clear the RTI flag to prevent immediate
            STAA TFLG2           interrupt on enabling RTI
*
            LDAA TMSK2
            ORAA #%01000000     Enable RTI interrupt bit
            STAA TMSK2
*
            RTS                 RTI is now running.
*
*************************************************************
*           Real Time Interrupt Service Routine
* This routine is called every 16.38 millisec and simply flips
*  bit PA6
* The RTI will run behind the Buffalo monitor.
*
RTI_SERVICE LDAA #%01000000  Clear the RTI flag
            STAA TFLG2
*
            LDAA PORTA       Get port A
            EORA #%01000000   and flip PA6
            STAA PORTA        and put it back so we see PA6 flipping
*
            NOP              Any other RTI tasks go here
            RTI              Done with Real Time Interrupt
```

6.4 State Machine Techniques

In many microprocessor control problems, it is useful to regard a software module as having a group of well-defined states. The classic example is a traffic light, which has a red state, an amber state and a green state. The software controlling the traffic light maintains a variable to indicate the current state and a set of rules that define how the module gets from one state to another, the so-called state transition rules.

When an event occurs that is relevant to the state machine, the action of the state machine is determined by the nature of the event and the current state. In the traffic light example, the colour of the next light depends on the current light and a timer event.

There are a number of advantages to the state machine model:

- The design process becomes much more systematic and less ad-hoc, thereby an aid the comprehension of a limited human intellect. This results in a more reliable design and one that is easier to troubleshoot if it misbehaves.

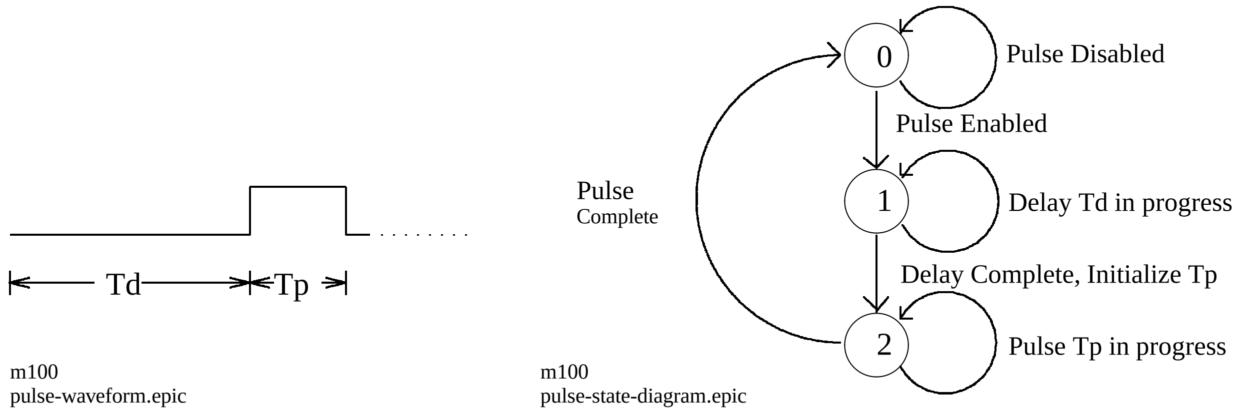
- It is possible to consider all possible combinations of input conditions and current state, thereby avoiding the unpleasant surprizes that occur when unexpected input and state combinations occur.

- For large state machines, it is possible to incorporate the current state, next state and input conditions into a table that is then read by the procedural code in the module. This makes the behavior easy to maintain and modify. It also turns out that such a data-driven state machine is very fast in execution [73].

6.4.1 State Machine Technique 1: Output Pulse Waveform

In section 6.0.4, we pointed out that a control program cannot operate correctly if it contains delay loops. We now discuss a state machine technique that avoids the use delay loops.

Consider that we wish to generate the pulse waveform shown in figure 6.6(a).



(a) Pulse Waveform (b) State Diagram

Figure 6.6: Pulse Generator Example

The naive approach to this is to write two software loops, one for the delay time T_d and a second for the pulse time T_p. The processor would lower the output line, delay T_d in a software loop, raise the output line, delay T_p in a software loop, and continue.

There are a number of problems with this approach. First, software delay loops are dependent on processor clock speed and the number of cycles in each machine instruction. As a result, they change with each new generation of processor[7] and should be avoided.

Second, each delay loop ties up the processor in unproductive looping, when it could be doing something useful, such as servicing the rest of the main loop.

A better approach is to use hardware timers or the real-time interrupt to generate the delays. Then they are a predictable function of clock speed. (If you change the processor crystal, all the delay times will change proportionally, and it will be necessary to adjust the hardware timer prescaler factor.)

Consider first the real-time clock method of generating the delay.

[7] DOS software that was written with suitable software delay loops for the 286 processor is completely unuseable on a 486 machine because the 486 is so much faster.

As shown in Chapter 6.3, the real-time-interrupt may be used to service any routines that depend on service at regular intervals. If this interval is 60 Hz, then the service occurs every 16.6 milliseconds.

We wish to generate a time delay of 0.5 seconds followed by a pulse of 0.1 seconds, so the pulse routine will be called by the real-time interrupt service routine and will have two timing counters. The delay counter T_d will be preset to $0.5/16.6^{-3} \approx 30$. The pulse counter T_p will be preset to $0.1/16.6^{-3} \approx 6$. Pulse generation is a two stage process. First, lower the output line and count down the delay timer. When it is zero, raise the output line and count down the pulse timer. When it is zero, repeat the delay time. And so on.

A systematic way of sequencing the pulse counts is with the state machine shown in figure 6.6(b). The state machinery is implemented with a variable PSTATE which contains the current state of the pulse generator state machine. Each time the pulse generator routine is called from the real-time interrupt, the processor uses the value of the current state (0, 1 or 2) to determine what it should do.

Pseudocode for the pulse generator routine is shown below:

```
Data Registers:     OUTPUT_ENABLED, byte  0 to disable output, 1 to enable
                    PSTATE, byte          Current state
                    DELAY_COUNT, byte     Delay count
                    PULSE_COUNT, byte     Pulse count

STATE_DISPATCHER    IF PSTATE=0 THEN
                        GOTO STATE_0_SERVICE
                    IF PSTATE=1 THEN
                        GOTO STATE_1_SERVICE
                    IF PSTATE=2 THEN
                        GOTO STATE_2_SERVICE
                    ELSE
                        SIGNAL STATE ERROR
                        RETURN

STATE_0_SERVICE     Td := 30 Initialize the delay timer
                    Tp := 6  Initialize the pulse timer
                    IF OUTPUT IS ENABLED THEN
                        PSTATE=1
                        MAKE OUTPUT LOW
                    RETURN

STATE_1_SERVICE     DECREMENT THE DELAY COUNTER Td
                    IF Td=0 THEN
                        PSTATE = 2
                        MAKE OUTPUT HIGH
                    RETURN

STATE_2_SERVICE     DECREMENT THE PULSE COUNTER Tp
                    If Tp=0 THEN
                        PSTATE = 0
                        MAKE OUTPUT LOW
                    RETURN
```

There are several points to notice about this routine.

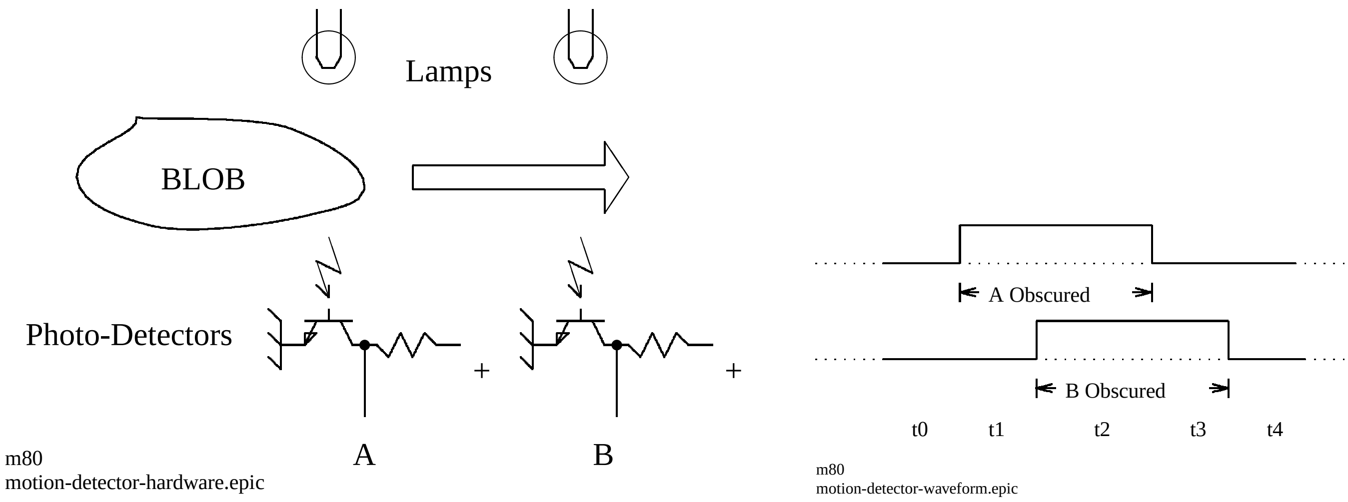
- The delay and pulse counts are both byte sized, so the maximum delay and pulse intervals would be $255 \times 16.6^{-3} = 4.23$ seconds. The resolution of the intervals is the real time clock pulse interval, 16.6 milliseconds.
- The `STATE_DISPATCHER` routine will be called by the real-time interrupt, 60 times per second. The dispatcher routes control to the appropriate state service routine, based on the current state, `PSTATE`. It is possible to design the state machine without the state dispatcher and current state variable, but having a variable to reflect the current state makes this approach easy to debug. For example, the state of the pulse generator task can be determined at any time by examining `PSTATE`. Read it out to a display, and you have a winking indication of the operation of the pulse generator.
- Notice what happens if the pulse generator is disabled by changing `OUTPUT_ENABLED` to zero. As the pulse generator is designed, it completes its current pulse and then discovers that output has been disabled when it returns to state 0. This holds it in state 0 until the output is enabled again. When the output is re-enabled, the task starts off in a known state, with the time delay part of the pulse.

- Because the pulse generator routine is in handled in software, it is capable of very long delays by increasing the size of `DELAY_COUNT` and `PULSE_COUNT` to multiple bytes. However, generation of a pulse waveform in software is not well suited for delays of less than some 10's of milliseconds. In that case, the 68HC11 hardware timers should be used.

6.4.2 State Machine Technique 2: Motion Detector

Now let us consider a more complex state machine application.

This task is required to count the number of objects past a point. Two photo-transistors are used to detect the motion of objects: humans in and out of a building, for example[8]. A suitable arrangement of hardware is shown in figure 6.7(a).



(a) Hardware (b) Waveforms

Figure 6.7: Motion Detector

When a photo-transistor is illuminated, the light causes collector current to flow. Assuming a correct value of collector resistor and illumination, the transistor is saturated, and the output voltage is approximately zero.

As an object goes past the photo-transistors, it blocks them from the light sources. When the light is removed from a photo-transistor, the collector current drops to zero and the collector voltage rises up to the supply voltage, +5 volts in this case.

As the blob object[9] travels from left to right, the output voltages of photo-transistor A and B follow the sequence shown in figure 6.7(b).

Treating zero volts as logic 0 and +5 volts as logic 1, we can show figure 6.7 as a table:

[8] A common use for this circuit is to monitor the number and direction of rotation of a motor or control shaft. Two photo-interrupters detect the passage of holes in a disk.

[9] I know, it doesn't look like a human. Use your imagination.

Blob Moves Right			
Time	**A**	**B**	**State Code**
t_0	0	0	0
t_1	1	0	2
t_2	1	1	3
t_3	0	1	1
t_4	0	0	0

If the blob moves in the opposite direction, we have

Blob Moves Left			
Time	**A**	**B**	**State Code**
t_0	0	0	0
t_1	0	1	1
t_2	1	1	3
t_3	1	0	2
t_4	0	0	0

Now the trick: treat each of these A:B combinations as a state number to obtain the state diagram shown in figure 6.8. (Notice the order of the states: they are not sequential. They order of the states is a Gray Code, in which one and only one bit changes with each number.)

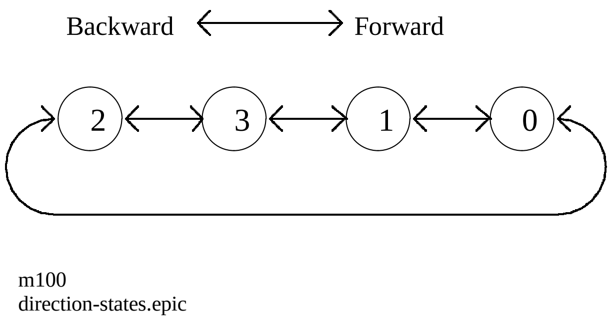


Figure 6.8: Motion Detector States

The next task is to have the microprocessor monitor signals **A** and **B** and detect the passage of an object, either forward or backward. The simplest scheme is to attach the **A** and **B** signals to two input port lines. If the speed of the program sufficiently fast that all changes in the levels can be reliably detected, the port may simply be read by the processor with each trip around the main loop. The current state of the two lines is then compared with the previous state to determine which way the object is moving.

Alternatively, changes in either of the **A** and **B** signals too rapid to be detected by checking from the main loop, may be detected by interrupt. In addition to the port input lines, the signals would be connected to the 68HC11 input capture lines IC0 and IC1. The timer edge interrupt is then set up to generate an interrupt on any edge. Every time **A** or **B** changes, this generates a processor interrupt, which causes the processor to go to the motion detector interrupt service routine. The interrupt service routine then reads the current levels of **A** and **B** and determines the new count and direction.

Each time the photo-interrupter lines change, the direction of the object movement that caused this change may be determined by comparing the current state of the **A** and **B** lines with their previous state. This is summarized in the following table: (– means no change).

Direction Transition Table				
Current State	**New State** 0	1	2	3
0	–	Left	Right	Error
1	Right	–	Error	Left
2	Left	Error	–	Right
3	Error	Right	Left	–

This table is known as a state transition table because it summarizes all possible transitions that can occur, and their possible significance. It considers all possible inputs, it should prevent the situation where a combination of input signals, unanticipated by the programmer, crashes the program.

With the state transition table in hand, we can now write the pseudocode. This is similar in concept to the previous example. A variable maintains the current state and a state dispatcher routes control to different sections of the program based on the current state.

We have one more concern to take care of, and that is knowing when to increment the blob counter. We might decide that an increment should occur when the blob moves out of the sensing area to the right, and a decrement should occur when the blob moves out of the sensing area to the left.

Then the increment and decrement actions occur for the following state transitions:

Count Transition Table				
Current State	**New State** 0	1	2	3
0	–	–	–	Error
1	Increment	–	Error	–
2	Decrement	Error	–	–
3	Error	–	–	–

With both the state transition tables in hand, we can now write the pseudocode. This is similar in concept to the example of section 6.3. A variable maintains the current state and a state dispatcher routes control to different sections of the program based on the current state.

```
Data Registers:     CURRENT_STATE, byte   Current state of input lines: 0,1,2 or 3
                    NEW_STATE,     byte   New state of input lines: 0,1,2 or 3
                    DIRECTION,     byte   0=Left, 1=Right
                    COUNT,         byte   Count of objects passing through

STATE_DISPATCHER    IF CURRENT_STATE=0 THEN
                        GOTO STATE_0
                    IF CURRENT_STATE=1 THEN
                        GOTO STATE_1
                    IF CURRENT_STATE=2 THEN
                        GOTO STATE_2
                    IF CURRENT_STATE=3 THEN
                        GOTO STATE_2
                    ELSE
                        SIGNAL STATE ERROR

STATE_0             IF NEW_STATE =0 THEN
                        NO CHANGE
                        RETURN
                    IF NEW_STATE=1 THEN
                        DIRECTION := LEFT
                        CURRENT_STATE := NEW_STATE
                        RETURN
                    IF NEW_STATE=2 THEN
                        DIRECTION := RIGHT
                        CURRENT_STATE := NEW_STATE
                        RETURN
                    ELSE ERROR
                        RETURN

STATE_1             IF NEW_STATE =0 THEN
                        INCREMENT COUNT
                        DIRECTION := RIGHT
                        CURRENT_STATE := NEW_STATE
                        RETURN
                    IF NEW_STATE=1 THEN
                        NO CHANGE
                        RETURN
                    IF NEW_STATE=2 THEN
                        ERROR
                        RETURN
                    IF NEW_STATE=3 THEN
                        DIRECTION := LEFT
                        CURRENT_STATE :=NEW_STATE
                        RETURN

STATE_2             IF NEW_STATE=0 THEN
                        DECREMENT COUNT
                        DIRECTION := LEFT
                        CURRENT_STATE := NEW_STATE
                        RETURN
                    IF NEW_STATE=1 THEN
```

```
                    ERROR
                    RETURN
                IF NEW_STATE=2 THEN
                    NO CHANGE
                    RETURN
                IF NEW_STATE=3 THEN
                    DIRECTION := RIGHT
                    CURRENT_STATE :=NEW_STATE
                    RETURN

STATE_3         IF NEW_STATE=0 THEN
                    ERROR
                    RETURN
                IF NEW_STATE=1 THEN
                    DIRECTION := RIGHT
                    CURRENT_STATE := NEW_STATE
                    RETURN
                IF NEW_STATE=2 THEN
                    DIRECTION := LEFT
                    CURRENT_STATE := NEW_STATE
                    RETURN
                IF NEW_STATE=3 THEN
                    NO CHANGE
                    RETURN
```

On the basis that there are only two situations where increment and decrement occur, you might expect that there is a simpler routine that will have the same effect. The state machine approach, however, instantly detects any change of direction and can deal with such pathological situations as the blob moving into the sensing area and then backing out again. If this happens, the blob should not be counted, and is not, using this routine. This routine is also quite robust at detecting illegal transitions in the input signals. This could be handy detecting hardware problems with the detectors.

Although there is a significant amount of code here it is a direct translation of the state transition tables into code: there is nothing tricky or unusual. In fact, it is possible to eliminate nearly all of the conditional branch instructions and have the program read the state transition tables directly, as discussed in reference [73].

As in the previous example, the `CURRENT_STATE` and `NEW_STATE` variables are handy in debugging: they indicate the current and new status of the machine.

Incidentally, you will notice that this pseudo-code has a zillion `RETURN` statements in it. To adhere to our one entrance, one exit rule for procedures, all of these `RETURN` statements should be translated into `JMP EXIT`, where `EXIT` is the address of a final `RTS` instruction.

6.5 The Button Interface

Eventually, we will need several buttons (for example) to

- select the display
- set the time-of-day clock
- reset minimum and maximum readings to zero

For now, however, let us start with one pushbutton that selects the current display. Each time this `SELECT` button is pushed, a different display is shown.

This seems simple enough, but there are several important considerations in the design of the button software.

One Event per Push The display should change once and only once per button push. Now, the software will likely be fast enough to detect a pushed button many times during the interval it is depressed by the human operator. A routine that simply changes the display each time it detects a depressed pushbutton will then change the display several times with ach button push, not what is wanted.

The button routine must therefore check for the release of the button as well as its actuation. A suitable state diagram for the button is shown below:

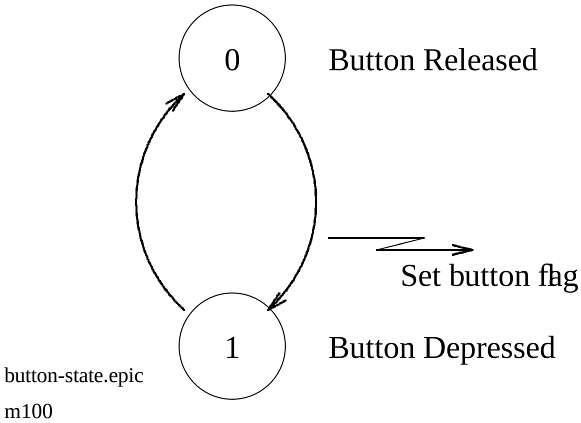


Figure 6.9: Pushbutton State Diagram

Scan Rate The button must be checked sufficiently frequently to detect the most rapid button jab by a human operator, or pressing the button must cause an interrupt. We choose to scan the button by polling software since we will be adding switches and have a limited number of available interrupts.

The main loop cannot be used to scan the button. The execution of the main loop varies as the tasks change, and so a minimum scan time for the pushbutton cannot be guaranteed.

On the other hand, if the switch scan routine is added to the real time interrupt service routine, the switch may be checked at a constant rate, 60 times per second. It turns out that this is frequent enough to detect the briefest possible actuation by a human.

Button Bounce When a mechanical switch closes, the contacts bounce open and closed for a few milliseconds after the initial contact. As a result, the switch circuit generates a train of pulses as shown in figure 6.10 below.

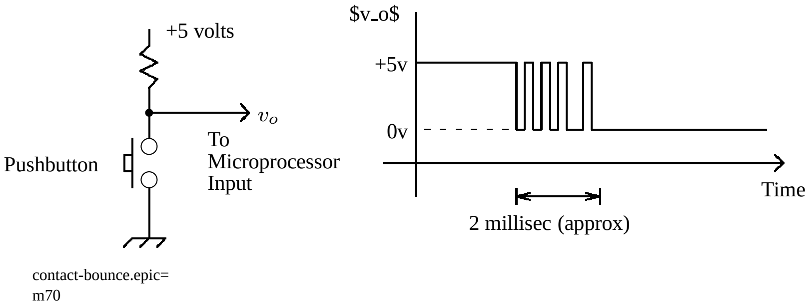


Figure 6.10: Contact Bounce

The train of pulses caused by contact bounce may erroneously be seen by the software as multiple button pushes. A simple fix is to delay and reconfirm that the switch signal has not changed. The delay is typically 3 milliseconds or so, short enough that it may be implemented with a software loop.

6.5.1 Button Switch Pseudocode

```
      B_STATE: Byte  {Keeps track of button state}
      B_FLAG:  Byte  {Set whenever button is pushed}
      RELEASED: 0    {Definition of states}
      PRESSED: 1

Initialization: B_STATE := 0
                B_FLAG := Cleared

{Real Time Interrupt causes entry here}

Routine: Case B_STATE Of    {Sends the program to the correct state }
         0: B_RELEASED      { handler software}
         1: B_PRESSED

         B_RELEASED: If button released
                        Then Exit
                        Else
                            Delay 3 milliseconds
                            If button is (still) pushed
                            Then
                                Raise B_FLAG
                                B_STATE := PRESSED
                                Exit

         B_PRESSED: If button pressed
                       Then Exit
                       Else
                           B_STATE := RELEASED
                           Exit

         EXIT: Continue servicing RTI or Return from Interrupt
```

6.5.2 Button Interface to Display Manager Task (Main Loop)

The Main Loop checks the button flag `B_FLAG` on each cycle around the main loop. If the button flag is set, the main loop increments a current display counter. For example, a current display of 0 might select temperature, a current display of 1 humidity, and so on.

Then, to indicate that it has detected the button event message, the main loop clears the button flag.

If button events occur faster than the main loop can accept them (perhaps because the main loop is preoccupied with lengthy calculations), the button task will attempt to set the button flag a second time, before it has been cleared by the main loop. This could be used to generate an error message.

Chapter 7

Instrumentation Laboratory Outline

7.1 Lab Description

The Instrumentation Laboratory is based on a single project – an instrument for measuring properties of the atmospheric environment.

The instrument uses a microprocessor and various analogue sensors to detect and record

- Ambient temperature
- Humidity
- Wind speed and direction
- Barometric air pressure

Readings are displayed on a liquid crystal display. The software includes a facility to store and display a history of past readings.

A minimum level project will consist of a device to measure and display three quantities, as specified by your instructor, from the above list. Advanced students will add other sensors to the system or may investigate new approaches to sensor design. For example, an advanced student may wish to build a wind velocity sensor that combines the measurement of wind direction and speed.

7.1.1 Project Objectives

The student will:

- design an electronic system using analogue and microprocessor components.
- design circuits requiring care in selection of components and grounding, shielding and decoupling.
- gain experience in calibrating and verifying the performance of an analogue system.
- design and build software for an embedded microprocessor system.

7.1.2 Project Management

This project is managed at a level known as structured design, ie, between a structured laboratory exercise and an open ended project (as in the thesis project). Students will be provided with a project manual which will include extensive information on the project and approaches to the design of components in the project. Since the lab will in some areas be in advance of the lecture material, enough theoretical background will be provided that the students can design circuitry used in the project.

Students will also be expected to have the Motorola 68HC11 Reference Manual.

Construction of project hardware and software should be done outside of lab hours. It is recommended that lab time be used for debugging nasty problems with the help of the instructor, calibration of sensors, and demonstrations. Students in previous sessions of the lab have spent at least 6 hours per week outside the lab on this project.

7.1.3 Microprocessor

Emphasis in the project is on analogue circuit design, construction and verification, and on microprocessor software design and debugging. To this end, a microprocessor kit the MPP Board will be available to purchase for the microprocessor section of the project. After assembling the kit, the student will have a functional 68HC11 microprocessor system. An undedicated area of the board may be used for analogue signal conditioning circuitry. Detailed specifications for the microprocessor board are given in a separate manual, [100].

The microprocessor module may be considered as a ’single chip micro’ development system. It is provided with RS232 terminal port and monitor software so that the application program may be debugged. Eventually, a single chip version of the micro could be used in the final product, in which the weather station software is burned into an EPROM.

7.1.4 Project Features

The microprocessor based weather station includes the following features:

Ambient Temperature Measurement The student is required to design and build temperature measuring circuitry. The student will consider the various engineering tradeoffs in using solid state temperature sensors vs thermistors or other devices. The final circuit will be tested against lab calibration standards. The drift performance of the measuring circuit must be predicted.

Lessons: Temperature measurement, nonlinear correction (thermistor), amplification and offset signal conditioning (solid state sensor). Use of lookup tables or floating point calculations.

Humidity Measurement The measuring circuit must determine the relative humidity of the air. This may be accomplished using a sensor which changes dielectric constant with humidity or using an electronic hygrometer. Lessons: measuring frequency and converting to measured quantity.

Air Speed Measurement The air speed measurement circuit may be devised by the standard ’whirling cups’ mechanical anemometer or by means of a thermal feedback circuit (preferred because it has no moving parts).

Lessons: Differential amplifier measurements, thermal time constants, thermal transfer effects.

Wind Direction Measurement A conventional wind vane with rotational sensor may be used to sense wind direction.

Lessons: Interfacing to a rotating sensor (gray scale sensing or other analogue rotational sensor).

Air Pressure A solid state absolute air pressure transducer may be used to generate a (small) signal proportional to barometric pressure. Conditioning this signal (a total variation of 1.67 mV over a typical change of air pressure for the Motorola MPX100A) is a challenging exercise in analogue circuit design.

Lessons: small signal amplification, control of noise, chopper stablized amplifier techniques, autocalibration technique, use of lookup tables or floating point package.

Sensor Readouts The instrument must have selectable readout displays, which should read out to 1 % precision, with (hopefully) corresponding accuracy. Calibration factors are stored in the microprocessor EEPROM.

7.1.5 Grading Criteria

The student project will be graded according to the following criteria:

- Meeting the checkpoint deadlines
- Overall integrity of the design
- Accuracy of measurement
- Ease of operator use (user interface)
- Software structure, methodology and style
- Construction technique: packaging neatness, serviceability, grounding technique
- Creativity in design and measurement technique
- Design analysis and documentation
- Quality and completeness of the final report

Marks will be assigned to different parts of the project as shown below:

MPP Board	Completed on time	5%	
	Functional	5%	10%
Sensor #1	Completed on time	5%	
	Accuracy	5%	
	Construction technique	5%	
	Software	5%	20%
Sensor #2	Completed on time	5%	
	Accuracy	5%	
	Construction technique	5%	
	Software	5%	20%
Sensor #3	Completed on time	5%	
	Accuracy	5%	
	Construction technique	5%	
	Software	5%	20%
System Integration	Completed on time		10%
Final Report	Completeness and Quality		20%
		Total:	100%

7.1.6 Tools and Equipment

Students are expected to have their own hand tools for assembling electronic circuits.

PAL and EPROM programmers are available.

Computers will be available in the instrumentation lab, though it is strongly recommended that the student have access to their own computer after hours. This way, software development can proceed on evenings and weekends, outside lab time.

Software development may be done with 68HC11 assembly language. Alternatively, students may prefer to use the C programming language. A suitable C compiler that runs under the Windows operating system is available from Imagecraft (http://www.imagecraft.com).

7.1.7 Resource Materials

Manuals for the microprocessor board and sensors are available in electronic form on the computers in the instrumentation lab.

Postscript files (suffix `.ps`) may be printed on the network laser printers.

Material currently available includes

- Construction and operation manual '`mpp_manual.ps`' for the Microprocessor Project Platform (MPP), the 68HC11 microprocessor board.
- Instrumentation lab manual, '`insmain.ps`'.
- Miscellaneous schematics, parts lists and assembly instructions.

- Software for the 68HC11.

You are strongly advised to obtain copies of this material and read it carefully.

7.1.8 Lab Schedule

Week #	Week of	Activity	Checkpoint
1	Sept 6	Orientation	
2	Sept 13	Construct MPP Board	
3	Sept 20	Install MPP Software	Demo MPP Board
4			
4	Sept 27	Build Sensor #1 Hardware	
5	Oct 4	Write Sensor #1 Software	
6	Oct 11	Calibrate Sensor #1	Demo Sensor #1
7	Oct 18	Build Sensor #2 Hardware	
8	Oct 25	Write Sensor #2 Software	
9	Nov 1	Calibrate Sensor #2	Demo Sensor #2
10	Nov 8	Build Sensor #3 Hardware	
11	Nov 15	Write Sensor #3 Software	
12	Nov 22	Calibrate Sensor #3	Demo Sensor #3
13	Nov 29	Integrate complete software	Demo Complete System
-	Nov 29	Write Report	Submit Report

- All checkpoint demonstrations must be completed by the ending time of the students lab on the due date.
- A Checkpoint Demonstration includes a demonstration of operational hardware, properly documented software and a calibration report. Calibration constants must be stored in EEPROM, not as part of the application software. Incomplete demonstrations will receive part marks. Students may be quizzed on aspects of the project, such as the software.
- A penalty of 5% per week is applied to late projects to a minimum of half the mark for that project.
- The completed unit must be demonstrated by 5 PM on the day of the lab in week 13. No work will be accepted after that date.
- The report must be submitted to the office by 4:30 PM on the due date. Late reports will not be accepted.
- Students are encouraged to help each other solve problems, including hunting for software bugs. However, plagarism (copying) of software is strictly forbidden and will be treated extremely seriously.

7.1.9 Project Report

The Lab Report documents the intellectual property that you have generated during your work on the project. There is no requirement to duplicate information, such as Ohm's Law or the MPP Manual, which is available elsewhere or is known to your instructor. At the conclusion of the project, you are required to submit a lab report which includes the following information:

- Properly drawn and labelled schematics
- Detailed parts list
- Properly commented and documented software listings
- Calibration curves and measurement results

The lab report should observe the following guidelines. (These pointers are based on the reports from previous iterations of this course.)

- The report must include a table of contents.
- Show calibration results, preferably in graphical form.
- Show all wiring diagrams, including connections between the analog sections and the microprocessor.
- Show a parts layout drawing with parts designations for each hardware section you constructed (not the MPP Board).
- Include a parts list, with quantity, designation, description, supplier and price of all items. (The microprocessor board may be treated as one item.)
- Schematics should be complete with designations and parts values. Any potentiometers should be labelled with their function.
- Formulae should have all terms defined.
- Software should be organized as a heirachy of main routine and subroutine tasks (modules).
- Software modules should be short: less than one page.
- Each software module should have a header and include a pseudocode description of its operation. A header should never go in the center of a software module.
- Document the software algorithm used to convert quantities before display.
- Software modules should be structured as 'black boxes' with well defined inputs and outputs. Input and output parameters must be specified in the header of each module.
- The emphasis on software documentation should be in the header. Try to specify the intent of the module, as opposed to a blow-by-blow description of its operation.
- If a lookup table is used, the method of generating the table should be shown as part of the header documentation.
- It is bad practice to disable interrupts. Try to use mechanisms, such as state machines, that do not require disabling interrupts.
- One and only one software module, the display manager, should write to the display. Other modules communicate with the display manager subroutine to write to the display.

- Avoid unnecessary ORG statements in code. As much as possible, allow the assembler to determine the memory address of code.
- Do not make statements you can't prove. For example the statement, "The circuit worked perfectly" is meaningless. A better statement is "The circuit met specifications", where the specifications have been defined and the actual performance measured and documented.
- Do not use more than two font types, and stick to plain fonts such as Helvetica or Times. Medieval script is not appropriate for a technical report, even the title.
- Do not use justification unless your word processing program has full, intelligent justification. Most programs don't, so it is better to leave the right edge of text ragged.
- The report must be in full electronic format. It is not acceptable to paste pieces of paper in the report or hand-draw graphs.
- The report must be printed on a laser or inkjet printer. Dot matrix is not acceptable quality.
- The preferred publishing tools are LaTeX, using **xfig** for drawings and **Matlab** or **gnuplot** for graphs. These tools have require some learning to use, but will also be extremely useful in generating your thesis project report. Documentation for LaTeXand xfig is available on the computer network.
- Reports are expected to be the original work of the person submitting the report. Any material, such as figures, graphs or software, which has been generated by some other party must be given credit in the report. Failure to do this constitutes plagiarism. Material from another party that is contains mistakes, whether attributed or not, will result in a substantial lowering of marks.

7.2 Laboratory Exercises

7.2.1 Overview

This document contains instructions for the instrumentation projects for the Instrumentation laboratory. It should be read in conjuction with the Course Outline for the Lab, and the Lab Theory Notes.

7.2.2 Calibration Constants

Many of these circuits require calibration constants to allow the microprocessor to generate the correct reading for a given electrical input. **These calibration constants should be stored in EEPROM and marks will be deducted otherwise.** Such calibration constants should **not** be written into the source code program.

Notice that it is a simple matter to manually write a calibration value into EEPROM using the Buffalo monitor. The computer program may then read the EEPROMed value during the execution of the program.

7.2.3 Humidity Sensor

Your first assignment is to develop a humidity sensor for use in the weather station. Read over Chapter 2 of the **Instrumentation System Design** manual. You may use any of the following three methods to measure humidity:

Wet-Dry Bulb (Hygrometer) Two thermistor temperature sensors are interfaced to microprocessor A-D channels. Software reads the two A-D channels and calculates the temperature of each thermistor.

One thermistor, the dry bulb, measures air temperature. The other thermistor, the wet bulb, is cooled by a muslin wick, dampened from a water resevoir, in a fan cooled air stream. A formula is used to calculate the relative humidity from the two temperatures.

This approach requires some mechanical construction but the software is quite straightforward, especially if one thermistor is already used to measure ambient temperature.

Humidity-Capacitance Sensor: Oscillator A humidity sensor is available as Philips part number 691-90001 from Electrosonic Supply for about $10. The capacitance of the sensor, which varies with relative humidity is used to control a variable frequency oscillator. The resultant frequency or period is measured by the microprocessor and a formula used to calculate relative humidity.

This approach requires the use of the timer input and interrupts on the 68HC11 microprocessor. The hardware is very simple, but the software will require careful design and coding.

Because of the variability of the sensor, the sensor must be calibrated using atmospheres of known humidity. At least three jars containing known humidity atmospheres will be provided. Students should measure the oscillator frequency at these three humidity points and then fit a curve to the measured data. The sensor response time is in the order of 5 minutes, so each measurement should allow sufficient time for the reading to stabilize.

The sensor itself generally cannot be extended on wires because the added wiring capacitance affects the calibration constant of the sensor. It is preferable to extend the entire oscillator circuit on wiring, and allow that to hang in the calibration atmosphere.

Humidity-Capacitance Sensor: Charge Amplifier This approach will appeal to students who are comfortable with hardware construction and wish to avoid interrupt programming. The variable capacitance of the Philips humidity sensor is converted into a variable analog voltage which may be read by the microprocessor A-D converter.

Suitable circuits are shown in the Linear Technology Linear Applications Handbook, application note 3. (One of these circuits is shown in Operational Amplifier Circuits, Theory and Applications, by E.J.Kennedy, Holt Rinehard and Winston.) A copy of the circuit diagrams should be available from your lab instructor.

The Linear Technology Switched Capacitor LT1043 part used in this circuit is available from Electrosonic Supply. The operational amplifier may be the Linear Technology LT1056 or some suitable substitute such as the National LMC660CN.

Humidity-Resistance Sensor A home-made humidity sensor, which relies on the change of resistivity of table salt with humidity, is described in the October 1972 issue of Popular Electronics: 'Electro-Chemical Hygrometer: Low Cost Unit measures Relative Humidity, by Joseph Giannelli'. This is a very simple circuit - it uses only two transistors - but it would have to be adapted to supply a signal to the 68HC11 A-D input.

Dew Point Sensor The Peltier Cell is a semiconductor device that may be used to cool objects. (Peltier cells may be available from Active Surplus on Queen Street). A mirror mounted on and cooled by a Peltier cell will collect a surface of moisture (dew) from the atmosphere.

In a dew point sensor, a lamp reflects a light beam off the surface of the mirror into a phototransistor. A simple feedback amplifier with input from the phototransistor drives the Peltier cell, such that dew is just beginning to form on the mirror surface. The temperature of the mirror is then at the current dew point temperature.

Relative humidity may be calculated from the ambient temperature and the dew point temperature (See the **Instrumentation System Design** manual).

Deliverables

On the due date, the student should show

- A working humidity measurement circuit. The instructor will breathe heavily onto the sensor and a change in reading should be apparent.
- Properly planned and documented software listing.
- For devices using the Philips humidity sensor, a three point calibration curve and equation.

Bonus Marks

A bonus mark of 10% will be awarded to students who use the Wet-Dry Bulb, Humidity-Capacitance Sensor: Charge Amplifier, Humidity-Resistance Sensor or Dew Point Sensor techniques. For a bonus mark to be awarded, the circuit must be substantially functional.

7.2.4 Barometric Air Pressure Sensor

Your second assignment is to develop a barometric air pressure for use in the weather station. Read over Chapter 5 of the **Instrumentation System Design** manual.

All designs will used the same sensor, the Motorola MPI100AP sensor, available from Electrosonic Supply. (Be certain to get the 'P' version, which has a pressure port connection, which is required for calibrating the circuit). The interface circuit drives the microprocessor A-D converter, and suitable software determines the air pressure.

Suitable operational amplifiers are the National LMC660CN or LM324N. The LMC660CN is a 'rail to rail' device and can work directly from the +5 volt logic supply, but is somewhat prone to spurious oscillations, especially with any capacitive loading. Its output should be carefully checked with an oscilloscope to ensure that it is not oscillating.

To drive the A-D to its full +5 volt positive limit, the LM3224 must be operated from a +6.5 volt supply. This may be derived from the raw DC input voltage to the MPP board, using a three terminal regulator (and suitable resistors to program it for 6.5 volts) such as the National LM317T or LM317L.

The barometric sensor circuit is a high gain instrumentation amplifier. The design and parts layout should be carefully planned. The circuit should be constructed carefully, neatly and with attention to proper analog grounding (ask your instructor.)

The barometric sensor can be calibrated using a water manometer apparatus available in the laboratory.

Deliverables

On the due date, the student should show

- A working air pressure measurement circuit. The student will blow positive and negative pressure into the sensor and a change in pressure reading should be apparent.
- A minimum of three points on a calibration plot, and the fitted equation relating air pressure and A-D sensor reading.
- Properly planned and documented software listing.

Bonus Marks

The lab theory manual describes how the circuit may be modified for automatic calibration of air pressure. A bonus mark of 10% will be awarded to circuits using this technique. You may wish to use the 6 bit discrete component D-A converter shown in the notes or you may prefer to purchase an integrated circuit 8 bit D-A converter such as the DAC0800. The DAC0800 and many other IC D-A converters require an op-amp current-voltage converter and a negative power supply, which may be stolen from the LT1081 RS232 driver circuit.

We strongly recommend that you first get the basic sensor circuit, using potentiometer calibration, working properly. Then add the auto-calibration circuit, carefully checking the operation of the D-A converter independently of the sensor circuit. Finally, jumper the D-A into the offset adjust point to test for auto-calibration.

For a bonus mark to be awarded, the entire circuit must be substantially functional.

7.2.5 Wind Speed Sensor

Your third assignment is to develop a wind speed sensor for use in the weather station. Wind speed is a new sensor design this year.

Read over Chapter 3 of the **Instrumentation System Design** manual.

Three wind speed sensor designs are possible:

Cup Anemometer For students who enjoy constructing mechanical devices, the Rotating Cup Anemometer may be attractive. A high quality DC motor, such as the Edmund Scientific #37590 could be used as a tachogenerator. Alternatively, a pulse frequency may be generated using a segmented disk and photo-interrupter.

In either case, the Rotating Cup Anemometer must be carefully calibrated. It may be possible to use one of the wind tunnels in the Department of Mechanical Engineering, or it may be necessary to attach the anemometer to a vehicle. In the latter case, the anemometer must be mounted well away from the vehicle body to avoid slipstream effects. It is also necessary to compensated for ambient wind velocity by measuring output vs vehicle velocity in both directions on a stretch of road. (Please choose a quiet location: not the Don Valley Parkway.)

It is to be expected that the output of a tachogenerator may require some low pass filtering to remove noise and some type of amplitude adjustment (voltage amplifier or voltage divider attenuator) to fit the dynamic range of the tachogenerator to the 5 volt range of the microprocessor A-D converter.

A careful graph of output vs wind speed should be measured. The distance constant should be calculated and the time constant estimated. The method of calibration must be carefully and completely documented.

Constant Temperature: Transistor A constant temperature anemometer, using junction transistors to sense temperature, is shown in the notes. The output from this circuit must be conditioned by a difference amplifier before it is connected to the microprocessor A-D converter.

This design will require a separate +20 volt, -5 volt power supply, since the voltages are not the same as the MPP board and current requirements are considerable. (Lab power supplies are acceptable for demonstrating circuit operation.)

It is expected that a miniature wind tunnel will be available to calibrate the circuit. A graph of output vs wind speed should be measured. The time constant should also be measured and documented.

Constant Temperature: Hot Wire A Hot Wire Anemometer circuit is also shown in the notes. A #328 lamp has its glass envellope removed to expose the filament. This is then used to measure wind speed.

This design requires a separate +15, -15 volt power supply, since the voltages are not the same as the MPP board and current requirements are considerable. (Lab power supplies are acceptable for demonstrating circuit operation.)

It is expected that a miniature wind tunnel will be available to calibrate the circuit. A graph of output vs wind speed should be measured. The time constant should also be measured and documented.

Deliverables

On the due date, the student should show

- A working air speed measurement circuit. Air will be caused to move past the sensor and a change in wind speed reading should be apparent.
- A detailed calibration plot with description of the calibration method, and the fitted equation relating wind speed and A-D sensor reading.
- Properly planned and documented software listing.

7.2.6 System Integration

The final assignment is to demonstrate the weather station functioning with all three sensors operational at the same time.

If the software for each sensor has been constructed in a modular fashion and properly documented, integration of the three sensors should be straightforward. Conceptually, the main loop should be viewed as a display manager. The display manager determines what should appear on the display, queries the sensors, formats the information, and builds the readout information.

The display can cycle through the three readings in sequence, or all three readings can occur at the same time, or a pushbutton may be added to cause the display to sequence through the various readings.

Deliverables

On the due date, the student should show

- A weather station system with three sensors functional.
- Properly planned and documented software listing for the entire system.

Bonus Marks

A bonus mark of 10% will be awarded for additional readouts provided in the final working system. These could be readouts for ambient temperature or wind direction, for example.

In the case of bonus marks for ambient temperature readout, an additional bonus of 5% will be awarded if the final system shows displays of dew point temperature, and humidex reading.

Bibliography

[1] The Canadian Encyclopedia
Second Edition
Hurtig Publishers Ltd.
Edmonton, Canada, 1988

[2] The World Book Encyclopedia
World Book, Inc, 1985
Wind, pp 278-279

[3] The Climates of Canada
David Phillips
Environment Canada, 1990
Available from:
Canadian Government Publishing Centre
Supply and Services Canada
Ottawa, Canada, K1A 0S9 Catalogue # EN56-1/1990E

[4] Piping Down the Valleys Wild:
A merry mix of verse for all ages
Edited by Nancy Larrick
Dell, 1968

[5] Mathematical Handbook of Formulas and Tables
Murray R. Spiegel
Schaum's Outline Series, McGraw Hill, 1968

[6] Practical Robotics
Bill Davies
Werd Technology Inc.
Unit 35B, Suite 155
10520 Yonge Street
Richmond Hill, Ontario, L4C 3C7

[7] Build a Computerized Weather Station
Steve Ciarcia
Byte Magazine, February 1982, PP 38-68

[8] Weather Station
Ronald M. Jackson
Electronics Now, October 1993, pp 31-38

[9] Absolute Zero and the Conquest of Cold
Tom Schachtman
Mariner Books, 1999

Temperature: Thermistors

[10] Philips Components
1990-91 Data Book
Philips Passive Components Group
2001 W. Blue Heron Blvd.
P.O.Box 10330
Riviera Beach, Fl. 33404, USA

[11] Omega Temperature Measurement Handbook
Omega International Corp
P.O.Box 22284
Stanford, CT., 06906 USA

[12] Thermistors
E.D.Macklen
Electrochemical Publications, 1979

[13] Calibration Curves for Thermistors
J.S.Steinhart and R.C.Hart
Deep Sea Research and Oceanographic Abstracts
Volume 15, 1968, pp 497-503

[14] Design Temperature Converters with Thermistor-Based Oscillators Harry L. Trietley
EDN, February 3, 1983, pp 115-118

[15] Direct-reading Converter Yeilds Temperature
James Williams and Thomas Durgavich
Electronics, April 3, 1975, pp101,103

[16] Thermistors Make Good Thermometers
C.S.Molee
Electronic Design 8, April 12, 1978, pp 90-92

Temperature: Diodes and Diode-Connected Transistors

[17] Surface Temperature Thermometer
L. Nelson-Jones
Wireless World, April 1969, pp 180-183

[18] Electronic Thermometer
A.S.Henderson
Wireless World, June 1981, pp 50-52

[19] Semiconducting Temperature Sensors and their Application
Herbert B. Sachse
Wiley and Sons, Inc, 1975

[20] The Silicon Transistor as a Temperature Sensor
T.C.Verster
Temperature, Its Measurement and Control in Science and Industry
Volume 2, Part 2, Instrument Society of America
Pittsburgh, 1973, pp 1125-1134

[21] Data Sheet, MTS102 Silicon Temperature Sensor
Motorola Semiconductors P.O.Box 20912, Phoenix, Arizona,85036, USA

[22] An Exact Expression for the Thermal Variation
of the Emitter Base Voltage of Bi-Polar Transistors
Robert J. Widlar
IEEE Proceedings, January, 1967, pp 96,97

[23] Accurate Analysis of Temperature Effects in Ic-Vbe
Characteristics with Application to Bandgap Reference Sources
Yannis P. Tsividis
IEEE Journal of Solid State Circuits, Vol SC-15, No. 6, December 1980, pp 1076-1084

[24] LM34/LM35 Precision Monolithic Temperature Sensors
Application Note AN-460
Linear Applications Handbook
National Semiconductor, 1994

[25] Transistors– A hot tip for accurate temperature sensing
Pat O'Neil and Carl Derrington
Electronics, October 11,1979, pp 137-141

[26] A Precision Silicon Transistor Thermometer
Akira Ohte and Michiaki Yamagata
IEEE Transactions on Instrumentation and Measurement
Vol IM-26, No. 4, December 1977, pp 335-341

[27] Operational Amplifier Databook, 1993 Edition
National Semiconductor Corporation, 2900 Semiconductor
Drive, P.O.Box 58090, Santa Clara, California, 95052-8090, USA

[28] LM34/LM35 Precision Monolithic Temperature Sensors
National Semiconductor Application Note 460
Linear Applications Handbook
National Semiconductor Corporation, 2900 Semiconductor
Drive, P.O.Box 58090, Santa Clara, California 95052-8090, USA

[29] Integrated Temperature Transducers
Michael J. Riezenman
Electronics, November 14, 1974, pp 130-132

[30] Operational Amplifier Circuits, Theory and Applications
E.J.Kennedy
Holt, Rinehart and Winston, 1988, pp 498-503

[31] LM134/LM234/LM334 3-Terminal Adjustable Current Sources
Data Sheet, Linear Data Book 2, 1988 Edition
National Semiconductor Corporation

[32] An RS-232 Thermometer
Don McLane
Home Automation and Building Control
October 1995, pp71-73

Voltage-Frequency Conversion

[33] 1Hz to 30MHz Voltage to Frequency Converter
Keith Williams
Electronics and Wireless World
September, 1985, page 58

[34] LM131/LM231/LM331 Precision Voltage-to-Frequency Converters
Application Circuit, 'Temperature to Freqency Converter'
Data Sheet, Linear Data Book 2, 1988 Edition
National Semiconductor Corporation

[35] Voltage-Frequency Converters
Eugene L. Zuch
Electronics, May 15, 1975, pp 91-95

[36] Versatile Monolithic V-F's Can Compute
Robert A. Pease
Electronic Design 25, Dec 6, 1978, pp 70-76

[37] Designing V-F Converters to Handle Bipolar Signals
Robert A. Pease
Electronics, June 7, 1979, pp 139-145

Humidity

[38] Meterological Instruments
W.E. Knowles Middleton and Athelstan F. Spilhaus
University of Toronto Press, 1953
Chapter 4, The Measurement of Atmospheric Humidity

[39] A Course in Elementary Meteorology
Meteorological Office
Her Majesty's Stationery Office, 1962
(The Campfield Press, St. Albans)

[40] A Thermistor Hygrometer
B. Bollen
Wireless World, December 1969, pp 557-561

[41] Elements of Meteorology
Albert Miller and Jack C. Thompson
Charles E. Merrill Publishing Company, 1970

[42] Relative Humidity Guage
Anthony J. Caristi
Electronic Hobbyists Handbook, Fall 1994
pp 84-88

[43] Electro Chemical Hydrometer
Low Cost Unit Measures Relative Humidity
J. Giannelli
Popular Electronics, October 1972, pp 33-35

[44] Relative Humidity Sensor Signal Conditioner
Jim Williams
Application note 3.
Linear Applications Handbook, Volume 1
Linear Technology, 1990

[45] Data Sheet, Series 691 Relative Humidity Sensor
Philips Components
1990-91 Data Book, page 684
(see reference [10] above)

[46] The Art of Electronics
Paul Horowitz and Winfield Hill
Cambridge University Press, 1989

[47] Troubleshooting Analog Circuits
EDN Series For Design Engineers
Robert. A. Pease
Butterworth-Heinemann, 1991

[48] Astable and Monostable Oscillators
Using RCA COS/MOS Digital Integrated Circuits
D.V.DiMassimo and A.R.Maslowski
RCA Application Note ICAN-6466
RCA Solid State COS/MOS Integrated Circuits Data Book,1980

[49] CMOS Cookbook
Don Lancaster
Howard W. Sams, 1977
Astable circuits, pp225-235

[50] Measure, Display Relative Humidity
W. Freeman
Electronic Design, January 1990, p136-137

[51] The Humidity FAQ
sci.geo.meteorology SIG
Richard Clark
rclark@lpl.arizona.edu or rdc@gas.uug.arizona.edu

[52] Small Scale Dewpoint Humidity Sensor
A.C.Easty and S.Young
Journal of Physics E, Scientific Instruments
Volume 9, No. 2, pp106-110, February 1976

[53] Vapourware: Revealing Your Humidity
Jeff Bachiochi
Circuit Cellar INK
Issue 57, April 1995, pp92-96

Wind Speed

[54] Meterological Instruments
W.E. Knowles Middleton and Athelstan F. Spilhaus
University of Toronto Press, 1953
The Cup Anemometer, pp 143-146

[55] Instruments for Physical Environmental Measurements
J.Y.Wang
Milieu Information Service
Cup anemometers, pp 190-193

Cooling effect sensors

[56] Cupless Anemometer Has Diode Wind Sensor
J.P.Scoseria
Electronics, August 30, 1979, p 156

[57] Measuring the Wind with Hot Metal
Shawn Carlson
Scientific American, November 1995, pp 98-99

[58] On the Convection of Heat from Small Cylinders in a Stream of Fluid: Determination of the Convection Constants of Small Platinum Wires, with Applications to Hot Wire Anemometry
Lois Vessot King
Proceedings of the Royal Society of London
Vol XC, September 1914, pp563-570

[59] Thermal Techniques in Measurement and Control Circuitry
Jim Williams
Application Note 5
Linear Applications Handbook, Volume 1
Linear Technology, 1990

[60] A Digital Anemometer
Ken Okamoto et al
IEEE Transactions on Instrumentation and Measurement
Vol 43, No.2, April 1994, pp 116-120

[61] Dynamic Behaviour of Thermistor Flowmeters
Andrea Taroni and Gianni Zanarini
IEEE Transactions on Industrial Electronics and Control Instrumentation
Vol IECI-22, No.3, August 1975, pp 391-396

[62] Experimental Results on Dynamic Behaviour of Thermistor Flowmeters
Bruno Morten, Maria Prudenziati, Andrea Taroni and Gianni Zanarini
IEEE Transactions on Industrial Electronics and Control Insntrumentation
Vol IECI-25, No.3, September 1976, pp 232-234

[63] Log Converters
Application Notes 29, 30, 31, 211,264
National Semiconductor Linear Applications Handbook, 1986

[64] Translinear Circuits, A Proposed Classification
Barry Gilbert
Electronic Letters, 9 January 1975, Vol 11, No 1, pp 14-16

[65] A New Wide Band Amplifier Technique
Barry Gilbert
IEEE Journal of Solid State Circuits
Vol SC-3, No 4, December 1068, pp 353-365

[66] Supermatched Bipolar Transistors Improve DC and AC Designs
Carl T. Nelson
EDN, January 5 1980, pp 115-120

[67] Analog Multipliers Determine True Wind Speed
J.A.Connelly and M.B.Lundberg
EDN Magazine, April 20, 1974, pp 69-74

[68] The Gunnplexer Cookbook
Bob Richardson
Ham Radio Publishing Group, 1981

[69] The ARRL Handbook, 1991 Edition
The American Radio Relay League

Wind Direction

[70] The Weather for a Hobby
Raymond F. Yates
Dodd, Mead and Company, 1947
Selsyn weather vane, p 85

[71] Synchro Engineering Handbook
A.R.Upson and J.H.Batchelor
CRC Press, 1965

[72] Synchro and Resolver Conversion
Geoffry S. Boyes
Analog Devices, 1980

[73] Measuring Transducer Directivity
Peter D. Hiscocks
Wireless World, August 1978, pp 61-66

[74] Microelectronic Circuits
Adel S. Sedra and Kenneth C. Smith
Oxford University Press, 1991
Chapter 12: Signal Generators and Waveform Shaping Circuits

Air Pressure

[75] Meterological Instruments
W.E. Knowles Middleton and Athelstan F. Spilhaus
University of Toronto Press, 1953
Chapter 2, The Measurement of Atmospheric Pressure

[76] Instruments for Physical Environmental Measurements
J.Y.Wang
Milieu Information Service
Mercury Barometers, pp 172-173

[77] Solid State Barometer
Sudhir K. Gupta
Radio Electronics, June 1984, pp41-44

[78] Electronic Altimeter
Frank Ogden
Wireless World, May 1985, pp 37-41

[79] Digital Altimeter
Frank Ogden
Wireless World (issue unknown, after May 1985)

[80] Temperature Compensation Methods for the
Motorola X-ducer Pressure Sensor Element
C. Swartz, C. Derrington, J. Cragg
Motorola Application Note AN-840
(included in Industrial Control Applications
Motorola Publication DL412/D)

[81] Using the Motorola X-ducer Pressure Sensor Data Sheet
H. Nagel, R. Frank
Motorola Application Note AN-919
(included in Industrial Control Applications
Motorola Publication DL412/D)

[82] Temperature Compensation, Calibration and Applications
of Motorola's X-ducer Pressure Sensor
H. Nagel
Motorola Application Note AN-922
(included in Industrial Control Applications
Motorola Publication DL412/D)

[83] MPX Pressure Sensors Used For Switch Applications
G. Swensen
Motorola Application Note AN-962
(included in Industrial Control Applications
Motorola Publication DL412/D)

[84] Calibration Free Pressure Sensor System
M. Burri
Motorola Application Note AN1097
(included in 8 Bit MCU Applications Manual
Motorola Publication DL408/D Rev 1)

[85] CMOS DACs in the Voltage Switching Mode
Steve Stephenson
Analog Dialog, 14-1, 1980, pp 16-18

[86] High Speed CMOS Logic Data Book
Texas Instruments, 1984
Data Sheet, SN74HC273, page 3-239
Specifications for HC MSI Circuits, page 2-10
Designers Information, page 7-17

[87] A Solid State Barometer for the HCS II
John Morley
Circuit Cellar Ink
October 1995, pp 63-69

Rain Guage

[88] The Weather for a Hobby
Raymond F. Yates
Dodd, Mead and Company, 1947
Rain guage, p 134

Passive Components

[89] A knowledge of capacitor subtleties helps solve capacitor-based troubles
Robert A. Pease
EDN, February 16, 1989, pp 127-134

[90] Understand capacitor soakage to optimize analog systems
Robert A. Pease
Electronics+Wireless World
October 1992, pp 832-835
also in
EDN, October 13, 1982, page 125

[91] Capacitors
Ray Marston
Electronics Now, March 1993, pp 57-64

[92] Under The Hood III: Capacitors
Brian Bergeron
QST, January 1994, pp 45-48

[93] Resonant Frequencies of Capacitors
Radio Communication Handbook
Radio Society of Great Britain, 1991

[94] Capacitors: A survey of present day technology and applications
R. A. Fairs
Wireless World, December 1974, pp 510-514

[95] Resistors: A survey of present day technology and applications
R. A. Fairs
Wireless World, October 1975, pp 487-492

[96] Troubleshooting gets down to the component level
Robert A. Pease
EDN, February 2, 1989, pp 176-184

[97] Tools for Schools: Math Packages for Education
IEEE Spectrum, September 1966, pp 41-45

68HC11 Microprocessor

[98] Motorola 68HC11 Reference Manual
Motorola Publication M68HC11RM/AD,Rev 3
General applications information on the 68HC11
family of microprocessors. A 'must have'.

[99] Motorola 68HC11 E Series
Motorola Publication M68HC11E/D, Rev2.0
Contains data on the E series
of devices, including electrical characteristics.

[100] 68HC11 Microcomputer
Technical Manual
Peter D. Hiscocks
Syscomp Electronic Design Limited, 2001
`www.syscompdesign.com`