High-Speed Assembly Language (80386/80387) Programming for Laser Spectra Scan Control and Data Acquisition Providing Improved Resolution Water Vapor Spectroscopy

Robert J. Allen
Allen Associates
Newport News, Virginia


NASA
National Aeronautics and Space Administration
Scientific and Technical Information Division

1988
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>WATER VAPOR SPECTROSCOPY EXPERIMENTS</td>
<td>1</td>
</tr>
<tr>
<td>CHOICE OF ASSEMBLY LANGUAGE PROGRAMMING</td>
<td>2</td>
</tr>
<tr>
<td>DESCRIPTION OF PROGRAMS</td>
<td>3</td>
</tr>
<tr>
<td>FORMULATING THE PROGRAMS</td>
<td>5</td>
</tr>
<tr>
<td>ACKNOWLEDGMENT</td>
<td>6</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>6</td>
</tr>
</tbody>
</table>

## LIST OF FIGURES

### Figure

1. BLOCK DIAGRAM, WATER VAPOR SPECTROSCOPY EXPERIMENT .......................... 7

2. FLOW DIAGRAMS:
   A. OF THE BASIC PROGRAM ........................................................................ 8
   B. OF THE ASSEMBLY LANGUAGE PROGRAM ................................................. 9

3. REPRESENTATIVE WATER (H2O) SPECTRUM CURVES:
   A. Cross Section vs. Wavenumber showing pressure shift in air for the line centered at 13914.9502 cm⁻¹ .................................................. 11
   B. Cross Section vs. Wavenumber showing pressure shift in air for the line centered at 13947.2608 cm⁻¹ .................................................. 12

4. SCOPE PHOTOGRAPHS OF 5-VOLT PULSES GENERATED FROM BINARY OUTPUTS BO-0 TO BO-3. PULSE WIDTHS AND VARIOUS PROGRAM TIMES ARE INCLUDED ..................... 13
LIST OF TABLES

Table | page
------|------
I. PROGRAM TIMES DETERMINED USING BINARY OUTPUT PULSES SHOWN IN FIGURE 4 | 15

II. BASIC PROGRAM (ALLENB7R.BAS)
   A. LISTING OF PROGRAM | 16
   B. DISPLAY PRODUCED BY RUNNING PROGRAM | 19

III. ASSEMBLY LANGUAGE PROGRAMS:
   A. ALLEN7R.LST (Microsoft Macro Assembler, Version 4.00) | 20
   B. SYMDEB ALLEN7R.EXE (Microsoft Symbolic Debug Utility, Version 4.00) | 27
ABSTRACT

An assembly language program using the Intel 80386 CPU and 80387 math co-processor chips was written to increase the speed of data gathering and processing, and provide control of a scanning CW ring dye laser system. This laser system is used in high resolution (better than 0.001 cm^{-1}) water vapor spectroscopy experiments. Laser beam power is sensed at the input and output of white cells and the output of a Fabry-Perot. The assembly language subroutine is called from Basic, acquires the data and performs various calculations at rates greater than 150 time faster than could be performed by the higher level language. The width of output control pulses generated in assembly language are 3 to 4 microseconds as compared to 2 to 3.7 milliseconds for those generated in Basic (about 500 to 1000 times faster).

Included are a block diagram and brief description of the spectroscopy experiment, a flow diagram of the basic and assembly language programs, listing of the programs, scope photographs of the computer generated 5-volt pulses used for control and timing analysis, and representative water spectrum curves obtained using these programs.
HIGH-SPEED ASSEMBLY LANGUAGE (80396/80387) PROGRAMMING
FOR LASER SPECTRA SCAN CONTROL AND DATA ACQUISITION
PROVIDING IMPROVED RESOLUTION WATER VAPOR SPECTROSCOPY

By
Robert J. Allen

INTRODUCTION

This report describes an Intel 80386 and 80387 assembly language program written to increase the speed of data gathering and processing, and provide control of a scanning CW ring dye laser system. This laser system is used in high resolution Water Vapor Spectroscopy Experiments. The 80386 is a 32-bit CPU chip operating at 16 MHz in a Compaq 386 computer, and the 80387 is a high speed math coprocessor. The assembly language subroutine is called from BASICA.

The program was originally written for the Compaq Portable 286 computer using the 80286 and 80287 chips. It was later updated for use with the faster Compaq 386.

WATER VAPOR SPECTROSCOPY EXPERIMENTS

A block diagram of the spectroscopy experiment is shown in Fig. 1.# An argon laser pumps a tunable CW ring dye laser. After receiving a Laser Scan Trigger, the Scanning Electronics will cause a very narrow line-width (0.0001 pm) laser beam to be outputted at wavelengths within pre-selected limits, for

# Figures 1 and 3 were provided by Dr. Benoist Grossmann, project scientists conducting the water vapor spectroscopy experiments.
example, 726.000 to 726.050 nm. This 50 pm scanning range can be selected anywhere between 725 nm (13,793.1 cm⁻¹) to 730 nm (13,698 cm⁻¹). A portion of the output beam is directed by beamsplitters to a Monochrometer for absolute wavelength calibration, a Fabry-Perot and photodiode P4 to provide the wavelength markers, and P3 as the Power Reference. The remaining beam is routed through White Cells #1 and #2 and sensed by photodiodes P1 and P2.

The outputs from the four photodiodes, P1-P4, are amplified and routed to four analog input channels multiplexed into a 10 KHz analog-to-digital converter with 12-bit resolution (A/D Board). The four multiplexed digital outputs are then processed as described later in this report. The A/D Board is part of an IBM Data Acquisition and Control Adapter card that fits inside the Compaq. This Adapter also contains two analog output channels; a 16-bit digital input port; a 16-bit digital output port; a 32-bit timer; and a 16-bit, externally-clocked, timer/counter. One bit of the digital output port is available for use as the Laser Scan Trigger.

**CHOICE OF ASSEMBLY LANGUAGE PROGRAMMING**

Since the four readings from photodiodes P1-P4 averaged N times are a single point on a rapidly changing plot of a single water vapor line, this data set should be obtained as rapidly as the hardware allows. More time, if needed, can be used to start the repeat (R) of subsequent data sets since this time is the separation between data points. Separation time, however,
should also be kept to a minimum since it is associated with the resolution of the curve.

As indicated in the general literature and by Rollins (1985), "assembly language is by far the fastest, most flexible, and most compact of all programming languages. It shows how and why higher-level languages operate. It gives access to features of a machine that are inaccessible with other languages." An assembly language matrix multiplication routine published by Startz (1985) is about 150 times faster than pre-8087 Basic.

Fortunately, assembly language routines are easily combined with either interpreted or compiled Basic, as well as with programs written in other high-level languages. The main program illustrated in this report was written in Basic (BASICA.COM) calling the assembly language routine for acquiring N sets of the four photodiode (P1-P4) readings and performing required calculations.

DESCRIPTION OF PROGRAMS

The basic program (Table II) and assembly language program (Table III) were prepared as part of this task and provided to the Project Scientists. He incorporated them into his Basic program which included plotting features for use in the high resolution spectroscopy experiments.

Figure 2 contains a flow diagram of both programs. Basic is used first (prior to scanning) to calculate and display the 'mean' value of the background noise (W, X, Y & Z) from each of the four photodiodes (P1-P4). Scanning is initiated and the assembly language program called. The output from each of the
photodiodes are read and the corresponding background noise subtracted. The differences from P1, P2 and P4 are then normalized to the power reference difference P3. This process is repeated N times and the 'mean' of each of the three ratios \(((P1-X)/(P3-W)),\ (P2-Y)/(P3-W),\ and\ (P4-Z)/(P3-W))\) calculated. The process then returns to Basic where the 'mean' of these three ratios are displayed. Each of the ratios constitute a single data point on each of three curves. The above process is then repeated R times obtaining new data points while the laser continues to scan.

BASICA line numbers are then displayed for the background sample size \(M\), data sample size \(N\) and number of repeats during laser scanning \(R\). The results, when coupled with the plotting program, are the curves shown in Figures 3A and B.

Five-volt pulses, BO-0 and BO-1, are generated in Basic prior to scanning and can be used to trigger a scope and to initiate scanning. Binary output pulses, BO-2 and BO-3, are generated in Assembly Language and were used to measure the time for different processes during program development such as the: (1) greater-than 20 microsecond delay time required to allow transients to settle following each multiplexed operation, (2) total time used to read the four analog input from the photodiodes and subtract the backgrounds from each, and (3) time to calculate the mean of the ratios. Scope photographs of pulses BO-0 to BO-3, their pulse widths, and various times are shown in Figure 4 and Table I.
FORMULATING THE PROGRAMS

The assembly language program was written using WordStar, assembled using Microsoft Macro-Assembler version 4.00 (MASM ALLEN7R) and linked for high memory residence (LINK/H ALLEN7R). Debug was called (SYMDEB ALLEN7R.EXE) and Basic entered using NBASICA.COM. L and G. ALLENB7R.BAS was then loaded and a binary file created using DEF SEG = &H9FCA followed by BSAVE "ALLEN7R.BIN",0,&h178. Running ALLENB7.BAS produced the display shown in Table IIB.
ACKNOWLEDGMENT

Publications by Lafore (1984) and Startz (1985) provided excellent references supporting the assembly language portions of this task. The author wishes to thank Dr. Jerry Tucker of NASA Langley Research Center for his helpful suggestion in connection with the assembly language programming, Dr. Benoist Grossmann for providing the information concerning the experiments and review of this report, and Patrick Ponsardin for his review and comments concerning this report. Both Benoist and Patrick are affiliated with Old Dominion University Research Foundation (on leave from Electricite de France).

REFERENCES


FIGURE 2A. FLOW DIAGRAM OF THE BASIC PROGRAM.
TO FIG. 2A

**FIGURE 2B. FLOW DIAGRAM OF THE 80386/80387 ASSEMBLY LANGUAGE PROGRAM.**
File: A9149002.J05

H20 Spectrum

Pressure Shift In Air

10 torr H20
10 torr H2O + 1 Atm Air
Fabry-Perot FSR = 0.100 cm⁻¹

Relative Wavenumber (FSR = 0.100 cm⁻¹)

FIGURE 3A. CROSS SECTION VS. WAVENUMBER SHOWING PRESSURE SHIFT IN AIR FOR THE LINE CENTERED AT 13914.9502 cm⁻¹.
FIGURE 3B. CROSS SECTION VS. WAVENUMBER SHOWING PRESSURE SHIFT IN AIR FOR THE LINE CENTERED AT 13947.2608 cm⁻¹.
(A) 5V 3.7 ms wide
Binary Output
pulse BO-0

2V/div 0.5 ms/div

(added program: 1035 GOTO 80)

ORIGINAL PAGE IS
OF POOR QUALITY

(B) 5V 2 ms wide
Binary Output
pulse BO-1

2V/div 1 ms/div

(added program: 1105 GOTO 1060)

(C) BO-0

2V/div 1 ms/div

(D) BO-1 (scope trig)

2V/div 1 ms/div

(added program: 1105 GOTO 80)

FIGURE 4. SCOPE PHOTOGRAPHS OF 5-VOLT PULSES GENERATED FROM
BINARY OUTPUTS BO-0 TO BO-3. PULSE WIDTHS AND VARIOUS
PROGRAM TIMES ARE LISTED IN TABLE I.
(E) BO-1
2V/div 100 ms/div
(added program: 1395 GOTO 1060)

(F) BO-2
2V/div 1 ms/div

(G) BO-3
2V/div 1 ms/div
(added program: 1185 GOTO 1160)

(H) BO-1 ADDED TO INTENSIFIED BO-2
(added program: 1185 GOTO 1060)

ORIGINAL PAGE IS OF POOR QUALITY.
TABLE I. PROGRAM TIMES DETERMINED USING BINARY OUTPUT PULSES SHOWN IN FIGURE 4

<table>
<thead>
<tr>
<th>BASICA LINE NUMBERS</th>
<th>ADDED PROGRAM STEP *</th>
<th>GENERATED BINARY OUTPUT PULSE</th>
<th>TIME FROM START OF BO-1 TO START OF BO-j</th>
</tr>
</thead>
<tbody>
<tr>
<td>80-140</td>
<td>1035 GOTO 80</td>
<td>BO-0, 5V 3.7 ms wide (FIG. 4A)</td>
<td>BO-0 to BO-0: 200 ms (M=1) 412 ms (M=10) 875 ms (M=30) 2500 ms (M=100)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1060-1100</td>
<td>1105 GOTO 1060</td>
<td>BO-1, 5V 2 ms wide (FIG. 4B)</td>
<td>BO-1 to BO-1: 3 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80-1100</td>
<td>DELETE 1040</td>
<td>BO-0 and BO-1 (FIG. 4C&amp;D)</td>
<td>BO-0 to BO-1: 200 ms (M=1) BO-1 to BO-0: 6.8 ms</td>
</tr>
<tr>
<td>1105 GOTO 80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1060-1390</td>
<td>1395 GOTO 1060</td>
<td>BO-1 (FIG. 4E)</td>
<td>BO-1 to BO-1: 770 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1160-1185</td>
<td>1185 GOTO 1160</td>
<td>BO-2 &amp; BO-3 3.5V into 10K (FIG. 4F&amp;G)</td>
<td>BO-2 to BO-2: 4.5 ms (N=1) 6.5 ms (N=3) 8.5 ms (N=5) 43.2 ms (N=40) FOR N=5: BO-2 to BO-3 -- 0.31 ms BO-31 to BO-32 -- 1.0 ms BO-35 to BO-2 -- 4.2 ms</td>
</tr>
<tr>
<td>includes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>total</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>assembly</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>language</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subroutine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1060-1185</td>
<td>1185 GOTO 1060</td>
<td>BO-1 ADDED TO INTENSIFIED (FIG. H)</td>
<td>BO-1 to BO-2: 29.6 ms BO-2 to BO-1: 10.4 ms BO-1 to BO-1: 40 ms</td>
</tr>
<tr>
<td>includes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>total</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>assembly</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>language</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subroutine</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Added program to produce a repetitive pulse for timing analyses and scope photographic purposes.
### TABLE IIA  BASIC PROGRAM ALLENB7R.BAS

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td><code>CLS '--- WATER VAPOR SPECTROSCOPY EXPERIMENT, CONTROL &amp; DATA PROCESSING ---'</code></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>`GENERATES BINARY OUTPUT PULSE BO 0, CALCULATES THE MEAN OF M SETS OF THE</td>
</tr>
<tr>
<td></td>
<td>BACKGROUND DATA FROM AD0-AD3, POOKES THE FOUR BACKGROUND VALUES &amp; THE DATA</td>
</tr>
<tr>
<td></td>
<td>SAMPLE SIZE N INTO MEMORY, CONVERTS THE FOUR INTO A VOLTAGE, AND DISPLAYS THEIR</td>
</tr>
<tr>
<td></td>
<td>VALUES.</td>
</tr>
<tr>
<td>40</td>
<td>`CALLS THE ASSEMBLY LANGUAGE PROGRAM. AFTER RETURNING TO BASIC, DIVIDES</td>
</tr>
<tr>
<td></td>
<td>THE THREE CALCULATED DATA RATIOS (corrected for background noise) BY 10,000,</td>
</tr>
<tr>
<td></td>
<td>ANDDISPLAYS THESE RATIOS.</td>
</tr>
<tr>
<td>50</td>
<td>` THE ASSEMBLY LANGUAGE PROGRAM IS AGAIN CALLED AND THE PROCESS REPEATED</td>
</tr>
<tr>
<td></td>
<td>R TIMES. THE PROGRAM LISTING FOR THE BACKGROUND SAMPLE SIZE (M), DATA SAMPLE</td>
</tr>
<tr>
<td></td>
<td>SIZE (N) AND THE NUMBER OF REPEATS DURING LASER SCANNING (R) ARE DISPLAYED FOR</td>
</tr>
<tr>
<td></td>
<td>EASE OF CHANGE.</td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>`****** GENERATE BINARY OUTPUT PULSE BO 0 ***************'</td>
</tr>
<tr>
<td>80</td>
<td>`OUT 49890!,8 'Selects Reg. 12, Device 8'</td>
</tr>
<tr>
<td>90</td>
<td>`OUT 8930,0 'Zero volts from BO 0 (low byte)'</td>
</tr>
<tr>
<td>100</td>
<td>`OUT 8931,0 'Zero volts from BO 0 (high byte)'</td>
</tr>
<tr>
<td>110</td>
<td>`OUT 8930,1 '5 volts from BO 0 (low byte)'</td>
</tr>
<tr>
<td>120</td>
<td>`(high byte) SCOPE TRIG +</td>
</tr>
<tr>
<td>130</td>
<td>`OUT 8930,0 'Zero volts from BO 0 (low byte)'</td>
</tr>
<tr>
<td>140</td>
<td>`(high byte) START MEASURING BACKGROUND</td>
</tr>
<tr>
<td>150</td>
<td>`********** TAKE THE MEAN OF BACKGROUND READINGS FROM AD0-AD3 **********'</td>
</tr>
<tr>
<td>160</td>
<td>`OUT 49890!,9 'BASIC ADDRESS OF DATA ACQ ADAPTER 0 = 2E2H (AD# INPUTS)'</td>
</tr>
<tr>
<td>170</td>
<td>`M = 30 'BACKGROUND SAMPLE SIZE, M'</td>
</tr>
<tr>
<td>180</td>
<td>`AL=0:AH=0:BL=0:BH=0:CL=0:DL=0:DH=0'</td>
</tr>
<tr>
<td>190</td>
<td>`WL=0:WH=0:XL=0:XH=0:YL=0:YH=0:ZL=0:ZH=0'</td>
</tr>
<tr>
<td>200</td>
<td>`FOR J=1 TO M'</td>
</tr>
<tr>
<td>210</td>
<td>`OUT 738,0'</td>
</tr>
<tr>
<td>220</td>
<td>`OUT 738,0 'SELECT AD 0 (P1)'</td>
</tr>
<tr>
<td>230</td>
<td>`OUT 739,0'</td>
</tr>
<tr>
<td>240</td>
<td>`OUT 738,0'</td>
</tr>
<tr>
<td>250</td>
<td>`OUT 739,0'</td>
</tr>
<tr>
<td>260</td>
<td>`OUT 738,0'</td>
</tr>
<tr>
<td>270</td>
<td>`OUT 739,0'</td>
</tr>
<tr>
<td>280</td>
<td>`AL=INP (8930) 'READ AD 0'</td>
</tr>
<tr>
<td>290</td>
<td>`WL=WL+AL'</td>
</tr>
<tr>
<td>300</td>
<td>`AH=INP (8931)'</td>
</tr>
<tr>
<td>310</td>
<td>`WH=WH+AH'</td>
</tr>
<tr>
<td>320</td>
<td>`OUT 738,0 'SELECT/READ/SUM AD 1 (P2)'</td>
</tr>
<tr>
<td>330</td>
<td>`OUT 739,1'</td>
</tr>
<tr>
<td>340</td>
<td>`OUT 738,1'</td>
</tr>
<tr>
<td>350</td>
<td>`OUT 739,1'</td>
</tr>
<tr>
<td>360</td>
<td>`OUT 738,0'</td>
</tr>
<tr>
<td>370</td>
<td>`OUT 739,1'</td>
</tr>
<tr>
<td>380</td>
<td>`BL=INP (8930)'</td>
</tr>
<tr>
<td>390</td>
<td>`XL=XL+BL'</td>
</tr>
<tr>
<td>400</td>
<td>`BH=INP (8931)'</td>
</tr>
<tr>
<td>410</td>
<td>`XH=XH+BH'</td>
</tr>
<tr>
<td>420</td>
<td>`OUT 738,0 'SELECT/READ/SUM AD 2 (P3)'</td>
</tr>
<tr>
<td>430</td>
<td>`OUT 739,2'</td>
</tr>
<tr>
<td>440</td>
<td>`OUT 738,1'</td>
</tr>
<tr>
<td>450</td>
<td>`OUT 739,2'</td>
</tr>
<tr>
<td>460</td>
<td>`OUT 738,0'</td>
</tr>
<tr>
<td>470</td>
<td>`OUT 739,2'</td>
</tr>
<tr>
<td>480</td>
<td>`CL=INP (8930)'</td>
</tr>
<tr>
<td>490</td>
<td>`YL=YL+CL'</td>
</tr>
<tr>
<td>500</td>
<td>`CH=INP (8931)'</td>
</tr>
<tr>
<td>510</td>
<td>`YH=YH+CH'</td>
</tr>
</tbody>
</table>

16
520 OUT 738,0  'SELECT/READ/SUM AD 3 (P4)
530 OUT 739,3
540 OUT 738,1
550 OUT 739,3
560 OUT 738,0
570 OUT 739,3
580 DL=INP (8930)
590 ZL=DL+DL
600 DH=INP (8931)
610 ZH=ZH+DH
620 NEXT J  'REPEAT SELECT/READ/SUM DATA SET AD 0 - AD 3
630 '
640 WL=WL/M  'MEAN OF BACKGROUND READINGS FROM AD 0
650 WH=WH/M
660 XL=XL/M  'MEAN OF BACKGROUND READINGS FROM AD 1
670 XH=XH/M
680 YL=YL/M  'MEAN OF BACKGROUND READINGS FROM AD 2
690 YH=YH/M
700 ZL=ZL/M  'MEAN OF BACKGROUND READINGS FROM AD 3
710 ZH=ZH/M
720 '
730 N = 5  'DATA SAMPLE SIZE, N
740 '
750  ' ********** STORE MEAN BACKGROUND READINGS AND N IN MEMORY **********
760 DEF SEG = &H9FCA  'SEGMENT ADDRESS (page 426)
770 ADDR = &H200
780 POKE ADDR, WL  'SAVE AD 0 BACKGROUND IN MEMORY 200H
790 ADDR=ADDR+1
800 POKE ADDR, WH
810 ADDR=ADDR+1
820 POKE ADDR, XL  'SAVE AD 1 BACKGROUND IN MEMORY 202H
830 ADDR=ADDR+1
840 POKE ADDR, XH
850 ADDR=ADDR+1
860 POKE ADDR, YL  'SAVE AD 2 BACKGROUND IN MEMORY 204H
870 ADDR=ADDR+1
880 POKE ADDR, YH
890 ADDR=ADDR+1
900 POKE ADDR, ZL  'SAVE AD 3 BACKGROUND IN MEMORY 206H
910 ADDR=ADDR+1
920 POKE ADDR, ZH
930 ADDR = &H208
940 POKE ADDR, N  'SAVE DATA SAMPLE SIZE, N, IN MEMORY 208H
950 '
960  ' ********** CONVERT TO VOLTAGE AND DISPLAY ***************
970 W=.00244*(256*WH+WL)
980 X=.00244*(256*XH+XL)
990 Y=.00244*(256*YH+YL)
1000 Z=.00244*(256*ZH+ZL)
1010 PRINT "BACKGROUND", "W", "X", "Y", "Z"
1020 PRINT ,W,X,Y,Z
1030 PRINT
1040 STOP
1050  ' ********** GENERATE BINARY OUTPUT PULSE BO1 **************
1060 OUT 498901,8  'Selects Reg. 12, Device 8
1070 OUT 8930,2  '5 volts from BO 1 (low byte)
1080 OUT 8931,0  ' (high byte) 4-CHANNEL MARKER
1090 OUT 8930,0  'Zero volts from BO 1 (LOW BYTE)
1100 OUT 8931,0  'Zero volts from BO 1 (high byte)
1110 '############ PREPARE FOR ASSEMBLY LANGUAGE PROGRAM ############
1120 BLOAD "ALLEN7R.BIN",0     
1130 R = 300          'NUMBER OF REPEATS DURING LASER SCANNING, R
1140 R = 3           'NUMBER OF REPEATS DURING LASER SCANNING, R
1150 FOR S=1 TO R
1160 CALL ALLEN7R
1170 ' ############ GO TO ASSEMBLY LANGUAGE PROGRAM ############
1180 '----------- RETURN FROM ASSEMBLY LANGUAGE PROGRAM -------
1190 '********** DISPLAY DATA CALCULATED IN ASSEMBLY LANGUAGE **********
1200 DEF SEG = &H9FCA
1210 ADDR = &H270
1220 PRINT "DATA OFFSET ADDRESS", "RATIO"
1230 FOR K=1 TO 3
1240 EL = PEEK(ADDR)
1250 EH = PEEK(ADDR+1)
1260 E=(256*EH+EL)
1270 PRINT "HEX$(ADDR), "
1280 PRINT USING "####.####";E
1290 F=E/10000
1300 PRINT,,F
1310 ADDR = ADDR+8
1320 NEXT K
1330 NEXT S
1340 PRINT
1350 ' ************ DISPLAY SAMPLE SIZE OF CALCULATED DATA ************
1360 PRINT "170 M_"M_" BACKGROUND SAMPLE SIZE, M
1370 N = PEEK(&H208)
1380 PRINT "730' N_"N_" DATA SAMPLE SIZE, N
1390 PRINT "1140 R_"R_" NUMBER OF REPEATS DURING LASER SCANNING, R
1400 END
1410 'SAVE"ALLEN7R 11/3/87
OK
TABLE IIB  DISPLAY PRODUCED BY RUNNING ALLENB7R.BAS

<table>
<thead>
<tr>
<th>DATA OFFSET</th>
<th>ADDRESS</th>
<th>RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>270</td>
<td>%21154.0</td>
<td>2.1154</td>
</tr>
<tr>
<td>278</td>
<td>%31532.0</td>
<td>3.1532</td>
</tr>
<tr>
<td>280</td>
<td>%42287.0</td>
<td>4.2287</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA OFFSET</th>
<th>ADDRESS</th>
<th>RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>270</td>
<td>%21053.0</td>
<td>2.1053</td>
</tr>
<tr>
<td>278</td>
<td>%31361.0</td>
<td>3.1361</td>
</tr>
<tr>
<td>280</td>
<td>%42071.0</td>
<td>4.2071</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA OFFSET</th>
<th>ADDRESS</th>
<th>RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>270</td>
<td>%21055.0</td>
<td>2.1055</td>
</tr>
<tr>
<td>278</td>
<td>%31418.0</td>
<td>3.1418</td>
</tr>
<tr>
<td>280</td>
<td>%42143.0</td>
<td>4.2143</td>
</tr>
</tbody>
</table>

170 M = 30  'BACKGROUND SAMPLE SIZE, M
730' N = 5  'DATA SAMPLE SIZE, N
1140 R = 3  'NUMBER OF REPEATS DURING LASER SCANNING, R

Break in 1040
Ok
CONT

OK
TABLE IIIA  ASSEMBLY LANGUAGE PROGRAM ALLEN7R.LST

Microsoft (R) Macro Assembler Version 4.00 10/29/87 14:49:17
Page 1-1

;ALLEN7R --.EXE file to be controlled by BASIC

; Generates a BINARY OUTPUT PULSE (BO2) to
; indicate the START of taking data, followed
; by a BO3 pulse to be used as a marker
; at the start of each set of ADC INPUTS.
; Reads data from AD0 and subtracts W (poked
; into RAM previously from BASICA) and holds
; the DIFFERENCE, then reads data from AD1
; and subtracts X and holds this DIFFERENCE,
; etc. for AD2-Y and AD3-Z. This is repeat N
; time where 2>N>100 which was previously
; poked from BASICA. A maximum of four
; memory locations (200H, 202H, 204H, & 206H)
; are required for the BACKGROUND data pre-
; viously averaged in BASIC; and four memory
; locations (210H, 218H, 220H, & 228H) for
; subtracted data, three memory locations
; (230H, 238H, & 240H) for the RATIO data,
; one (248H) for the value of N, three
; (250H, 258H, & 260H) for the summation of
; RATIOS, and the same three (250H, 258H,
; & 260H) for the MEAN of the RATIOS all
; obtained in this ASSEMBLY LANGUAGE program.
; The values are then multiplied by 10,000d
; to eliminate the decimal point. The prog-
; ram then returns to basic.

; BASICA PROGRAM -- ALLENB7R.BAS

;**********************************************************************

ST SEG SEGMENT STACK ;DEFINE STACK SEGMENT
   DB 20 DUP ('STACK ')

ST SEG ENDS

;**********************************************************************

PROGNAM SEGMENT ;DEFINE CODE SEGMENT

MAIN PROC FAR ;MAIN PART OF PROGRAM

ASSUME CS:PROGNAM, DS:NOTHING

START: ;START EXECUTION ADDR

20
;SET UP STACK FOR RETURN (p 439)
PUSH BP ;SAVE BP
PUSH DS ;SAVE OLD DATA SEGMENT
PUSH SI ;SAVE OLD SI
MOV AX,CS ;DATA SEGMENT SAME AS
MOV DS,AX ;CODE SEGMENT
SUB AX,AX ;ZERO AX
PUSH AX ;SAVE IT ON STACK

;SELECT/HOLD DATA ACQ/CTRL ADAPT--BINARY
MOV DX,0C2E2H ;REG 12,ADPTR 0,LO B
MOV AX,8 ;BINARY DEVICE 8
OUT DX,AL ;LOW BYTE OUTPUT
MOV DX,0C3E3H ;REG 12,ADPTR 0,HI B
MOV AX,0 ;BINARY DEVICE 8
OUT DX,AL ;HIGH BYTE OUTPUT

;Select Binary Output (Write), register 2
; Set BO 2 TTL output low
MOV DX,22E2H ;REG 2,BIN OUT,LO B
MOV AX,0 ;BO 2 = 0 (LO & HI)
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,22E3H ;REG 2,BIN OUT,HI B
OUT DX,AL ;OUTPUT HIGH BYTE

; Set BO 2 TTL output high
MOV DX,22E2H ;REG 2,BIN OUT,LO B
MOV AL,4 ;BO 2 = 1, LO B
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,22E3H ;REG 2,BIN OUT,HI B
MOV AL,0 ;HIGH BYTE OF BO 2=0
OUT DX,AL ;OUTPUT HIGH BYTE

; Set BO 2 TTL output low
MOV DX,22E2H ;REG 2,BIN OUT,LO B
MOV AL,0 ;BO 2 = 0 (LO & HI)
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,22E3H ;REG 2,BIN OUT,HI B
OUT DX,AL ;OUTPUT HIGH BYTE

; Fill memory offset locations
MOV BX,0210H ;OFFSET FILL ADDR.
MOV DI,BX ; INTO REG. DI
MOV CX,80H ; # LOCATIONS TO FILL
MOV AX,0000H ;PUT # INTO AX
FILL: MOV [DI],AX ;FILL MEMORY WITH #s
INC DI ;INCREASE ADDR. BY 1
LOOP FILL ;REPEAT FILLING

;Prepare to Take 4N readings
MOV BX,0208H ;N POKED FROM BASIC 1s
MOV CX,[BX] ; # OF READ. TO AVERAGE
MOV BX,0210H ;OFFSET DATA ADDR.
MOV DI,BX ; INTO REG. DI
MOV AL,4 ;# OF ADC CHANNELS
NEWSET: PUSH CX ;HOLD # OF READING 4*N

;SELECT/HOLD DATA ACQ/CTRL ADAPT--BINARY
MOV DX,0C2E2H ;REG 12,ADPTR 0,LO B
MOV AX,8 ;BINARY DEVICE 8
OUT DX,AL ; LOW BYTE OUTPUT
MOV DX,0C3E3H ;REG 12,ADPTR 0,HI B
MOV AX,0 ;BINARY DEVICE 8
OUT DX,AL ; HIGH BYTE OUTPUT

;Select Binary Output (Write), register 2
; Set BO 3 TTL output high (WAS LOW)
MOV DX,22E2H ;REG 2,BIN OUT,LO B
MOV AL,8 ;BO 3 = 1, LO B
OUT DX,AL ; LOW BYTE OUTPUT
MOV DX,22E3H ;REG 2,BIN OUT,HI B
MOV AL,0 ;HIGH BYTE OF BO 3=0
OUT DX,AL ;OUTPUT HIGH BYTE

; Set BO 3 TTL output low
MOV DX,22E2H ;REG 2,BIN OUT,LO B
MOV AL,0 ;BO 3 = 0 (LO & HI)
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,22E3H ;REG 2,BIN OUT,HI B
OUT DX,AL ;OUTPUT HIGH BYTE

;SELECT/HOLD DATA ACQ/CTRL ADAPT--ANALOG INPUT
MOV DX,0C2E2H ;REG 12,ADPTR 0,LO B
MOV AL,9 ;DEVICE 9 LOW BYTE
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,0C2E3H ;REG 12,ADPTR 0,HI B
MOV AL,0 ;DEVICE 9 HIGH BYTE
OUT DX,AL ;OUTPUT HIGH BYTE

;PREPARE TO TAKE ANALOG INPUT DATA
SUB BX,BX ;ZERO BL (AD# = AD0)

;Select Analog Input Control Reg 0 (AD#)
; Connect AD#. Hold conv (reset bit #0 to 0)
NEXTAD: MOV DX,02E2H ;REG 0,AD# INPUT,LO
MOV AL,0 ;HOLD CONVERT
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,02E3H ;REG 0,AD# INPUT,HI
MOV AL,BL ;CODE FOR AD# 0 thru 3
OUT DX,AL ;OUTPUT HIGH BYTE

; Delay >20 us (allow transients to settle)
MOV CX,0022H ;MEASURED DELAY: 26us
TRANS: LOOP TRANS ;EXECUTE DELAY #1

; Start Conversion (set bit #0 to 1)
MOV DX,02E2H ;REG 0, AD# INPUT, LO
MOV AL,1 ;START CONVERT, LOW B
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,02E3H ;REG 0, AD# INPUT, HI
MOV AL,BL ;CODE FOR AD# 0 thru 3
OUT DX,AL ;OUTPUT HIGH BYTE

; Wait for BUSY STATE bit 0 = 0 (Polling)
MOV DX,02E2H ;REG 0, LOW BYTE
CONV: IN AX,DX ;IN FROM AI STATUS REG
AND AX,0001H ;MASK ALL BUT BIT 0
JNZ CONV ;REPEAT IF BIT 0 NOT 0

; Enable Reading of AD# Channel
MOV DX,02E2H ;REG 0, AD# INPUT, LO
MOV AX,0 ;ENABLE READING, LO B
OUT DX,AL ;OUTPUT LOW BYTE
MOV DX,02E3H ;REG 0, A/D 0 INPUT, HI
MOV AL,BL ;CODE FOR AD# 0 thru 3
OUT DX,AL ;OUTPUT HIGH BYTE

; Read AD# Input (register 2--see p71)
MOV DX,02E2H ;REG 0, AD# INPUT, LO
IN AX,DX ;INPUT READING

; Subtract background from A/D inputs
PUSH DI ;SAVE DATA ADDR.
MOV DX,0200H ;BACKGROUND DATA
ADD DX,BX ;ADDRESS
ADD DX,BX ;+ 2 TIMES
MOV DI,DX ;BL (AD#)
MOV DX,[DI] ;GET BACKGROUND
SUB AX,DX ;DIFFERENCE IN AX
POP DI ;DIFFERENCE ADDRESS
MOV [DI],AX ;DIFFERENCE IN MEMORY
ADD DI,8 ;INCREASE ADDR BY 8

;Obtain another AD# (channel # 1, 2 or 3)
INC BL ;INCREASE AD# BY 1
CMP BL,4 ;IS BL > 3 i.e. (= 4)
JNZ NEXTAD ;NO? THEN REPEAT

; Take three ratios:
MOV BX,0210H ;ADDR OF DIVISOR AD0-W
MOV AX,BX ;COPY INTO AX
ADD AX,8 ;ADDRESS OF DIVIDEND
MOV DI,AX ;AD1-X IN DI
MOV CX,BX ;ADDR OF AD1-X/AD0-W
ADD CX,20H ;OFFSET FROM DIVISOR
MOV SI,CX ; BY 20H (IN SI)
MOV CX,3 ; 3 RATIOS
NXRAT: FILD DWORD PTR [DI] ; LD DIVIDEND-80287
      FIDIV DWORD PTR [BX] ; DIVIDE INTEGER
      FSTP DWORD PTR [SI] ; STORE REAL & POP
      ADD DI,8 ; NEXT AD# DIFFERENCE
      ADD SI,8 ; NEXT AD# RATIO ADDR
      LOOP NXRAT ; CALCULATE NEXT RATIO

; Sum the three ratios
      ADD BX,20H ; ADDR OF AD1-X/ADO-W
      MOV DI,BX ; IN DI
      MOV AX,BX ; ADDR SUM AD1-X/ADO-W
      ADD AX,20H ; 20H HIGHER ADDR
      MOV SI,AX ; IN SI
      MOV CX,3 ; 3 SUMS
      NXSUM: FLD DWORD PTR [DI] ; LD REAL SUM
        FADD DWORD PTR [SI] ; ADD REAL
        FADD DWORD PTR [SI] ; ADD OF NEXT RATIO
        ADD SI,8 ; ADDR OF NEXT SUM
        LOOP NXSUM ; CALCULATE NEXT SUM

; Prepare to obtain another set of 4 AD# read.
      MOV BX,0210H ; OFFSET DATA ADDR.
      MOV DI,BX ; INTO PTR DI
      MOV BX,0000H ; SELECT AD#
      MOV AX,BX ; # READINGS REMAINING
      DEC CX ; DECREASE BY 1
      JCCXZ MEAN ; END IF CX = 0
      JMP NEWSET ; OTHERWISE LOOP

; Calculate mean (divide each sum by N)
MEAN: MOV CX,3 ; 3 GROUPS OF SUMS
      MOV BX,00208H ; ADDRESS
      MOV BX,0250H ; OFFSET SUM ADDR.
      MOV SI,BX ; INTO PTR SI
      NXMEAN: FILD DWORD PTR [SI] ; LOAD INTO 80287
        FIDIV DWORD PTR [DI] ; DIVIDE BY N
        FSTP DWORD PTR [SI] ; STORE & POP MEAN
        ADD SI,8 ; NEXT OFFSET SUM ADDR
        LOOP NXMEAN ; REPEAT FOR NEXT MEAN

; Multiply by 10000d and change to integer
      MOV BX,0268H ; ADDRESS OF MULTIPLIER
      MOV AX,2710H ; MOVx 100000d MULTIPL.
      MOV [BX],AX ; INTO ADDR. 0268H
      MOV DX,0250H ; OFFSET MEAN ADDR
      MOV DI,DX ; POINTED TO BY DI
      MOV DX,0270H ; ADDRESS OF ANSWER
      MOV SI,DX ; POINTED BY SI
      MOV CX,3 ; REPEAT FOR 3 MEANS
CONINT: FLD DWORD PTR [DI]; LOAD MEAN IN 80287
FIMUL DWORD PTR [BX]; MULTIPLY X 10000D
FISTP DWORD PTR [SI]; STORE & POP ANS
ADD DI, 8; POINT NEXT MEAN
ADD SI, 8; POINT NEXT ANSWER
LOOP CONINT; NEXT CONVERT2INTEGER

; Stop After 3 ANSWERS in memory--short integer
POP AX
POP SI
POP DS
POP BP
NOP
NOP
RET; RTN BASIC

MAIN ENDP; END OF MAIN PART OF PROGRAM
DB 400D DUP(?); RESERVE DATA AREA

;------------------------------------------
; PROGNAM ENDS
;********************************************************
END START; END ASSEMBLY
Segments and Groups:

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Align</th>
<th>Combine</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROGNAM</td>
<td>0308</td>
<td>PARA</td>
<td>NONE</td>
<td></td>
</tr>
<tr>
<td>ST_SEG</td>
<td>00A0</td>
<td>PARA</td>
<td>STACK</td>
<td></td>
</tr>
</tbody>
</table>

Symbols:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
<th>Attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONINT</td>
<td>L NEAR</td>
<td>015E</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>CONV</td>
<td>L NEAR</td>
<td>00A8</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>FILL</td>
<td>L NEAR</td>
<td>0044</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>MAIN</td>
<td>F PROC</td>
<td>0000</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>MEAN</td>
<td>L NEAR</td>
<td>012E</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>NEWSET</td>
<td>L NEAR</td>
<td>0055</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>NEXTAD</td>
<td>L NEAR</td>
<td>0088</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>NXMEAN</td>
<td>L NEAR</td>
<td>013B</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>NXRAT</td>
<td>L NEAR</td>
<td>00EE</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>NXSUM</td>
<td>L NEAR</td>
<td>010E</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>START</td>
<td>L NEAR</td>
<td>0000</td>
<td>PROGNAM</td>
</tr>
<tr>
<td>TRANS</td>
<td>L NEAR</td>
<td>0097</td>
<td>PROGNAM</td>
</tr>
</tbody>
</table>

288 Source Lines
288 Total Lines
35 Symbols

49350 Bytes symbol space free

0 Warning Errors
0 Severe Errors
<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>Operation</th>
<th>R Value</th>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>9FCA:0000</td>
<td>55</td>
<td>PUSH</td>
<td>BP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0000</td>
<td>55</td>
<td>PUSH</td>
<td>AX</td>
<td>0000</td>
<td></td>
</tr>
<tr>
<td>9FCA:0007</td>
<td>2BC0</td>
<td>SUB</td>
<td>AX, AX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0009</td>
<td>60</td>
<td>MOV</td>
<td>AX, CS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:000D</td>
<td>B80800</td>
<td>MOV</td>
<td>AX, 0008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0010</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0011</td>
<td>BAE3C3</td>
<td>MOV</td>
<td>DX, C3E3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0014</td>
<td>B80000</td>
<td>MOV</td>
<td>AX, 0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0017</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0018</td>
<td>BAE222</td>
<td>MOV</td>
<td>DX, 22E2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:001B</td>
<td>B80000</td>
<td>MOV</td>
<td>AX, 0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:001E</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:001F</td>
<td>BAE322</td>
<td>MOV</td>
<td>DX, 22E3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0022</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0023</td>
<td>BAE222</td>
<td>MOV</td>
<td>DX, 22E2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0026</td>
<td>B004</td>
<td>MOV</td>
<td>AL, 04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0028</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0029</td>
<td>BAE322</td>
<td>MOV</td>
<td>DX, 22E3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:002C</td>
<td>B000</td>
<td>MOV</td>
<td>AL, 00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:002E</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:002F</td>
<td>BAE222</td>
<td>MOV</td>
<td>DX, 22E2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0032</td>
<td>B000</td>
<td>MOV</td>
<td>AL, 00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0034</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0035</td>
<td>BAE322</td>
<td>MOV</td>
<td>DX, 22E3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0038</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0039</td>
<td>BB1002</td>
<td>MOV</td>
<td>BX, 0210</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:003C</td>
<td>8BFB</td>
<td>MOV</td>
<td>DI, BX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:003E</td>
<td>B98000</td>
<td>MOV</td>
<td>CX, 0080</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0041</td>
<td>B80000</td>
<td>MOV</td>
<td>AX, 0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0044</td>
<td>8905</td>
<td>MOV</td>
<td>[DI], AX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0046</td>
<td>47</td>
<td>INC</td>
<td>DI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0047</td>
<td>E2FB</td>
<td>LOOP</td>
<td>0044</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0049</td>
<td>BB0802</td>
<td>MOV</td>
<td>BX, 0208</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:004C</td>
<td>8B0F</td>
<td>MOV</td>
<td>CX, [BX]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:004E</td>
<td>BB1002</td>
<td>MOV</td>
<td>BX, 0210</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0051</td>
<td>8BFB</td>
<td>MOV</td>
<td>DI, BX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0053</td>
<td>B004</td>
<td>MOV</td>
<td>AL, 04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0055</td>
<td>61</td>
<td>PUSH</td>
<td>CX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0056</td>
<td>BAE2C2</td>
<td>MOV</td>
<td>DX, C2E2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0059</td>
<td>B80800</td>
<td>MOV</td>
<td>AX, 0008</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:005C</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:005D</td>
<td>BAE3C3</td>
<td>MOV</td>
<td>DX, C3E3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0060</td>
<td>B80000</td>
<td>MOV</td>
<td>AX, 0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9FCA:0063</td>
<td>EE</td>
<td>OUT</td>
<td>DX, AL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MOV DX, 22E2
MOV AL, 08
OUT DX, AL
MOV DX, 22E3
MOV AL, 00
OUT DX, AL
MOV DX, 22E2
MOV AL, 09
OUT DX, AL
MOV DX, 2CE2
MOV AL, 00
OUT DX, AL
SUB BX, BX
MOV DX, 02E2
MOV AL, 00
OUT DX, AL
MOV DX, 02E3
MOV AL, BL
OUT DX, AL
MOV CX, 0022
LOOP 0097
MOV DX, 02E2
MOV AL, 01
OUT DX, AL
MOV DX, 02E3
MOV AL, BL
OUT DX, AL
MOV DX, 02E2
IN AX, DX
AND AX, 0001
JNZ 00A8
MOV DX, 02E2
MOV AX, 0000
OUT DX, AL
MOV DX, 02E3
MOV AL, BL
OUT DX, AL
MOV DX, 22E2
IN AX, DX
PUSH DI
MOV DX, 0200
ADD DX, BX
ADD DX, BX
MOV DI, DX
MOV DX, [DI]
SUB AX, DX
POP DI
MOV [DI], AX
ADD DI, +08
INC BL
CMP BL, 04
JNZ 0088
9FCA:00DA BB1002
9FCA:00DD 8BC3
9FCA:00DF 050800
9FCA:00E2 8BF8
9FCA:00E4 8BCB
9FCA:00E6 83C120
9FCA:00E8 8BF1
9FCA:00EB B90300
9FCA:00EE 8BF8
9FCA:00F0 DB05
9FCA:00F2 DA37
9FCA:00F4 9B
9FCA:00F5 D91C
9FCA:00F7 83C708
9FCA:00FA 83C608
9FCA:00FD E2EF
9FCA:00FF 83C320
9FCA:0102 8BF0
9FCA:0104 8BC3
9FCA:0106 052000
9FCA:0109 8BF1
9FCA:010B B90300
9FCA:010E 9B
9FCA:010F D904
9FCA:0111 9B
9FCA:0112 D805
9FCA:0115 D91C
9FCA:0117 83C708
9FCA:011A 83C608
9FCA:011D E2EF
9FCA:011F BB1002
9FCA:0122 8BFB
9FCA:0124 BB0000
9FCA:0127 59
9FCA:0128 49
9FCA:0129 E303
9FCA:012B E927FF
9FCA:012E B90300
9FCA:0131 BB0802
9FCA:0134 8BFB
9FCA:0136 BB5002
9FCA:0139 8BF3
9FCA:013B 9B
9FCA:013C D904
9FCA:013D 9B
9FCA:013F DA35
9FCA:0141 9B
9FCA:0142 D91C
9FCA:0144 83C608
9FCA:0147 E2F2
9FCA:0149 BB6802
9FCA:014C B81027
9FCA:014F 8907

MOV BX, 0210
MOV AX, BX
ADD AX, 0008
MOV DI, AX
MOV CX, BX
ADD CX, +20
MOV SI, CX
MOV CX, 0003
WAIT
FILD DWord Ptr [DI]
WAIT
FIDIV DWord Ptr [BX]
WAIT
FSTP DWord Ptr [SI]
ADD DI, +08
ADD SI, +08
LOOP 00EE
ADD BX, +20
MOV DI, BX
MOV AX, BX
ADD AX, 0020
MOV SI, AX
MOV CX, 0003
WAIT
FLD DWord Ptr [SI]
WAIT
FADD DWord Ptr [DI]
WAIT
FSTP DWord Ptr [SI]
ADD DI, +08
ADD SI, +08
LOOP 010E
MOV BX, 0210
MOV DI, BX
MOV BX, 0000
POP CX
DEC CX
JCCXZ 012E
JMP 0055
MOV CX, 0003
MOV BX, 0208
MOV CX, 0003
MOV BX, 0250
MOV CX, 0000
WAIT
FLD DWord Ptr [SI]
WAIT
FIDIV DWord Ptr [DI]
WAIT
FSTP DWord Ptr [SI]
ADD SI, +08
LOOP 013B
MOV BX, 0268
MOV AX, 2710
MOV [BX], AX

29
9FCA:0151 BA5002
9FCA:0154 8BFA
9FCA:0156 BA7002
9FCA:0159 8BF2
9FCA:015B B90300
9FCA:015E 9B
9FCA:015F D905
9FCA:0161 9B
9FCA:0162 DA0F
9FCA:0164 9B
9FCA:0165 DB1C
9FCA:0167 83C708
9FCA:016A 83C608
9FCA:016D E2EF
9FCA:016F 58
9FCA:0170 5E
9FCA:0171 1F
9FCA:0172 5D
9FCA:0173 90
9FCA:0174 90
9FCA:0175 CB
9FCA:0176 90
9FCA:0177 90

MOV DX, 0250
MOV DI, DX
MOV DX, 0270
MOV SI, DX
MOV CX, 0003
WAIT
FLD DWord Ptr [DI]
WAIT
FIMUL DWord Ptr [BX]
WAIT
FISTP DWord Ptr [SI]
ADD DI, +08
ADD SI, +08
LOOP 015E
POP AX
POP SI
POP DS
POP BP
POP
POP
POP
POP
RETF
NOP
NOP
NOP
-
An assembly language program using the Intel 80386 CPU and 80387 math coprocessor chips was written to increase the speed of data gathering and processing, and provide control of a scanning CW ring dye laser system. This laser system is used in high resolution (better than 0.001 cm⁻¹) water vapor spectroscopy experiments. Laser beam power is sensed at the input and output of white cells and the output of a Fabry-Perot. The assembly language subroutine is called from Basic, acquires the data and performs various calculations at rates greater than 150 faster than could be performed by the higher level language. The width of output control pulses generated in assembly language are 3 to 4 microseconds as compared to 2 to 3.7 milliseconds for those generated in Basic (about 500 to 1000 times faster).

Included are a block diagram and brief description of the spectroscopy experiment, a flow diagram of the basic and assembly language programs, listing of the programs, scope photographs of the computer generated 5-volt pulses used for control and timing analysis, and representative water spectrum curves obtained using these programs.