

PART V - SAMPLE PROGRAMS

The following programs include short samples of how to implement various Input/Output combination functions, as well as listings of the three programs stored in EPROM memory.

5.1 Discrete and Timed Analog I/O Control Program

This example uses the program model described in Section 1.7.6.

5.1.1 PROGRAM DESCRIPTION

The hypothetical process control problem involves mixing solvent which must be kept hot. Assume that the hot fluid mix is flowing out at a constant rate, and equal volumes of two fluids (A and B) flow into the mixing pot and maintain a fairly constant level.

The control program for this simplified example must read temperature and the state of all input switches, control all the outputs, and display status on a remote CRT. An Auto/Manual switch allows operator control without shutting down the machine. All discrete switches except Emergency Stop are alternate action (toggle) type switches.

The Heater can be switched On or Off in Manual or Automatic mode. Heating power level is controlled with the Analog output, from 0 to 10 VDC. Temperature is measured with an Analog input. For the sake of simplicity, assume that the temperature sensor output ranges from 0 to 10 VDC, from 0 to 127 degrees. BCD thumbwheel switches are used to input temperature set point, from 0 to 99 degrees. The mixer must be on when heat is on.

In Auto Mode, a crude proportional control algorithm is used to set the heater output based on the temperature deviation from set point. This algorithm is not recommended for general use. In Manual Mode, the set point value is output directly to the heater.

To avoid over control of temperature, the heat is adjusted once every two minutes.

Interrupts are used for the timing function and the Emergency Stop switch.

5.1.2 I/O DESCRIPTION

Model 52 Inputs

<u>Function</u>	<u>Assigned inputs</u>
<i>ISOLATED:</i>	
Emergency Stop	<i>IN D</i>
<i>PARALLEL:</i>	
BCD Temperature Set Point (0 to 99)	0 to 7
Auto/Manual	8
Heater Enable/Disable switch	9
Manual Mixer On/Off switch	10
Manual Heat On/Off	11
<i>ANALOG:</i>	
Temperature Sensor (0 to 10VDC)	<i>IN#1</i>

Model 52 Outputs

	<u>Assigned outputs</u>
<i>PARALLEL:</i>	
Auto Mode Enabled Indicator	0
Heater On	1
Mixer On	2
Temperature High indicator	3
Temperature Low indicator	4
<i>ANALOG:</i>	
Heater Control	<i>ANALOG OUT</i>
<i>RS-232:</i>	
Status Display	<i>PRINTER</i>

5.1.3 PROGRAM LISTING

Initialization

```

10 CALL 0           (shut off all outputs just in case)
20 PUSH 1,10:CALL 3 (set analog input range, 10 VDC)
30 BAUD 9600        (set Baud rate to remote display)
40 CALL 23          (enable E. Stop switch interrupt)
50 ONEX1 600        (enable E. Stop program interrupt)
60 DIO E           (enable outputs)
70 T1=0:T2=0       (declare variables)
80 TIME=0:CLOCK1    (start the real time clock)
90 ONTIME 120,400   (enable 2 minute timer interrupt)
    
```

Main Program

```

100 SCAN 10=2,24
110 SCAN 10&9=1,25      (see SCAN discussion below)
120 SCAN 10&8=0,26
140 GOTO 100
    
```

Timed Heat Control Interrupt

```

400 CALL 4:POP T1      (read BCD Setpoint from inputs 0-7)
410 ANLG I,B,T2,1     (read temperature from analog IN#1)
420 E1=INT(T1-(T2/2)) (calculate error)
430 DIO R,8,A1        (read Auto/Manual switch)
440 IF A1=0 THEN ANLG O,B,T1 (output set point if manual)
450 IF A1<>0 THEN ANLG O,B,E1 (calculated output if auto)
460 IF E1<>0 THEN 470
470 DIO C,3:DIO C,4:GOTO 500 (indicators off if no error)
480 IF E1<0 THEN DIO C,27 ELSE DIO S,27
490 SCAN 27=!3,4      (turn one indicator off, one on)
500 GOSUB 800         (display status)
510 IF TIME>=3600 THEN H1=H1+1:TIME=TIME-3600 (count hours)
520 ONTIME TIME+120,400:RETI (re-enable, end interrupt)
    
```

Emergency Stop Interrupt

```

600 CALL 0           (shut off all outputs)
610 PRINT# "**** EMERGENCY STOP! ****",CHR(7) (beep)
620 END             (program must be restarted)
    
```

Remote Display Subroutine

```
600 PRINT# "Mixer ",
610 DIO R,24,I1:GOSUB 700          (read Mixer output state)
620 PRINT# "Heater ",
630 DIO R,25,I1:GOSUB 700        (read Heater output state)
630 PRINT# "Temperature: ",T2," Degrees"
640 PRINT# "Set Point: ",T1," Degrees"
650 PRINT# "Operating Mode: ",
660 DIO R,26,I1                  (read Auto/Manual state)
670 IF I1=0 THEN PRINT# "MANUAL" ELSE PRINT# "AUTO"
680 PRINT# "ELAPSED TIME: ",H1,":",      (display hours)
690 M1=TIME/60:PRINT# INT(M1)           (display minutes)
690 RETURN

700 IF I1=0 THEN PRINT# "OFF" ELSE PRINT# "ON"
710 RETURN
```

Sample Display Output

```
Mixer: ON
Heater: ON
Temperature: 87 Degrees
Set Point: 89 Degrees
Operating Mode: AUTO
Elapsed time: 1:36
```

5.1.4 USE OF THE SCAN INSTRUCTION

The program above provides some examples of how the SCAN instruction can be used to reduce program size. Lines 100 through 120 each do an input to output conversion that would otherwise require many lines of IF...THEN statements.

Each of the three lines takes advantage of "Dummy" I/O where dummy outputs are used to record the state of real outputs. Dummy outputs can be read as inputs also, such that the program is able to test the state of real outputs by checking their corresponding dummy inputs. The dummy I/O used have the following significance.

```
24 the Mixer is [ON or OFF]
25 the Heater is [ON or OFF]
26 the system is in [Auto or Manual] Mode
27 the fluid temperature is [High or Low]
```

When Heater output 1 is turned ON in line 110, dummy output

25 is also turned ON. In line 630, dummy input 25 indicates whether the Heater output is ON or OFF.

In line 490, the SCAN instruction is used to control the "Temperature High" and "Temperature Low" indicators connected to outputs 3 and 4. Most of the time these two will have opposite state. Line 490 uses a dummy input to establish under or over temperature and efficiently converts that to two outputs with opposite state using the invert symbol on output 3.

These examples are simplified by the use of alternate action switches. When momentary switches are used to turn a function on or off, like the Heater, the program must keep track of whether the switch means on or off.

Example:

```
100 SCAN 25&10@28=2,28
110 SCAN !10=25
```

Here, the Exclusive OR function is used to invert the state of Heater output 2 every time Heater ON/OFF switch 10 goes ON. Once again, dummy variable 28 is used to represent the state of output 2. When input 10 first goes ON, "10@28" will result in an inversion of 28 (and 2). When 10 is OFF, the result is no change.

If the switch is ON too long, the output might be inverted several times as the program loops through the SCAN statements. Dummy output 25 is used to prevent the inversion from taking place again before the switch goes OFF. When input 10 goes ON, input 25 goes OFF due to line 110 above. Then, the AND function "25&10" in line 100 prevents a second Exclusive OR inversion of the Heater output.

The combination of a dummy input with the Exclusive OR function is very handy. When the input is true (ON), the Exclusive OR inverts. When the input is false (OFF), it doesn't.

5.2 Sample Parts Counting and Motor Control Program

This program takes advantage of the STEP and DIR outputs to jog a step motor. Analog inputs are used for set point and position feedback. Parts are counted and events are timed.

5.2.1 PROGRAM DESCRIPTION

The Model 52 is in charge of painting widgets which pass by hanging from a conveyor. The speed at which they pass is controlled elsewhere, but the B52 must control the pressure used to drive the spray paint according to line speed.

Each time 100 widgets have passed, the B52 must provide a one second signal to a transfer line which controls packaging.

A 1,000 step motor is used to drive a pressure valve controlling the spray rate. The valve has 320 degrees of travel. A potentiometer is mounted to the back shaft, and wired to produce -10 VDC when the valve is closed, +10 VDC when full open. The valve should be half open when widgets are passing once per second.

The stepper translator is driven by the STEP and DIR outputs. Jog switches are required for manual valve control.

The operator wants to be able to adjust spray rate, so another potentiometer with the same voltage range is wired to provide a set point.

A photoelectric detector senses the passing widgets.

Interrupts are used for widget timing, widget counting, and output signal timing.

5.2.2 I/O DESCRIPTION

Model 52 Inputs

	<u>Assigned input</u>
<i>ISOLATED:</i>	
Run/Stop Switch	<i>IN C</i>
Jog CW Switch	<i>IN B</i>
Jog CCW Switch	<i>IN A</i>
<i>ENCODER:</i>	
Widget detector	<i>CH B</i>
Widget detector	<i>CH Z</i>
<i>ANALOG:</i>	
Motor Position	<i>IN#1</i>
Set Point	<i>IN#2</i>

Model 52 Outputs

	<u>Assigned output</u>
<i>ISOLATED:</i>	
Widget count complete	<i>OUT A</i>
Fault	<i>OUT B</i>
<i>MOTOR:</i>	
Motor Step	<i>STEP</i>
Direction	<i>DIR</i>

5.2.3 PROGRAM LISTING

Initialization

10 CALL 0	(shut off all outputs)
20 A1=0:I1=0:T1=0	(declare CTR, ANLG variables)
30 DIO E	(enable discrete outputs)
40 A2=.5:GOSUB 600	(close spray pressure valve)
50 DIO R,C,I1	(read Run/Stop switch)
60 IF I1>0 THEN 100	(go to Main Program if Run)
70 GOSUB 700:GOSUB 800	(otherwise check Jog switches)
80 GOTO 50	

Main Program

```

100 CALL 21:ONEX1 300      (enable widget interrupt)
110 CTR 1,C,D,100        (enable interrupt on 100 widgets)
120 ANLG I,B,A1,2        (read set point)
130 A2=(A1+T1)/2        (average the set point with line rate)
140 ANLG I,B,A1,1        (read valve position)
150 IF A1<A2 THEN GOSUB 600      (open the valve)
160 IF A1>A2 THEN GOSUB 500      (or close the valve)
170 DIO R,C,I1          (check Run/Stop input)
180 IF I1=1 THEN 120
190 CALL 22:CLEARI:GOTO 40      (disable interrupts if Stop)

```

Interrupt Service routine

```

300 CALL 1:POP X1          (fetch interrupt source code)
310 IF X1=0 THEN RETI      (return if no more interrupts)
320 X2=2                   (initialize logical mask for interrupt bits)
330 FOR I2=1 TO 5         (test interrupt bits 1 to 5)
340 IF X1.AND.X2=0 THEN 360      (ignore bit if zero)
350 ON I2 GOSUB 430,490,450,400  (go process the interrupt)
360 X2=X2+X2              (increment bit mask)
370 NEXT:GOTO 300

400 DIO S,A                (signal 100 widgets)
410 CTR 1,C,D,100        (restart widget counter)
420 CTR 3,T,D,1000      (start 1 second signal timer)
430 RETURN

450 CTR 2,R,T1           (read widget timer)
460 CTR 2,T,T           (restart widget timer)
470 T1=128/T1          (scale time to valve position)
480 RETURN

490 DIO C,A:RETURN      (shut off 100 widget signal)

```

Motor Control Subroutines

500 DIO C,E	(set motor direction to Close valve)
510 DO:PWM 200,200,5	(send 5 steps to the translator)
520 ANLG R,B,A1,1	(read the valve position pot)
530 UNTIL A1>A2	(repeat until feedback is correct)
540 RETURN	
600 DIO S,E	(set motor direction to Open valve)
610 DO:PWM 200,200,5	(send 5 steps to the translator)
620 ANLG R,B,A1,1	(read the valve position pot)
630 UNTIL A1<A2	(repeat until feedback is correct)
640 RETURN	
700 DIO S,E	(set motor direction to Open valve)
710 DIO R,B,I1	(read the Jog switch)
720 IF I1=0 RETURN	(return if it has gone OFF)
730 PWM 2000,2000,10	(send 10 steps to the translator)
740 GOTO 710	(repeat)
800 DIO C,E	(set motor direction to Close valve)
810 DIO R,A,I1	(read the Jog switch)
820 IF I1=0 RETURN	(return if it has gone OFF)
830 PWM 2000,2000,10	(send 10 steps to the translator)
840 GOTO 810	(repeat)

5.3 The Directory Program

The following program is stored permanently in the B52's permanent EPROM memory. It is available for running for inspecting memory contents and running stored programs by issuing the LOAD instruction while in the Direct command mode.

```

10 REM *** MODEL 52 DIRECTORY PROGRAM ***
20 PRINT:PRINT:PRINT
30 PRINT "Compumotor Model 52 Program Directory"
40 PRINT " (use UPPER CASE, please)"
50 N1=1:IF XBY(8010H)=55H THEN 300 (any EEPROM programs?)
60 PRINT "No Programs are Saved in Memory"
70 N1=0:GOTO 300 (55h starts EEPROM programs)

100 A1=8011H (N1 is the program counter)
110 PRINT "Press 'Escape' to stop the search"
120 PRINT:GOSUB 200 (display 1st program)
130 DO (start search loop)
140 A2=GET:IF A2=27 THEN A1=0BFF0H (test for Escape)
150 IF XBY(A1)<>01H THEN 180 (test for end of program)
160 IF XBY(A1+1)=55H THEN 170 (test for next program)
165 A1=0BFF0H:GOTO 180 (end)
170 N1=N1+1:A1=A1+2:GOSUB 200:GOTO 180 (count & display)
180 N2=XBY(A1):A1=A1+N2 (advance one program line)
190 UNTIL A1>=0BFF0H (repeat search for 01h)
195 PRINT "That's all":GOTO 300

200 PRINT "Program No.",N1," Starts at address",
210 PH0. A1-1, (display program no.& start address)
220 IF XBY(A1+3)=96H THEN 240 (test for REM statement)
230 PRINT " (no title)":RETURN (no REM)
240 PRINT:PRINT "Title: ", (REM found)
250 A2=XBY(A1)-5:A3=A1+3 (juggle character pointers)
260 FOR I1=1 TO A2
270 PRINT CHR(XBY(A3+I1)), (display REM remark)
280 NEXT:PRINT:RETURN

```

```

300 PRINT:PRINT "Press a Key to Select an Option:"
310 PRINT "A. Run the I/O Test Program"
320 PRINT "B. Run the CX/372/2100 Program"
325 PRINT "C. Stop the Program"          (display options)
330 IF N1=0 THEN 350                    (include EEPROM if any)
340 PRINT "D. Run a Stored Program"
345 PRINT "E. Search memory for a Stored Program":PRINT
350 A=GET:IF A=0 THEN 350                (wait for keystroke)
360 IF A=65 THEN CALL 50                 (test for A key)
370 IF A=66 THEN CALL 51                 (test for B key)
375 IF A=67 THEN PRINT:PRINT:STOP       (test for C key)
380 IF N1=0 THEN PRINT CHR(7);:GOTO 350 (beep, wrong key)
385 IF A=68 THEN 400                     (test for D key)
390 IF A=69 THEN 100                     (test for E key)
395 PRINT CHR(7);:GOTO 350               (beep, wrong key)

400 PRINT:PRINT "Enter the Program Number (10 max)"
410 INPUT "(enter 0 to start over)",I1   (select program)
420 IF I1>N1 THEN PRINT CHR(7);:GOTO 400
430 IF I1<=0 THEN 20                     (test for valid no.)
435 IF I1>10 THEN PRINT CHR(7);:GOTO 400
440 ON I1 GOTO 1,450,451,452,453,454,455,456,457,458,459
450 RROM1
451 RROM2
452 RROM3
453 RROM4
454 RROM5                               (branch to run selected EEPROM program)
455 RROM6
456 RROM7
457 RROM8
458 RROM9
459 RROM10

```

Notable features of the above program include the use of the 8052's start-of-program and end-of-program characters (55 hex and 01 hex, respectively). To give any stored program a title, the first line must be a REM statement with the title. CALL 35 loads a program from EPROM to RAM and runs it.

5.4 The Test and Demonstration Program

The following program is stored permanently in the B52's permanent EPROM memory. This program is especially useful in troubleshooting situations when a B52 is installed and mysterious malfunctions occur. I/O function can be tested in many cases by connecting inputs to outputs.

This program allows testing the various I/O functions without loading any other diagnostic programs. It is accessible through the Directory program above by issuing the **LOAD** instruction while in the Direct command mode.

5.4.1 PROGRAM DESCRIPTION

The program has several menus of I/O operations to exercise the various B52 I/O devices. Messages displayed by this program are explained below, followed by the program listing.

Opening message:

**** Model 52 Test and Demonstration ****

READ/WRITE MEMORY SIZE = 15770 BYTES
PROGRAM SIZE = 6735 BYTES
MEMORY AVAILABLE = 8523 BYTES

STRIKE A KEY TO PROCEED

(More memory is available when programs are run from EEPROM)

Main Menu:

PRESS A LETTER KEY TO SELECT A FUNCTION

A. ANALOG I/O TEST
E. ENCODER TEST
I. ISOLATED I/O TEST
M. MOTOR OUTPUT TEST
P. PARALLEL I/O TEST
R. RS-232 COMMUNICATIONS TEST

For this and subsequent menus, it is only necessary to press a key once. Do not press RETURN.

Capital letters are required for this program; press the CAPS LOCK key if required. It is easy to get out of any menu chosen.

Pressing "A" results in the Analog test menu:

ANALOG I/O TEST

PRESS S TO SET THE OUTPUT VOLTAGE
PRESS R TO READ THE INPUTS
PRESS A FOR AUTOMATIC CYCLE
PRESS + OR - TO INCREMENT THE OUTPUT
PRESS I TO CHANGE THE INCREMENT
PRESS Space TO EXIT ANALOG TESTING

This section of the program allows setting the output voltage, and displaying the input voltage on all four channels.

The "+" and "-" keys will increase or decrease the output voltage by a fixed increment. This increment can be adjusted by pressing "I".

The "A" key allows either repetitive incrementing of the output from minimum to maximum voltage, or a running display of input voltage on all four channels, or both. The output increment is adjustable with the "I" option.

Pressing the Space Bar returns the program to the Main Menu. If "E" is then pressed, the following display appears:

ENCODER TEST

PRESS A KEY TO RESTART THE COUNT
PRESS Space TO EXIT ENCODER TESTING

TOTAL QUADRATURE COUNT:

0

Once an incremental encoder is powered up and connected to the *ENCODER* inputs, the program will display any change in the position of the encoder.

If the Main Menu "I" option is selected, the following menu is displayed:

ISOLATED I/O TEST

PRESS O TO INVERT AN OUTPUT
PRESS I TO CHECK THE INPUTS
PRESS A FOR OUTPUT CYCLE
PRESS Space TO EXIT ISOLATED I/O TEST

This is actually the only program option that has any visible results when no I/O is connected. The four output indicators will turn on when their associated output is turned on. The "I" option displays the state of all four ISOLATED inputs. The "A" option cycles through all four outputs in sequence, and displays the state of all four ISOLATED inputs each time.

STEP OUTPUT TEST

PRESS R TO REVERSE MOTOR DIRECTION
PRESS F FOR FREQUENCY OUTPUT
PRESS M FOR MOTOR CYCLE
PRESS Space TO EXIT

The Main Menu "M" option results in the above display. If a Compumotor motor drive is connected to the STEP and DIR outputs, either the "F" or "M" options will move the motor.

PARALLEL I/O TEST

PRESS O TO INVERT AN OUTPUT
PRESS F TO CHANGE INPUT FORMAT
PRESS I TO CHECK THE INPUTS
PRESS A FOR AUTOMATIC CYCLE
PRESS Space TO EXIT PARALLEL TESTING

The Main Menu "P" option results in this display. Any of the 24 outputs (0 through 23) on the *PARALLEL OUTPUT* connector can be turned on or off using the "O" key. "I" results in a display of all 24 discrete inputs.

The "F" option provides several format display options for the 24 *PARALLEL INPUTS*. The inputs may be displayed in 6 digit BCD format, or the lower 16 inputs may be displayed as a binary number. Input state can be inverted.

The "A" option results in a cycle of all the outputs with the inputs displayed for each output change. Connecting outputs to inputs provides a function test of both input and output.

RS232 PORT TEST

CURRENT RS232 PARAMETERS:

BAUD RATE: 9600
PARITY: NONE
CHARACTER LENGTH: 8 BITS
NO. OF STOP BITS: 1
MODE: FULL DUPLEX
CURRENT COMMUNICATIONS PORT= 0

PRESS C TO CHANGE PORTS
PRESS P TO CHANGE PARAMETERS
PRESS A FOR AUTO STRING TRANSMISSION
PRESS K FOR KEYBOARD TRANSMISSION
PRESS Space TO EXIT PORT TEST

This display follows selection of the "R" option from the Main Menu. Either port, *RS232#0* or *RS232#1*, may be chosen for communications.

Communications protocol can be set using the "P" option. For communicating with devices that do not echo what they receive, use the "E" option when setting protocol.

The "A" option will automatically transmit a string of 93 ASCII characters out at a high rate. The "K" option sends *CONSOLE* keyboard characters out the port, and sends any characters coming into the port to the *CONSOLE* display.

This is the mode to use for testing interactive remote devices.

5.4.2 PROGRAM LISTING

Opening Message

```
10 PRINT "*** Model 52 Test and Demonstration ** "  
20 CALL 0:DIO E (outputs off, enabled)  
30 STRING 82,79:X1=0 (reserve string space)  
40 PRINT "READ/WRITE MEMORY SIZE =",M TOP+1," BYTES"  
50 PRINT "PROGRAM SIZE =",LEN," BYTES"  
60 PRINT "MEMORY AVAILABLE =",FREE," BYTES"  
70 PRINT:PRINT "STRIKE A KEY TO PROCEED"  
80 A=GET:IF A=0 THEN 80 (wait for Console input)
```

Main Menu

```

90 GOSUB 260
100 PRINT "PRESS A LETTER KEY TO SELECT A FUNCTION"
110 PRINT:PRINT "A. ANALOG I/O TEST"
120 PRINT "E. ENCODER TEST"
130 PRINT "I. ISOLATED I/O TEST"
140 PRINT "M. MOTOR OUTPUT TEST"
150 PRINT "P. PARALLEL I/O TEST"
160 PRINT "R. RS-232 COMMUNICATIONS TEST"
170 A=GET:IF A=0 THEN 170
180 IF A=65 THEN GOSUB 500:GOTO 90           (test for A key)
190 IF A=69 THEN GOSUB 1000:GOTO 90        (test for E key)
200 IF A=77 THEN GOSUB 1200:GOTO 90        (test for M key)
210 IF A=73 THEN GOSUB 1600:GOTO 90        (test for I key)
220 IF A=80 THEN GOSUB 2100:GOTO 90        (test for P key)
230 IF A=82 THEN GOSUB 2600:GOTO 90        (test for R key)
240 IF A>96 THEN IF A<123 THEN PRINT "UPPER CASE, PLEASE"
250 PRINT CHR(7);:GOTO 170                 (beep, wrong key)

260 FOR I=0 TO 5:PRINT:NEXT I:RETURN

270 FOR I=0 TO 38:PRINT "~";:NEXT I
280 PRINT:RETURN
    
```

Analog I/O Section

```

500 GOSUB 260:N1=.3125
510 PRINT "ANALOG I/O TEST":GOSUB 270
520 PRINT:PRINT "PRESS S TO SET THE OUTPUT VOLTAGE"
530 PRINT "PRESS R TO READ THE INPUTS"
540 PRINT "PRESS A FOR AUTOMATIC CYCLE"
550 PRINT "PRESS + OR - TO INCREMENT THE OUTPUT"
560 PRINT "PRESS I TO CHANGE THE INCREMENT"
570 PRINT "PRESS Space TO EXIT ANALOG TESTING":PRINT
580 A=GET:IF A=0 THEN 580
590 IF A=32 THEN PRINT USING(0):GOTO 90     (if space, exit)
600 IF A=43 THEN GOSUB 670                  (test for "+")
610 IF A=45 THEN GOSUB 690                  (test for "+")
620 IF A=65 THEN 710                        (test for "-")
630 IF A=73 THEN 820                        (test for "I")
640 IF A=82 THEN GOSUB 840:PRINT:GOTO 520  (test for "R")
650 IF A=83 THEN GOSUB 900:GOTO 520        (test for "S")
660 PRINT CHR(7);:GOTO 580
    
```

Output Increment Routine

```

670 V1=V1+N1:IF V1>10 THEN V1=10           (increment voltage)
680 GOTO 930                                (output voltage)
690 V1=V1-N1:IF V1<-10 THEN V1=-10        (decrement voltage)
700 GOTO 930                                (output voltage)

```

Automatic Cycle Routine

```

710 V1=0:PRINT "PRESS O FOR OUTPUT ONLY"
720 PRINT "PRESS I FOR INPUT ONLY"
730 PRINT "OR PRESS Space FOR BOTH"
740 B=GET:IF B=0 THEN 740
750 PRINT:PRINT "PRESS ANY KEY TO STOP":PRINT
760 M=0:IF B=79 THEN M=2:GOTO 780          (test for output only)
770 GOSUB 840:PRINT CR,:IF B=73 THEN M=1   (set up for input)
780 ON M GOSUB 800,810,950                 (branch to I/O routine)
790 A=GET:IF A=0 THEN 780 ELSE 510        (test for exit)
800 GOSUB 950                              (increment output)
810 GOSUB 860:RETURN                       (display inputs)

820 PRINT:PRINT "VOLTAGE INCREMENT=",N1
830 INPUT "NEW INCREMENT? ",N1:GOTO 520

```

Input Display Routine

```

840 PRINT "INPUT VOLTAGE:"
850 PRINT "CHANNEL 1 2 3 4"
860 PRINT SPC(7), : FOR I=1 TO 4
870 ANLG I,V,X1,I                          (read channel volts)
880 PRINT USING(##.##),X1,                 (format display)
890 NEXT:RETURN

```

Set Output Routine

```

900 INPUT "OUTPUT VOLTAGE? (-10 TO +10) ",V1
910 IF V1<=10 THEN IF V1>=-10 THEN 930    (test for valid range)
920 PRINT CHR(7),:GOTO 900
930 ANLG O,V,V1                             (set output voltage)
940 RETURN

```

Output Cycle Routine

```

950 V1=V1+N1
960 IF V1>10 THEN V1=-10
980 GOTO 930

```

Encoder Section

```

1000 GOSUB 260:PRINT "ENCODER TEST"
1010 GOSUB 270:PRINT "PRESS A KEY TO RESTART THE COUNT"
1020 PRINT:PRINT "PRESS Space TO EXIT ENCODER TEST"
1030 PRINT:PRINT "TOTAL QUADRATURE COUNT:"
1040 PRINT SPC (5),"0", SPC (10), CR ,
1050 QUAD C (start counting)
1060 QUAD R,X1 (read the count)
1070 IF X1=X2 THEN 1100 (ignore if no change)
1080 PRINT SPC (5),X1, CR , (display new count)
1090 X2=X1
1100 A=GET:IF A=0 THEN 1060 (repeat if no Console)
1110 IF A=32 THEN 90 (exit if Space)
1120 GOTO 1010 (else restart)

```

Motor Control Section

```

1200 GOSUB 260:PRINT "STEP OUTPUT TEST"
1210 GOSUB 270
1220 PRINT "PRESS F FOR FREQUENCY OUTPUT"
1230 PRINT "PRESS M FOR MOTOR CYCLE"
1240 PRINT "PRESS Space TO EXIT"
1250 A=GET:IF A=0 THEN 1250
1260 IF A=70 THEN 1400 (branch on F)
1270 IF A=77 THEN 1280 ELSE RETURN (test for M)
1280 PRINT:PRINT "PRESS ANY KEY TO STOP"
1290 A=5120:B=16 (set initial values)
1300 DO:PWM 20,A,B (output B pulses)
1310 A=A/2:B=B*2 (reduce pulse width, increase freq.)
1320 UNTIL A=20 (loop)
1330 PWM 20,20,10000 (output 10,000 pulses a max rate)
1340 DO:PWM 20,A,B
1350 A=A*2:B=B/2 (increase pulse width, reduce freq.)
1360 UNTIL A>=5120 (loop)
1370 A=GET:IF A<>0 THEN 1390 (exit if Console)
1380 DIO I,E:GOTO 1290 (invert DIR)
1390 RETURN

```

Frequency Output Routine

```

1400 PRINT:INPUT "ENTER FREQUENCY (10HZ-20KHZ) ",F1
1410 IF F1<0 THEN 1400 ELSE IF F1>20000 THEN 1400
1420 N=INT(1000/(F1*.001085)/2)           (calculate pulse width)
1430 PRINT "ACTUAL FREQUENCY WILL BE ",1/(N*2*.000001085)
1440 PRINT "ENTER NUMBER OF PULSES (65,535 MAX)"
1450 INPUT "(0 FOR REPEAT) ",P
1460 IF P<0 THEN 1450                     (limit no. pulses)
1470 IF P>65535 THEN 1450
1480 IF P<>0 THEN F1=P
1490 PWM N,N,F1                           (P=0 repeats every second)
1500 IF P>0 THEN 1510
1505 A=GET:IF A=0 THEN 1490 ELSE 1210     (stop on Console)
1510 PRINT "PRESS Space TO REPEAT"
1520 A=GET:IF A=0 THEN 1520
1530 IF A=32 THEN 1490 ELSE 1210
    
```

Isolated I/O Section

```

1600 F=1:GOSUB 260
1610 PRINT "ISOLATED I/O TEST":GOSUB 270
1620 PRINT:PRINT "PRESS O TO INVERT AN OUPUT"
1630 PRINT "PRESS I TO CHECK THE INPUTS"
1640 PRINT "PRESS A FOR OUTPUT CYCLE"
1650 PRINT "PRESS Space TO EXIT ISOLATED I/O TEST":PRINT
1660 A=GET:IF A=0 THEN 1660
1670 IF A=65 THEN GOSUB 1830:GOTO 1620     (test for A)
1680 IF A=73 THEN GOSUB 1720:PRINT:GOTO 1620 (test for I)
1690 IF A=79 THEN GOSUB 1780:GOTO 1620     (test for O)
1700 IF A=32 THEN RETURN                   (exit on Space)
1710 GOTO 1660
    
```

Input Display Routine

```

1720 PRINT "OPTICAL INPUTS: A B C D"
1730 PRINT SPC (15),
1740 FOR I1=65 TO 68                       (65 to 68 are ASCII A to D)
1750 DIO R,I1,X1                           (read input to X1)
1760 PRINT X1,:NEXT                        (display)
1770 RETURN
    
```

Output Set Routine

```
1780 PRINT "SELECT OUTPUT (A-D)"
1790 X1=GET:IF X1=0 THEN 1790      (wait for Console)
1800 IF X1<65 THEN 1790 ELSE IF X1>68 THEN 1790 (test for A to D)
1810 DIO I,X1                      (invert output)
1820 RETURN
```

Automatic Cycle Routine

```
1830 PRINT "PRESS ANY KEY TO STOP:"
1840 GOSUB 1720:X2=65:PRINT CR,    (set up input display)
1850 DIO S,X2                      (output On)
1860 FOR I=0 TO 10:NEXT           (wait)
1870 GOSUB 1730:PRINT CR,
1880 DIO C,X2                      (output Off)
1890 X2=X2+1:IF X2>68 THEN X2=65 (next output)
1900 A=GET : IF A=0 THEN 1850     (exit on Console)
1910 GOTO 1600
```

Parallel Input Format subroutine

```
1920 PRINT:PRINT "SELECT INPUT FORMAT:"
1930 PRINT "PRESS B FOR BINARY (16 BIT)"
1940 PRINT "PRESS C FOR BCD"
1950 PRINT "PRESS D FOR DISCRETE"
1960 PRINT "PRESS I TO INVERT"
1970 A=GET:IF A=0 THEN 1970
1980 IF A=66 THEN F1=0:PRINT "INPUT FORMAT: BINARY"
1990 IF A=67 THEN F1=2:PRINT "INPUT FORMAT: BCD"
2000 IF A=68 THEN F1=1:PRINT "INPUT FORMAT: DISCRETE"
2010 IF A<>73 THEN RETURN
2020 PRINT "INPUT FORMAT: INVERT"
2030 CALL 14:F1=-F1:GOTO 1970
```

Parallel I/O Section

```

2100 F1=1:GOSUB 260
2110 PRINT "PARALLEL I/O TEST"
2120 GOSUB 270
2130 PRINT "PRESS O TO INVERT AN OUTPUT"
2140 PRINT "PRESS F TO CHANGE INPUT FORMAT"
2150 PRINT "PRESS I TO CHECK THE INPUTS"
2160 PRINT "PRESS A FOR AUTOMATIC CYCLE"
2170 PRINT "PRESS Space TO EXIT PARALLEL TESTING"
2180 A=GET:IF A=0 THEN 2180
2190 PRINT:IF A=65 THEN GOSUB 2440:GOTO 2130
2200 IF A=70 THEN GOSUB 1920:GOTO 2180           (test for F)
2210 IF A=73 THEN GOSUB 2240:PRINT:GOTO 2130    (test for I)
2220 IF A=79 THEN GOSUB 2340:GOTO 2180         (test for O)
2230 IF A<>32 THEN 2180 ELSE RETURN             (exit on Space)

```

Discrete Input Display Routine

```

2240 PRINT "PARALLEL INPUTS:"
2250 IF ABS(F1)<>1 THEN 2390           (branch if not discrete format)
2260 PRINT "  0-7      8-15      15-23"
2270 FOR I=0 TO 2                      (loop 3 ports)
2280 FOR I2=0 TO 7                      (loop 8 inputs each)
2290 I3=I1*8+I2
2300 DIO R,I3,X1                        (read input)
2310 PRINT CHR(X1+48),                  (display state, 0 or 1)
2320 NEXT:PRINT " ",:NEXT:PRINT CR ,
2330 RETURN

```

Output Set Routine

```

2340 INPUT "SELECT OUTPUT (0-23) ",C1
2350 IF C1<0 THEN 2340
2360 IF C1>23 THEN 2340
2370 DIO I,C1                           (invert output, 0 to 23)
2380 GOTO 2240

```

Numerical Input Display Routine

```

2390 PRINT "THE NUMBER ON THE INPUTS IS:"
2400 ON ABS(F1) GOTO 2410,2410,2430
2410 CALL 12:POP X1
2420 PRINT X1," BINARY (" ,           (display 16 inputs as binary no.)
2425 PHO. X1,:PRINT ")", CR ,:RETURN    (and Hex)
2430 CALL 8:POP X1
2435 PRINT X1," BCD",CR ,:RETURN (display 24 inputs as BCD no.)

```

Automatic Cycle Routine

```

2440 PRINT "PRESS ANY KEY TO STOP"
2450 GOSUB 2240:X2=0           (set up input display)
2460 B=GET:IF B<>0 THEN RETURN (exit on Console)
2470 DIO S,X2                 (turn On output)
2480 GOSUB 2270               (display inputs)
2490 DIO C,X2                 (turn Off output)
2500 X2=X2+1                  (next output)
2510 IF X2>23 THEN X2=0
2520 GOTO 2460
    
```

RS-232 Communications Test

```

2600 GOSUB 260:PRINT "RS232 PORT TEST"
2610 GOSUB 270:A1=0FA00H      (set Port 0 address)
2620 P2=0:B1=4:S1=0:C1=0:P1=0:M1=0 (set protocol variables)
2630 GOSUB 3390:GOSUB 3690    (enable communications)
    
```

Protocol Display Routine

```

2635 PRINT "CURRENT RS232 PARAMETERS:"
2640 PRINT "BAUD RATE:",TAB(25),2**(9-B1)*300 (B1 is Baud rate)
2650 PRINT "PARITY:",TAB(25),
2660 IF P1=1 THEN PRINT "ODD"                 (P1 is Parity)
2670 IF P1=2 THEN PRINT "EVEN"
2680 IF P1=0 THEN PRINT "NONE"
2690 PRINT "CHARACTER LENGTH:", TAB (25),      (C1 is Data bits)
2700 IF C1=0 THEN PRINT "8 BITS" ELSE PRINT "7 BITS"
2710 PRINT "NO. OF STOP BITS:",TAB(25),        (S1 is Stop bits)
2720 IF S1=0 THEN PRINT "1" ELSE PRINT "2"
2730 PRINT "MODE:",TAB(25),                    (M1 is Mode)
2740 IF M1=0 THEN PRINT "FULL DUPLEX", ELSE PRINT "HALF DUPLEX",
2750 IF M1=2 THEN PRINT " WITH ECHO" ELSE PRINT
2760 FOR I=1 TO 1000:NEXT I
    
```

Execution Menu

```
2770 PRINT "CURRENT COMMUNICATIONS PORT= ",P2
2780 PRINT:PRINT "PRESS C TO CHANGE PORTS"
2790 PRINT "PRESS P TO CHANGE PARAMETERS"
2800 PRINT "PRESS A FOR AUTO STRING TRANSMISSION"
2810 PRINT "PRESS K FOR KEYBOARD TRANSMISSION"
2820 PRINT "PRESS Space TO EXIT RS-232 TEST"
2830 A=GET:IF A=0 THEN 2830
2840 PRINT
2850 IF A=65 THEN 3050 (test for A)
2860 IF A=75 THEN 2980 (test for K)
2870 IF A=67 THEN 2910 (test for C)
2880 IF A=80 THEN 3160 (test for P)
2890 IF A<>32 THEN 2900 ELSE RETURN (exit on Space)
2900 PRINT CHR(7);:GOTO 2830
```

Port Change Routine

```
2910 PRINT "NEW COMMUNICATIONS PORT (1 OR 0) "
2920 A=GET:IF A=0 THEN 2920
2930 IF A<48 THEN 2910
2940 IF A>49 THEN 2910 (test for 1 or 0)
2950 P2=A-48 (P2 identifies Port)
2960 A1=0FA00H+P2*200H
2970 GOTO 2630
```

Interactive Console Routine

```
2980 PRINT:PRINT "ALL KEYSTROKES WILL BE TRANSMITTED"
2990 PRINT " (PRESS ESC TO EXIT)"
3000 IF M1=0 THEN PRINT "RECEIVED ECHO:",
3010 A=GET:IF A=0 THEN GOSUB 3110:GOTO 3010 (check Console)
3020 IF A=27 THEN PRINT:GOTO 2770 (exit if Escape)
3030 GOSUB 3080 (transmit char.)
3040 GOTO 3010
```

Automatic String Routine

```
3050 FOR A=20H TO 70H
3060 GOSUB 3080 (transmit 93 ASCII characters)
3070 NEXT A:PRINT:GOTO 2770
```

Transmit and Receive Routines

```

3080 IF P2=0 THEN XMT 0,$A,      (transmit char. with code A)
3090 IF P2=1 THEN XMT 1,$A,
3100 IF M1=0 THEN 3110 ELSE PRINT CHR(A),(display if half duplex)
3110 IF P2=1 THEN 3120
3120 CALL 31:POP C1              (test for received chars)
3130 IF C1<>0 THEN RCV 0,0,C1:PRINT $(0), (display if any)
3135 RETURN
3140 CALL 32:POP C1              (test for received chars)
3150 IF C1<>0 THEN RCV 1,0,C1:PRINT $(0), (display if any)
3155 RETURN
    
```

Set Baud Rate Routine

```

3160 INPUT "NEW BAUD RATE (300-19200)",B1
3170 B1=INT(B1/300)              (Baud rate is encoded)
3180 I=0:DO:B1=B1/2:I=I+1        (and written to UART registers)
3190 UNTIL B1<1:B1=10-I:PRINT    (B1 is the code, see 3690)
    
```

Set Parity Routine

```

3200 PRINT "PRESS A LETTER KEY TO SET PARITY"
3210 PRINT " (E FOR EVEN, O FOR ODD, N FOR NONE)"
3220 A=GET:IF A=0 THEN 3220
3230 P1=0:C1=40H                 (set 7 data bits)
3240 IF A=69 THEN P1=2:GOTO 3300 (skip data bits if Even)
3250 IF A=79 THEN P1=1:GOTO 3300 (skip data bits if Odd)
    
```

Set Data Bits Routine

```

3260 PRINT "PRESS A NUMBER KEY FOR CHARACTER LENGTH"
3270 PRINT " (7 OR 8 BITS)"      (no. of Data bits is encoded)
3280 A=GET:IF A=0 THEN 3280      (and written to UART registers)
3290 IF A<>55 THEN C1=0          (C1 is the code, see 3690)
    
```

Set Stop Bits Routine

```

3300 PRINT:PRINT "PRESS A NUMBER KEY FOR STOP BITS"
3310 PRINT " (1 OR 2)":PRINT     (no. of Stop bits is encoded)
3320 A=GET:IF A=0 THEN 3320      (and written to UART registers)
3330 IF A=50 THEN S1=20H ELSE S1=0 (S1 is the code, see 3690)
    
```

Set Mode Routine

```

3340 PRINT "PRESS F FOR FULL, H FOR HALF DUPLEX"
3350 PRINT "PRESS E TO ECHO RECEIVED CHARACTERS"
3355 A=GET:IF A=0 THEN 3355
3360 M1=0:IF A=69 THEN M1=2      (test for E)
3370 IF A=72 THEN M1=1          (test for H)
3380 GOTO 2630
    
```

Configure Receiver/Transmitter Routine

The following portion of the Test and Demonstration Program works out the various permutations of possible port and protocol combinations selected above. This is necessary because the COM instruction does not allow variable parameters. Subroutine 3690 below reduces the huge number of permutations by directly writing Stop bits, Data bits, and Baud rate information directly to the 8256 UART control registers.

```

3390 ON P2 GOTO 3400,3540      (select Port 0 or 1)
3400 ON M1 GOTO 3410,3410,3480 (select Echo or not)
3410 ON P1 GOTO 3420,3440,3460 (select Parity)
3420 COM 0,,N,,R
3430 RETURN
3440 COM 0,,O,,R
3450 RETURN
3460 COM 0,,E,,R
3470 RETURN
3480 COM 0,,N,,E
3490 RETURN
3500 COM 0,,O,,E
3510 RETURN
3520 COM 0,,E,,E
3530 RETURN
3540 ON M1 GOTO 3550,3550,3620
3550 ON P1 GOTO 3560,3580,3600
3560 COM 1,,N,,R
3570 RETURN
3580 COM 1,,O,,R
3590 RETURN
3600 COM 1,,E,,R
3610 RETURN
3620 ON P1 GOTO 3630,3650,3670
3630 COM 1,,N,,E
3640 RETURN
3650 COM 1,,O,,E
3660 RETURN
3670 COM 1,,E,,E
3680 RETURN
    
```

Direct UART Control Routine

```
3690 PRINT
3700 A=XBY(A1)      (read 1st UART register)
3710 A=A.AND.0FH    (erase 4 upper bits)
3720 A=A.OR.C1      (set # Data bits)
3730 A=A.OR.S1      (set # Stop bits)
3740 XBY(A1)=A      (write to UART)
3750 A=XBY(A1+1)    (read 2nd UART register)
3755 A=A.AND.0F0H   (erase 4 lower bits)
3760 A=A.OR.B1      (set Baud rate)
3770 XBY(A1+1)=A    (write to UART)
3780 RETURN
```

5.5 The CX/372/2100 Control Program

This program allows the testing and control of Compumotor RS-232 indexing equipment from Auxiliary RS-232 port *RS232#0*.

The program allows control of Compumotor models CX, 372, and 2100. Menu help is provided for those unfamiliar with indexer commands, as well as interactive hands on control.

Program Initialization

```

10 CALL 0:STRING 106,20          (reserve string space)
20 COM 0                          (enable communications)
30 A1=10:V1=5:D1=1:P1=25000:A9=1 (set default parameters)
35 $(3)="Preset":$(4)="Continuous"
40 PRINT "CX/372/2100 Indexer Control Program":PRINT
50 PRINT "Press the Space bar for the CX,"
60 PRINT "Press RETURN for the 372 or 2100"
70 A=GET:IF A=0 THEN 70
80 IF A=32 THEN 90                (branch if CX)
85 XBY(0FA00H)=XBY(0FA00H).OR.20H:GOTO 100 (set 2 stop bits)
90 P1=12800:A9=8:C2=1            (set CX parameters)
95 $(0)=" LD3 MPI SCA3 ":XMT 0,0:RCV 0,0 (CX initialization)
100 $(0)=" E MN":XMT 0,0:RCV 0,0 (2100 initialization)
110 GOSUB 540:GOSUB 590:GOSUB 640 (download parameters)
115 PRINT:GOSUB 900              (display parameters)

```

Main Menu

```

120 PRINT:PRINT "Press a key to select a command option:"
125 PRINT "  A: change device Address"
130 PRINT "  C: send an execution command"
140 PRINT "  K: select direct keyboard control"
160 PRINT "  P: change motion parameters"
175 PRINT "  S: Stop the Program"
180 A=GET:IF A=0 THEN 180
190 IF A=67 THEN 1500              (test for C)
200 IF A=75 THEN 1000             (test for K)
210 IF A=80 THEN GOSUB 300:GOTO 120 (test for P)
220 IF A<>83 THEN 240             (test for S)
230 PRINT:PRINT:PRINT "Enter 'LOAD' for the Directory":STOP
240 IF A<>65 THEN PRINT CHR(7),:GOTO 180 (test for A)
250 INPUT "New Address? ",A9
260 IF A9>0 THEN IF A9<9 THEN 120
270 GOTO 250

```

Parameter Change Routine

```

300 GOSUB 900
305 PRINT "Press a key to select a parameter:"
310 PRINT "  A: change Acceleration"
320 PRINT "  V: change Velocity"
330 PRINT "  P: change Position (Distance)"
340 PRINT "  D: change Direction"
350 PRINT "  M: select the ",$(4)," mode"
360 PRINT "  X: exit"
370 A=GET:IF A=0 THEN 370
380 IF A=65 THEN GOSUB 500:GOTO 300      (test for A)
390 IF A=68 THEN GOSUB 650:GOTO 300      (test for D)
400 IF A=77 THEN GOSUB 700:GOTO 300      (test for M)
410 IF A=80 THEN GOSUB 600:GOTO 300      (test for P)
420 IF A=86 THEN GOSUB 550:GOTO 300      (test for V)
430 IF A=88 THEN RETURN                  (test for X)
440 PRINT CHR(7);:GOTO 370

500 PRINT:PRINT "Current Acceleration: ",A1," rev/sec.sq"
510 INPUT "Enter new Acceleration (.01 to 999) ",A1
520 IF A1>=.01 THEN IF A1<=999 THEN 540
530 PRINT CHR(7);:GOTO 510
540 N1=A1:ASC$(0,1)=65:GOTO 850      (transmit new accel)

550 PRINT:PRINT "Current Velocity: ",V1," rev/sec"
560 INPUT "Enter new Velocity (.001 to 50) ",V1
570 IF V1>=.001 THEN IF V1<=50 THEN 590
580 PRINT CHR(7);:GOTO 560
590 N1=V1:ASC$(0,1)=86:GOTO 850      (transmit new velocity)

600 PRINT:PRINT "Current Parameter: ",P1," steps"
610 INPUT "Enter new Distance? (1 to 9,999,999) ",P1
620 IF A1>=-99999999 THEN IF A1<=99999999 THEN 640
630 PRINT CHR(7);:GOTO 610
640 N1=P1:ASC$(0,1)=68:GOTO 850      (transmit new distance)

650 PRINT:PRINT "Current Direction: ",
660 IF D1=0 THEN PRINT "CCW" ELSE PRINT "CW"
670 PRINT "Press '+' or '-' for CW or CCW rotation"
680 A=GET:IF A=0 THEN 680
685 IF A=43 THEN D1=1:$(0)="H+":GOTO 800
690 IF A=45 THEN D1=0:$(0)="H-":GOTO 800 (transmit direction)
695 PRINT CHR(7);:GOTO 680

700 IF ASC$(3,1)=80 THEN 730
710 $(3)="Preset":$(4)="Continuous"
720 $(0)="MN":GOTO 800                (transmit new mode)
730 $(3)="Continuous":$(4)="Preset"
740 $(0)="MC":GOTO 800                (transmit new mode)

```

Command Transmit Routine

```

800 XMT 0,0,,                (transmit string $(0))
810 XMT 0,$32,              (then a space)
830 PRINT:PRINT "Command transmitted: ",
840 GOSUB 1120:PRINT:RETURN (display echo)
    
```

Numerical Parameter Conversion Routine

```

850 STR N1,0,2              (convert N1 to string, add on to $(0))
860 GOTO 800                (transmit)
    
```

Current Parameter Display Routine

```

900 PRINT:PRINT "Current Parameters:"
910 PRINT "Acceleration: ",A1,
920 PRINT TAB(20),"Velocity: ",V1
930 PRINT "Direction: ",
945 IF D1=1 THEN PRINT "CW", ELSE PRINT "CCW",
940 PRINT TAB(20),"Distance: ",P1
950 PRINT "Mode: ",$(3),
960 PRINT TAB(20),"Device address: ",A9
970 FOR I=0 TO 1000:NEXT
980 RETURN
    
```

Interactive Control Routine

```

1000 PRINT "All keystrokes are transmitted,"
1010 PRINT " only echoes are displayed"
1005 PRINT "Press 'Esc' to exit":PRINT
1010 A=GET:IF A=0 THEN GOSUB 1120:GOTO 1010 (test receiver)
1020 IF A=27 THEN PRINT:GOTO 120          (exit on Escape)
1030 GOSUB 1100                          (transmit key)
1040 IF A=13 THEN PRINT CHR(10),         (add LF to CR)
1050 GOTO 1010
    
```

Transmit/Receive Routine

```

1100 XMT 0,$A,                (transmit keystroke)
1120 CALL 31:POP C1           (test receiver)
1130 IF C1=0 THEN RETURN     (exit if no characters)
1140 RCV 0,0,C1              (else receive to $(0))
1150 FOR I=1 TO C1
1160 IF ASC$(0,I)=13 THEN PRINT CHR(10),
1170 PRINT CHR$(0,I),        (display characters)
1180 NEXT:$(0)=" "
1190 RETURN
    
```

Execution Options Routine

```

1500 PRINT:PRINT "Press a key to select a command option:"
1510 PRINT " S: Stop the Motor"
1520 PRINT " G: Start the Motor (Go)"
1530 PRINT " H: Go Home (reverse direction)"
1540 PRINT " D: reverse Direction"
1550 PRINT " P: change Parameters"
1560 PRINT " R: Report Parameters and Status"
1580 PRINT " X: Exit",
1590 A=GET:IF A=0 THEN 1590
1600 IF A=83 THEN $(0)="S":GOSUB 800:GOTO 1500 (stop)
1610 IF A=71 THEN $(0)="G":GOSUB 800:GOTO 1500 (go)
1620 IF A<>72 THEN 1650
1630 IF D1=1 THEN $(0)="GH-1" ELSE $(0)="GH1" (go home)
1640 GOSUB 800:IF C2=0 THEN $(0)="X0" GOSUB 800
1645 GOTO 1500
1650 IF A<>68 THEN 1680
1660 $(0)="H":GOSUB 800:IF D1=1 THEN D1=0 ELSE D1=1 (reverse)
1670 GOTO 1500
1680 IF A=80 THEN GOSUB 300:GOTO 1500 (change parameters)
1690 IF A=82 THEN GOSUB 2000:GOTO 1500 (report parameters)
1740 IF A=88 THEN 120 (exit)
1750 PRINT CHR(7);:GOTO 1590

```

Status Report Routine

```

2000 GOSUB 900:F1=0:PRINT (display parameters)
2005 CALL 31:POP C1:RCV 0,0,C1 (clear receiver)
2010 ASC$(0,1)=A9+48 (set transmit address)
2020 ASC$(0,2)=82 (set R command)
2030 N1=2:N2=3:GOSUB 3000 (send 2, receive 3)
2035 IF ASC$(1,1)=0 THEN PRINT "NO ECHO!":RETURN
2040 PRINT "Status: ",
2050 IF ASC$(1,2)=82 THEN PRINT "Ready";:GOTO 2060
2055 F1=1:PRINT "Busy", (82=R=Ready)
2060 ASC$(0,2)=82
2070 ASC$(0,3)=65 (set RA command)
2080 N1=3:GOSUB 3000 (send 3, receive 3)
2090 PRINT TAB(17),"Limit: ",
2100 IF ASC$(1,2)=64 THEN PRINT "None":GOTO 2110
2105 PRINT "Active" (64=@=No Limits)
2110 ASC$(0,2)=84
2120 ASC$(0,3)=83 (set TS command)
2130 GOSUB 3000
2140 PRINT "Inputs: ",
2150 PRINT $(1), (print response)

```

```
2210 IF F1=1 THEN 1500           (skip if busy)
2220 PRINT TAB(17),"Position: ",
2230 IF C2=1 THEN 2260           (branch if CX)
2240 N2=10:ASC$(0),2)=88
2250 ASC$(0),3)=49:GOTO 2280     (set X1 command)
2260 N2=12:ASC$(0),2)=80
2270 ASC$(0),3)=82               (set PR command)
2280 GOSUB 3000                  (send command)
2290 PRINT $(1)                  (print response)
2300 GOTO 1500
```

Status Transmit/Receive Routine

```
3000 XMT 0,0,N1,                 (transmit N1 characters)
3010 XMT 0,$32,                  (transmit Space)
3020 FOR I=1 TO 100:NEXT         (stall)
3030 RCV 0,1,N1+1                (receive transmitted characters)
3040 RCV 0,1,N2                  (receive subsequent response)
3050 RETURN
```