To the Graduate Council:

I am submitting herewith a thesis written by Everette M. Eldridge entitled "Microcomputer Control for Feeding Material Into a Chemical Process." I recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Donald W. Bouldin

Dr. Donald W. Bouldin, Major Professor

We have read this thesis
and recommend its acceptance:

Robert W Rochelle

Roberta Bodenheimer

Accepted for the Council:

L Evans Haak

Vice Chancellor
Graduate Studies and Research

MICROCOMPUTER CONTROL FOR FEEDING MATERIAL INTO A CHEMICAL PROCESS

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Everette M. Eldridge

December 1981

ACKNOWLEDGEMENTS

I would like to thank my parents for instilling in me the fortitude to forge ahead. I would like to thank the Lord for giving me the strength and courage to continue the work. I would like to give a special thanks to Dr. Donald W. Bouldin for being there when I had a question and for responding quickly. Also, I would like to thank Dr. Larry B. Newman for his assistance and continuing interest.

Most of all, I owe a great deal to my wife, Gail, and my daughter, Jennifer. During the time that I was doing work on this thesis, they did nothing but encourage me. I know they sacrificed more than I did, and for this I owe them my complete appreciation and love.

ABSTRACT

The purpose of this thesis was to develop a control system to uniformly feed a material of variable consistency into a chemical process. To develop this system, the author used parts of the existing weighing system and added a microcomputer to correct for inaccurate weighing.

The development of the control system progressed in three stages. In the first stage, the author made use of the existing scale logic and did not optimize the design because of time constraints placed on the project. In the second and third stages, methods were developed to optimize the microcomputer system's performance. Of these two stages, only the third has been debugged and tested on a prototype system.

In this thesis, the author explains in detail the revisions made in each stage of the project. The explanation includes descriptions of the hardware used, the microcomputer programs, and advantages and disadvantages of each stage.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Background Information

The original scale system in this thesis was installed in early 1976. The system is used to feed a non-uniform bulk material into a chemical process. The original system incorporated standard analog electronics to control material weights. After the system was operated for a period of months, it became evident that the mechanical part of the scales was not able to weigh material consistently. In other words, one scale cycle might have ten percent more material than the setpoint, and the next cycle might have five percent less. When this happened, problems involving product quality were encountered downstream in the chemical process. In the original electronic control, there was no ability to correct for inaccurate weights, so a modification was required to correct this problem. After investigating the scale logic, it was decided that a microcomputer could be installed to accomplish the correction needed. In August of 1977, a project was initiated to design, install, and check out a microcomputer control system to correct this problem.

This thesis discusses in chronological order the developments and improvements which resulted in the proposed scale system.

Chapter Content

The second chapter contains a description of the original scale system's functions. It includes sketches showing the mechanical, as

well as electronic, portions of the scale. Last, it tells the deficiencies of this system.

The third chapter explains the original installation of a microcomputer to improve the weighing characteristics of the scale system. It describes the microcomputer system and individual electronic boards within it. The chapter explains the revisions made to the existing operator controls and new controls installed. It contains connection diagrams and block diagrams which show how the system is put together. Finally, it explains the improvements, as well as remaining deficiencies, in this system.

The fourth chapter explains how the second microcomputer installation affects the scale operation. A block diagram furnishes information on how devices are attached to the microcomputer. The chapter reveals the advantages of going to this method.

The fifth chapter describes the addition of a teletype-writer (TTY) input-output. It explains the system operation and has a pin-out connection diagram of the system. It explains why this is the best design for improved system operation.

The sixth chapter discusses the software considerations and implementations.

The seventh chapter explains hardware considerations.

The eighth chapter gives a summary and conclusions. It lists other additions to the microcomputer system which could be beneficial.

CHAPTER II

THE ORIGINAL INSTALLATION

Hardware Overview

Each weighing system[1] consists of two scale systems. Each of these scale systems consists of a bulk material supply bin, feed rolls, feed gates, dump gates, and electronic weight controls. For the remainder of the thesis, only one scale system will be discussed, because both are identical in operation.

Description of Mechanical Hardware

Figure 1 shows a simplified sketch of the mechanical portion of the system. Within the system, each device has a specific function to perform. The bulk material supply bin supplies material to the feed rolls. Operating at variable speed, the feed rolls pull material from the bottom of the supply bin and feed it onto the scale dump gates. After the scale has reached the desired weight of material, the feed rolls stop, and the feed gates close. The feed gates are used to prevent material from falling through the spaces between the rolls onto the dump gates. The scale dump gates supply material to the chemical process when opened and allow the material to be weighed when closed. A more detailed discussion of the original scale operation is presented in the remainder of this chapter.

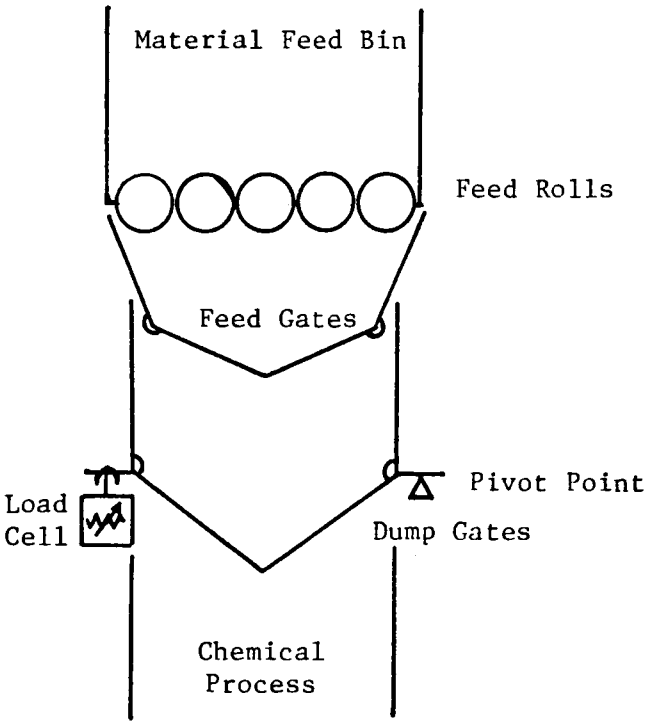


Figure 1. Diagram of Mechanical Equipment in Scale System

Description of Electrical Controls

The electrical controls consist of panel-mounted operator's controls such as pushbuttons, potentiometers, a timer, and indicators. Non-operator controls consist of a load cell for sensing scale weight and electronic comparing circuits which turn on and off scale rolls, feed gates, and dump gates. A block diagram of these controls is shown in Figure 2.

The cycle timer and setpoint potentiometer are by far the most critical controls. The cycle timer, adjustable from 0-60 seconds, sets the length of time between the start of the present cycle and the start of the next scale cycle. The setpoint potentiometer produces a 0-10 VDC signal which is fed into the electronic controls. The setpoint weight adjustment varies from 0-200 pounds, which corresponds to the voltage signal it outputs. From the cycle time and setpoint, one can calculate the feed rate of the scale as follows:

$$\underset{\text{(pounds/minute)}}{\text{Feed Rate}} = [\underset{\text{(pounds)}}{\text{Setpoint}} \times 60 \text{ seconds}] \div \underset{\text{(seconds)}}{\text{Cycle Time}}$$

The materials-in-suspension (MIS) potentiometer and the bulk/dribble selector (BDS) potentiometer control roll speed and gate closures. Both of the potentiometers use the voltage from the setpoint potentiometer to produce a voltage signal equal to 0-100 percent of the setpoint voltage. Each of these signals is input into the negative side of separate voltage comparators. The positive lead of the comparators is connected to the voltage from the load cell which indicates the weight. The BDS comparator is used to switch from bulk to dribble speed. The MIS comparator is used to close the feed gates and turn off the feed rolls.

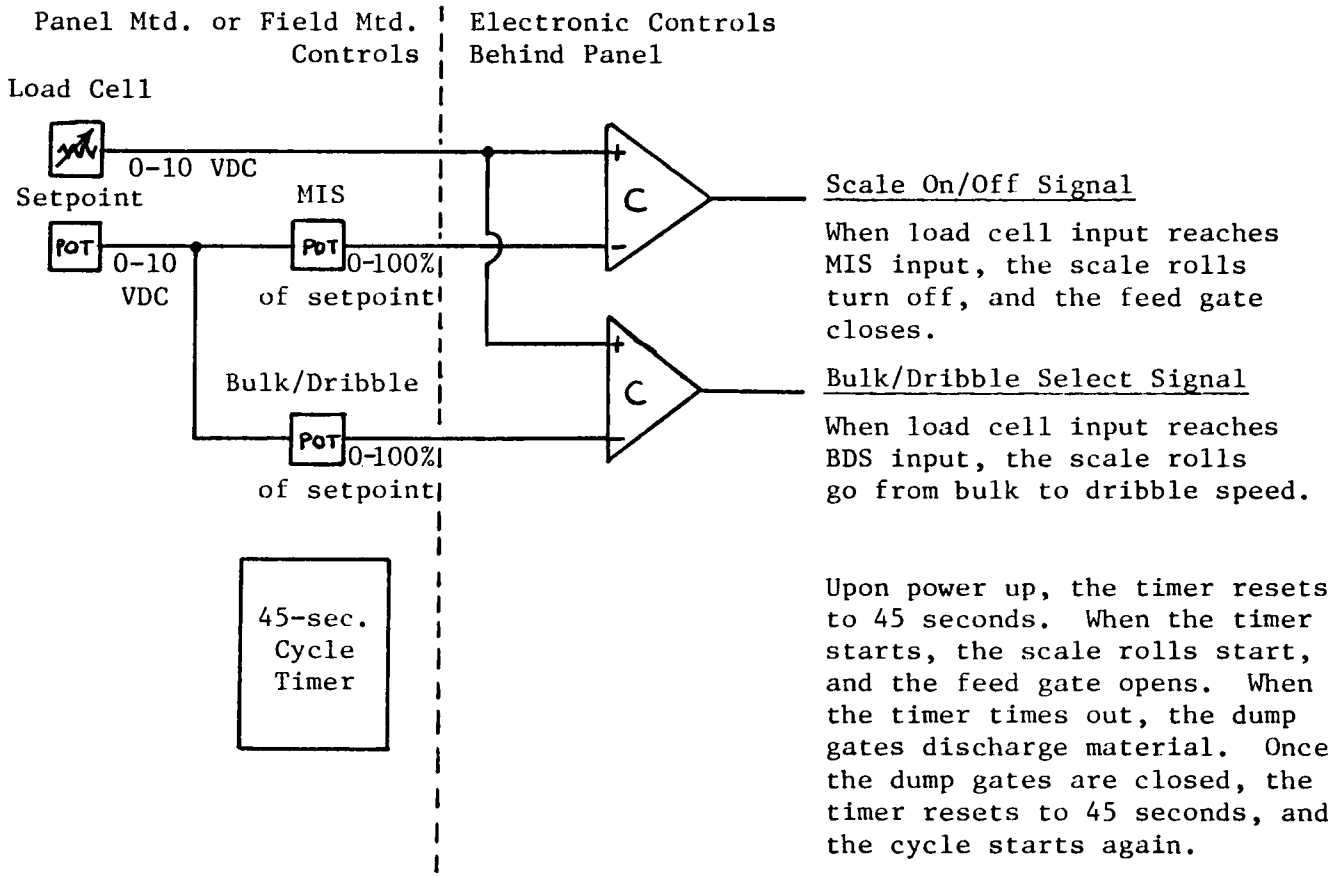


Figure 2. Block Diagram of Original Scale Control Electronics*

*Notes: 1. Sketch does not show relays energized by timer or comparators.
2. Roll speed potentiometers to set bulk and dribble speed are not shown.

Several devices have operations that are self-explanatory. The power on/off switch connects power to the scale controls. The weight indicator indicates the amount of material on the scales in pounds. The indicator lights indicate such things as when the power is on, dribble speed, bulk speed, and when the scale is discharging. There are also dribble speed and bulk speed potentiometers which control the roll speed of the variable speed drive. These miscellaneous controls will only be mentioned at this time and not discussed in detail because they are not germane to the thesis.

System Operation

When the power is turned on, the scale cycle timer resets to 45 seconds and starts timing down to zero. When the timer resets, the feed gates of the scale open, and the feed rolls begin feeding material onto the scale dump gates at the bulk speed. When the BDS comparator senses that the weight is equal to the BDS setpoint, the feed rolls change to the dribble speed. When the MIS comparator senses that the weight is equal to the MIS setpoint, the feed rolls stop, and the feed gates close. The scale remains in this state until the cycle timer times out. When the timer times out, the scale dump gates open, feeding the material into the chemical process. After the material is dumped, the dump gates close, the cycle timer resets, and the scale cycle repeats.

System Deficiencies

Several deficiencies exist in this system. First, the feed rate of the feed rolls is inconsistent. This causes overfilling and underfilling of the scales because the MIS control assumes a constant feed

rate of material. Second, if there is an overfill or an underfill, the scale has no way to correct for this inaccuracy.

Correction of these problems could be accomplished in one of two ways. One way would be to make the feeding mechanism feed more consistently. This method was tried by changing feed roll sizes, and the results were not successful. The second method would be to install a control that would adjust for weight inaccuracies. The design and installation of this method is discussed in the following chapters.

Three requirements had to met while installing this system. First, it must correct the problem with weight fluctuations. Second, it must be installed so that the old system can be used as a back-up. Finally, it must be installed and operational as quickly as possible.

CHAPTER III

ORIGINAL MICROCOMPUTER INSTALLATION

Reason for Microcomputer

The new system would need to be able to accept analog inputs, accept digital inputs, turn on and off digital devices, do mathematical computations, and store computed values. All of these functions can be obtained by using a microcomputer with various interface devices. Also, by using a microcomputer, most revisions to design can be accomplished through programming instead of wiring changes.

A microcomputer can provide the adjustment for incorrect weight by using one of two methods. In the first method, the microcomputer adjusts the setpoint of the next cycle to correct for the weight inaccuracy of the present cycle. In the second method, the microcomputer adjusts the cycle time of the existing cycle to maintain the correct feed rate, i.e., pounds of material per unit time. The latter method is used because it corrects the problem during the present cycle instead of the next cycle. The remainder of Chapter III describes the original microcomputer system, revisions to existing controls, additional controls, and results obtained.

System Configuration

The microcomputer system needs six digital inputs, eight digital outputs, and four analog inputs, each with a range of 0-10 VDC. Also, the microcomputer needs EPROM storage capacity for non-volatile program

storage and RAM storage for variables. The microcomputer boards used were chosen because of their availability and prior experience with them. The chosen microcomputer system consists of an 8080A-based microcomputer with an ADAC Model 735-SBC[2] analog-to-digital converter board. These units meet all the requirements described earlier. A pin-out connection diagram of this system is shown in Figure 3.

The microcomputer consists of an SBC-655[3] system chassis and an SBC 80/10A™[3] single board microcomputer. Features of the SBC 80/10A™ microcomputer which are necessary for the scale system include an 8080A CPU, a 2.048-megahertz system clock, 1 K bytes of RAM, sockets for 8 K bytes of EPROM using 2716 EPROM's, two 8255 programmable peripheral interfaces, sockets for I/O line drivers or terminators, and MULTIBUS™ control logic. A block diagram of this board is found in Figure 4. A schematic diagram can be found in the first five pages of Appendix A. For a more detailed description of the board and its operation, refer to the board's hardware reference manual.[3]

The ADAC 735-SBC analog-to-digital board used in the scales' microcomputer system employs the successive approximation method to convert an analog signal within the range of 0-10 VDC to a 12-bit binary code with the corresponding range of 0-0FFF hexadecimal. This board accepts up to 32 fully differential inputs. It has a relative accuracy of $\pm$ 1 bit and a sampling rate of 35,000 channels per second. The board surpasses the resolution and accuracy needed to obtain a scale cycle error of less than $\pm$ 0.5 pounds for a 0-200 pound range. A block diagram of an analog-to-digital board is shown in Figure 5. A pin-out connection diagram can be found in Appendix A. For a more detailed description of

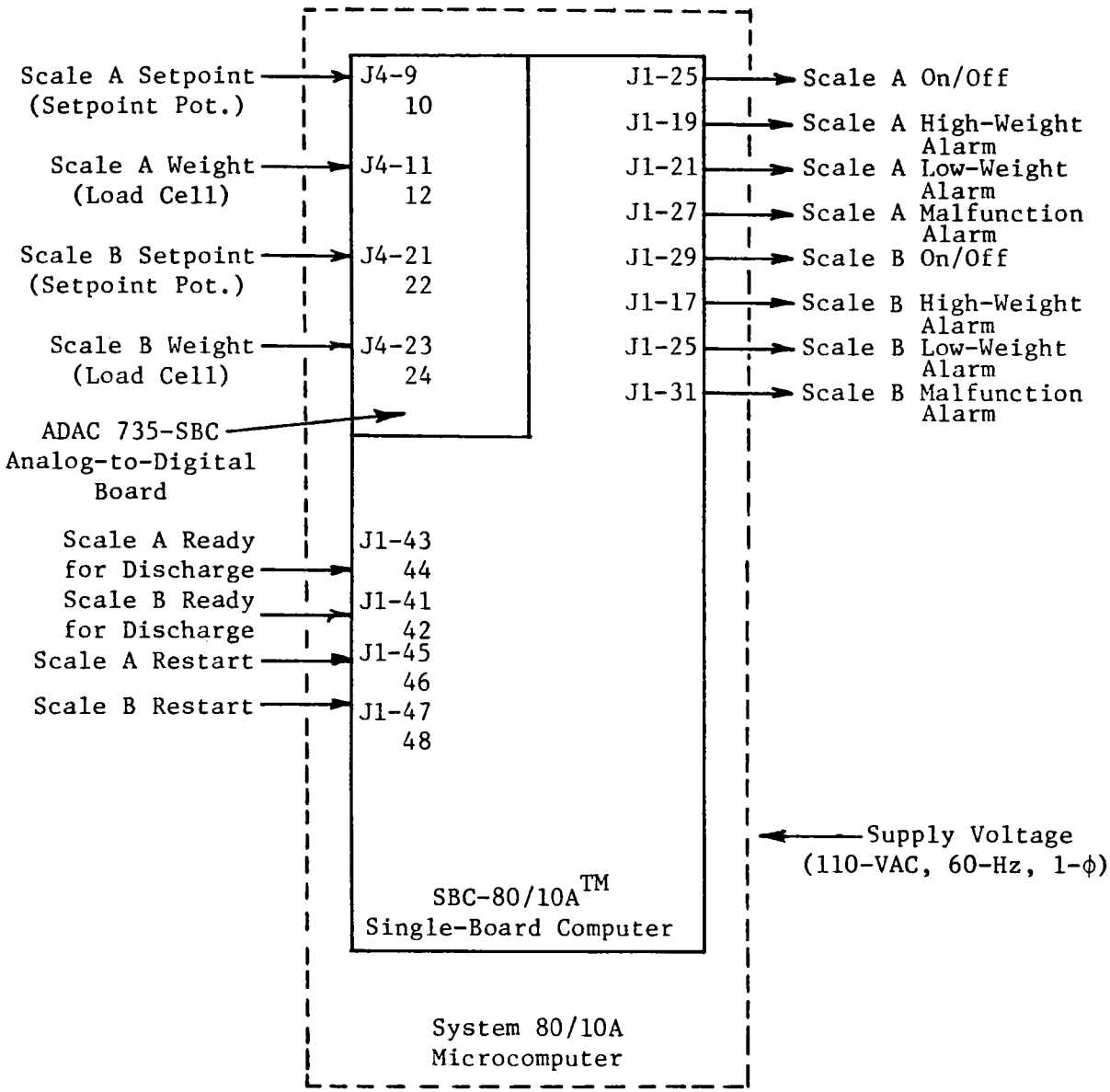
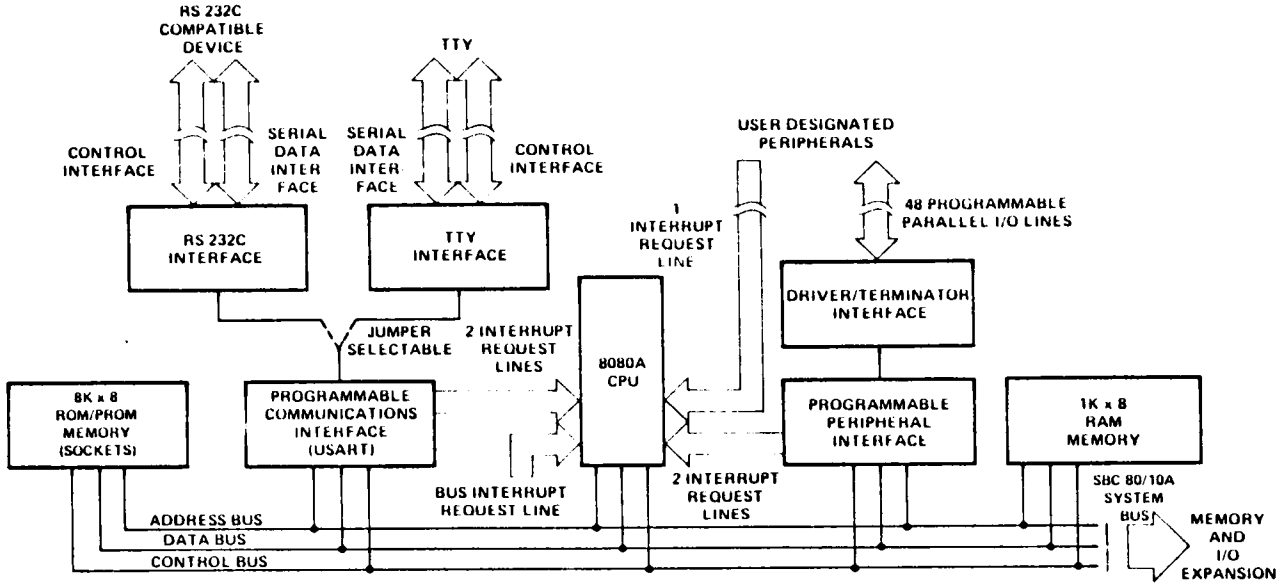


Figure 3. Pin-Out Connections to External Controls of Original Microcomputer Installation*

*Notes:
1. All odd-numbered terminals are for positive side of input or output.
2. J1 connector is for digital input/output TTL logic. J4 connector is for analog input (0-10 VDC).
3. All digital outputs are negative true logic. (OV out means output turned on.) All digital inputs are negative true logic. (OV in means input present.)



1. Interrupts originating from the Programmable Communications Interface and Programmable Peripheral Interface are jumper selectable.

Figure 4. SBC-80/10A™ Block Diagram

Source: Application Note AP-26, iSBC-80/10A--System 80/10 Single Board Computer Applications, Thomas Rolander.

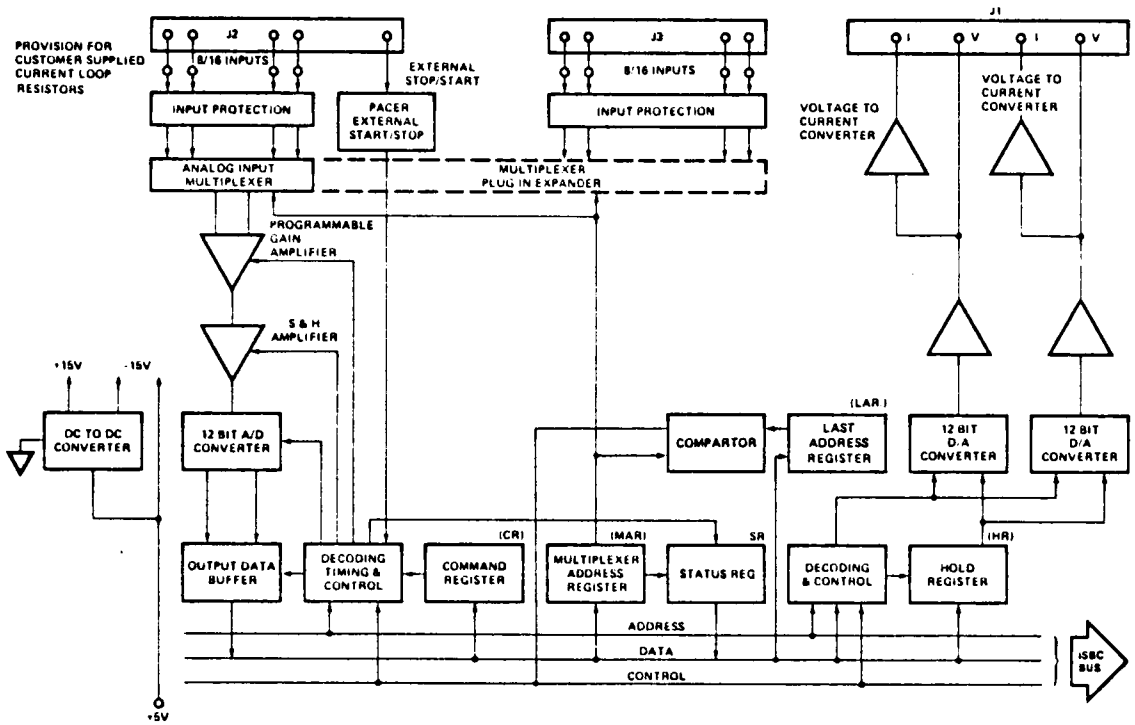


Figure 5. Block Diagram of Analog-to-Digital Board*

*Note: This diagram is of an A/D board similar in design to the ADAC 735-SBC.

Source: Intel System Data Catalog, August 1978.

the board's function and operation, refer to the board's instruction manual.[2]

The microcomputer system as installed adds several operator controls and modifies one existing control. The added controls include a microcomputer power on/off selector switch, a scale on/off microcomputer selector switch, a scale restart pushbutton, a high-weight alarm light, a low-weight alarm light, and a scale malfunction alarm light. The existing panel-mounted scale cycle timer requires a different setting when operated in the computer mode versus the non-computer mode.

The microcomputer power selector switch applies 110-VAC, 60-hertz, single-phase power to the microcomputer system when in the "on" position. The scale microcomputer selector switch puts the scale in the microcomputer mode when in the "on" position. When depressed and released, the scale restart pushbutton sets the scale timer count equal to the system timer count and starts the next scale cycle. The alarms indicate specific alarm conditions in the scale system.

All of these controls are duplicated for each scale, with the exception of the microcomputer power on/off selector switch, since both scales are operated from the same microcomputer. The functions of these controls are explained further in the remainder of this chapter.

System Operation

When the scale on/off microcomputer selector switch is in the "off" position, the scale operates as discussed in Chapter II. The panel-mounted cycle timer should be set so that the time from one cycle start to the next is equal to 45 seconds. When the switch is in the "on" position, the scale cycle timer is set at 30 seconds and controls

the time in which the scale rolls can complete feeding the material onto the dump gates. When the cycle timer times out, it sends a discharge signal to the microcomputer through a digital input.

When the discharge signal is received, the microcomputer inputs the weight setpoint value and weight received value through the analog-to-digital interface. After receiving these signals, the microcomputer resets the scale alarms and turns off the scale cycle timer, which allows the dump gate to open and the material to feed into the continuous process. Next, the microcomputer calculates the time needed for the present cycle using the following formula:

$$\text{Time (tenths of seconds)} = [\text{Weight Received} \div \text{Setpoint}) \times 450$$

The 450 value in this formula is the normal cycle time in tenths of seconds if the correct weight is received.

Once this calculation is made, the microcomputer checks for alarm conditions. If the calculated time is less than 40 seconds, a low-weight alarm is turned on. If the calculated time is greater than 50 seconds, a high-weight alarm is turned on. If the calculated time is greater than 90 seconds, a malfunction alarm is turned on, and a value of 45 seconds is substituted for the calculated time.

After the alarm checks are completed, the microcomputer adds the calculated time to the present cycle's start time. This value tells the microcomputer when the scale starts the next cycle. Once the time is reached by the microcomputer program, the scale cycle timer is energized, and the next cycle begins. The feed rolls and feed gates are controlled exactly as described in Chapter II.

The microcomputer program has some other features. The timing is done by the loop-count method. Each loop through the program equals approximately one-tenth of one second. When the loop is completed, the system timer increments its count. Another feature is a subtraction routine which prevents the timers from overrunning their 16-bit memory storage. When the system timer count reaches EC00 hexadecimal, the microcomputer subtracts C400 hexadecimal from the value of the system timer count and each scale timer count.

The microcomputer program operates both scale systems simultaneously. A structure chart is found in Appendix B, and a program listing is in Appendix C.

Results and Necessary Improvements

After gathering data, the author verified that the accuracy of the feed rate of the scale was calculated to be two-tenths of a pound per cycle. As a direct result of the improved accuracy, the microcomputer reduced product waste and improved product quality to where its installation cost was saved within the first two months of operation. Even with these improvements, there were still some control modifications which could and should be made.

One improvement was that the loop-count method could be removed by using an interrupt from a 10-hertz clock pulse to increment the system timer count. Also, a separate system timer count could be kept for each scale. This would remove the necessity of the subtraction routine to avoid storage register overflow. Finally, by having the microcomputer calculate when the rolls should be turned on or off, the microcomputer

could run the scales without the cycle timer interface. The following two chapters discuss ways to accomplish these and other revisions.

CHAPTER IV

THE FIRST REVISION TO THE MICROCOMPUTER

Options Available

When entering this portion of the design stage, the author came to a crossroads. The microcomputer could continue to receive its set-point through a potentiometer or could use a TTY to input values. There were advantages to each of these methods which are discussed later. For this reason, the author designed both methods. This chapter deals with continuing the use of potentiometers for input. The next chapter explains in detail the capabilities of using a TTY for input and output.

System Configuration

For this revision, the microcomputer system needs two digital inputs, 14 digital outputs, four analog inputs, a clock circuit for interrupt input, and EPROM program storage and RAM variable storage. All the above devices are similar to the ones used on the system in Chapter III except the clock circuit. This clock circuit consists of a 10-hertz crystal oscillator and a single shot tied directly to the Interrupt 0 line on the microcomputer. For a pin-out diagram of the clock circuit, see Figure 6. For a pin-out diagram of the microcomputer system, see Figure 7.

The operator controls for the new system remain the same with the following exceptions. First, the restart pushbutton for each scale is removed, and this function is incorporated in the scale on/off micro-computer selector switch. By turning the selector switch to the "off"

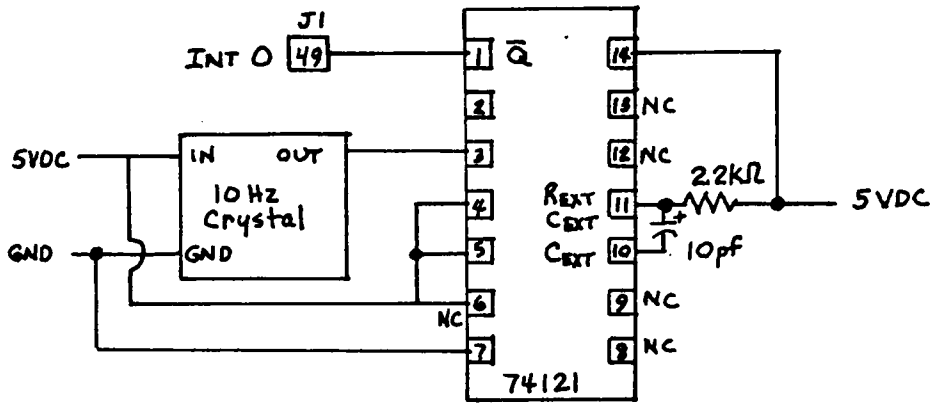


Figure 6. Pin-Out Diagram of Clock Input*

*Notes:
1. Minimum length of Interrupt Routine = 228 clock cycles x 488 ns/clock cycle = 111 µsec.
2. Range of Interrupt pulse width = 3 µs to 111 µs.
3. Pulse Width = t_w (sec) = $R_{EXT}C_{EXT}$ ln 2 = 2.2 kΩ x 10 pf x ln 2 = 15.25 µsec at Q(1).

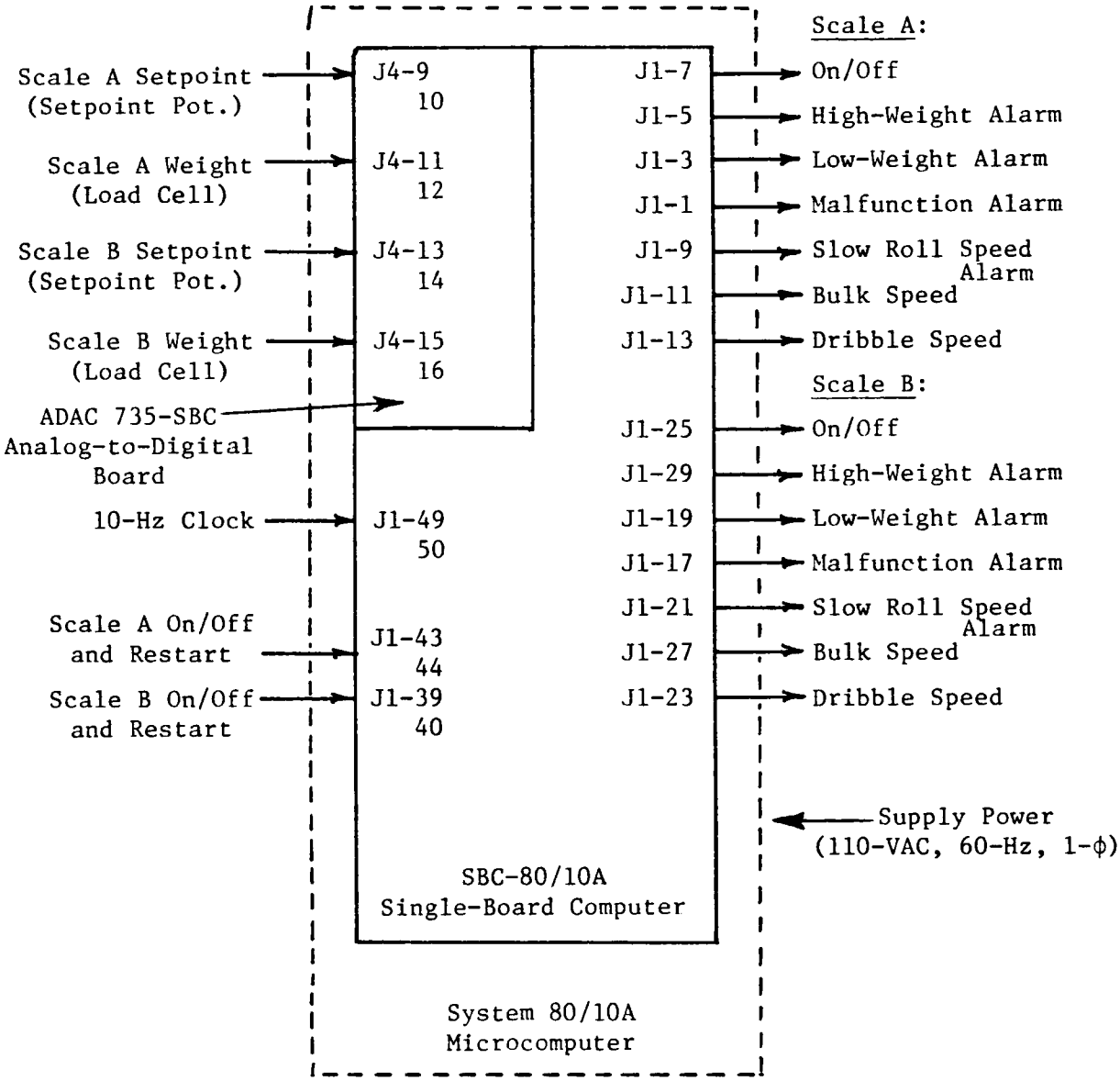


Figure 7. Pin-Out Connections to External Controls on First Revision of Microcomputer Installation*

*Notes:
1. All odd-numbered terminals are for positive of input or output. Even-numbered terminals are for digital or analog ground.
2. J1 connector is for digital I/O (5-VDC TTL). J4 connector is for analog inputs (0-10 VDC).
3. All digital outputs are negative true. (OV out means output turned on.) All digital inputs are negative true. (OV in means input present.)

position and then to the "on" position, the operator causes the microcomputer to sense a restart signal. Second, the panel-mounted cycle timer no longer controls the scale in the microcomputer mode. Last, the MIS potentiometer and BDS potentiometer are not used. Instead, these potentiometer settings are programmed into the microcomputer system as constants and can only be changed by reprogramming.

System Operation

In this microcomputer system, only the load cell input, setpoint potentiometer, and the scale feed roll speed controls remain from the original scale system. Upon starting the cycle, the microcomputer opens the feed gates and turns on the bulk speed of the feed rolls. When the weight of material reaches 70 percent of the setpoint, the bulk speed turns off, and the dribble speed turns on. When the weight equals 90 percent of the setpoint, the dribble speed turns off, and the scale feed gates close.

Next, the microcomputer waits five seconds to allow the scale weight to settle out. After the delay, the microcomputer inputs the weight and setpoint values through the analog-to-digital interface. After this, the scale dump gates open and discharge the material into the chemical process. The microcomputer calculates the time needed for the amount of weight received as follows:

$$\text{Time (tenths of seconds)} = [\text{Weight Received} \div \text{Setpoint}] \times 450$$

This equation is the same used in the original microcomputer system.

Once this calculation is completed, the microcomputer compares the calculated value with the stored alarm values and energizes the alarms if necessary. If the malfunction alarm is set, the microcomputer replaces the calculated time with 450. After running the alarm check, the microcomputer compares the calculated time with the scale clock time. When the scale clock becomes greater than or equal to the calculated time, a new cycle is started. At this time, the scale clock time is set to zero.

The scale clock time is generated through the use of an interrupt routine. A 10-hertz clock pulse is input into the INTR 0 line of the microcomputer. A low signal on this line gives a Restart 7 command to the microcomputer, which causes the interrupt routine at memory location 038H to be executed. This routine increments the scale clock time and checks to see if the scale should be started.

The microcomputer operates both scales simultaneously. All control devices are typical for each scale, so an explanation of one scale's operation is sufficient. A flow chart can be found in Appendix B, and a program listing can be found in Appendix C.

Possible Results and Improvements

If installed, this system would be easier to troubleshoot than the original system because there is less interfacing with existing logic. Since it does not require a discharge signal, the microcomputer program would eliminate the need for the panel-mounted scale cycle timer. This would avoid possible operator error of failure to reset the time, as can happen in the system described in Chapter III. The installation

would be easier and less expensive than the TTY revision discussed in the next chapter, since most of the components are already installed.

Even with these advantages, there were still some improvements to the system which the author felt were necessary. A print-out of the alarms would help the operator during a system upset. It could give a valuable record of what had happened. The addition of the capability to input values such as the BDS setting, the MIS setting, and the cycle time would return more control to the operator. Last, if the setpoint value of the scale could be input in digital form, no analog calibration of the setpoint signal would be required, resulting in a more accurate and reliable setting. All of these revisions are accomplished by adding a TTY interface to the system. This addition is discussed in the following chapter.

CHAPTER V

SECOND REVISION TO THE MICROCOMPUTER

System Configuration

The second revision consists of adding a serial communications interface to transmit data between a TTY and the microcomputer. The microcomputer system needs two digital inputs, eight digital outputs, a clock circuit for interrupt input, a 20-mA current loop serial interface, EPROM program storage, and RAM variable storage. A programmable communications interface is available on the SBC 80/10A board using an 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART).[5] All the remaining requirements are similar to those used on the system in Chapter IV. Figure 8 shows a pin-out diagram of this system.

System Operation

The addition of the TTY enables the operator to modify the set-point, MIS setting, BDS setting, or cycle time at any time while the system is running. To modify a value, the operator first chooses the scale by hitting a "Control A" for scale A or a "Control B" for scale B. The microcomputer acknowledges receipt of this by sending to the TTY the message "Scale '(A or B)' Chosen." After this message is received, the operator chooses what operation is needed. Table 1 shows what input is required for a given operation.

After this selection is made, the microcomputer sends a message telling the chosen constant's present value and asks if the operator

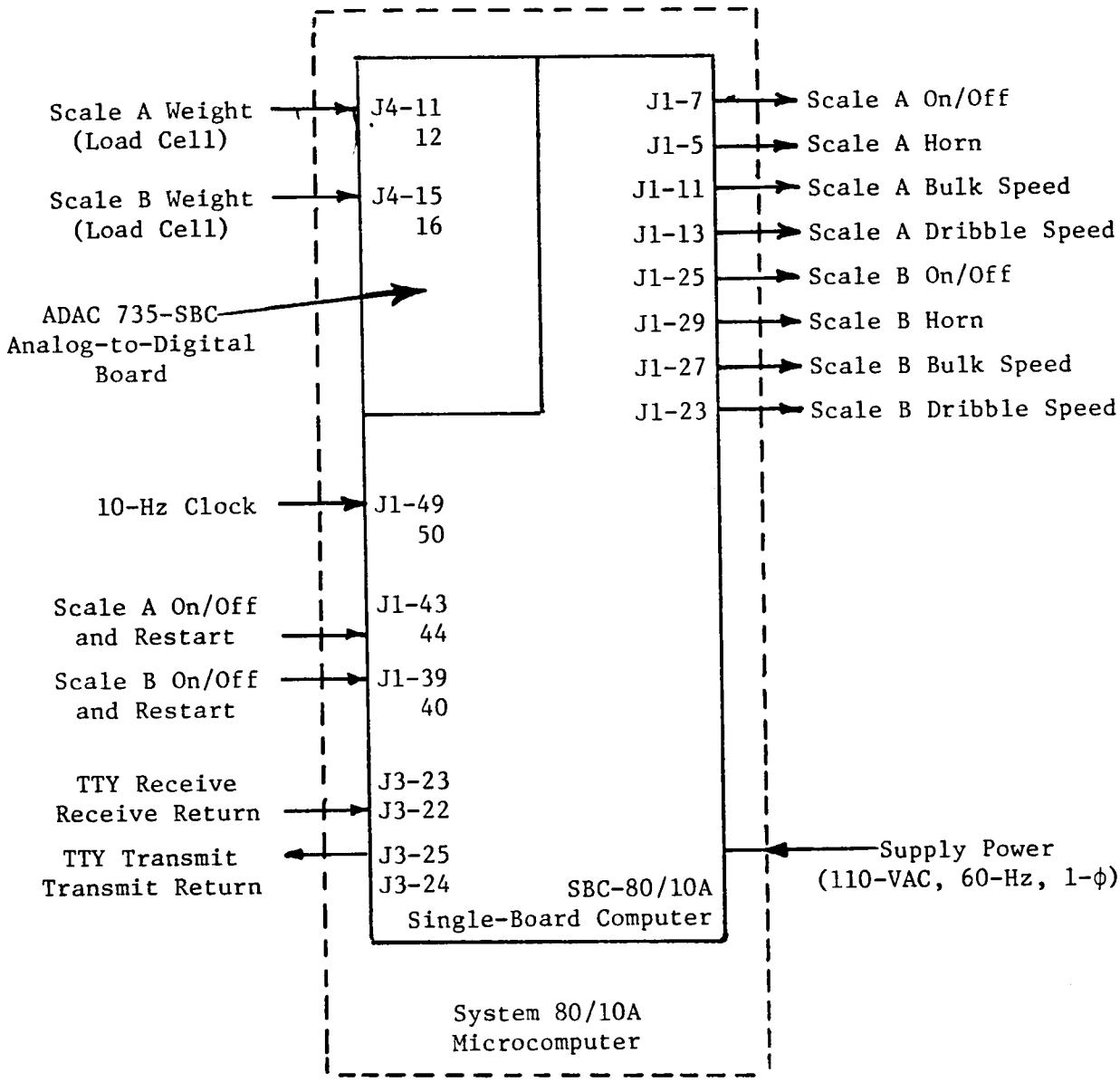


Figure 8. Pin-Out Connections to External Controls on Second Revision of Microcomputer Installation*

*Notes:
1. All odd-numbered terminals (except J3) are for positive of input or output. Even-numbered terminals (except J4) are for digital or analog ground.
2. J1 connector is for digital I/O (5-VDC TTL). J3 connector is for serial I/O (20-mA current loop). J4 connector is for analog inputs (0-10 VDC).
3. All digital outputs are negative true. (Grounded output means output on.) All digital inputs are negative true. (Grounded input means input present.)

TABLE 1

KEY COMMANDS FOR TTY

Key Input	ASCII	Operation Wanted
Control E	05H	MIS Input
Control D	04H	BDS Input
Control T	14H	Cycle Time Input
Control S	13H	Setpoint Input
Control A	01H	Scale A Chosen
Control B	02H	Scale B Chosen
Control L	0CH	List Constants
Control H	08H	List Alarms
Control Z	1AH	Calibration Check

Note: A scale must be selected before an operation can be initialized.

wants to change it. A "No (N)" answer causes the microcomputer to answer with "Goodbye" and return to the main program. A "Yes (Y)" response causes the microcomputer to request the new value of the constant. After the operator enters the new value, the microcomputer echoes the value entered and asks if it is correct. If the operator responds with "N," the microcomputer requests that the new value be entered again. If the operator answers "Y," the entered value is stored for calculations.

At this time, the microcomputer calculates and stores the new value of the constant. If this were a setpoint change, new values for the MIS constant and the BDS constant are calculated and stored. If

this were a cycle time change, new values for the alarm times are calculated and stored. After the calculations are complete, the microcomputer lists the constant values of the scale chosen and returns to the main program.

If after choosing the scale the operator hits a "Control L," the microcomputer lists on the TTY the present value of the constants of the scale chosen. If the alarm horn has been energized, the operator can check alarm outputs by hitting "Control H." If a calibration check is needed, the operator or maintenance person can press "Control Z." When this is done, the microcomputer will output to the TTY the present value of the analog-to-digital input of the scale chosen.

The roll speed control and alarm check routines are similar to those used in the first revision. One difference is instead of turning on individual lights for each alarm, this revision energizes a horn if any alarm is recognized. The operator then acknowledges the horn and checks which alarm is present. This input from the TTY causes the microcomputer to list on the TTY the scale alarms and values of each alarm. When the scale is turned off, the slow roll alarm energizes to warn the operator that this has been done.

Advantages

By having this communications capability, the operator can keep better track of the system operation. At any time during the operation, the operator can find out if there has been an alarm, how bad it was, and what the present constant values are. Not only does the operator get an indication, he also gets a hard-copy record of the degree of the

alarm. Also, there is no setpoint potentiometer calibration to check. Finally, the maintenance personnel can check the calibration of the A/D board while the system is in full operation.

CHAPTER VI

SOFTWARE CONSIDERATIONS

The programming of the microcomputer was accomplished by using a microcomputer development system[6,7] to assemble or compile the 8080 machine code from higher level languages. The program of the original computer installation was written in assembly language[8] and assembled to machine code. Calculations were accomplished by using an assembly language floating-point math package.[9] Both revisions were written in FORT//80,[10] a FORTRAN for the 8080 microprocessor, and compiled into machine code using the FORT//80 compiler of the development system.

By doing this project in several steps, the author found that the use of a higher level language such as FORTRAN is beneficial if the memory capacity is available. By being more readable, it is easier to revise by others not familiar with the original design. Although this is true, it takes 8 K of memory to store the object code of the second microcomputer revision and only 2 K of memory to store the object code of the original microcomputer installation. The additional memory cost of the second revision is negligible.

The original assembly language program has been installed and working in two weight systems since February 1978. The first revision, which is written in FORT//80, has not been debugged because a decision was made to go to the second revision before this was installed. The second revision, which is written in FORT//80, has been debugged using the in-circuit emulator[11] and is now operating on a prototype only.

Even without the actual installation into the scale system, one can see that this system is superior to the original installation because of reasons cited in Chapter V.

CHAPTER VII

HARDWARE CONSIDERATIONS

The digital inputs for the microcomputer system are negative-true. Each input is interfaced through a pull-up resistor. When the input is present, the signal is grounded through the contacts of the input device. With this in mind, all input contacts are connected between a J1 terminal and digital ground.

The digital outputs for the microcomputer system are negative-true. This is made possible by using an SN 7438 quad 2-input NAND buffer with open collector outputs for the interface chip. Solid-state relays which can be energized by 5 VDC at less than 48 mA are connected between the J1 connector and a 5-VDC supply. When the output is energized, the relay is grounded through the 7438, thus turning the relay on. These relays are used to isolate external voltages from microcomputer system voltages.

Throughout the thesis, all references to actual connections of wiring between the scale system and the microcomputer were omitted. This was done because information about these connections would add little to the explanation of scale operation. All external wiring connections were done with relay interfacing.

There are several microcomputer systems and analog-to-digital interfaces available which meet the hardware requirements of this system. The author does not want to imply that the hardware used is the only way to obtain the necessary results, but it is a good alternative. The hardware discussed previously was chosen because of availability, programming facilities, and the author's familiarity with it.

CHAPTER VIII

SUMMARY AND CONCLUSION

The original purpose of the project was to find a means to correct for inaccurate weighing in the scale system. The original microcomputer installation accomplished this. By comparing data obtained prior to the installation with data observed after the installation, it was evident that the modification had reduced product waste and aided in improving the product quality.

With this in mind, the author felt the need to design a scale system that would omit all the existing interfaces with the old scale electronics and improve the operator's controls. This effort resulted in the second revision to the microcomputer system. In this system, the operator has the ability to get printed records of control variables, not only receive alarms, but discern the magnitude, and change variables without depending on calibration accuracy of the original scale system.

Even though the author ended his design effort at this point, there are still some possible improvements which could be tried. One is the addition of an automatic tare subroutine which would correct for material build-up on the scale. Also, weight totalizing for inventory records could be accomplished with a small program change. Finally, the addition of automatic roll speed control could be investigated.

LIST OF REFERENCES

LIST OF REFERENCES

1. Instruction Manual for Thayer Scale. Thayer Scale, Hyer Industries, Penbroke, Massachusetts 02359.

2. Instruction Manual for ADAC Corporation Model 735-SBC Data Acquisition and Control System. ADAC Corporation, 70 Tower Office Park, Woburn, Massachusetts 01801.

3. Intel System Data Catalog. Copyright 1976, 1977, 1978. Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051, August 1978.

4. Rolander, Thomas. iSBC 80/10A--System 80/10 Single Board Computer Applications. Copyright 1978. Application Note AP-26, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

5. iSBC 80/10™ and iSBC 80/10A™ Single Board Computer Hardware Reference Manual. Copyright 1976, 1977, 1979. Manual No. 9800230-7, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

6. Intellec® 800 Microcomputer Development System Operator's Manual. Copyright 1975. Manual No. 9800129A, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

7. ISIS-II User's Guide. Copyright 1976, 1977, 1978. Manual No. 9800306D, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

8. 8080 Assembly Language Programming Manual. Copyright 1974, 1975, 1976. Manual No. 98-004C, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

9. Caserta, Keith J. 8080 Floating Point Package. The Procter and Gamble Company, Cincinnati, Ohio.

10. FORT//80 FORTRAN/IV for the 8080 Implementation Manual. Copyright 1979. Arkon Electronics Limited, Toronto, Ontario, Canada.

11. In/Circuit Emulator/80 Operator's Manual. Copyright 1975, 1976, 1977. Manual No. 98001850, Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051.

APPENDICES

APPENDIX A

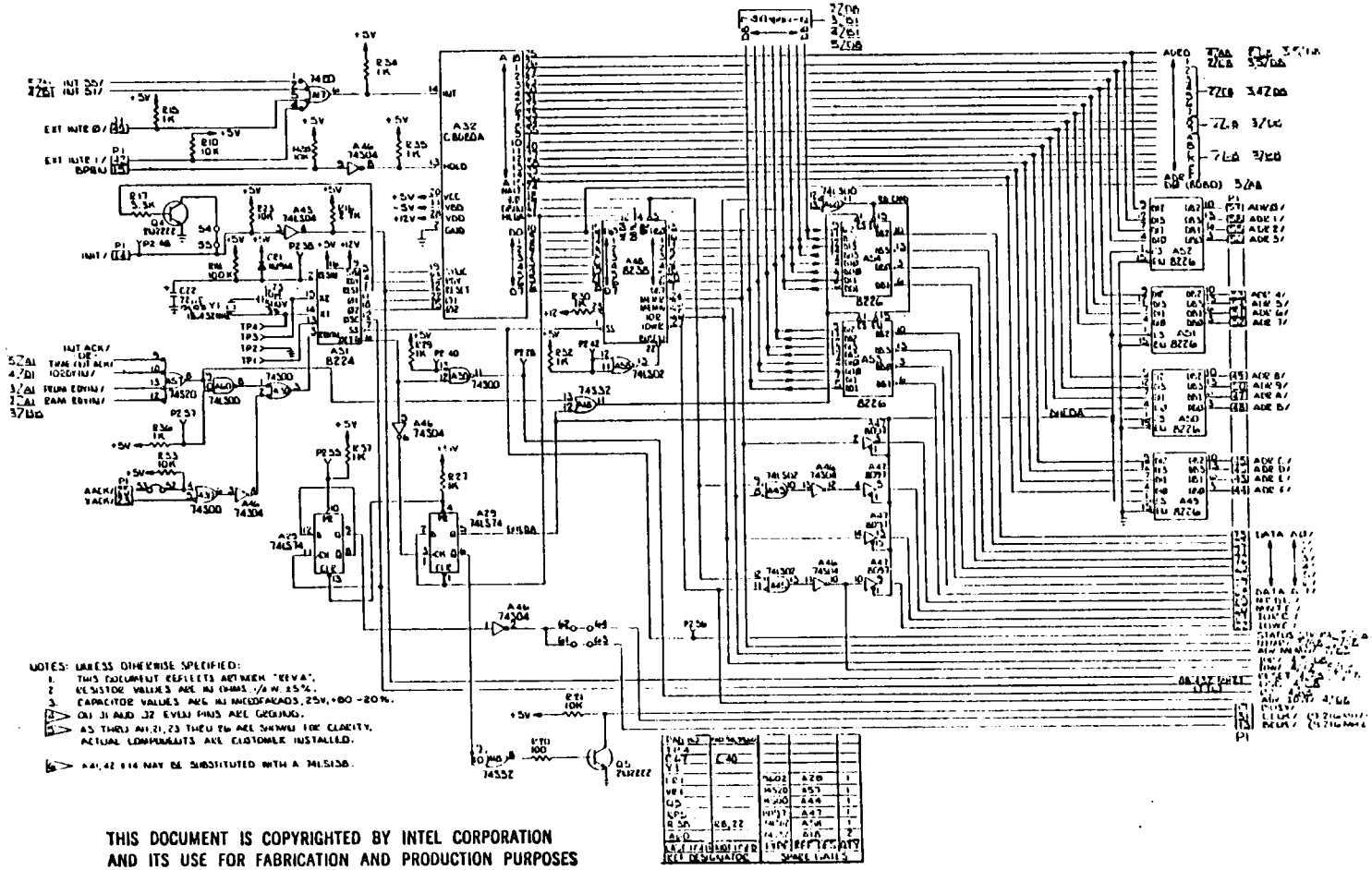


Figure A-1. Schematics for SBC-80/10A

Source: iSBC-80/10 and iSBC-80/10A Single Board Computer Hardware Reference Manual.[5]

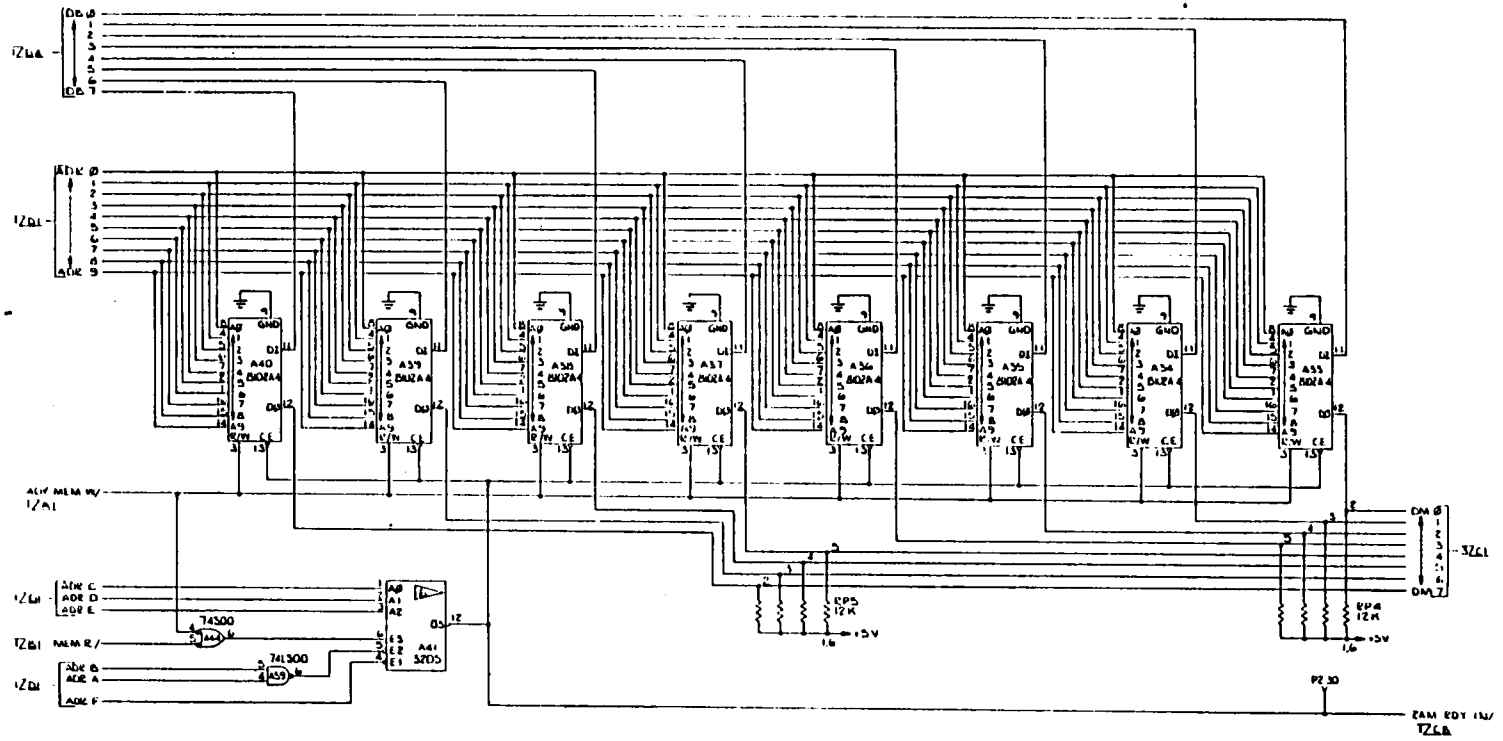


Figure A-1. (Continued)

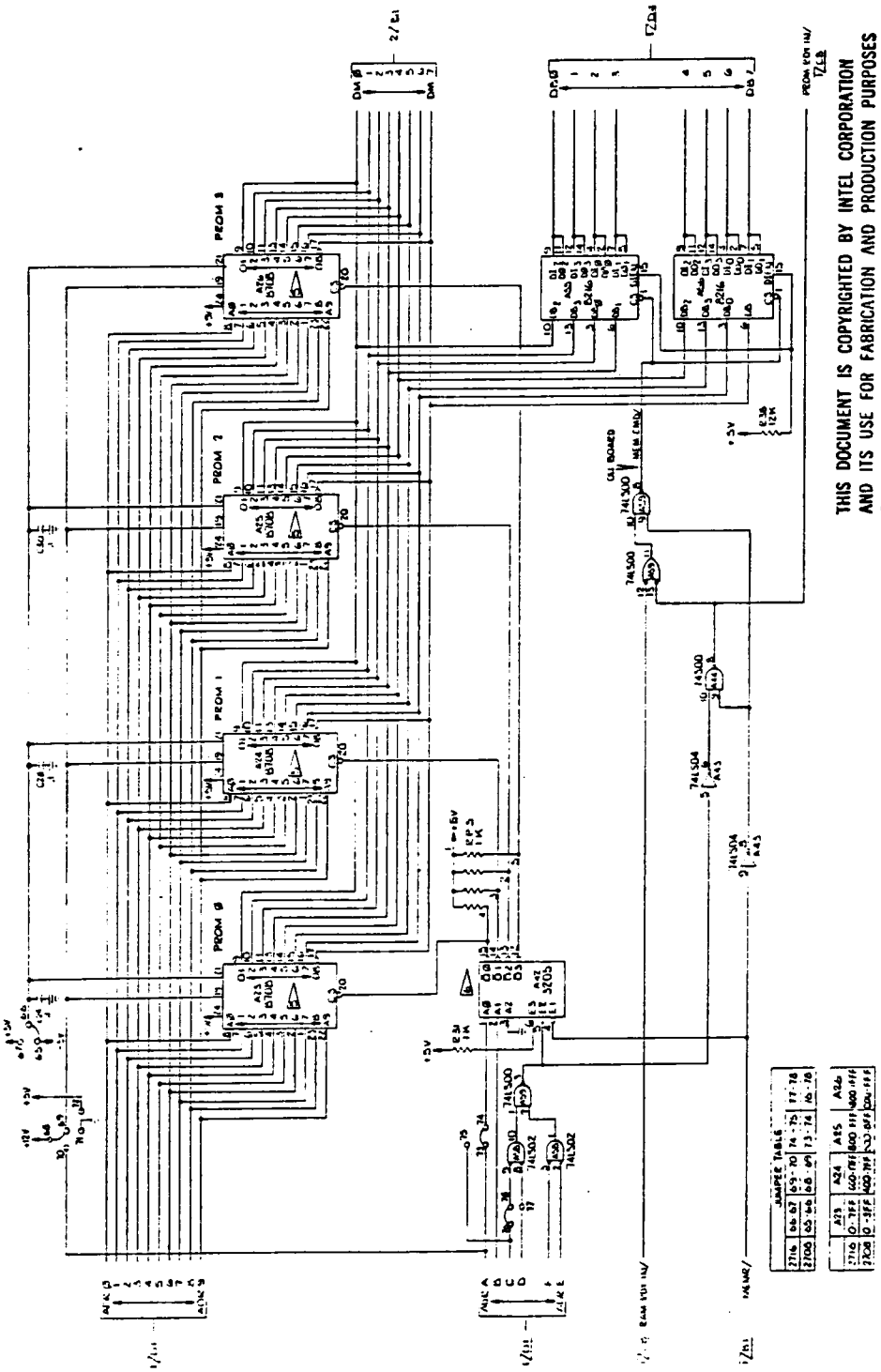


Figure A-1. (Continued)

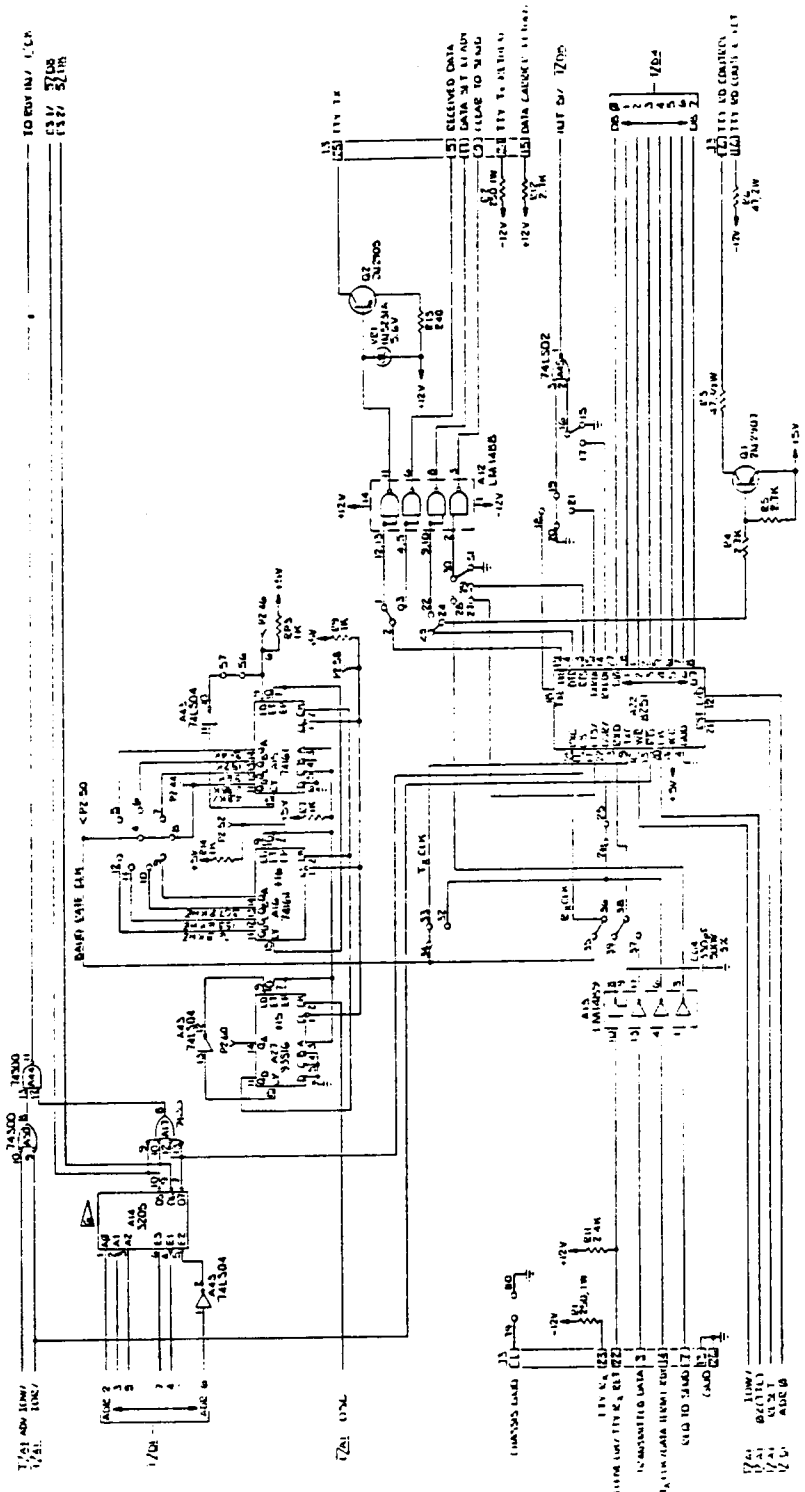


Figure A-1. (Continued)

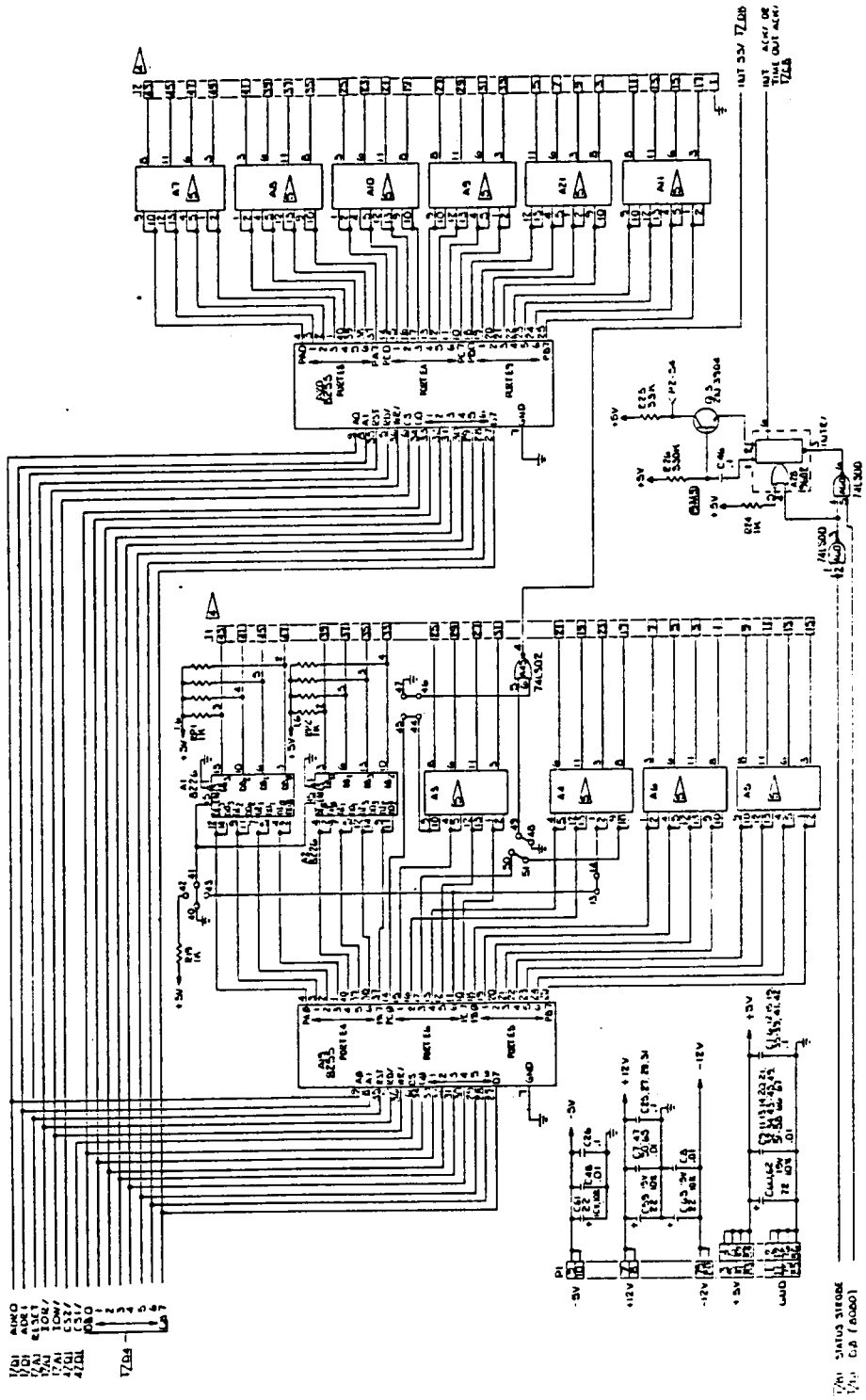


Figure A-1. (Continued)

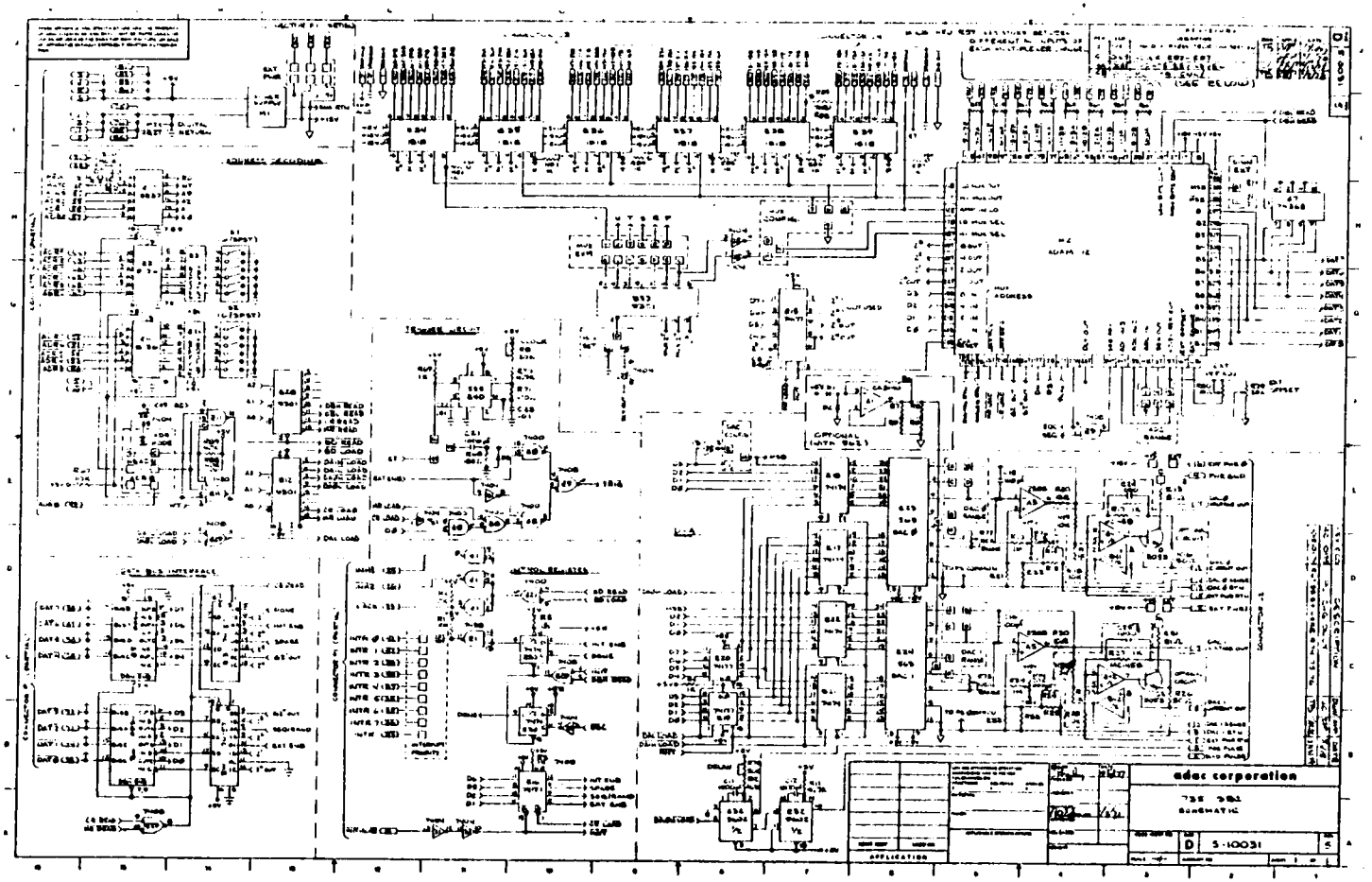


Figure A-2. Schematic for ADAC 735-SBC

Source: Instruction Manual for ADAC Corporation Model 735-SBC Data Acquisition and Control System.[2]

APPENDIX B

APPENDIX B

In this appendix, the author attempted to use structure charts to describe the flow of the programs. The original microcomputer program could be done with ease, since there were few, if any, jumps in the program. On the other hand, in the first and second revisions, the structure charts got very awkward due to frequent jumps in the programs. For this reason, the author described these programs in the form of flow charts.

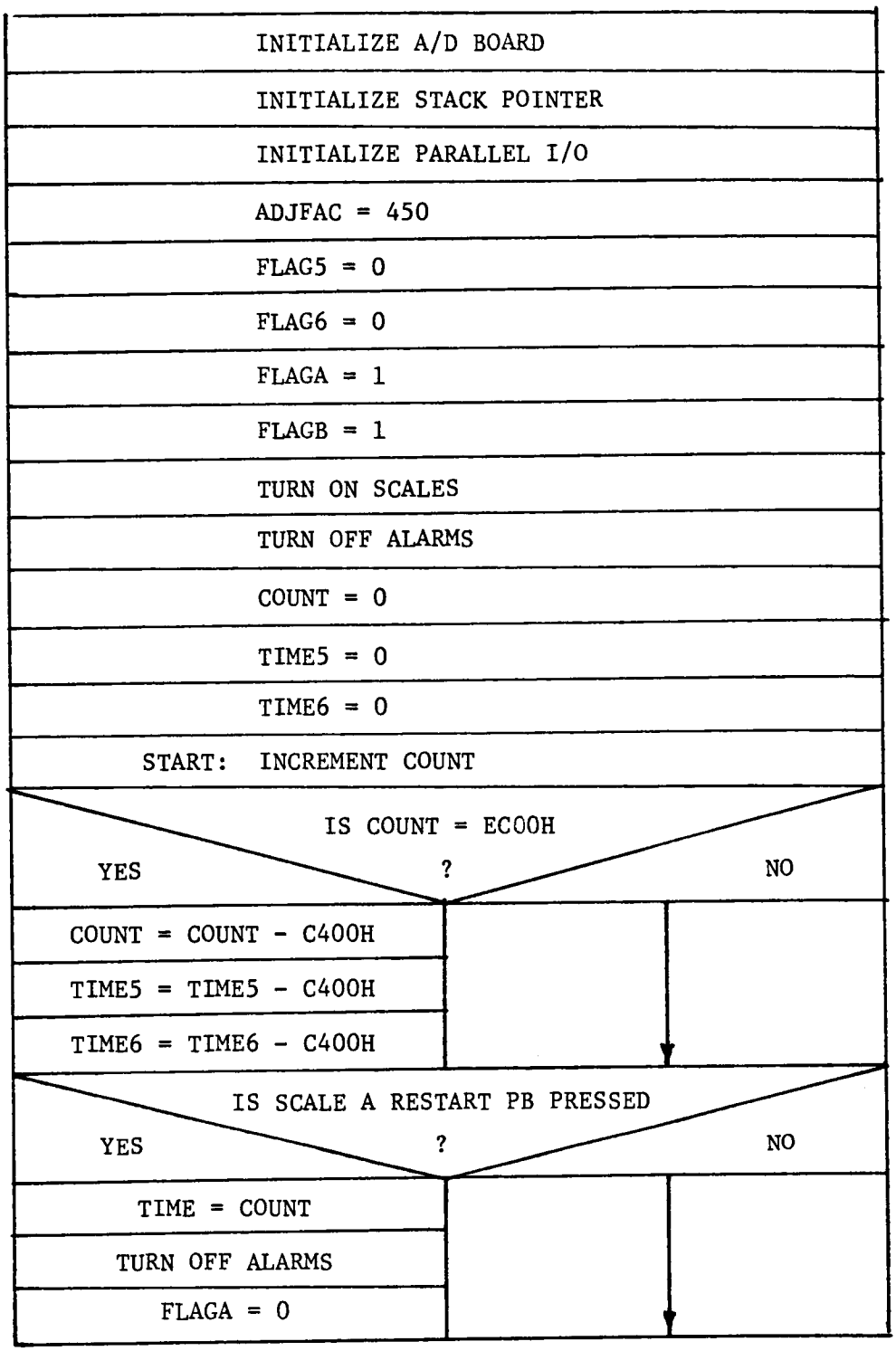


Figure B-1. Structure Chart for Original Microcomputer Installation

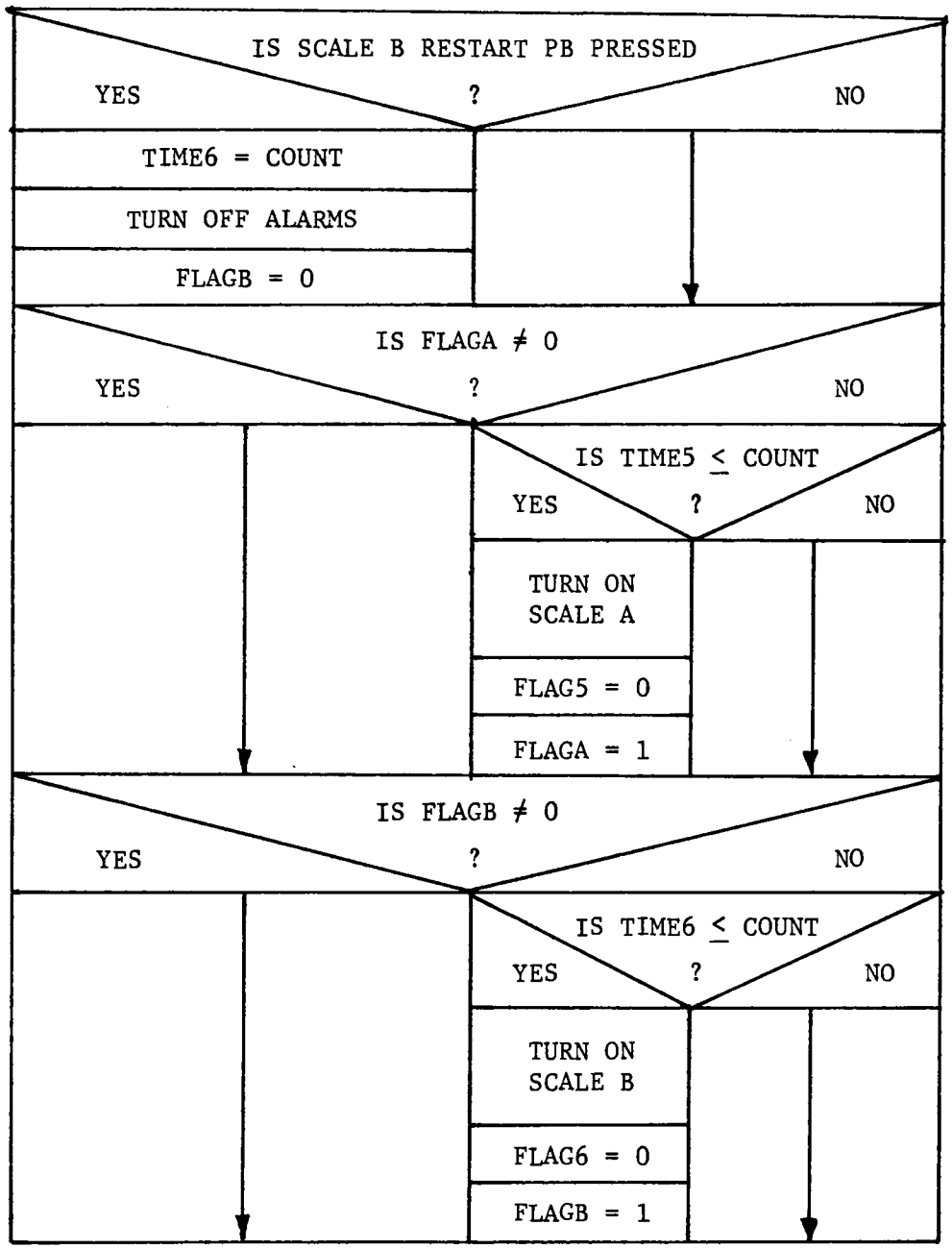


Figure B-1. (Continued)

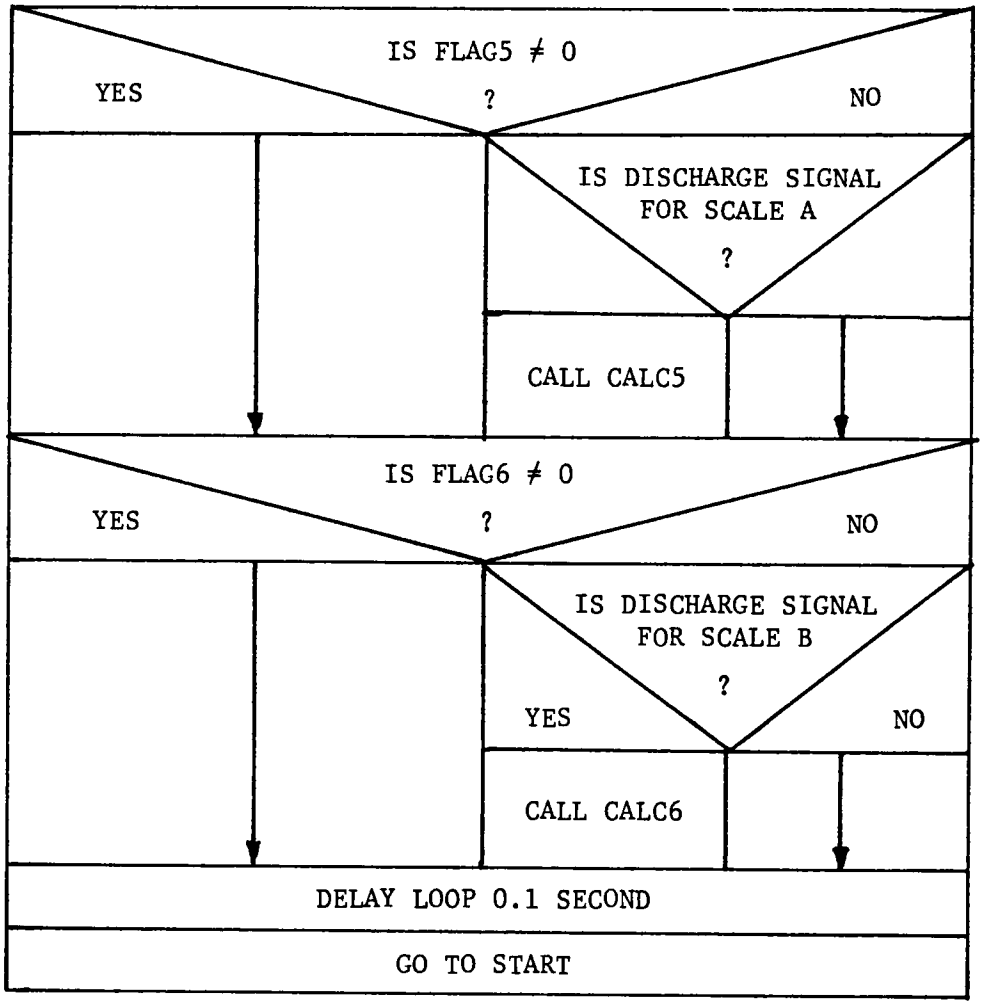


Figure B-1. (Continued)

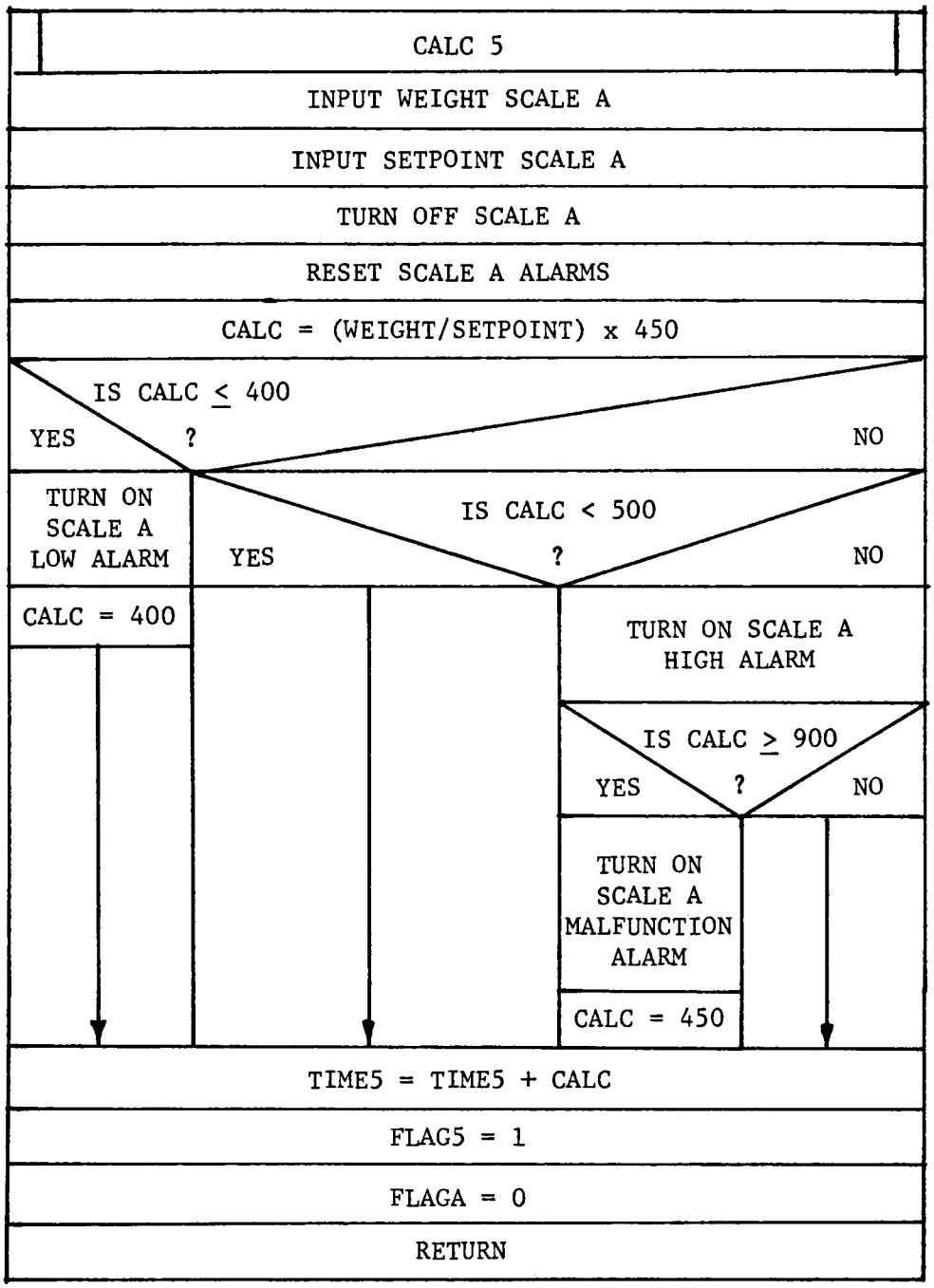


Figure B-1. (Continued)

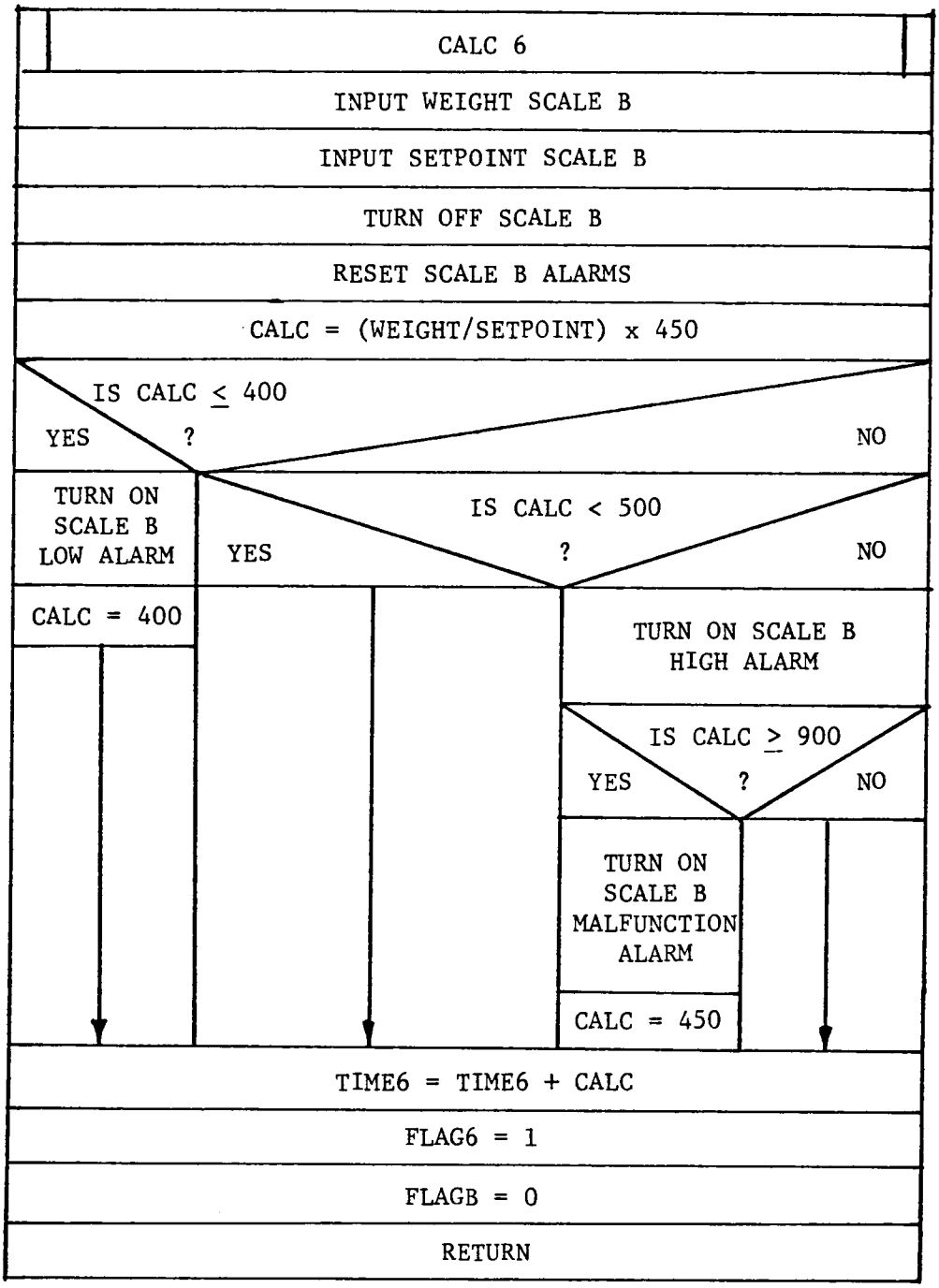


Figure B-1. (Continued)

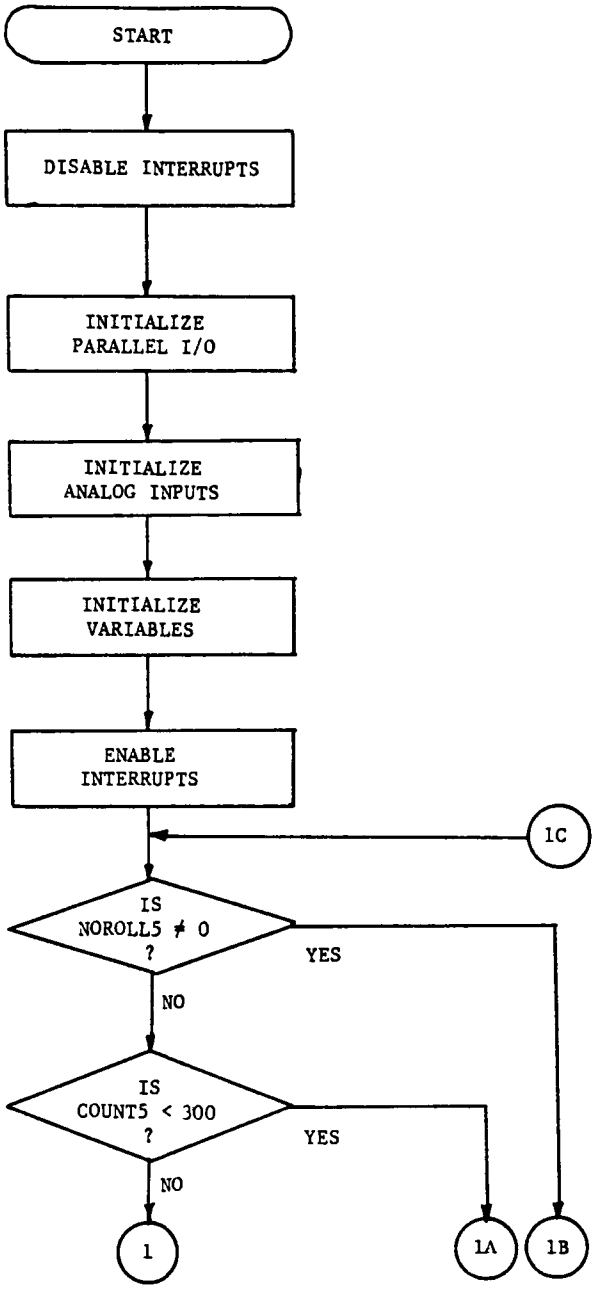


Figure B-2. Flow Chart for First Revision

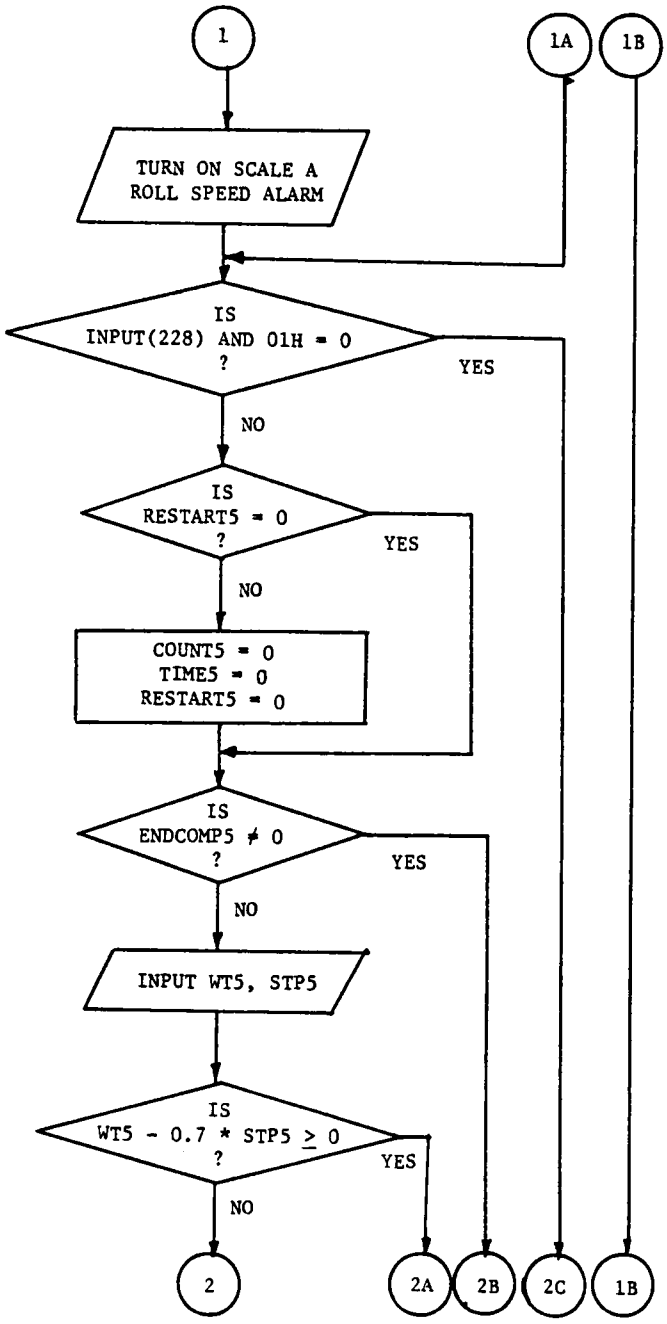


Figure B-2. (Continued)

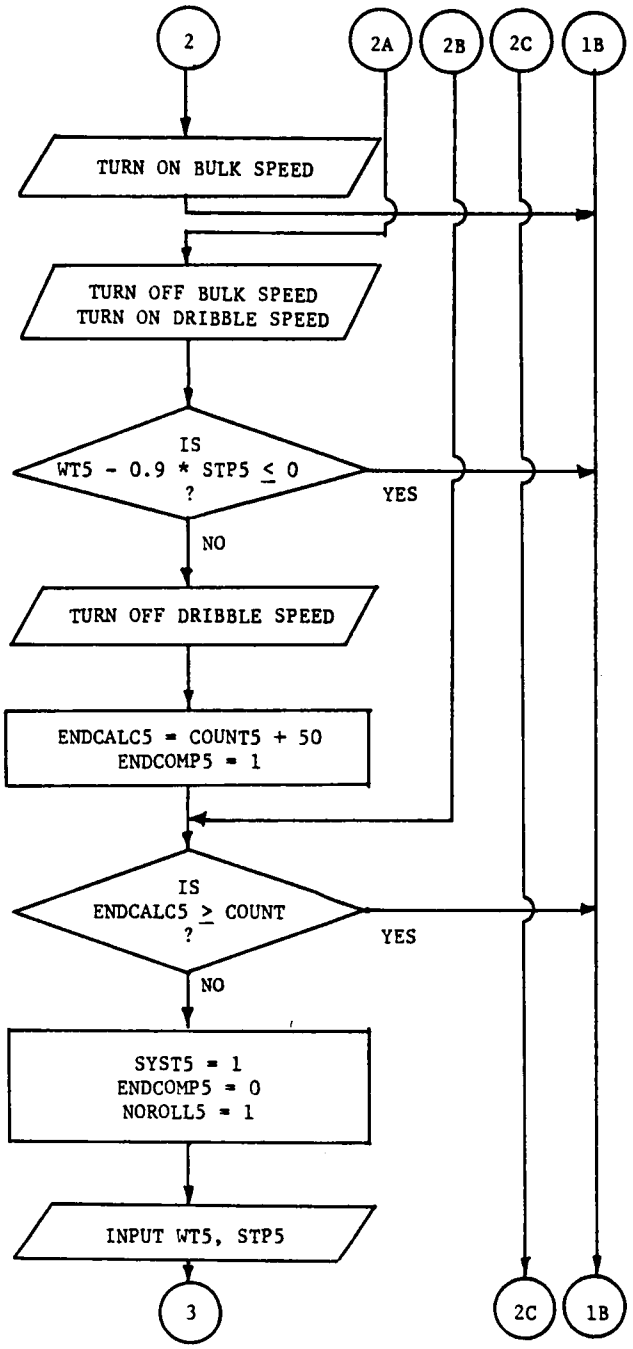


Figure B-2. (Continued)

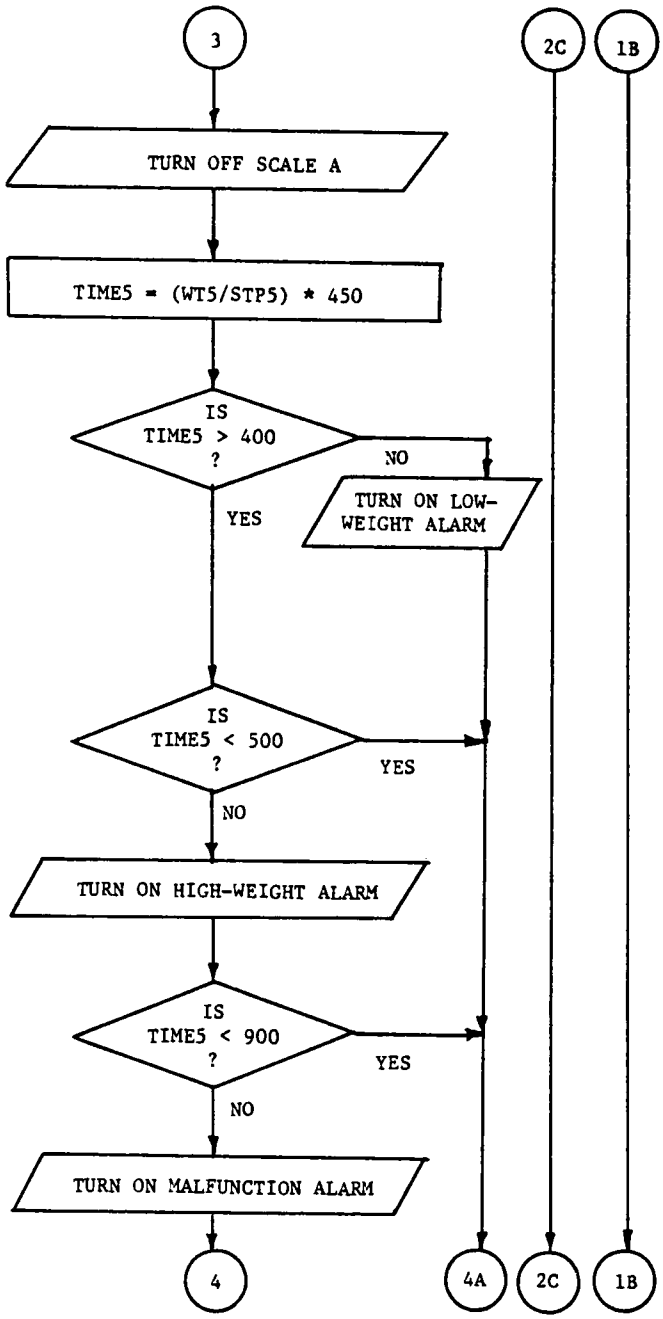


Figure B-2. (Continued)

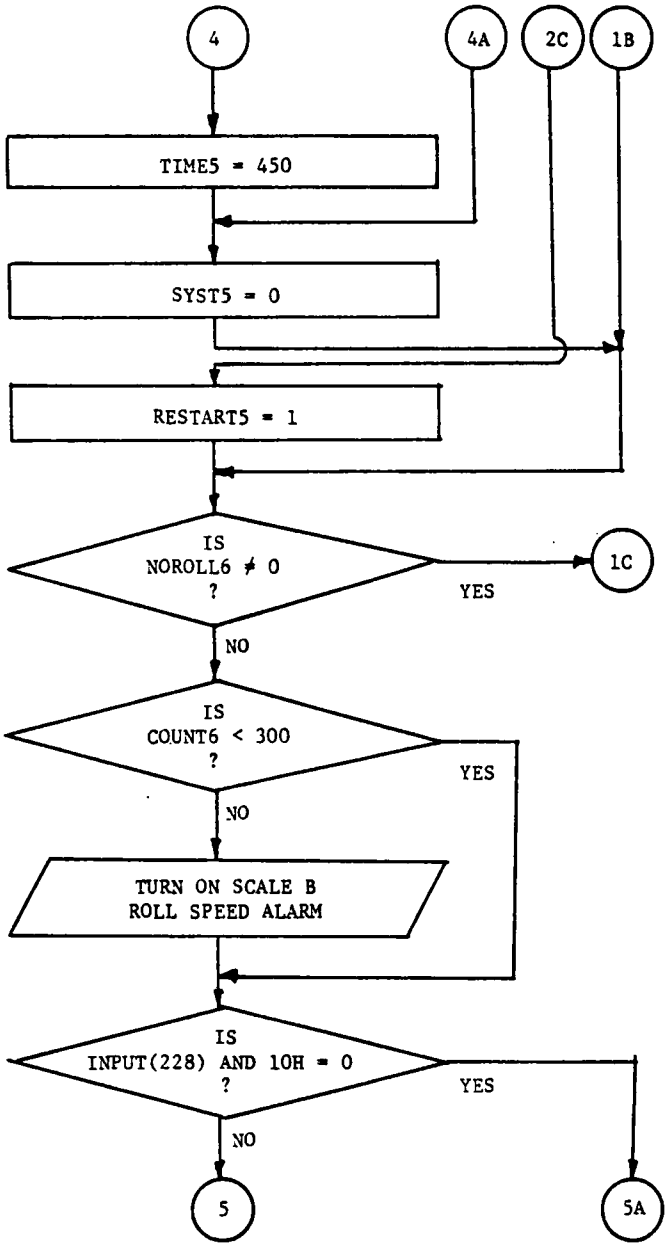


Figure B-2. (Continued)

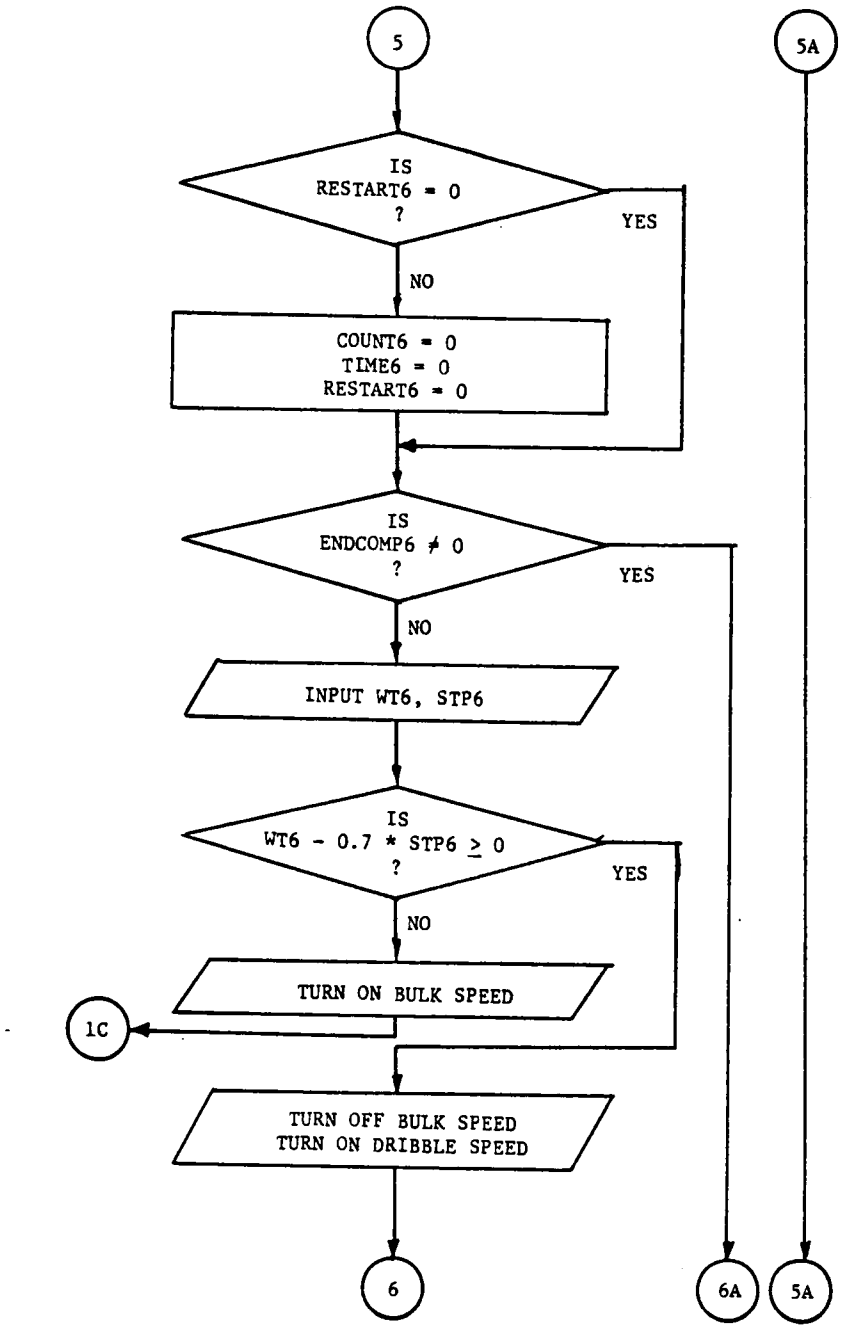


Figure B-2. (Continued)

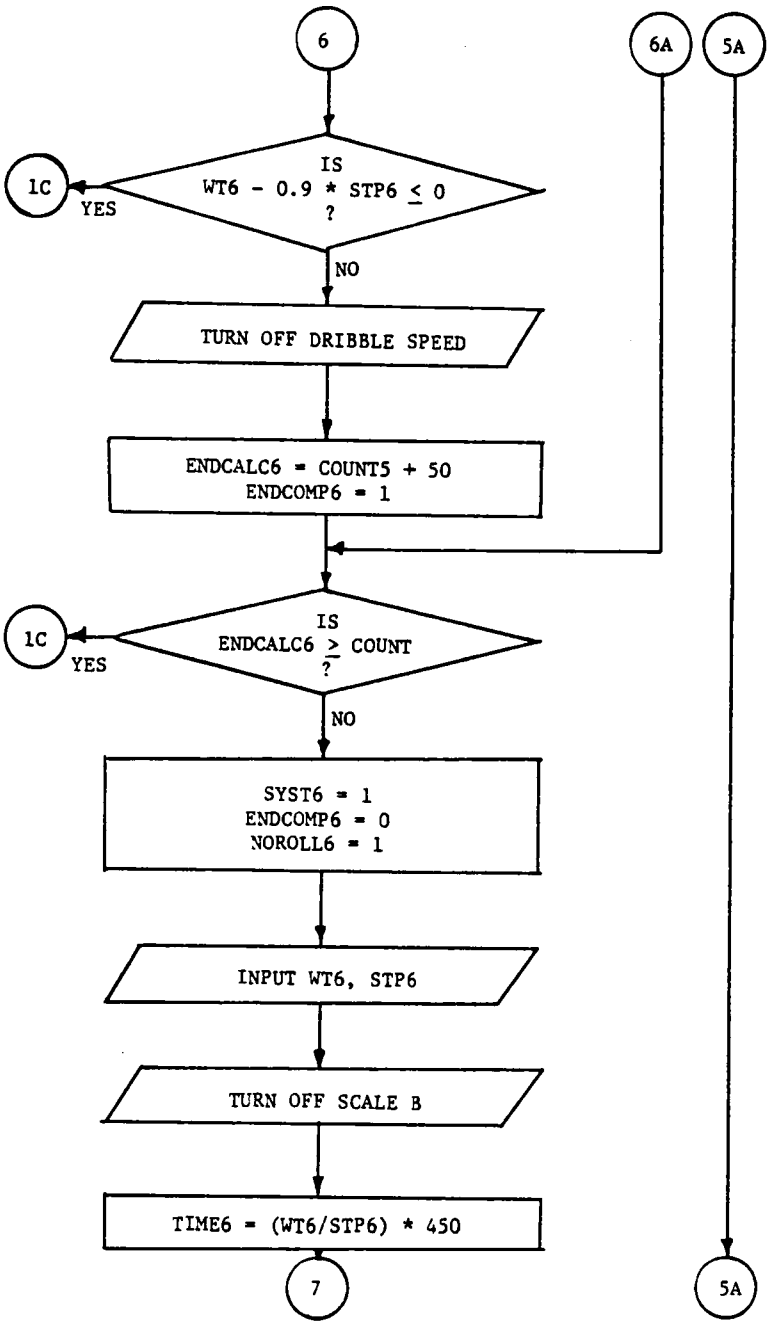


Figure B-2. (Continued)

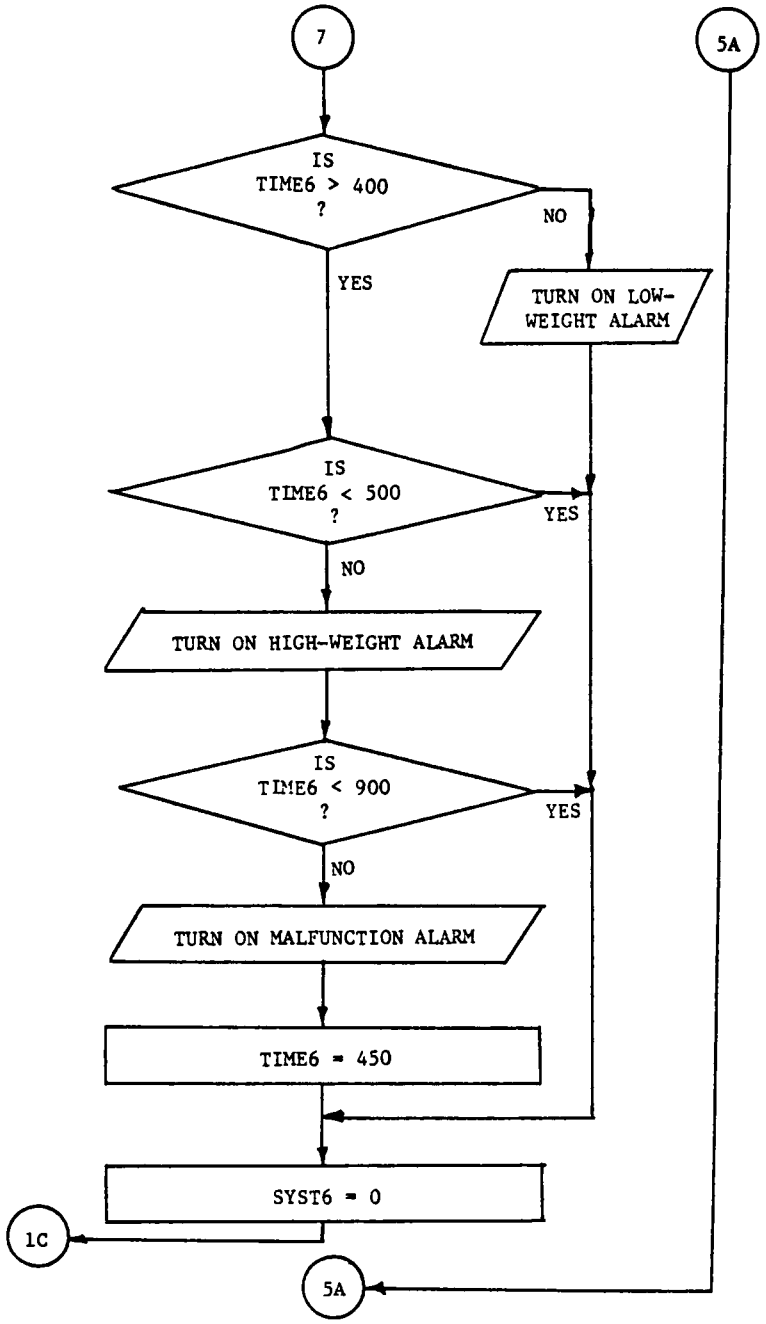


Figure B-2. (Continued)

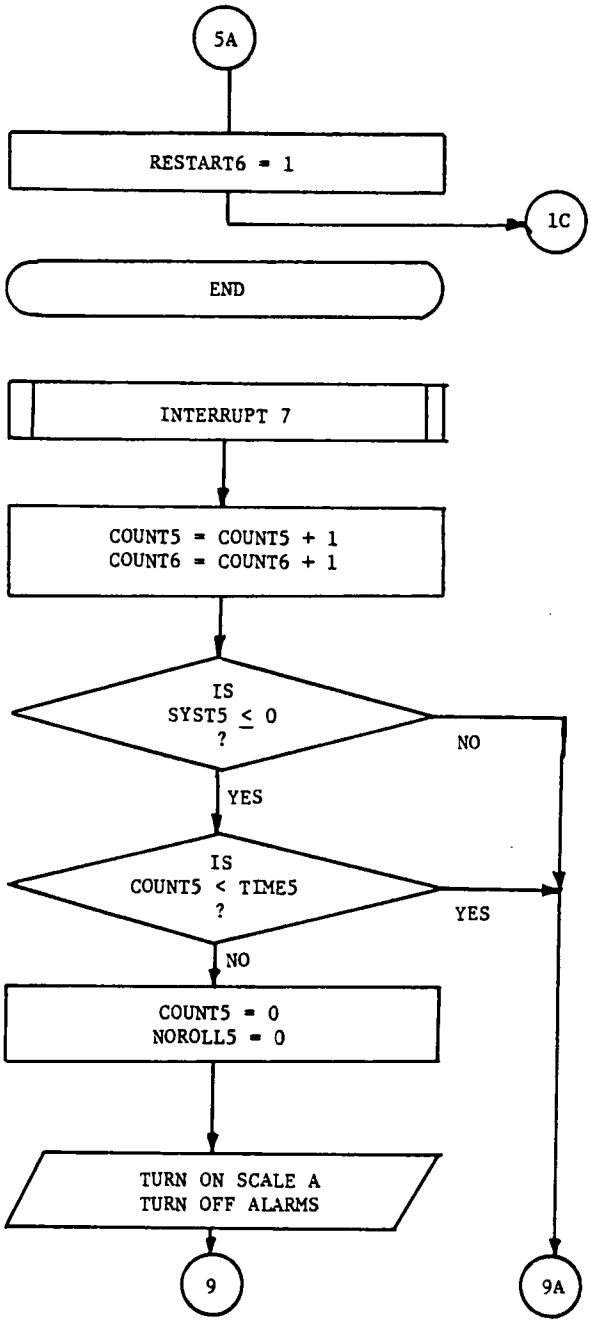


Figure B-2. (Continued)

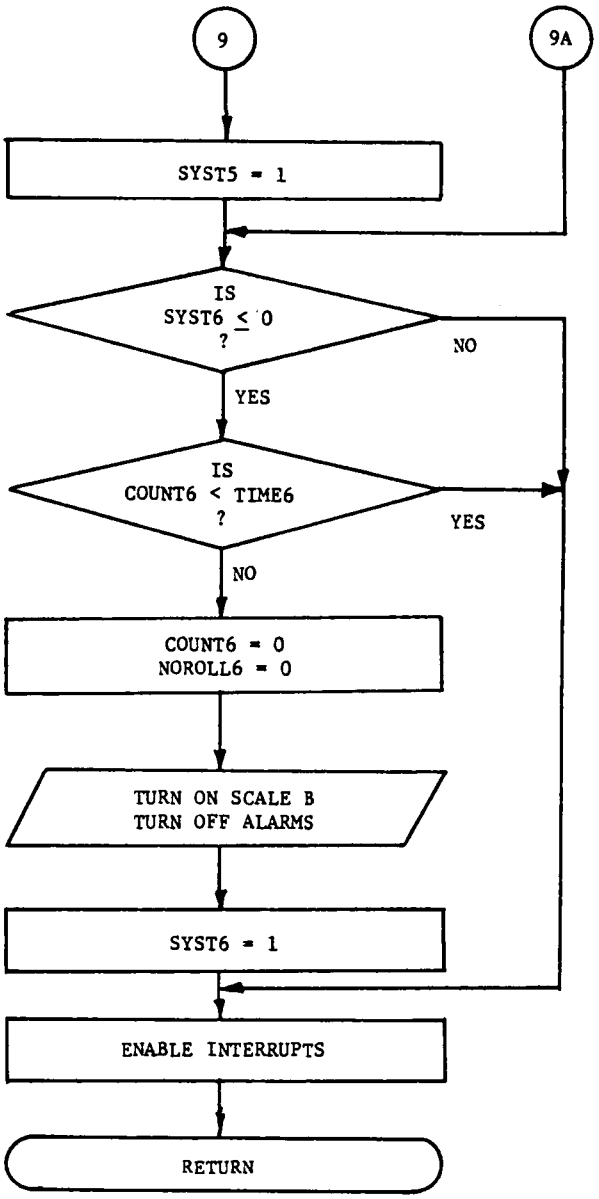


Figure B-2. (Continued)

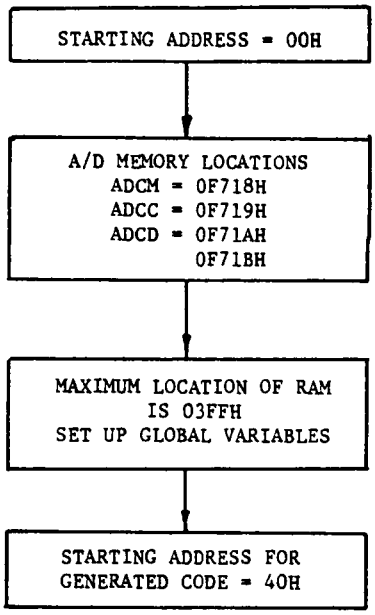


Figure B-3. Flow Chart for Second Revision

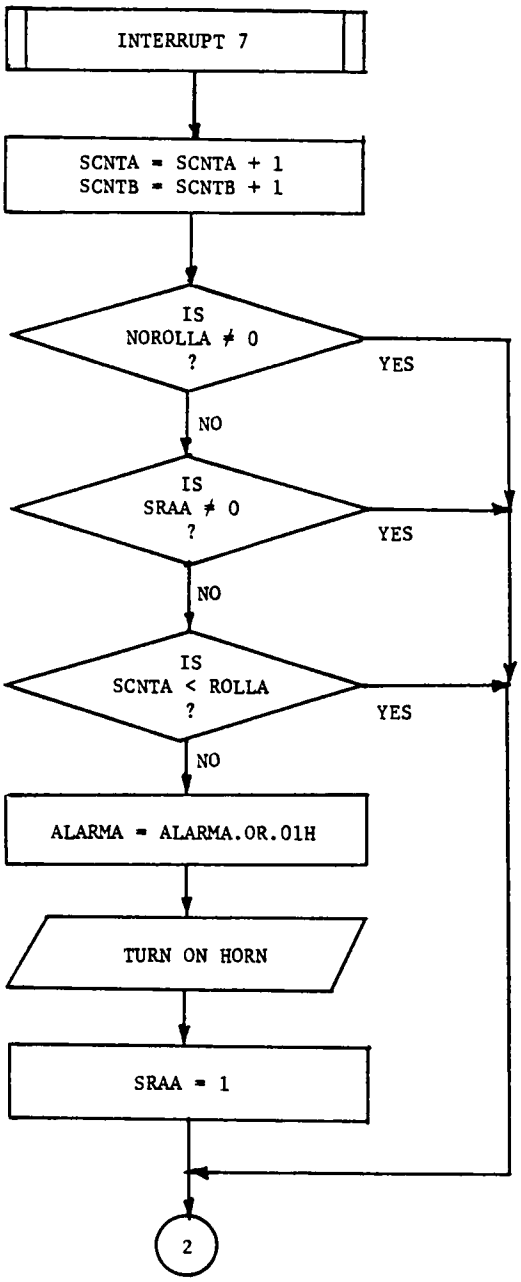


Figure B-3. (Continued)

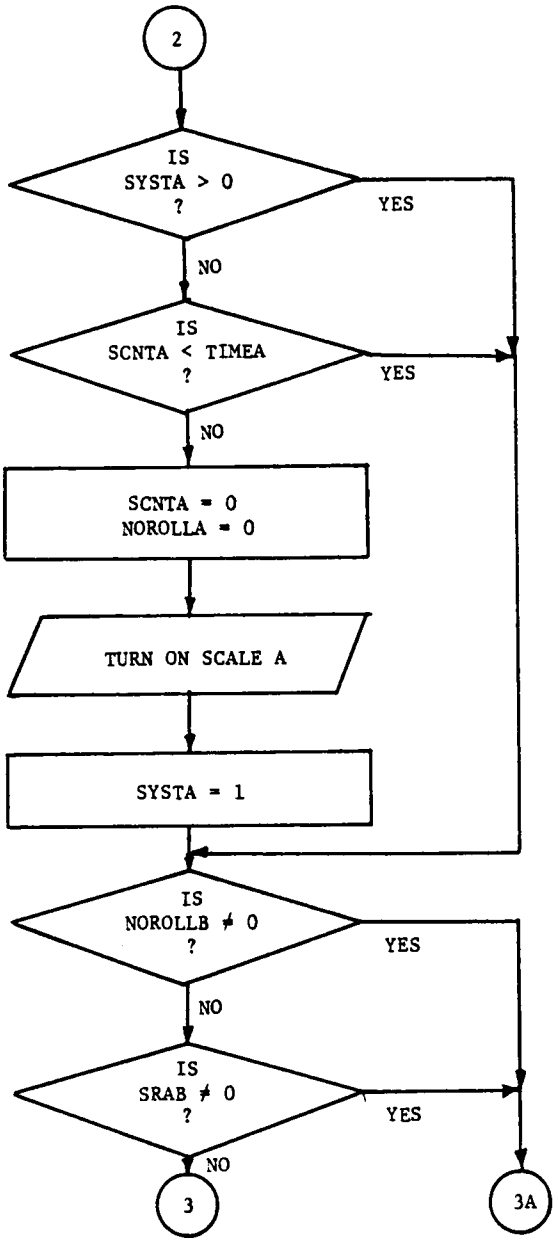


Figure B-3. (Continued)

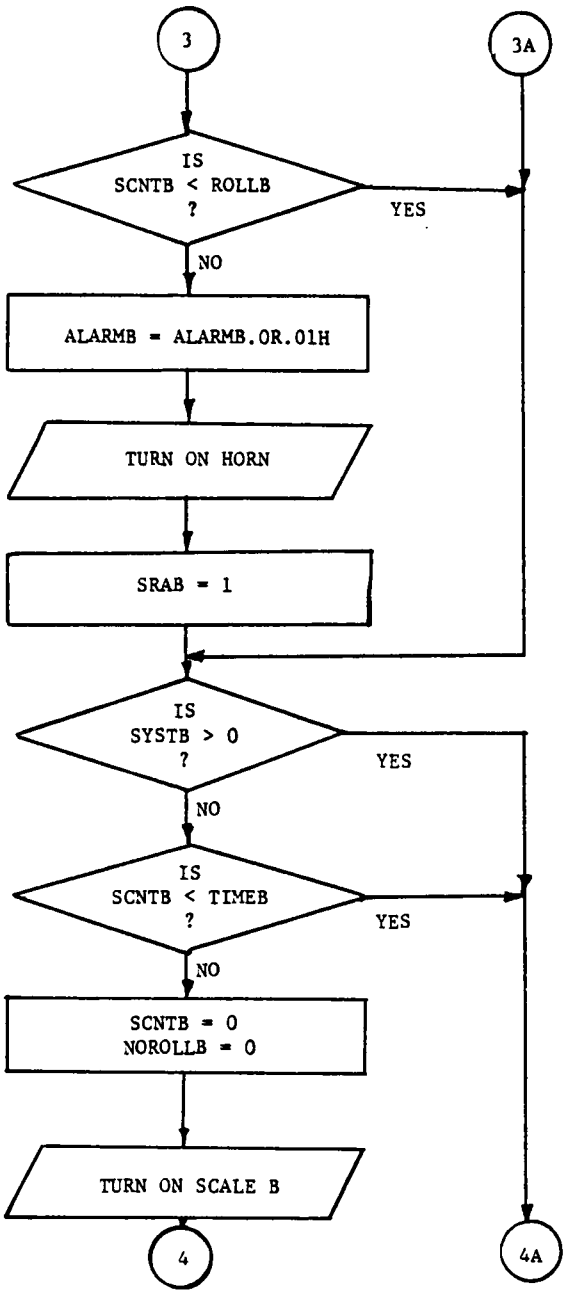


Figure B-3. (Continued)

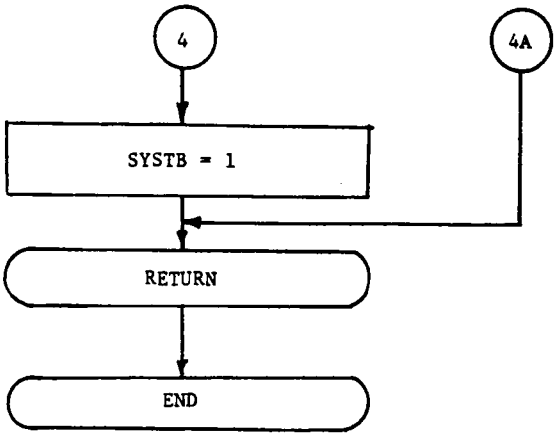


Figure B-3. (Continued)

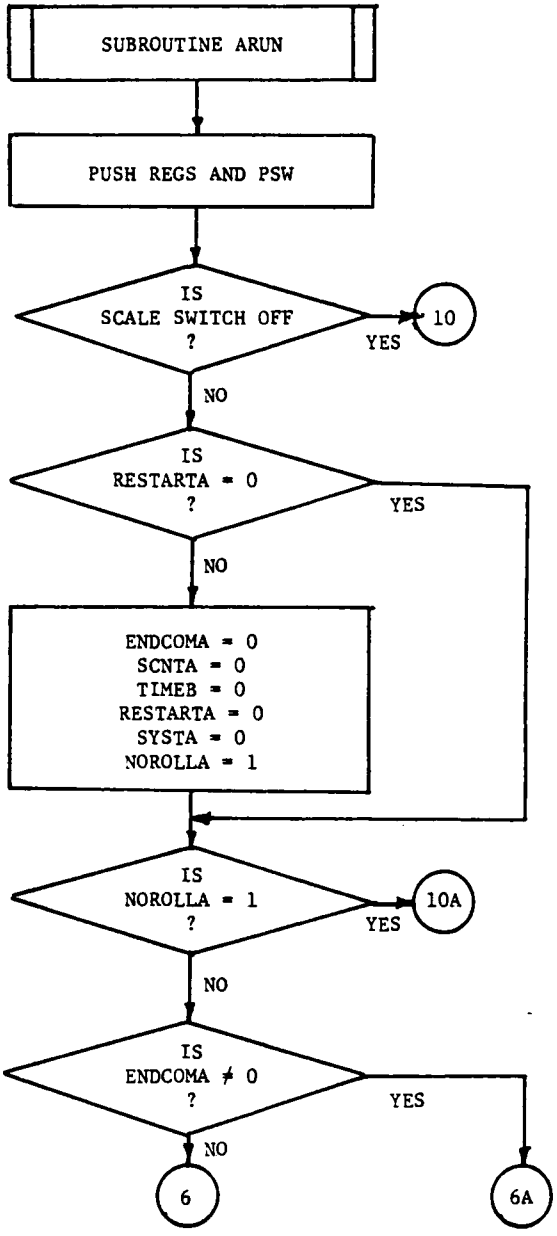


Figure B-3. (Continued)

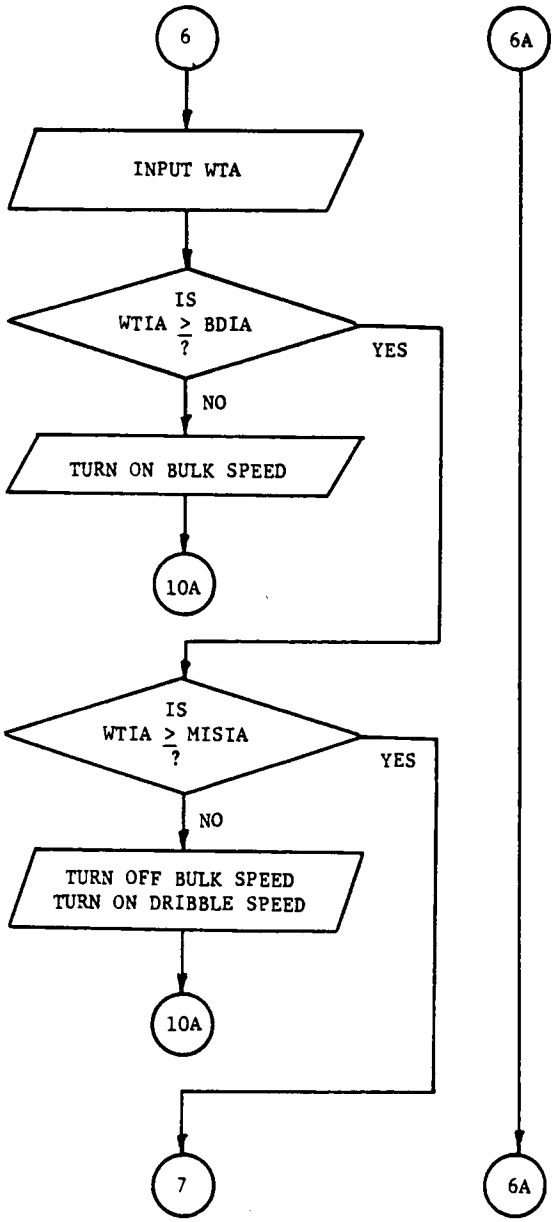


Figure B-3. (Continued)

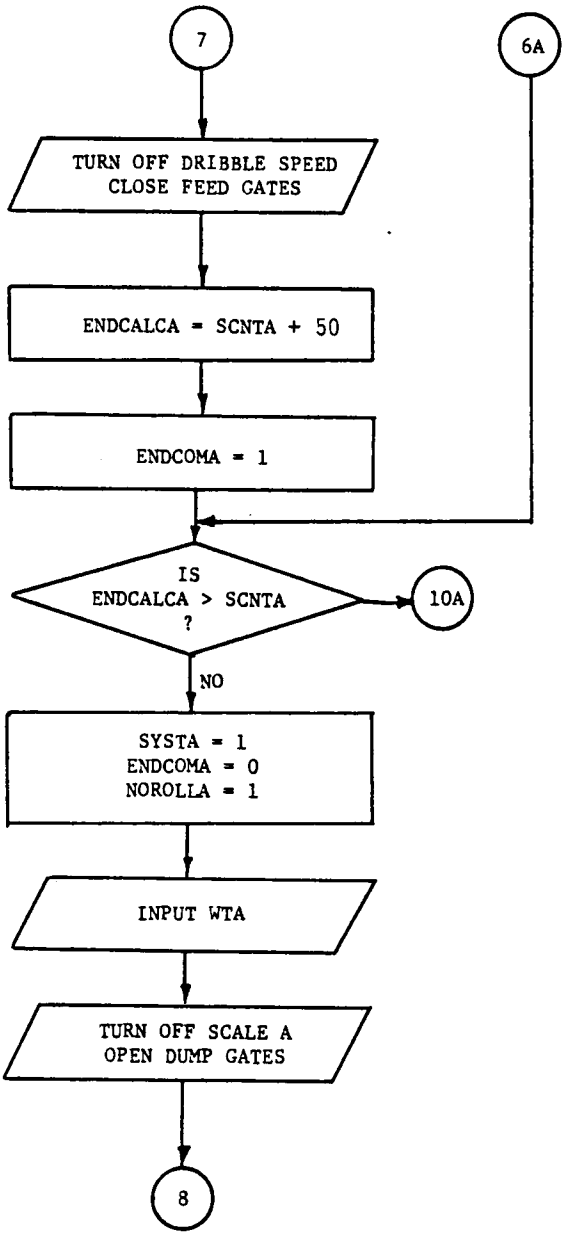


Figure B-3. (Continued)

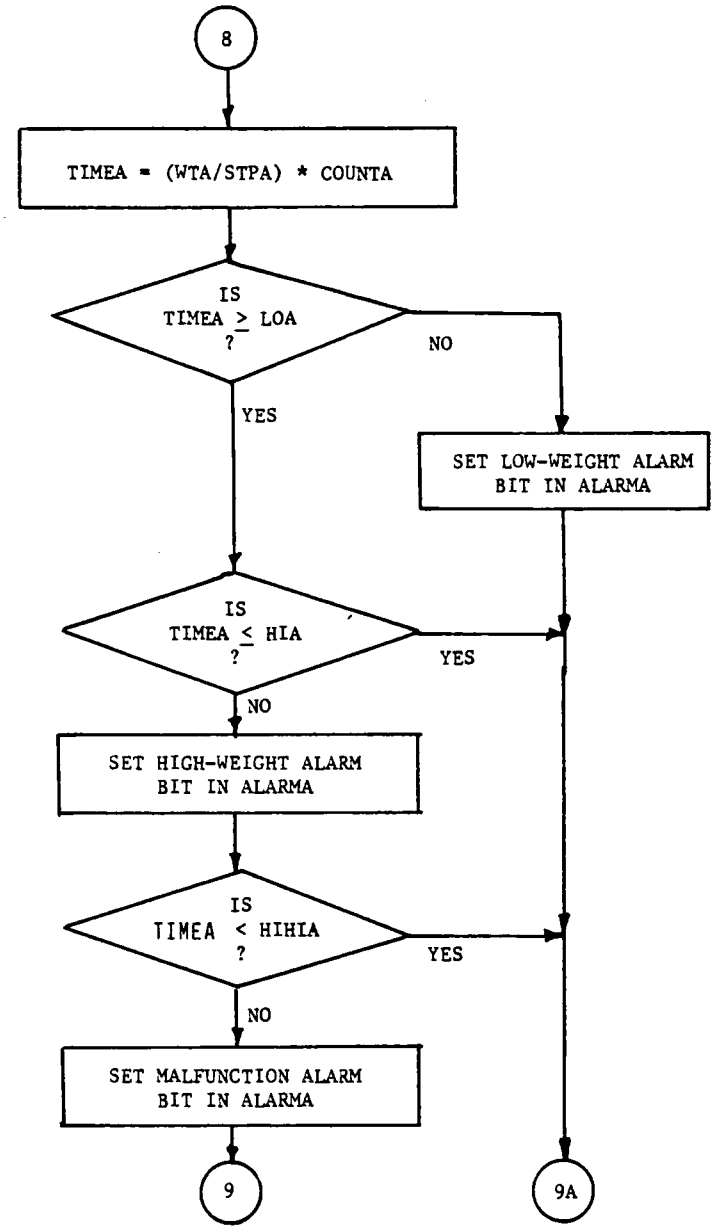


Figure B-3. (Continued)

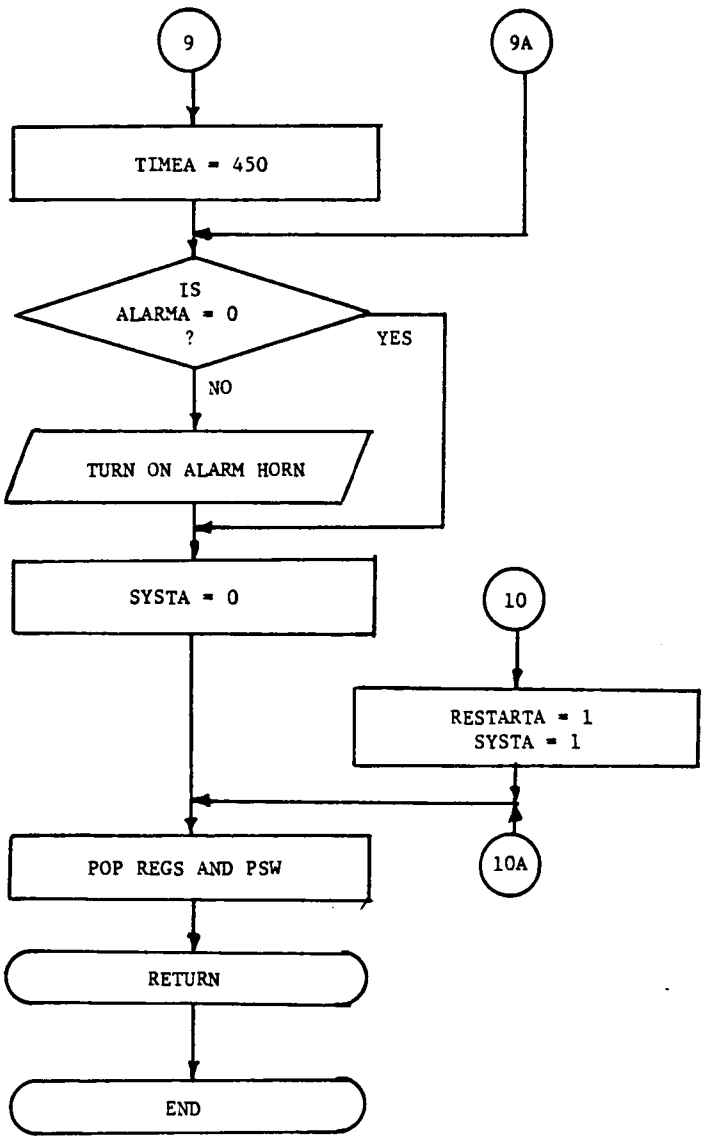
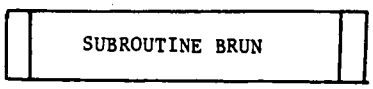


Figure B-3. (Continued)



Note: Subroutine BRUN is the same as ARUN except with scale B I/O and constants.

Figure B-3. (Continued)

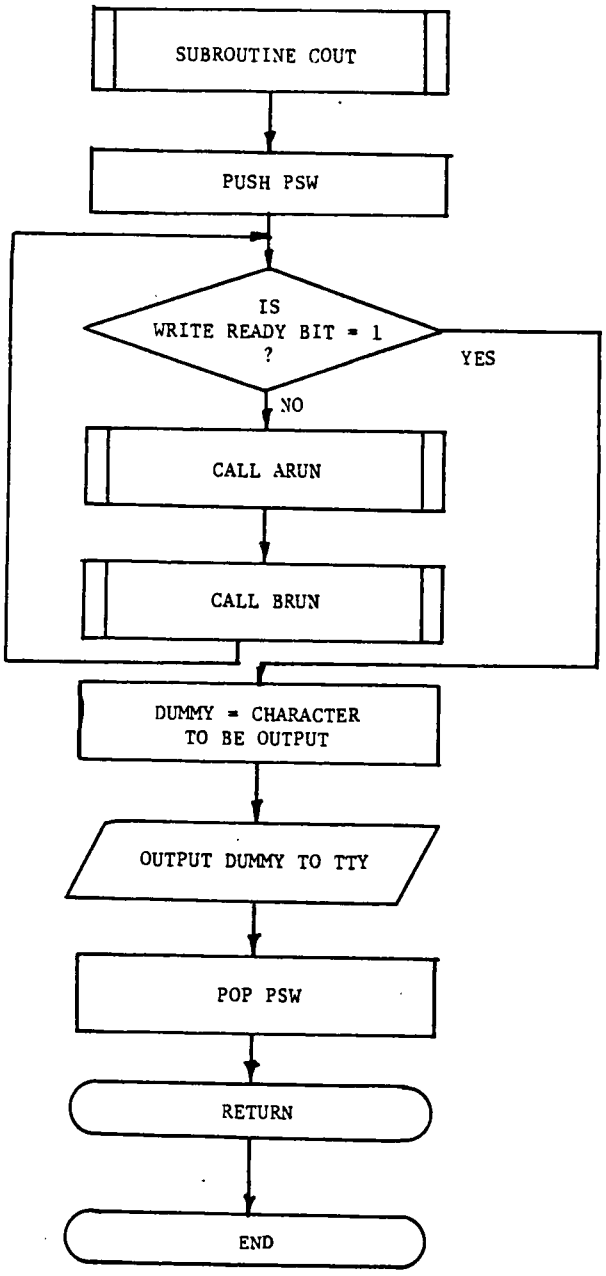


Figure B-3. (Continued)

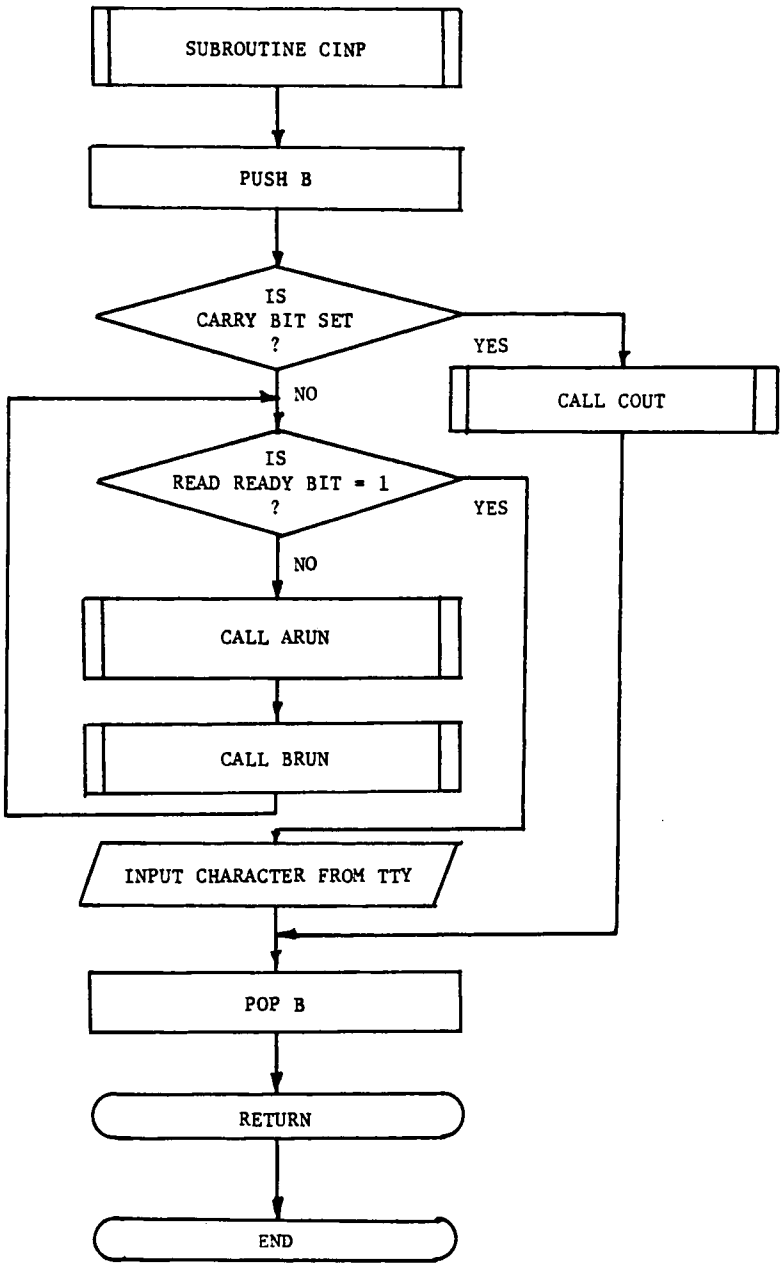


Figure B-3. (Continued)

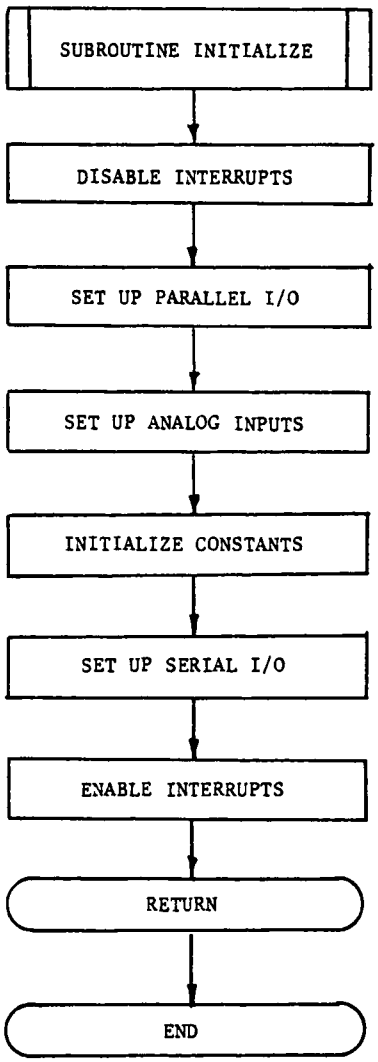


Figure B-3. (Continued)

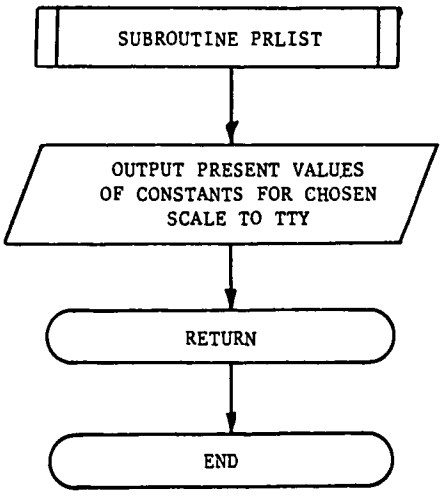


Figure B-3. (Continued)

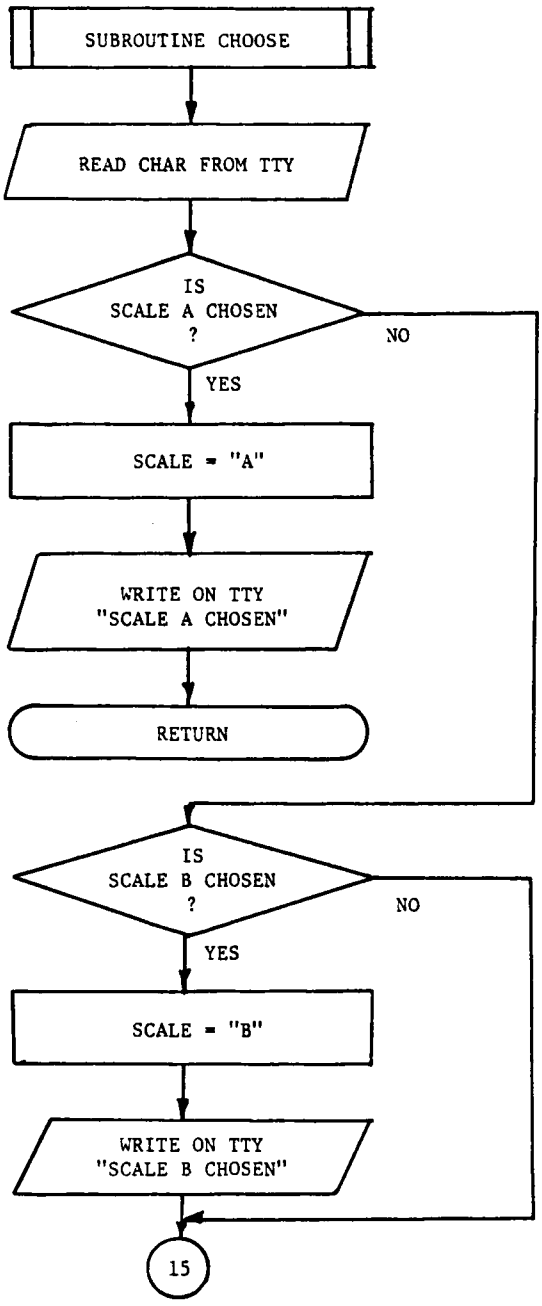


Figure B-3. (Continued)

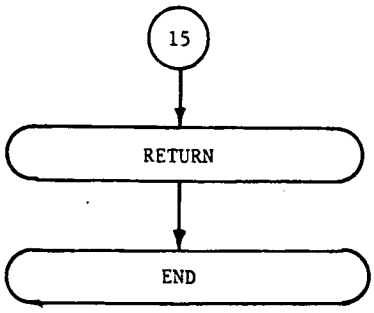


Figure B-3. (Continued)

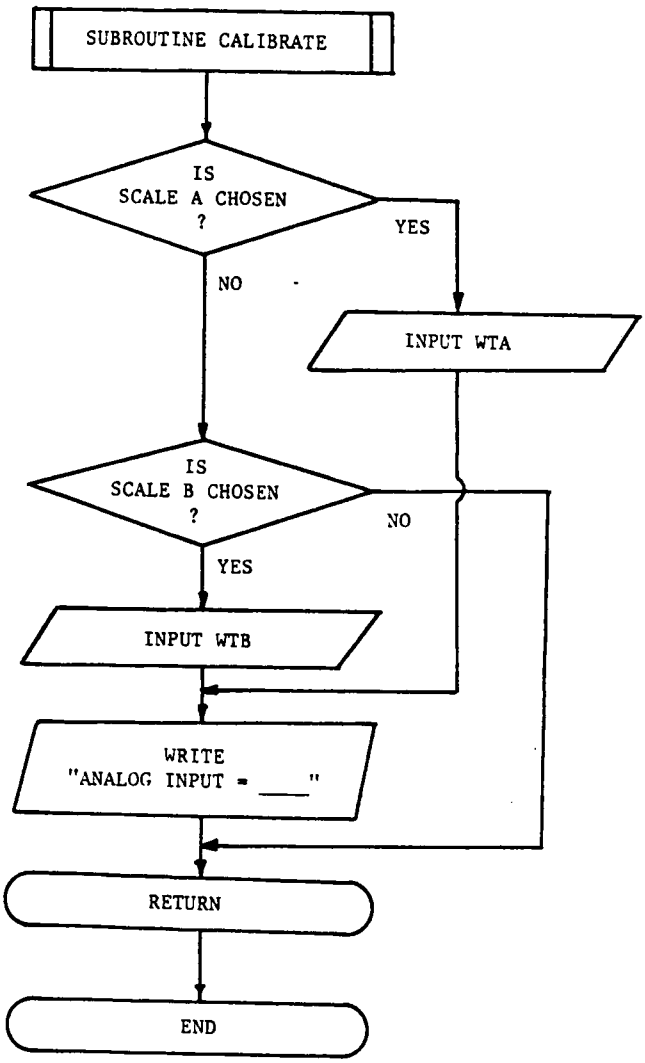


Figure B-3. (Continued)

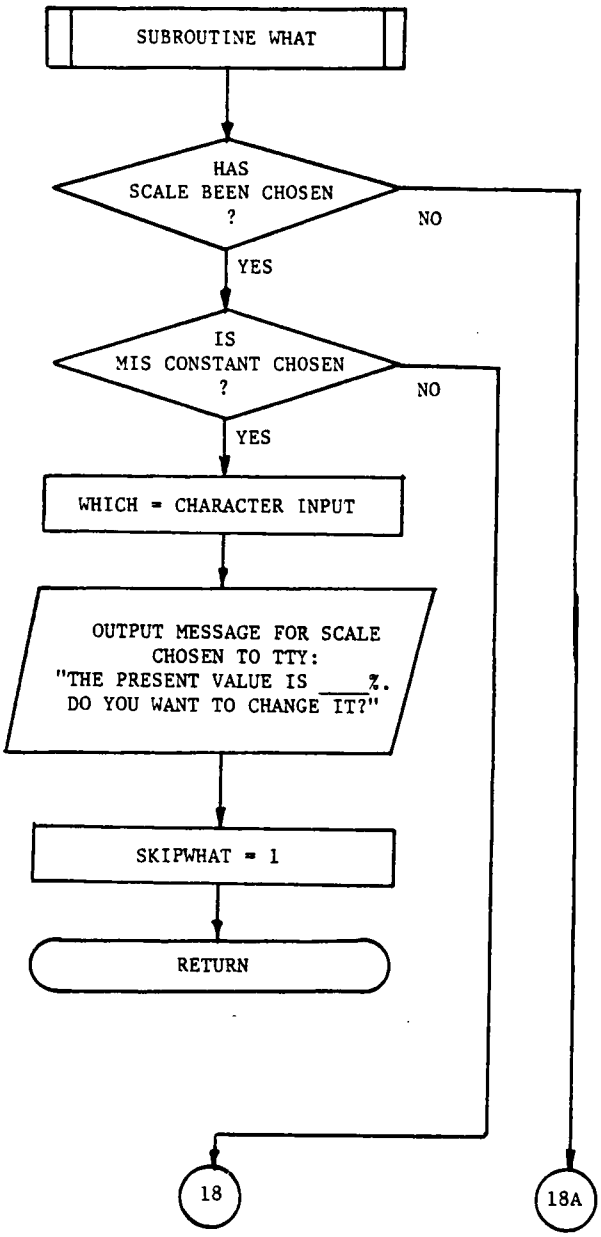


Figure B-3. (Continued)

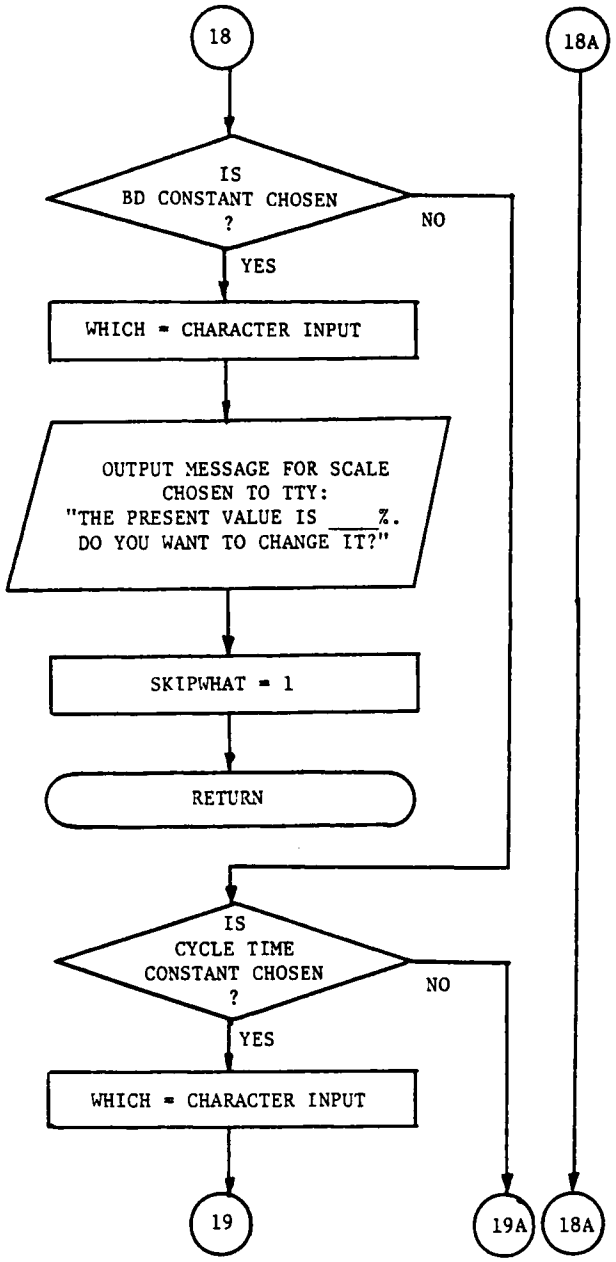


Figure B-3. (Continued)

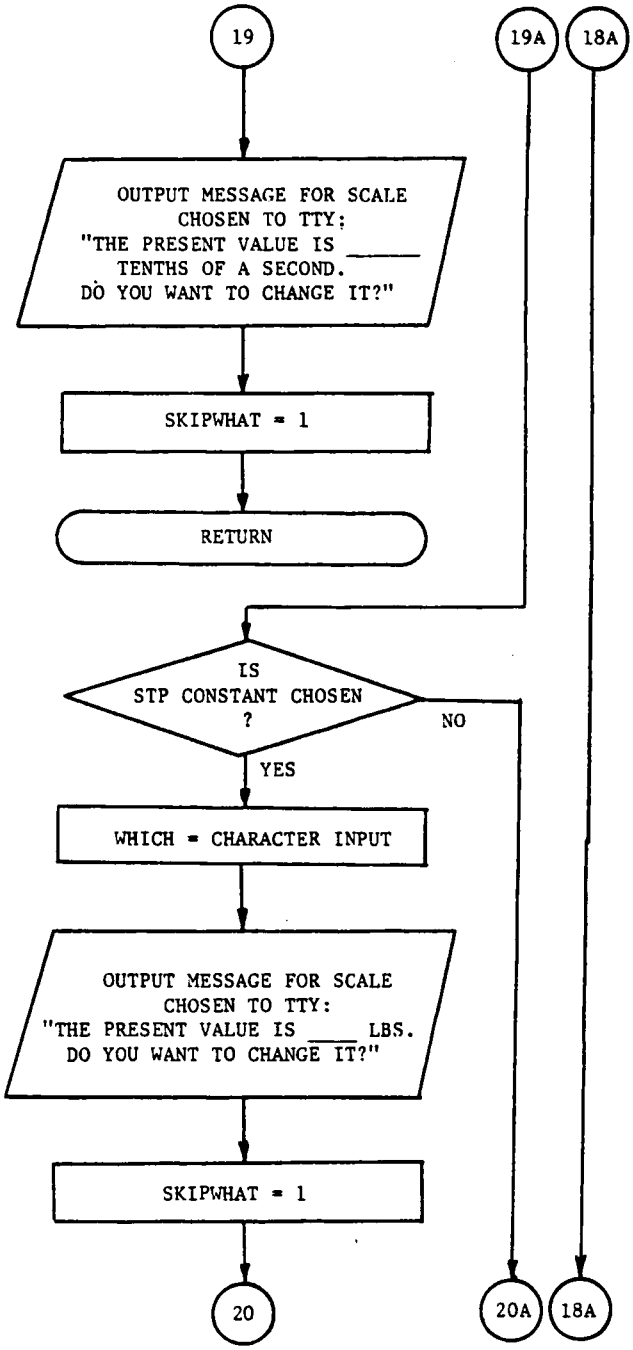


Figure B-3. (Continued)

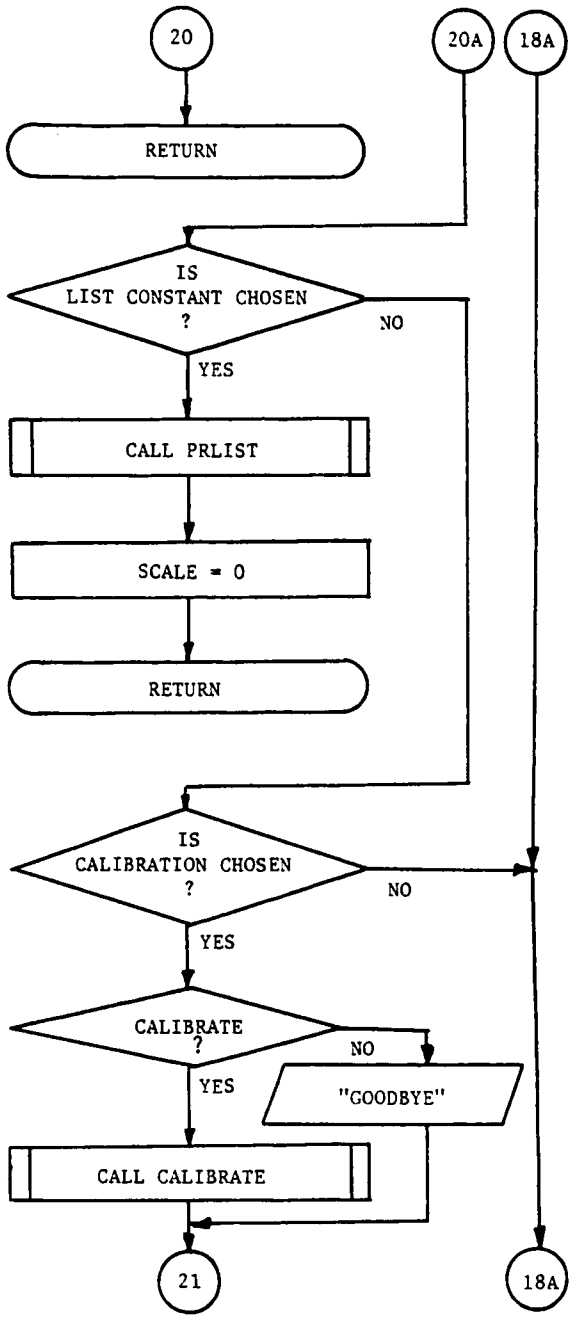


Figure B-3. (Continued)

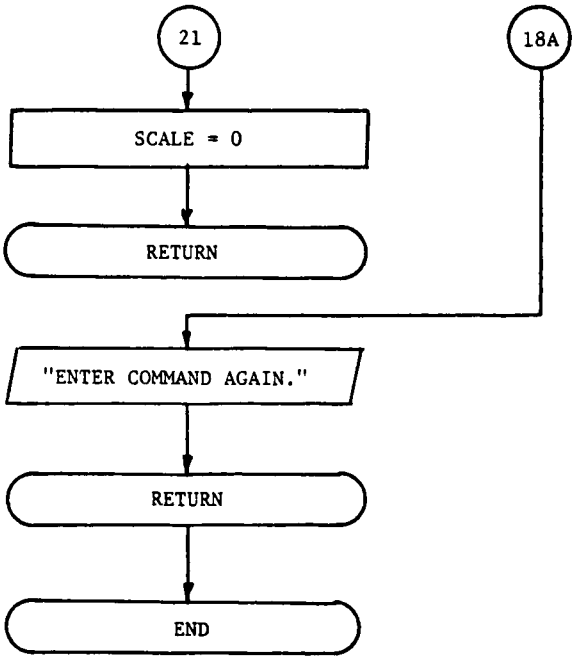


Figure B-3. (Continued)

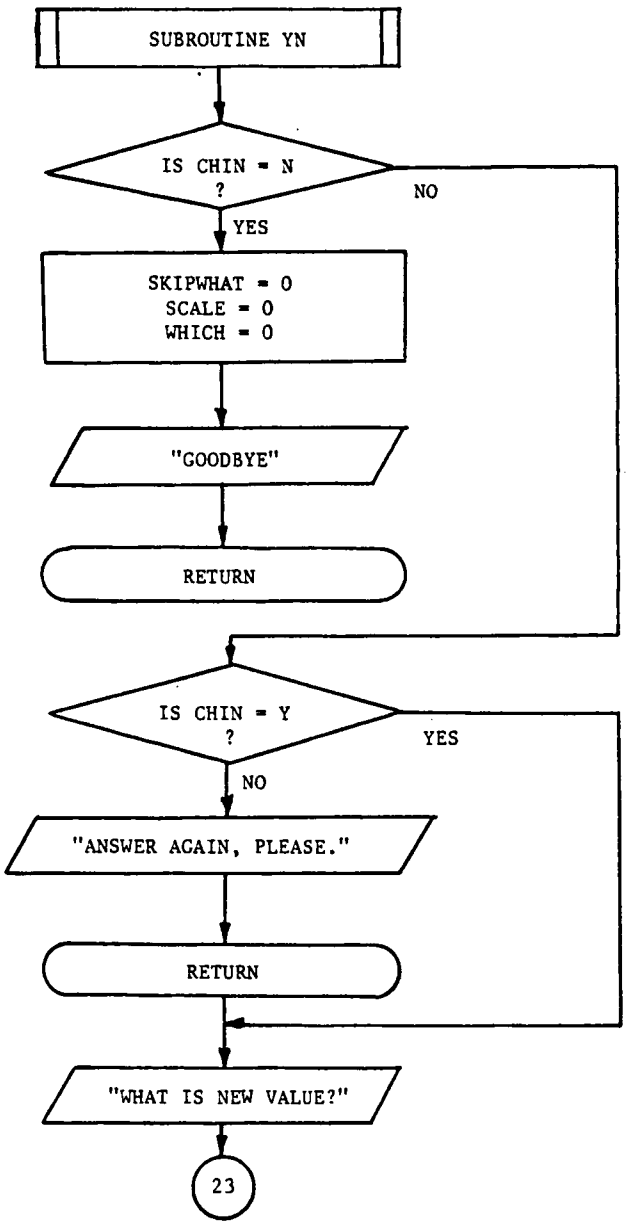


Figure B-3. (Continued)

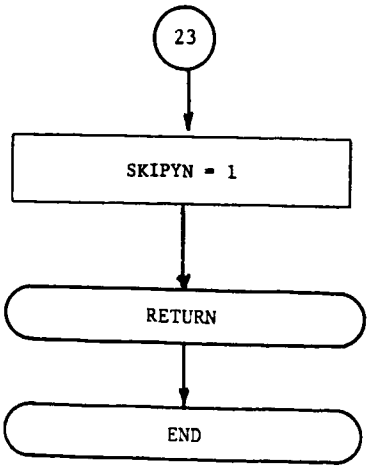


Figure B-3. (Continued)

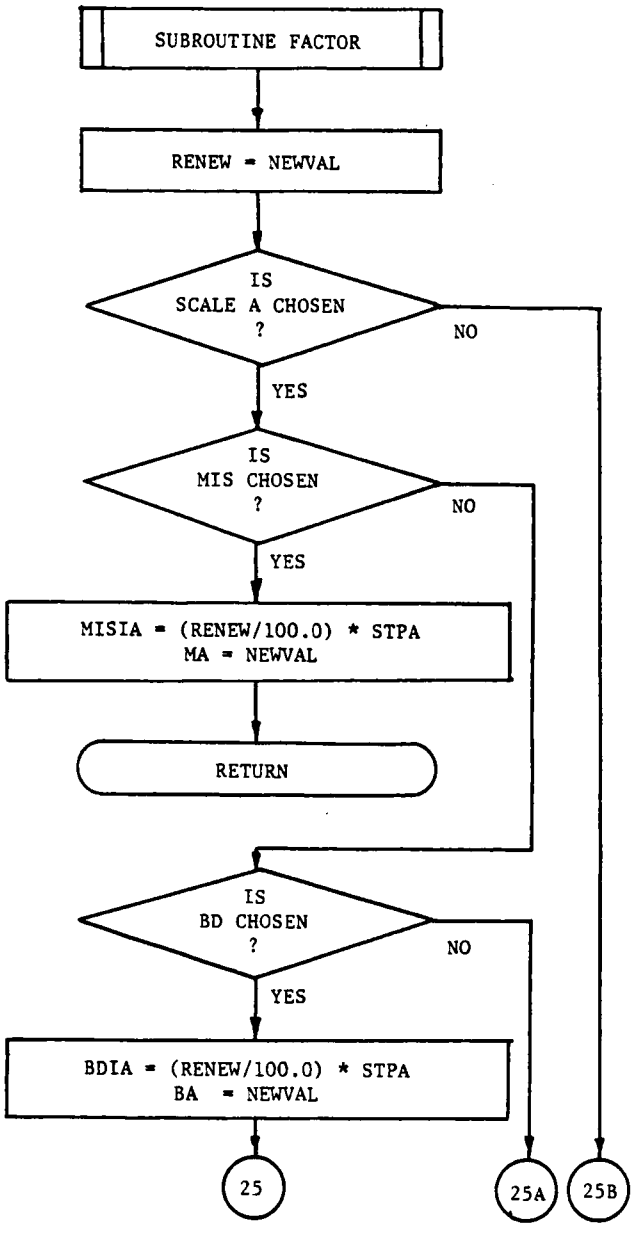


Figure B-3. (Continued)

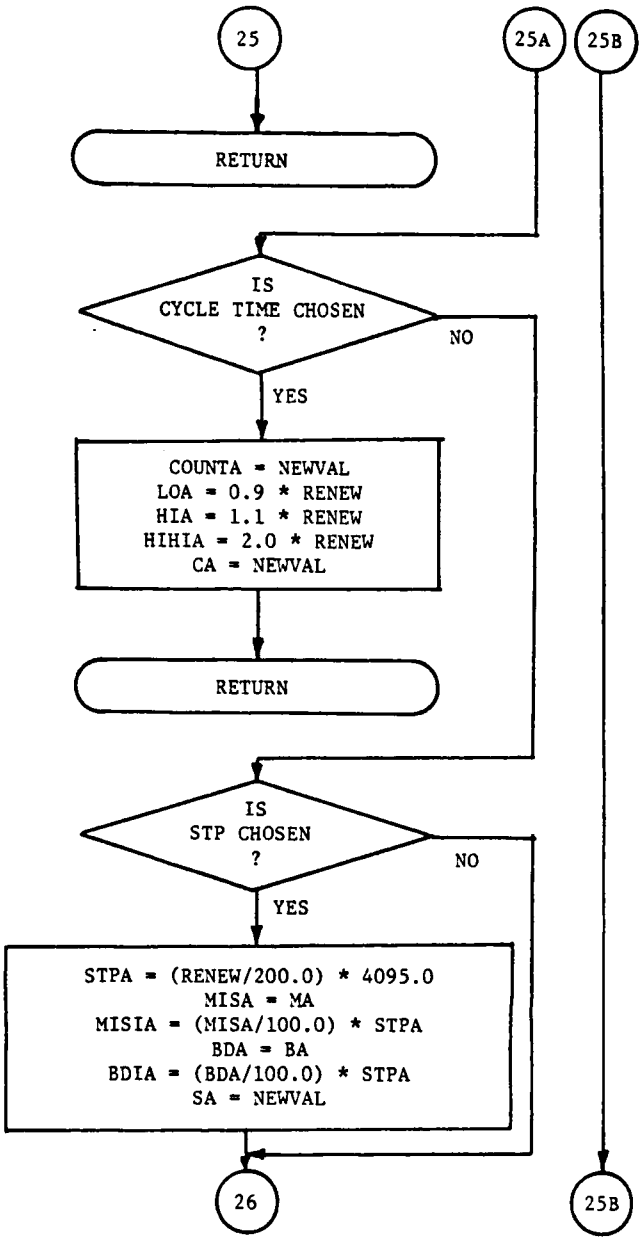


Figure B-3. (Continued)

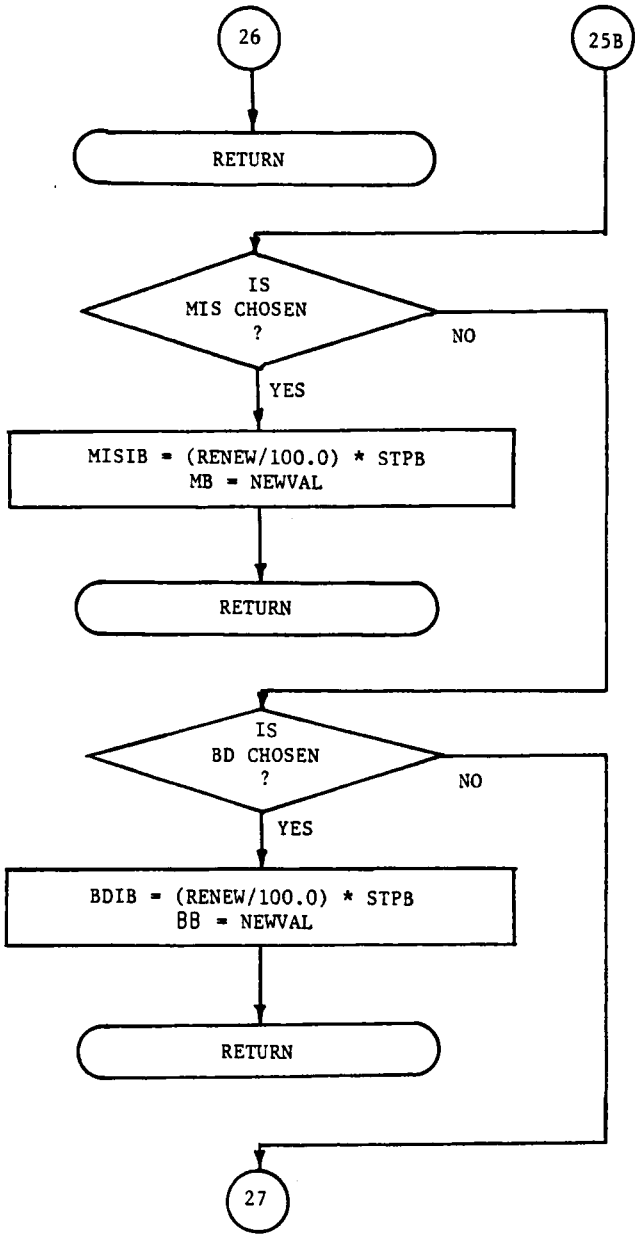


Figure B-3. (Continued)

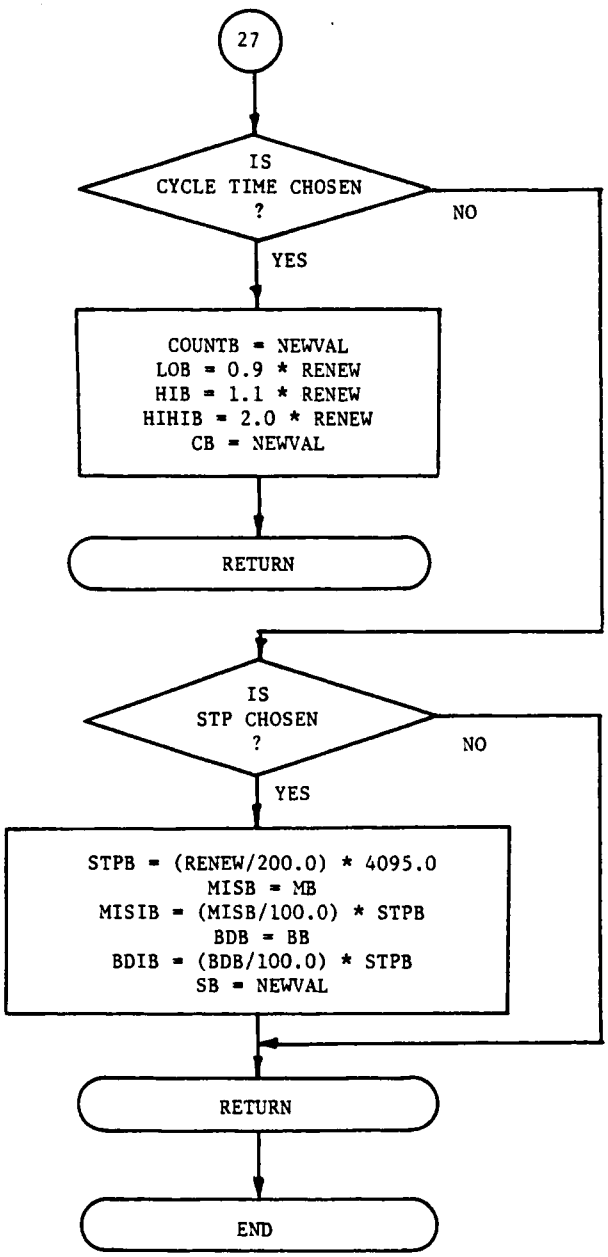


Figure B-3. (Continued)

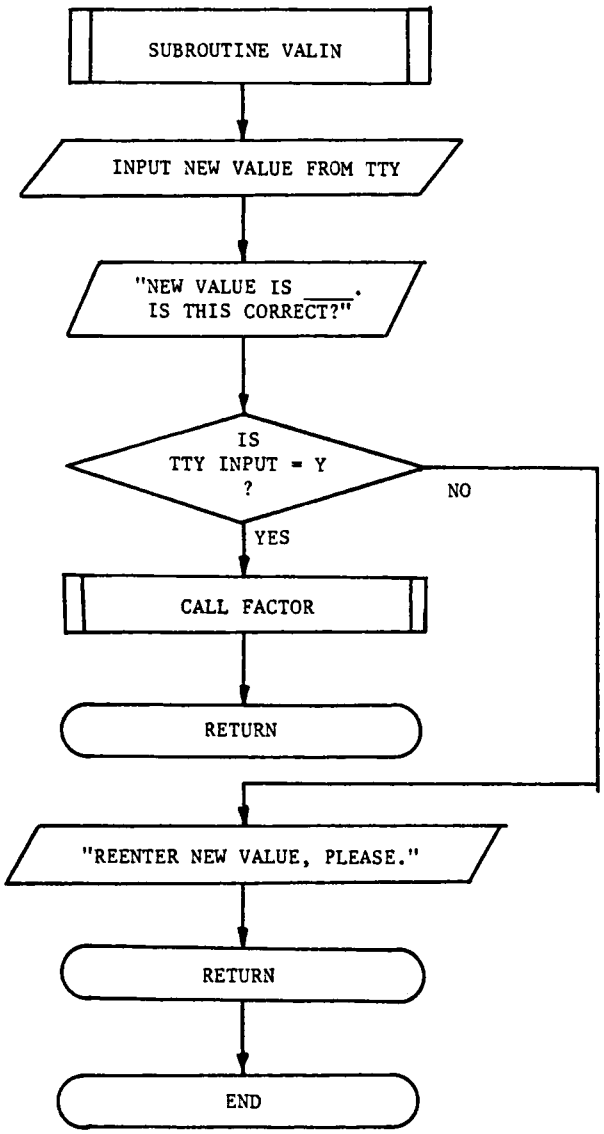


Figure B-3. (Continued)

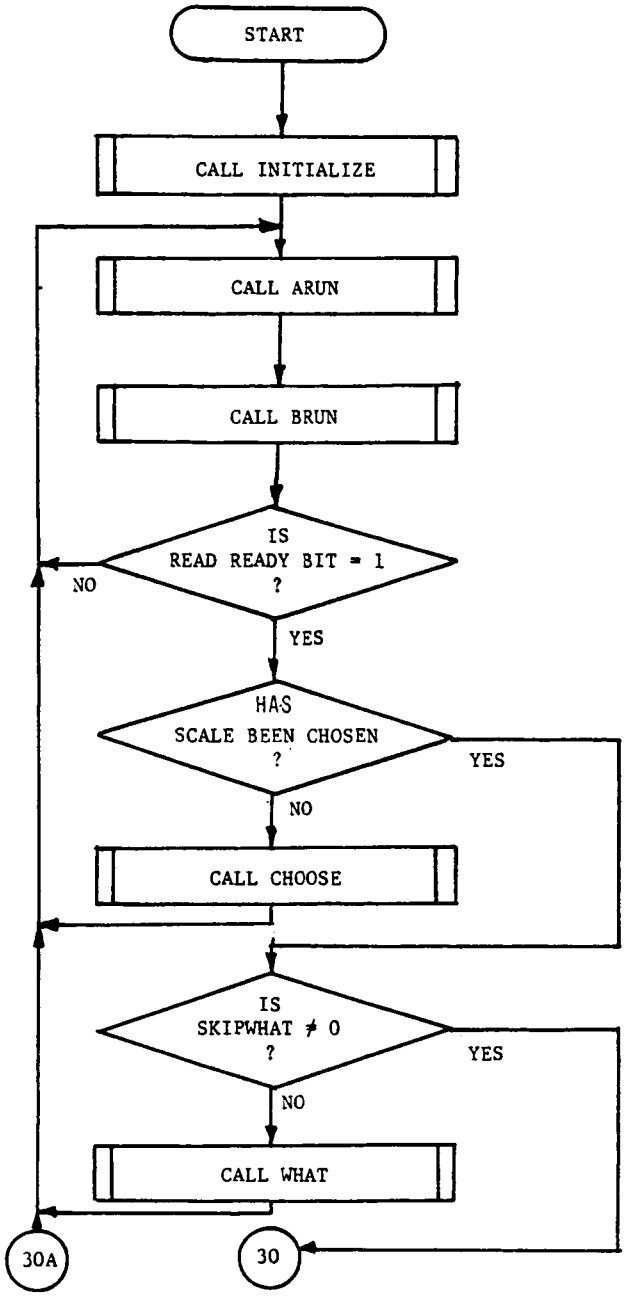


Figure B-3. (Continued)

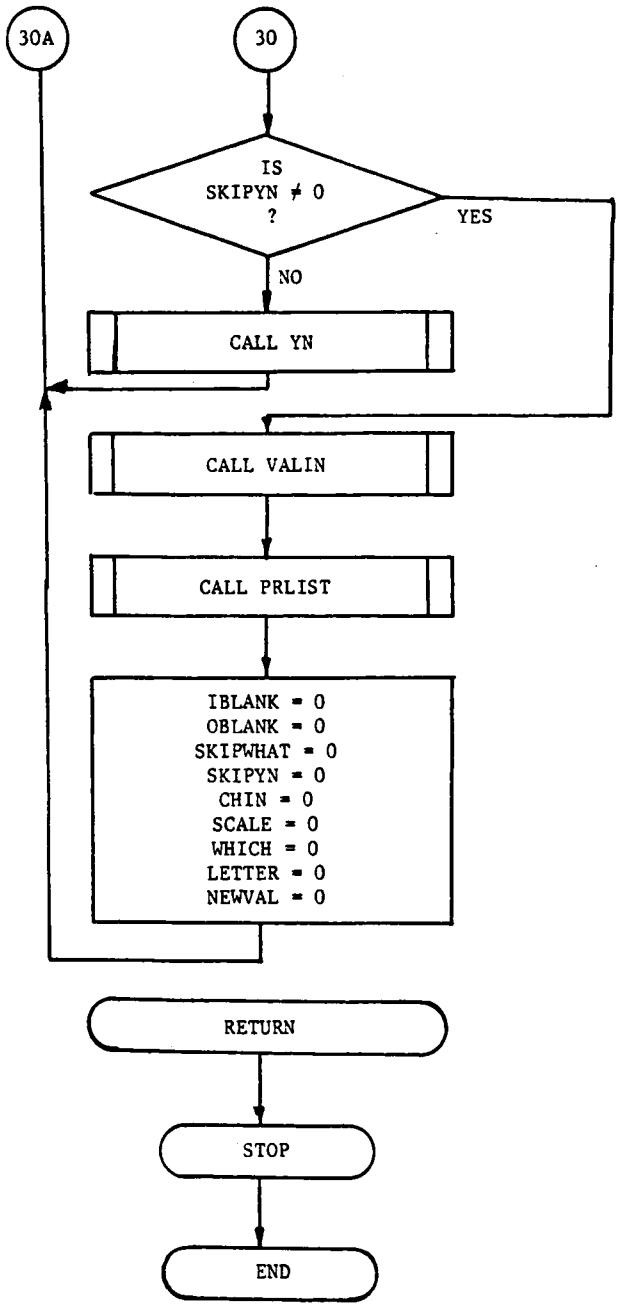


Figure B-3. (Continued)

APPENDIX C

```
ASM80 :F1:BULK.80


ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0         MODULE   PAGE    1

  LOC  OBJ          LINE        SOURCE STATEMENT

                      1 ;ORIGINAL INSTALLATION
                      2 ;BULK.80  WEIGHT CONTROL ON SCALE SYSTEM
                      3 ;AUTHOR: EVERETTE ELDRIDGE
                      4 ;DATE: 8/6/79
                      5 ;DESCRIPTION:  THIS PROGRAM TAKES ANALOG INPUTS FROM EXISTING
                      6 ;SCALE LOGIC USING AN ADAC ANALOG TO DIGITAL INTERFACE
                      7 ;AND CALCULATES THE CYCLE TIME NEEDED FOR THE WEIGHT RECEIVED.
                      8 ;THE SCALE IS RESTARTED AT THE END OF THIS CYCLE.  LOW WEIGHT,
                      9 ;HIGH WEIGHT, AND MALFUNCTION ALARMS ARE FURNISHED.
                     10 ;LOW AND HIGH TIME CALCULATIONS ARE PREVENTED BY AUTOMATIC
                     11 ;RESTART ROUTINES.
                     12 ;TWO SCALES ARE OPERATED BY THIS PROGRAM.
                     13 ;REVISIONS TO PROGRAM:
                     14 ;8-5-81   REVISED OUTPUTS TO AGREE WITH FIELD INSTALLATION
                     15 ;8-10-81  CHANGED ANALOG INPUTS FOR SCALE B AND TIMING
                     16 ;         LOOP COUNT TO 5000.
                     17 ;8-12-81  ADDED MINIMUM CYCLE TIME TO CALC ROUTINES
                     18 ;PARALLEL (I/O)  E4,E5(INPUTS)AND E6(OUTPUTS)
                     19 ;E4-BIT 0=DISCHARGE SCALE A
                     20 ;   BIT 1=DISCHARGE SCALE B
                     21 ;   BIT 2=RESTART SCALE A
                     22 ;   BIT 3=RESTART SCALE B
                     23 ;BIT 4 TO 7 NOT USED
                     24 ;E5-NONE USED
                     25 ;E6-BIT 0=SET, SCALE A ON
                     26 ;       1=SET, SCALE B ON
                     27 ;       2=NOT SET, SCALE A HIGH ALARM
                     28 ;       3=NOT SET, SCALE B HIGH ALARM
                     29 ;       4=NOT SET, SCALE A LOW ALARM
                     30 ;       5=NOT SET, SCALE A MALFUNCTION ALARM
                     31 ;       6=NOT SET, SCALE B LOW ALARM
                     32 ;       7=NOT SET, SCALE B MALFUNCTION ALARM
                     33 ;ANALOG INPUTS
                     34 ;CHANNEL 00H= SETPOINT FOR SCALE A
                     35 ;        01H= WEIGHT FOR SCALE A
                     36 ;        06H= SETPOINT FOR SCALE B
                     37 ;        07H= WEIGHT FOR SCALE B
  00E4               38 NAMIN EQU 0E4H
  00E6               39 NAMOUT EQU 0E6H
  F718               40 ADCM EQU 0F718H
  F719               41 ADCC EQU 0F719H
  F71A               42 ADCD EQU 0F71AH
  0000               43         ORG 0H
  0000 2119F7        44         LXI H,ADCC       ;LD A/D CONTROL REG MEM LOC INTO H&L REG
  0003 3618          45         MVI M,018H       ;MOVE CONTROL WORD INTO ABOVE LOCATION
  0005 31FD3F        46         LXI SP,3FFDH     ;INITIALIZE STACK POINTER TO MEM LOC 03FFDH
  0008 3E92          47         MVI A,092H       ;SET UP INPUT AND OUTPUT PORT A19
  000A D3E7          48         OUT 0E7H
                     49 ;INITIALIZING FUNCTIONS
  000C 11C201        50 ADFAC:  LXI D,01C2H      ;LD ADJUSTING FACTOR INTO D&E REG (450)
  000F CDBE03        51         CALL FLOT
  0012 210C3C        52         LXI H,ADJFAC     ;LD ADJUSTING FACTOR MEM. LOC. INTO H&L REG
  0015 71            53         MOV M,C          ;MOVE FLOATING POINT VALUE INTO MEM LOC.
  0016 23            54         INX  H
```

Figure C-1. Program Listing of Original Microcomputer Installation

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    2

  LOC  OBJ          LINE        SOURCE STATEMENT

  0017 73             55          MOV M,E          ;INVERT DE REG IN ORDER TO USE LHLD OCOMM.
  0018 23             56          INX H
  0019 72             57          MOV M,D
  001A 21213C         58          LXI H,FLAGS      ;INITIALIZE FLAGS FOR SKIP DISHARGE CHECK
  001D 3600           59          MVI M,00H
  001F 23             60          INX H
  0020 3600           61          MVI M,00H
  0022 23             62          INX H            ;SET TO FLAG TO SKIP CYCLE TIME CHECK
  0023 3601           63          MVI M,01H
  0025 23             64          INX H
  0026 3601           65          MVI M,01H
  0028 C32B00         66          JMP STR56        ;JUMP TO STATEMENT TO START SCALE TIMERS
  002B 3EFF           67 STR56:   MVI A,0FFH
  002D D3E6           68          OUT NAMOUT       ;OUT PORT 1 TO ENERGIZE TIMERS
  002F 21253C         69          LXI H,OUTPUT
  0032 77             70          MOV M,A
  0033 C33600         71          JMP INCTO        ;JUMP TO INITIALIZE COUNT SEQUENCE
  0036 210000         72 INCTO:   LXI H,0000H      ;INITIALIZE SYSTEM COUNT TO 0
  0039 22153C         73          SHLD COUNT       ;STORE COUNT IN MEM LOC
  003C 22173C         74          SHLD TIME5       ;INITIALIZE SCALE A TIME
  003F 22193C         75          SHLD TIME6       ;INITIALIZE SCALE B TIME
  0042 2A153C         76 STCNT:   LHLD COUNT       ;LD HL REG WITH LAST COUNT
  0045 23             77          INX H            ;INCREMENT COUNT
  0046 3EEC           78          MVI A,0ECH       ;CHECK IF COUNT EQUALS EC00
  0048 BC             79          CMP H
  0049 C26600         80          JNZ STOCT        ;JUMP OUT OF LOOP IF NOT EQUAL
  004C 06C4           81          MVI B,0C4H       ;SUBTRACT C400 FROM COUNT
  004E 7C             82          MOV A,H
  004F 90             83          SUB B
  0050 67             84          MOV H,A
  0051 22153C         85          SHLD COUNT       ;STORE NEW COUNT
  0054 21183C         86          LXI H,TIME5+01H  ;SUBTRACT  C400 FROM TIME5
  0057 7E             87          MOV A,M
  0058 90             88          SUB B
  0059 77             89          MOV M,A
  005A 211A3C         90          LXI H,TIME6+01H
  005D 7E             91          MOV A,M
  005E 90             92          SUB B
  005F 77             93          MOV M,A
  0060 2A153C         94          LHLD COUNT
  0063 C36900         95          JMP STNAT
  0066 22153C         96 STOCT:   SHLD COUNT
  0069 EB             97 STNAT:   XCHG             ;MOVE COUNT TO DE REG
  006A DBE4           98          IN NAMIN         ;CHECK SCALE A FOR RESTART SIGNAL
  006C E604           99          ANI 04H
  006E CA8D00        100          JZ RSTS6         ;IF RESTART SIGNAL NOT PRESENT CHECK B
  0071 DBE4          101 STNAT2:  IN NAMIN
  0073 E604          102          ANI 04H
  0075 C27100        103          JNZ STNAT2
  0078 CDE902        104          CALL DELAY
  007B 2A153C        105          LHLD COUNT       ;LD HL REG WITH PRESENT COUNT
  007E 22173C        106          SHLD TIME5       ;CHANGE TIME A TO PRESENT COUNT
  0081 21253C        107          LXI H,OUTPUT     ;RESET ALARM BITS FOR SCALE A
  0084 7E            108          MOV A,M
  0085 F634          109          ORI 34H
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    3

 LOC  OBJ          LINE        SOURCE STATEMENT

 0087 77            110         MOV M,A
 0088 21233C        111         LXI H,FLAGA      ;RESET SKIP CYCLE TIME CHECK FLAG
 008B 3600          112         MVI M,0
 008D DBE4          113 RSTS6:  IN NAMIN         ;CHECK A
 008F E608          114         ANI 08H
 0091 CAB000        115         JZ CH5
 0094 DBE4          116 RSTS62: IN NAMIN
 0096 E608          117         ANI 08H
 0098 C29400        118         JNZ RSTS62
 009B CDE902        119         CALL DELAY
 009E 2A153C        120         LHLD COUNT
 00A1 22193C        121         SHLD TIME6       ;CHANGE A TO PRESENT COUNT
 00A4 21253C        122         LXI H,OUTPUT     ;RESET ALARM BITS FOR SCALE B
 00A7 7E            123         MOV A,M
 00A8 F6C8          124         ORI 0C8H
 00AA 77            125         MOV M,A
 00AB 21243C        126         LXI H,FLAGB      ;RESET SKIP CYCLE TIME CHECK FLAG
 00AE 3600          127         MVI M,0
 00B0 21233C        128 CH5:    LXI H,FLAGA
 00B3 7E            129         MOV A,M
 00B4 E601          130         ANI 01H
 00B6 C2E000        131         JNZ CH6
 00B9 2A173C        132         LHLD TIME5       ;LD HL REG WITH TIME A
 00BC 7A            133         MOV A,D
 00BD BC            134         CMP H
 00BE DAE000        135         JC CH6           ;IF TIME5<=COUNT, TURN ON A
 00C1 C2C900        136         JNZ CH5A
 00C4 7B            137         MOV A,E
 00C5 BD            138         CMP L
 00C6 DAE000        139         JC CH6
 00C9 21253C        140 CH5A:   LXI H,OUTPUT     ;SET BIT TO TURN ON SCALE A
 00CC 7E            141         MOV A,M
 00CD F601          142         ORI 01H
 00CF 77            143         MOV M,A
 00D0 21213C        144         LXI H,FLAG5      ;RESET SKIP DISCHARGE CHECK FLAG
 00D3 3600          145         MVI M,00H        ;TO 0 FOR SCALE A
 00D5 2A153C        146         LHLD COUNT       ;SET BEGINNING COUNT OF SCALE A CYCLE
 00D8 22173C        147         SHLD TIME5
 00DB 21233C        148         LXI H,FLAGA      ;SET SKIP COUNT CHECK FLAG FOR
 00DE 3601          149         MVI M,01H        ;SCALE A
 00E0 21243C        150 CH6:    LXI H,FLAGB
 00E3 7E            151         MOV A,M
 00E4 E601          152         ANI 01H
 00E6 C21001        153         JNZ OUTTR
 00E9 2A193C        154         LHLD TIME6       ;CHECK TIME B TO SEE IF
 00EC 7A            155         MOV A,D          ;TIMER ENERGIZE READY
 00ED BC            156         CMP H
 00EE DA1001        157         JC OUTTR
 00F1 C2F900        158         JNZ CH6A
 00F4 7B            159         MOV A,E
 00F5 BD            160         CMP L
 00F6 DA1001        161         JC OUTTR
 00F9 21223C        162 CH6A:   LXI H,FLAG6      ;RESET SKIP DISCHARGE CHECK FLAG
 00FC 3600          163         MVI M,00H
 00FE 21253C        164         LXI H,OUTPUT     ;SET BIT TO TURN ON SCALE B
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0          MODULE   PAGE    4

 LOC  OBJ          LINE         SOURCE STATEMENT

 0101 7E            165          MOV A,M
 0102 F602          166          ORI 02H
 0104 77            167          MOV M,A
 0105 2A153C        168          LHLD COUNT        ;SET BEGINNING COUNT OF SCALE B
 0108 22193C        169          SHLD TIME6        ;CYCLE
 010B 21243C        170          LXI H,FLAGB       ;SET SKIP COUNT CHECK FLAG FOR
 010E 3601          171          MVI M,01H         ;SCALE B
 0110 21253C        172 OUTTR:   LXI H,OUTPUT
 0113 7E            173          MOV A,M
 0114 D3E6          174          OUT NAMOUT        ;OUTPUT TO PORT 1 TURNS ON TIMERS
                    175                            ;IF REQUIRED FOR A AND/OR B
 0116 C31901        176          JMP DISC5
 0119 21213C        177 DISC5:   LXI H,FLAG5       ;CHECK TO SEE IF SKIP FLAG IS SET
 011C 7E            178          MOV A,M
 011D E601          179          ANI 01H
 011F C22C01        180          JNZ DISC6
 0122 DBE4          181          IN NAMIN          ;CHECK TO SEE IF A HAS DISCHARGED
 0124 E601          182          ANI 01H
 0126 CA2C01        183          JZ DISC6          ;IF NO SIGNAL, CHECK B
 0129 CD5901        184          CALL CALC5        ;CALCULATE NEW TIME IF SIGNAL PRESENT
 012C 21223C        185 DISC6:   LXI H,FLAG6       ;CHECK TO SEE IF SKIP FLAG IS SET
 012F 7E            186          MOV A,M
 0130 E601          187          ANI 01H
 0132 C23F01        188          JNZ TIMLP         ;JUMP TO BEGINNING OF COUNT
 0135 DBE4          189          IN NAMIN          ;CHECK INPUT TO SEE IF
 0137 E602          190          ANI 02H           ;B HAS DISCHARGED
 0139 CA3F01        191          JZ TIMLP
 013C CD2102        192          CALL CALC6        ;CALCULATE NEW TIME
 013F 218813        193 TIMLP:   LXI H,5000        ;TIMING LOOP LD HL WITH 5000
 0142 2B            194 HNZ:     DCX H
 0143 06FF          195          MVI B,0FFH        ;APPROX .1 SEC
 0145 78            196          MOV A,B
 0146 05            197          DCR B
 0147 05            198          DCR B
 0148 A4            199          ANA H
 0149 C24201        200          JNZ HNZ
 014C 2D            201 LNZ:     DCR L
 014D 06FF          202          MVI B,0FFH
 014F 78            203          MOV A,B
 0150 05            204          DCR B
 0151 05            205          DCR B
 0152 A5            206          ANA L
 0153 C24C01        207          JNZ LNZ
 0156 C34200        208          JMP STCNT         ;AFTER TIMING LOOP JUMP TO START COUNTING
 0159 2118F7        209 CALC5:   LXI H,ADCM        ;CALL A/D 1 TO CALCULATE WEIGHT FOR A
 015C 3601          210          MVI M,01H
 015E 2119F7        211 CHKA1:   LXI H,ADCC        ;CHECK A/D 1 TO SEE IF DONE
 0161 7E            212          MOV A,M
 0162 E680          213          ANI 080H
 0164 CA5E01        214          JZ CHKA1
 0167 2A1AF7        215          LHLD ADCD         ;LD A/D WEIGHT INTO REG HL
 016A EB            216          XCHG
 016B CDBE03        217          CALL FLOI
 016E 21123C        218          LXI H,WT5         ;LD WEIGHT MEM LOC INTO H&L REG
 0171 71            219          MOV M,C           ;MOVE FLOATING POINT VALUE TO MEM LOC
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    5

 LOC  OBJ          LINE        SOURCE STATEMENT

 0172 23            220          INX H
 0173 73            221          MOV M,E          ;INVERT DE REGS TO BE ABLE TO USE
 0174 23            222          INX H            ;LHLD COMMAND
 0175 72            223          MOV M,D
 0176 C37901        224          JMP INSP5        ;JUMP TO INPUT SETPOINT SUBROUTINE
 0179 2118F7        225 INSP5:   LXI H,ADCM       ;CALL A/D 0 TO CALCULATE SETPOINT
 017C 3600          226          MVI M,00H
 017E 2119F7        227 CHKA0:   LXI H,ADCC       ;CHECK TO SEE IF A/D 0 IS DONE
 0181 7E            228          MOV A,M
 0182 E680          229          ANI 080H
 0184 CA7E01        230          JZ CHKA0
 0187 2A1AF7        231          LHLD ADCD        ;LD A/D SETPOINT IN REG HL
 018A EB            232          XCHG
 018B CDBE03        233          CALL FLOT
 018E 210F3C        234          LXI H,STP5       ;LD SETPOINT MEM LOC INTO HL REG
 0191 71            235          MOV M,C          ;MOVE FLOATING POINT VALUE TO MEM LOC
 0192 23            236          INX H
 0193 73            237          MOV M,E          ;INVERT DE REG TO ENABLE USE OF
 0194 23            238          INX H            ;LHLD COMMAND
 0195 72            239          MOV M,D
 0196 C39901        240          JMP DFTM5        ;JUMP TO DIVIDE FOR TIME SUBROUTINE
 0199 21253C        241 DFTM5:   LXI H,OUTPUT     ;LD HL REG WITH TIMER OUTPUT MEM
 019C 7E            242          MOV A,M
 019D F634          243          ORI 34H          ;RESETS ALA ONLY
 019F E6FE          244          ANI 0FEH
 01A1 D3E6          245          OUT NAMOUT
 01A3 77            246          MOV M,A          ;AFFECT NO. 6
 01A4 21123C        247          LXI H,WT5        ;LD ADDR OF HIGH BYTE OF WEIGHT IN HL
 01A7 46            248          MOV B,M          ;MOVE MEM CONTENTS TO REG B
 01A8 2A133C        249          LHLD WT5+01H     ;MOVE LOWER TWO BYTES
 01AB CDE003        250          CALL FDIV        ;DIVIDE WEIGHT BY SETPOINT
 01AE 210C3C        251          LXI H,ADJFAC     ;LD HL OF HIGH BYTE OF ADJUSTING FACTOR
 01B1 46            252          MOV B,M          ;MOVE CONTENTS TO B REG
 01B2 2A0D3C        253          LHLD ADJFAC+01H  ;MOVE LOWER BYTES TO HL REG
 01B5 CD5804        254          CALL FMPY        ;MULTIPLIES W/S TIMES ADJUSTING FACTOR
 01B8 CD9203        255          CALL FIXX        ;CHANGE VALUE TO FIXXED NUMBER
 01BB 219001        256 WD15:    LXI H,400        ;CHECK FOR LOW ALARM
 01BE 7C            257          MOV A,H
 01BF BA            258          CMP D
 01C0 DADA01        259          JC WD16
 01C3 C2CB01        260          JNZ WD15A
 01C6 7D            261          MOV A,L
 01C7 BB            262          CMP E
 01C8 DADA01        263          JC WD16
 01CB 21253C        264 WD15A:   LXI H,OUTPUT     ;IF <=400, TURN ON LOW ALARM
 01CE 7E            265          MOV A,M
 01CF E6EF          266          ANI 0EFH
 01D1 D3E6          267          OUT NAMOUT
 01D3 77            268          MOV M,A
 01D4 119001        269          LXI D,400        ;SET CALC TIME = 40 SECONDS
 01D7 C30F02        270          JMP WD18
 01DA 21F401        271 WD16:    LXI H,500        ;CHECK FOR HIGH ALARM
 01DD 7C            272          MOV A,H
 01DE BA            273          CMP D
 01DF DAEA01        274          JC WD16A
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0          MODULE   PAGE    6

 LOC  OBJ          LINE        SOURCE STATEMENT

 01E2 C20F02        275          JNZ WD18
 01E5 7D            276          MOV A,L
 01E6 BB            277          CMP E
 01E7 D20F02        278          JNC WD18
 01EA 21253C        279 WD16A:   LXI H,OUTPUT     ;IF >=500, TURN ON HIGH ALARM
 01ED 7E            280          MOV A,M
 01EE E6FB          281          ANI 0FBH
 01F0 D3E6          282          OUT NAMOUT
 01F2 77            283          MOV M,A
 01F3 218403        284 WD17:    LXI H,900        ;CHECK FOR MALFUNCTION(HI-HI)
 01F6 7C            285          MOV A,H
 01F7 BA            286          CMP D
 01F8 DA0302        287          JC WD17A
 01FB C20F02        288          JNZ WD18
 01FE 7D            289          MOV A,L
 01FF BB            290          CMP E
 0200 D20F02        291          JNC WD18
 0203 21253C        292 WD17A:   LXI H,OUTPUT     ;IF >=900, TURN ON MALFUNCTION
                    293                           ;ALARM
 0206 7E            294          MOV A,M
 0207 E6DF          295          ANI 0DFH
 0209 D3E6          296          OUT NAMOUT
 020B 77            297          MOV M,A
 020C 11C201        298          LXI D,450        ;RESET TIME5 TO 450
 020F 2A173C        299 WD18:    LHLD TIME5       ;SET TIME5 EQUAL TO DE
 0212 19            300          DAD D
 0213 22173C        301          SHLD TIME5
 0216 21213C        302 WD18A:   LXI H,FLAG5      ;RESET SKIP FLAGS
 0219 3601          303          MVI M,01H
 021B 21233C        304          LXI H,FLAGA
 021E 3600          305          MVI M,00H
 0220 C9            306          RET
 0221 2118F7        307 CALC6:   LXI H,ADCM       ;SAME AS CALC5 EXCEPT FOR B
 0224 3607          308          MVI M,07H
 0226 2119F7        309 CHKA3:   LXI H,ADCC
 0229 7E            310          MOV A,M
 022A E680          311          ANI 080H
 022C CA2602        312          JZ CHKA3
 022F 2A1AF7        313          LHLD ADCD
 0232 EB            314          XCHG
 0233 CDBE03        315          CALL FLOT
 0236 211B3C        316          LXI H,W16
 0239 71            317          MOV M,C
 023A 23            318          INX H
 023B 73            319          MOV M,E
 023C 23            320          INX H
 023D 72            321          MOV M,D
 023E C34102        322          JMP INSP6
 0241 2118F7        323 INSP6:   LXI H,ADCM
 0244 3606          324          MVI M,06H
 0246 2119F7        325 CHKA2:   LXI H,ADCC
 0249 7E            326          MOV A,M
 024A E680          327          ANI 080H
 024C CA4602        328          JZ CHKA2
 024F 2A1AF7        329          LHLD ADCD
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    7

 LOC  OBJ          LINE        SOURCE STATEMENT

 0252 EB            330         XCHG
 0253 CDBE03        331         CALL FLOT
 0256 211E3C        332         LXI H,STP6
 0259 71            333         MOV M,C
 025A 23            334         INX H
 025B 73            335         MOV M,E
 025C 23            336         INX H
 025D 72            337         MOV M,D
 025E C36102        338         JMP DFTM6
 0261 21253C        339 DFTM6:  LXI H,OUTPUT
 0264 7E            340         MOV A,M
 0265 F6C8          341         ORI 0C8H        ;TURN OFF B ALARMS
 0267 E6FD          342         ANI 0FDH        ;TURN OFF B ONLY
 0269 D3E6          343         OUT NAMOUT
 026B 77            344         MOV M,A
 026C 211B3C        345         LXI H,WT6       ;LD ADDR OF HIGH BYTE OF WEIGHT IN HL
 026F 46            346         MOV B,M         ;MOVE MEM CONTENTS TO REG B
 0270 2A1C3C        347         LHLD WT6+01H    ;MOVE LOWER TWO BYTES
 0273 CDE003        348         CALL FDIV       ;DIVIDE WEIGHT BY SETPOINT
 0276 210C3C        349         LXI H,ADJFAC    ;LD HL OF HIGH BYTE OF ADJUSTING FACTOR
 0279 46            350         MOV B,M         ;MOVE CONTENTS TO B REG
 027A 2A0D3C        351         LHLD ADJFAC+01H ;MOVE LOWER BYTES TO HL REG
 027D CD5804        352         CALL FMPY       ;MULTIPLIES W/S TIMES ADJUSTING FACTOR
 0280 CD9203        353         CALL FIXX       ;CHANGE VALUE TO FIXXED NUMBER
 0283 219001        354 WD19:   LXI H,400       ;CHECK FOR LOW ALARM
 0286 7C            355         MOV A,H
 0287 BA            356         CMP D
 0288 DAA202        357         JC WD20
 028B C29302        358         JNZ WD19A
 028E 7D            359         MOV A,L
 028F BB            360         CMP E
 0290 DAA202        361         JC WD20
 0293 21253C        362 WD19A:  LXI H,OUTPUT     ;IF <=400, TURN ON LOW ALARM
 0296 7E            363         MOV A,M
 0297 E6BF          364         ANI 0BFH
 0299 D3E6          365         OUT NAMOUT
 029B 77            366         MOV M,A
 029C 119001        367         LXI D,400       ;SET CALC TIME = 40 SECONDS
 029F C3D702        368         JMP WD22
 02A2 21F401        369 WD20:   LXI H,500       ;CHECK FOR HIGH ALARM
 02A5 7C            370         MOV A,H
 02A6 BA            371         CMP D
 02A7 DAB202        372         JC WD20A
 02AA C2D702        373         JNZ WD22
 02AD 7D            374         MOV A,L
 02AE BB            375         CMP E
 02AF D2D702        376         JNC WD22
 02B2 21253C        377 WD20A:  LXI H,OUTPUT     ;IF >=500, TURN ON HIGH ALARM
 02B5 7E            378         MOV A,M
 02B6 E6F7          379         ANI 0F7H
 02B8 D3E6          380         OUT NAMOUT
 02BA 77            381         MOV M,A
 02BB 218403        382 WD21:   LXI H,900       ;CHECK FOR MALFUNCTION(HI-HI)
 02BE 7C            383         MOV A,H
 02BF BA            384         CMP D
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    8

  LOC  OBJ          LINE        SOURCE STATEMENT

  02C0 DAC802        385          JC WD21A
  02C3 C2D702        386          JNZ WD22
  02C6 7D            387          MOV A,L
  02C7 BB            388          CMP E
  02C8 D2D702        389          JNC WD22
  02CB 21253C        390 WD21A:   LXI H,OUTPUT     ;IF >=900, TURN ON MALFUNCTION
                     391                           ;ALARM
  02CE 7E            392          MOV A,M
  02CF E67F          393          ANI 07FH
  02D1 D3E6          394          OUT NAMOUT
  02D3 77            395          MOV M,A
  02D4 11C201        396          LXI D,450        ;RESET TIME5 TO 450
  02D7 2A193C        397 WD22:    LHLD TIME6       ;SET TIME5 EQUAL TO DE
  02DA 19            398          DAD D
  02DB 22193C        399          SHLD TIME6
  02DE 21223C        400 WD22A:   LXI H,FLAG6      ;RESET SKIP FLAGS
  02E1 3601          401          MVI M,01H
  02E3 21243C        402          LXI H,FLAG8
  02E6 3600          403          MVI M,00H
  02E8 C9            404          RET
                     405 ;
                     406 ;
                     407 ;
  02E9 06FF          408 DELAY:   MVI B,0FFH
  02EB 05            409 DEL2:    DCR     B
  02EC 00            410          NOP
  02ED C2EB02        411          JNZ     DEL2
  02F0 C9            412          RET
                     413 ;
                     414 ;
                     415 ;
                     416 ;FLOATING POINT ROUTINE FSUB: THIS ROUTINE SETS UP THE PARAMETERS TO PASS
                     417 ;TO FADD WHICH RESULTS IN SUBTRACTION OF A NUMBER IN C-D-E FROM A
                     418 ;NUMBER IN B-H-L.  IT MUST BE LOADED INTO MEMORY IMMEDIATELY PRECEEDING
                     419 ;FADD.
  02F1 CDD603        420 FSUB:    CALL FNEG              ;NEGATE C-D-E
                     421 ;FLOATING POINT ROUTINE FADD: THIS ROUTINE ADDS A FLOATING POINT
                     422 ;NUMBER IN C-D-E TO A FLOATING POINT NUMBER IN B-H-L AND RETURNS WITH
                     423 ;THE RESULT IN C-D-E. IT SHOULD BE LOADED INTO MEMORY IMMEDIATELY
                     424 ;FOLLOWING FSUB.
  02F4 CDF005        425 FADD:    CALL ZRO         ;CHECK FOR ZERO C-D-E
  02F7 87            426          ADD A
  02F8 EB            427 FA00:    XCHG
  02F9 79            428          MOV A,C          ;EXCHANGE C AND B
  02FA 48            429          MOV C,B
  02FB 47            430          MOV B,A
  02FC C8            431          RZ               ;ZERO IMPLIES ONE NUMBER WAS ZERO,
                     432                           ;RESULT IN C-D-E
  02FD CDF005        433          CALL ZRO         ;CHECK NEXT VALUE (B-H-L)
  0300 87            434          ADD A
  0301 CAF802        435          JZ FA00          ;ZERO IMPLIES B-H-L WAS ZERO,
                     436                           ;RESULT FOR C-D-E
  0304 E5            437          PUSH H
  0305 21013C        438          LXI H,S2
  0308 79            439          MOV A,C
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE    9

  LOC  OBJ          LINE        SOURCE STATEMENT

  0309 77            440         MOV M,A         ;RESERVE EXPONENT
  030A E67F          441         ANI 7FH         ;MASK SIGN BIT
  030C 4F            442         MOV C,A
  030D 2B            443         DCX H
  030E 78            444         MOV A,B
  030F 77            445         MOV M,A
  0310 E67F          446         ANI 7FH
  0312 47            447         MOV B,A
  0313 B9            448         CMP C           ;MATCH EXPONENTS IF NOT EQUAL
  0314 CA4303        449         JZ FA03         ;IF Z EQU 1, EXPONENTS EQUAL, CONTINUE
  0317 DA2703        450         JC FA01         ;CARRY  EQU  1 IMPLIES C>B
  031A 41            451         MOV B,C         ;C<B, SWITCH TO MAKE B<C
  031B 4F            452         MOV C,A
  031C C5            453         PUSH B
  031D 46            454         MOV B,M
  031E 23            455         INX H
  031F 4E            456         MOV C,M
  0320 70            457         MOV M,B
  0321 2B            458         DCX H
  0322 71            459         MOV M,C
  0323 C1            460         POP B
  0324 E1            461         POP H
  0325 EB            462         XCHG            ;SWITCH MANTISSAE
  0326 E5            463         PUSH H
  0327 E1            464 FA01:   POP H
  0328 78            465         MOV A,B
  0329 3C            466 FA02:   INR A           ;INCREASE EXPONENT BY 1, NUMBER BY 2X
  032A 47            467         MOV B,A
  032B 37            468         STC             ;ZERO CARRY
  032C 3F            469         CMC
  032D 7C            470         MOV A,H
  032E 1F            471         RAR             ;DIVIDE HIGH MANTISSA BY 2
  032F 67            472         MOV H,A
  0330 7D            473         MOV A,L
  0331 1F            474         RAR             ;DIVIDE LOW MANTISSA BY 2
  0332 6F            475         MOV L,A
  0333 AF            476         XRA A           ;CLEAR A TO SET UP FOR ROUNDOFF
  0334 17            477         RAL             ;DROPPED BIT FROM CARRY INTO LSB OF A
  0335 32023C        478         STA S3
  0338 78            479         MOV A,B
  0339 B9            480         CMP C           ;EXPONENTS MATCHED?
  033A C22903        481         JNZ FA02        ;IF Z EQU 1, NO RETURN AND INCREMENT
  033D 3A023C        482         LDA S3
  0340 85            483         ADD L           ;ROUNDOFF
  0341 6F            484         MOV L,A
  0342 E5            485         PUSH H
  0343 E1            486 FA03:   POP H           ;EXPONENTS NOW MATCHED, CHECK SIGN OF
                     487                         ;NUMBER
  0344 3A013C        488         LDA S2
  0347 E680          489         ANI 80H         ;MASK ALL BUT SIGN BIT
  0349 4F            490         MOV C,A         ;STORE SIGN
  034A 3A003C        491         LDA S1
  034D 81            492         ADD C
  034E F27E03        493         JP FA10         ;IF THE ANSWER IS +, SIGN SAME, CONTINUE
  0351 7A            494 FA04:   MOV A,D         ;SIGN DIFFERENT, WHICH IS LARGER?
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0          MODULE   PAGE   10

 LOC  OBJ          LINE        SOURCE STATEMENT

 0352 BC            495         CMP H
 0353 CA7303        496         JZ FA09          ;EQUAL, COMPARE E AND L
 0356 DA5E03        497         JC FA06          ;C EQU 1 IMPLIES H>D, COMPLIMENT C-D-E AND
                    498                          ;ITS SIGN
 0359 EB            499 FA05:   XCHG             ;D>H, SWITCH NUMBERS
 035A 79            500         MOV A,C
 035B C36103        501         JMP FA07
 035E 79            502 FA06:   MOV A,C
 035F C680          503         ADI 80H
 0361 80            504 FA07:   ADD B            ;SET TOTAL EXPONENT
 0362 4F            505         MOV C,A          ;EXPONENT IN C
 0363 37            506         STC
 0364 3F            507         CMC
 0365 7B            508         MOV A,E          ;COMPLIMENT SMALLER (NOW IN CDE)
 0366 2F            509         CMA
 0367 5F            510         MOV E,A
 0368 7A            511         MOV A,D
 0369 2F            512         CMA
 036A 57            513         MOV D,A
 036B 13            514         INX D
 036C 19            515         DAD D            ;RESULT IN C-H-L, IGNORE OVERFLOW
 036D EB            516         XCHG             ;RESULT IN C-D-E
 036E 0610          517 FA08:   MVI B,10H
 0370 C3C003        518         JMP FL1          ;GO TO LEFT JUSTIFY ROUTINE IN FLOT
                    519                          ;ROUTINE
 0373 7B            520 FA09:   MOV A,E          ;D,H SAME, COMPARE E AND L
 0374 BD            521         CMP L
 0375 CA8C03        522         JZ FA11          ;EQUAL MAGNITUDE, OPPOSITE SIGN,
                    523                          ;RESULT ZERO
 0378 DA5E03        524         JC FA06
 037B C35903        525         JMP FA05
 037E 79            526 FA10:   MOV A,C          ;SIGN SAME, SET EXPONENT
 037F 80            527         ADD B
 0380 4F            528         MOV C,A
 0381 19            529         DAD D            ;ADD MANTISSAE
 0382 EB            530         XCHG
 0383 D0            531         RNC              ;CARRY NOT SET, RETURN
 0384 0C            532         INR C            ;CARRY SET, INCREMENT EXPONENT
 0385 7A            533         MOV A,D          ;LEFT JUSTIFY
 0386 1F            534         RAR
 0387 57            535         MOV D,A
 0388 7B            536         MOV A,E
 0389 1F            537         RAR
 038A 5F            538         MOV E,A
 038B C9            539         RET
 038C 0E40          540 FA11:   MVI C,40H        ;ANSWER ZERO, RETURN
 038E 110000        541         LXI D,0000H      ;SET ZERO ANSWER
 0391 C9            542         RET
                    543 ;FLOATING POINT ROUTINE FIXX: THIS ROUTINE CONVERTS A FLOATING POINT
                    544 ;NUMBER IN C-D-E TO A 16 BIT UNSIGNED FIXED POINT NUMBER IN D-E.
                    545 ;FRACTIONS ARE TRUNCATED. UPON RETURN, THE STATE OF THE A REGISTER,
                    546 ;WHEN DOUBLED, INDICATES THE CONDITION OF THE FIX.  A PROPER TEST
                    547 ;ROUTINE WOULD CONSIST
                    548 ;
                    549 ;       CALL FIXX        ;CALL FIX ROUTINE
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   11

 LOC  OBJ          LINE        SOURCE STATEMENT

                    550 ;       ADD A           ;DOUBLE A
                    551 ;       JZ OK           ;IF RESULT ZERO, FIX PERFORMED OK
                    552 ;       JM BN           ;IF NEGATIVE, IT TRIED TO FIX A NEGATIVE
                    553                         ;NUMBER
                    554 ;       JP BP           ;IF PLUS, IT TRIED TO FIX A NUMBER > OR  EQU
                    555                         ;2(16)
                    556 ;                       ;CANNOT BE HANDLED IN 16 BITS)
0392 AF             557 FIXX:   XRA A
0393 67             558         MOV H,A
0394 6F             559         MOV L,A
0395 81             560         ADD C
0396 FA9303         561         JM FX2          ;IF MINUS, FAULT TO RESULT
0399 C640           562         ADI 40H
039B F29B03         563         JP FX3          ;IF +, THE NUMBER IS A FRACTION, RESULT
                    564                         ;ZERO
039E E63F           565         ANI 3FH         ;MASK THE 2 MSBS OF THE EXPONENT
03A0 FE11           566         CPI 11H
03A2 D2BB03         567         JNC FX4         ;OVERLOADED IF >16
03A5 4F             568         MOV C,A
03A6 CDE005         569 FX1:    CALL DROT       ;ROTATE D-E
03A9 CDE905         570         CALL HROT       ;ROTATE H-L FROM D-E
03AC 0D             571         DCR C
03AD C2A603         572         JNZ FX1
03B0 AF             573         XRA A           ;CLEAR A IMPLIES RESULT SOUND
03B1 EB             574         XCHG            ;RESULT IN D-E
03B2 C9             575         RET
03B3 3E40           576 FX2:    MVI A,40H       ;WHEN DOUBLED, RESULT MINUS, IMPLIES
                    577                         ;MINUS FIX ATT
03B5 C9             578         RET
03B6 110000         579 FX3:    LXI D,0000H     ;RESULT ZERO
03B9 7A             580         MOV A,D
03BA C9             581         RET
03BB 3E01           582 FX4:    MVI A,01H       ;OVERFLOW, + A WHEN DOUBLED
03BD C9             583         RET
                    584 ;FLOATING POINT ROUTINE FLOT: THIS ROUTINE CONVERTS AN UNSIGNED, 16 BIT
                    585 ;NUMBER IN D-E TO A FLOATING POINT NUMBER AND RETURNS WITH THE RESULT
                    586 ;OF THE FLOAT IN C(EXP)-D-E.  THE UNSIGNED NUMBER, UPON FLOATING, IS
                    587 ;ASSIGNED A POSITIVE VALUE.  IF A NEGATIVE VALUE IS DESIRED, THE
                    588 ;NEGATE ROUTINE FNEG SHOULD BE CALLED UPON RETTRN.
03BE 0E50           589 FLOT:   MVI C,50H       ;C EQU EXPONENT OF +16 .
03C0 CDF405         590 FL1:    CALL ZRDE       ;SEE IF D EQU E=0
03C3 87             591         ADD A
03C4 CAD303         592         JZ FL3
03C7 AF             593 FL2:    XRA A
03C8 82             594         ADD D
03C9 3E00           595         MVI A,00H       ;ZERO A TO INDICATE A PROPER RETURN
03CB F8             596         RM
03CC CDE005         597         CALL DROT       ;ROTATE D-E
03CF 0D             598         DCR C           ;DECREASE EXPONENT BY 1
03D0 C3C703         599         JMP FL2         ;SEE IF LEFT JUSTIFIED YET
03D3 0E40           600 FL3:    MVI C,40H       ;RESULT ZERO, SET PROPER EXPONENT
03D5 C9             601         RET
                    602 ;FLOATING POINT ROUTINE FNEG: THIS ROUTINE NEGATES A FLOATING POINT
                    603 ;NUMBER WHOSE EXPONENT IS IN C. IF C-D-E HAS A VALUE OF ZERO,
                    604 ;NO OPERATION IS PERFORMED.
```

Figure C-1. (Continued)

```
SIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   12

 LOC  OBJ          LINE        SOURCE STATEMENT

 03D6 CDF005        605 FNEG:   CALL ZRO          ;CHECK FOR ZERO VALUE
 03D9 87            606         ADD A
 03DA C8            607         RZ                ;IF C-D-E IS ZERO, NO NEGATION PERFORMED
 03DB 3E80          608 FN1:    MVI A,80H
 03DD 81            609         ADD C             ;COMPLIMENT SIGN BIT
 03DE 4F            610         MOV C,A
 03DF C9            611         RET
                    612 ;FLOATING POINT ROUTINE FDIV: THIS ROUTINE DIVIDES A FLOATING POINT
                    613 ;NUMBER IN B-H-L BY A FLOATING POINT NUMBER IN C-D-E BY THE NONRESTORING
                    614 ;METHOD AND RETURNS WITH THE RESULT IN C-D-E. IF A DIVISION BY ZERO IS
                    615 ;ATTEMPTED, THE ROUTINE RETURNS WITH A '1' IN THE A REGISTER.
                    616 ;OTHERWISE, THE RETURN IS MADE WITH A ZERO A REGISTER.
 03E0 CDF005        617 FDIV:   CALL ZRO          ;CHECK FOR DIVISION BY ZERO
 03E3 87            618         ADD A
 03E4 3E01          619         MVI A,01H
 03E6 C8            620         RZ
 03E7 E5            621 FD00:   PUSH H
 03E8 21033C        622         LXI H,S4
 03EB AF            623         XRA A             ;ZERO S4 AND S5
 03EC 77            624         MOV M,A
 03ED 23            625         INX H
 03EE 77            626         MOV M,A
 03EF 23            627         INX H
 03F0 3611          628         MVI M,11H         ;17 IN ME
 03F2 23            629         INX H
 03F3 3601          630         MVI M,01H         ;FIRST POINTER IN MH
 03F5 23            631         INX H
 03F6 78            632         MOV A,B
 03F7 91            633         SUB C             ;DIVIDE EXPONENTS
 03F8 C641          634         ADI 41H           ;SET OFFSET
 03FA E67F          635         ANI 7FH           ;MASK SIGN BIT
 03FC 77            636         MOV M,A           ;STORE IN ML
 03FD 79            637         MOV A,C
 03FE E680          638         ANI 80H           ;MASK THE REST
 0400 4F            639         MOV C,A
 0401 78            640         MOV A,B
 0402 E680          641         ANI 80H           ;SET UP SIGN
 0404 81            642         ADD C
 0405 86            643         ADD M
 0406 77            644         MOV M,A
 0407 AF            645         XRA A
 0408 47            646         MOV B,A
 0409 4F            647         MOV C,A
 040A E1            648         POP H
 040B B9            649 FD01:   CMP C
 040C 37            650         STC
 040D C21104        651         JNZ FD02          ;NOT ZERO IMPLIES SIGNS DIFFERENT,
                    652                           ;Q(N) EQU 0
 0410 3F            653         CMC
 0411 3F            654 FD02:   CMC               ;CARRY CONTAINS THE CURRENT QUOTIENT BIT
 0412 E5            655         PUSH H
 0413 21033C        656         LXI H,S4
 0416 7E            657         MOV A,M
 0417 17            658         RAL
 0418 77            659         MOV M,A
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   13

 LOC  OBJ          LINE        SOURCE STATEMENT

 0419 23            660          INX H
 041A 7E            661          MOV A,M
 041B 17            662          RAL
 041C 77            663          MOV M,A
 041D 23            664          INX H
 041E 35            665          DCR M
 041F CA4804        666          JZ FD06          ;WHEN ZERO, DONE, SIGN BIT IN CARRY
 0422 23            667          INX H
 0423 35            668          DCR M
 0424 E1            669          POP H
 0425 CA2B04        670          JZ FD03          ;ZERO IMPLIES FIRST PASS
 0428 29            671          DAD H            ;DOUBLE DIVIDENT OR REMAINDER
 0429 17            672          RAL
 042A 47            673          MOV B,A
 042B 3A033C        674 FD03:    LDA S4
 042E E601          675          ANI 01H          ;MASK ALL BUT MORE RECENT Q BIT
 0430 87            676          ADD A
 0431 CA3F04        677          JZ FD04          ;Q(N) EQU 0 IMPLIES GO ADD 2R AND Y
 0434 7D            678          MOV A,L          ;Q(N) EQU 1 IMPLIES SUBTRACT Y FROM 2R
 0435 93            679          SUB E
 0436 6F            680          MOV L,A
 0437 7C            681          MOV A,H
 0438 9A            682          SBB D
 0439 67            683          MOV H,A
 043A 78            684          MOV A,B
 043B 99            685          SBB C
 043C C34204        686          JMP FD05
 043F 19            687 FD04:    DAD D            ;RESULT IN H-L
 0440 78            688          MOV A,B
 0441 89            689          ADC C
 0442 E601          690 FD05:    ANI 01H          ;MASK ALL BUT SIGN
 0444 47            691          MOV B,A
 0445 C30B04        692          JMP FD01
 0448 E1            693 FD06:    POP H            ;CLEAR THE STACK
 0449 21073C        694          LXI H,ML
 044C 4E            695          MOV C,M
 044D 2B            696          DCX H
 044E 2B            697          DCX H
 044F 2B            698          DCX H
 0450 56            699          MOV D,M          ;QUOTIENT LSB IN E
 0451 2B            700          DCX H
 0452 5E            701          MOV E,M          ;QUOTIENT MSB IND
 0453 0610          702          MVI B,10H        ;BEGIN TO LEFT JUSTIFY
 0455 C3C003        703          JMP FL1          ;LEFT JUSTIFY IN FLOT ROUTINE
                    704 ;FLOATING POINT ROUTINE FMPY: THIS ROUTINE MULTIPLIES A FLOATING
                    705 ;POINT NUMBER IN B-H-L BY A FLOATING POINT NUMBER IN C-D-E
                    706 ;AND RETURNS WITH THE RESULT IN C-D-E. IT DOES NOT CHECK FOR
                    707 ;EXPONENTIAL OVERFLOW.
 0458 78            708 FMPY:    MOV A,B          ;STORE ONE MULTIPLIER
 0459 32053C        709          STA ME
 045C 7C            710          MOV A,H
 045D 32063C        711          STA MH
 0460 7D            712          MOV A,L
 0461 32073C        713          STA ML
 0464 21003C        714          LXI H,S1         ;ADDRESS OF S1
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0         MODULE   PAGE   14

 LOC  OBJ          LINE        SOURCE STATEMENT

 0467 AF            715         XRA A
 0468 77            716         MOV M,A
 0469 77            717         MOV M,A          ;ZERO S1-S5
 046A 23            718         INX H
 046B 77            719         MOV M,A          ;S1-S4 TEMPORARY SUM STORAGE
 046C 23            720         INX H
 046D 77            721         MOV M,A
 046E 23            722         INX H
 046F 77            723         MOV M,A
 0470 23            724         INX H
 0471 77            725         MOV M,A          ;S5 IS FINISH POINTER
 0472 23            726         INX H
 0473 7E            727         MOV A,M          ;BRING IN EXPONENT OF MULTIPLIER
 0474 E680          728         ANI 80H          ;MASK ALL BUT SIGN BIT
 0476 47            729         MOV B,A
 0477 7E            730         MOV A,M
 0478 81            731         ADD C            ;MULTIPLY EXPONENTS, SET SIGN
 0479 C640          732         ADI 40H          ;SET OFFSET
 047B E67F          733         ANI 7FH          ;MASK SIGN
 047D 77            734         MOV M,A
 047E 79            735         MOV A,C
 047F E680          736         ANI 80H          ;MASK ALL BUT SIGN BIT
 0481 80            737         ADD B            ;SET SIGN IN EXPONENT
 0482 86            738         ADD M
 0483 77            739         MOV M,A          ;RESULT IN ME
 0484 23            740         INX H
 0485 23            741         INX H
 0486 46            742         MOV B,M          ;LSB MULTIPLIER IN B
 0487 0E08          743         MVI C,08H        ;DECREMENTER  EQU  8
 0489 210000        744         LXI H,0000H      ;ZERO H-L
 048C 37            745         STC
 048D 3F            746         CMC
 048E C39704        747         JMP FM2
 0491 CDE005        748 FM1:    CALL DROT        ;ROTATE D-E PAIR
 0494 CDE905        749         CALL HROT        ;ROTATE H-L FROM D-E
 0497 78            750 FM2:    MOV A,B          ;ADDEND NOW IN H-L-D-E
 0498 1F            751         RAR              ;CURRENT MULTIPLIER IN CARRY
 0499 47            752         MOV B,A
 049A D2B104        753         JNC FM3          ;BIT ZERO, DONT ADD
 049D C5            754         PUSH B           ;STORE B-C
 049E 01003C        755         LXI B,S1         ;S1 ADDRESS IN B-C
 04A1 0A            756         LDAX B           ;SUM
 04A2 83            757         ADD E
 04A3 02            758         STAX B
 04A4 03            759         INX B
 04A5 0A            760         LDAX B
 04A6 8A            761         ADC D
 04A7 02            762         STAX B
 04A8 03            763         INX B
 04A9 0A            764         LDAX B
 04AA 8D            765         ADC L
 04AB 02            766         STAX B
 04AC 03            767         INX B
 04AD 0A            768         LDAX B
 04AE 8C            769         ADC H
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   15

 LOC  OBJ          LINE        SOURCE STATEMENT

 04AF 02            770         STAX B          ;ALL SUMMED IN (MSB)S4-S3-S2-S1
 04B0 C1            771         POP B           ;RESTORE B-C
 04B1 0D            772 FM3:    DCR C
 04B2 C29104        773         JNZ FM1         ;FINISH MULTIPLIER
 04B5 01043C        774         LXI B,S5        ;S5 ADDRESS IN B-C
 04B8 0A            775         LDAX B
 04B9 3D            776         DCR A
 04BA CAC904        777         JZ FM4          ;Z EQU 1 IMPLIES ALL DONE, LEFT JUSTIFY
 04BD 3C            778         INR A
 04BE 3C            779         INR A
 04BF 02            780         STAX B          ;SET S5 TO INDICATE LAST PASS COMPLETE
 04C0 03            781         INX B
 04C1 03            782         INX B
 04C2 0A            783         LDAX B          ;MH TO A
 04C3 47            784         MOV B,A         ;MH TO B, MULTIPLY BY HIGH WORD
 04C4 0E08          785         MVI C,08H       ;RESTORE POINTER TO C
 04C6 C39104        786         JMP FM1         ;FINISH MULTIPLICATION, MSB MULTIPLIER
 04C9 21053C        787 FM4:    LXI H,ME        ;SUM NOW IN S4-S1
 04CC 7E            788         MOV A,M
 04CD 2B            789         DCX H
 04CE 2B            790         DCX H           ;S4 ADDRESS
 04CF 46            791         MOV B,M         ;S4 IN B
 04D0 2B            792         DCX H           ;S3 IN C
 04D1 4E            793         MOV C,M
 04D2 2B            794         DCX H
 04D3 56            795         MOV D,M         ;S2 IN D
 04D4 2B            796         DCX H
 04D5 5E            797         MOV E,M         ;S1 IN E
 04D6 60            798         MOV H,B
 04D7 69            799         MOV L,C
 04D8 4F            800         MOV C,A         ;EXPONENT IN C
 04D9 0610          801         MVI B,10H
 04DB AF            802 FM5:    XRA A
 04DC 84            803         ADD H
 04DD FAEE04        804         JM FM6          ;IF MINUS, LEFT JUSTIFIED, CONTINUE
 04E0 CDE005        805         CALL DROT       ;ROTATE D-E
 04E3 CDE905        806         CALL HROT       ;ROTATE H-L
 04E6 0D            807         DCR C           ;REDUCE EXPONENT
 04E7 05            808         DCR B
 04E8 C2DB04        809         JNZ FM5
 04EB C38C03        810         JMP FA11        ;RESULT ZERO, RETURN BY WAY OF ZERO
                    811                         ;ANSWER
 04EE EB            812 FM6:    XCHG            ;RESULT IN C-D-E
 04EF C9            813         RET
                    814 ;FLOATING POINT ROUTINE FBCD: THIS ROUTINE CONVERTS A
                    815 ;FLOATING POINT NUMBER IN C-D-E TO A FLOATING POINT
                    816 ;BCD NUMBER IN C-D-E.  IT RETURNS WIT THE REGISTERS ASSIGNED AS FOLLOWS:
                    817 ;
                    818 ;       REGISTER                        BIT
                    819 ;                                 7  6  5  4  3  2  1  0
                    820 ;        C                       SN SE EH EH EL EL EL EL
                    821 ;        D                       X4 X4 X4 X4 X3 X3 X3 X3
                    822 ;        E                       X2 X2 X2 X2 X1 X1 X1 X1
                    823 ;
                    824 ;WHERE SN IS THE SIGN OF THE NUMBER (0 IMPLIES PLUS), SE IS
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   16

 LOC  OBJ          LINE       SOURCE STATEMENT

                    825 ;THE SIGN OF THE EXPONENT, EH IS THE TWO BIT BCD UPPER EXPONENT DIGIT,
                    826 ;EL IS THE BCD LOWER EXPONENT DIGIT, AND X4 AND X1 IS THE BCD MANTISSA
                    827 ;AS X.XXX.
                    828 ;
 04F0 79            829 FBCD:   MOV A,C
 04F1 E680          830         ANI 80H          ;MASK ALL BUT SIGN
 04F3 F5            831         PUSH PSW
 04F4 3E03          832         MVI A,03H        ;3 IMPLIES EXPONENT OF 10(3)
 04F6 32083C        833         STA MX
 04F9 79            834         MOV A,C
 04FA E67F          835         ANI 7FH          ;MASK SIGN
 04FC 4F            836         MOV C,A
 04FD 3E4E          837 FB1:    MVI A,4EH
 04FF B9            838         CMP C
 0500 DA1505        839         JC FB2           ;C IMPLIES 4E<REG C, DIVIDE BY 10
 0503 C22705        840         JNZ FB3          ;NZ IMPLIES THE NUMBER MAY BE TOO
                    841                          ;SMALL, CHECK
 0506 3E9C          842         MVI A,9CH
 0508 BA            843         CMP D
 0509 DA1505        844         JC FB2           ;C IMPLIES 9C<REG D, DIVIDE BY 10
 050C C24905        845         JNZ FB4          ;NZ IMPLIES D<9C, NUMBER OK
 050F 3E3C          846         MVI A,3CH
 0511 BB            847         CMP E
 0512 D24905        848         JNC FB4          ;NC IMPLIES 3C> OR  EQU  REG E, NUMBER OK
 0515 063D          849 FB2:    MVI B,3DH        ;LOAD 0.1
 0517 26CC          850         MVI H,0CCH
 0519 6C            851         MOV L,H
 051A CD5804        852         CALL FMPY        ;X 0.1
 051D 3A083C        853         LDA MX
 0520 3C            854         INR A
 0521 32083C        855         STA MX           ;INCREMENT EXPONENT POINTER
 0524 C3FD04        856         JMP FB1
 0527 3E4A          857 FB3:    MVI A,4AH
 0529 B9            858         CMP C
 052A DA4905        859         JC FB4           ;C IMPLIES 4A<REG C, NUMBER IS OK
 052D C23605        860         JNZ FB3A         ;NZ IMPLIES REG C < 4AH, MULT
 0530 3EF9          861         MVI A,0F9H
 0532 BA            862         CMP D
 0533 DA4905        863         JC FB4           ;C IMPLIES F9<REG D, NUMBER IS OK
 0536 0644          864 FB3A:   MVI B,44H        ;LOAD 10
 0538 26A0          865         MVI H,0A0H
 053A 2E00          866         MVI L,00H
 053C CD5804        867         CALL FMPY        ;X 10
 053F 3A083C        868         LDA MX
 0542 3D            869         DCR A
 0543 32083C        870         STA MX           ;DECREMENT EXPONENT
 0546 C32705        871         JMP FB3
 0549 CD6F05        872 FB4:    CALL TROT        ;ROTATE D-E TO H-L
 054C AF            873         XRA A
 054D CD8005        874         CALL BCD         ;CONVERT H-L TO BCD, RETURN WITH BCD
                    875                          ;IN H-L
 0550 F1            876         POP PSW          ;RESTORE # SIGN TO A
 0551 E5            877         PUSH H           ;SAVE BCD
 0552 E5            878         PUSH H           ;SAVE BCD
 0553 4F            879         MOV C,A          ;PUT SIGN OF NUMBER IN C
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   17

 LOC  OBJ          LINE        SOURCE STATEMENT

 0554 3A083C        880         LDA MX
 0557 47            881         MOV B,A
 0558 AF            882         XRA A
 0559 80            883         ADD B
 055A F26705        884         JP FB5          #P IMPLIES EXPONENT IS +
 055D 2F            885         CMA
 055E 3C            886         INR A           #CHANGE MINUS EXPONENT TO 2S
                    887                         #COMPLIMENT
 055F 67            888         MOV H,A
 0560 3E40          889         MVI A,40H
 0562 81            890         ADD C
 0563 4F            891         MOV C,A         #SET SIGN OF EXPONENT  EQU  -
 0564 C36805        892         JMP FB6
 0567 67            893 FB5:    MOV H,A
 0568 3E07          894 FB6:    MVI A,07H       #SIGN OF # AND EXPONENT IN C, EXPONENT
                    895                         #IN H
 056A CD8005        896         CALL BCD        #RETURN WITH EXPONENT BCD AND ALL
                    897                         #SIGNS IN C
 056D D1            898         POP D           #RESTORE BCD MANTISSA TO D-E
 056E C9            899         RET
                    900 #SUBROUTINE TROT:THIS SUBROUTINE ROTATES D-E AND H-L UNTIL THE
                    901 #NUMERICAL EXPONENT IS REDUCED TO ZERO.
 056F 210000        902 TROT:   LXI H,0000H
 0572 CDE005        903 TRT1:   CALL DROT
 0575 CDE905        904         CALL HROT
 0578 0D            905         DCR C
 0579 3E40          906         MVI A,40H
 057B B9            907         CMP C
 057C C8            908         RZ
 057D C37205        909         JMP TRT1
                    910 #SUBROUTINE BCD: THIS SUBROUTINE CONVERTS A BINARY NUMBER IN
                    911 #H-L TO A BCD NUMBER IN H-L. IF IT IS ENTERED WITH THE A REGISTER
                    912 #EQUAL TO ZERO, IT CONVERTS THE NUMBERICAL MANTISSA IN H-L. IF THE
                    913 #A REGISTER IS NONZERO, IT CONVERTS THE NUMBERICAL EXPONENT IN H.
 0580 F5            914 BCD:    PUSH PSW
 0581 110000        915         LXI D,0000H
 0584 EB            916         XCHG
 0585 87            917         ADD A
 0586 CA9205        918         JZ BC1          #Z IMPLIES MANTISSA TO BE CONVERTED
 0589 0606          919         MVI B,06H
 058B 7A            920         MOV A,D
 058C 17            921         RAL
 058D 17            922         RAL
 058E 57            923         MOV D,A
 058F C39405        924         JMP BC2
 0592 0610          925 BC1:    MVI B,10H
 0594 CDE005        926 BC2:    CALL DROT
 0597 CDE905        927         CALL HROT
 059A 05            928         DCR B
 059B CAB905        929         JZ BC6
 059E 7D            930         MOV A,L         #CHECK FIRST DECADE
 059F E60F          931         ANI 0FH
 05A1 FE05          932         CPI 05H
 05A3 DAAE05        933         JC BC3          #C IMPLIES DECADE <5, SKIP
 05A6 3E03          934         MVI A,03H
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   18

 LOC  OBJ          LINE        SOURCE STATEMENT

 05A8 85            935         ADD L
 05A9 6F            936         MOV L,A
 05AA 3E00          937         MVI A,00H
 05AC 8C            938         ADC H
 05AD 67            939         MOV H,A
 05AE 7D            940 BC3:    MOV A,L          ;CHECK SECOND DECADE
 05AF E6F0          941         ANI     0F0H
 05B1 FE50          942         CPI 50H
 05B3 DABE05        943         JC BC4           ;C IMPLIES DECADE <5, SKIP
 05B6 3E30          944         MVI A,30H
 05B8 85            945         ADD L
 05B9 6F            946         MOV L,A
 05BA 3E00          947         MVI A,00H
 05BC 8C            948         ADC H
 05BD 67            949         MOV H,A
 05BE 7C            950 BC4:    MOV A,H          ;CHECK THIRD DECADE
 05BF E60F          951         ANI 0FH
 05C1 FE05          952         CPI 05H
 05C3 DACA05        953         JC BC5           ;C IMPLIES DECADE <5, SKIP
 05C6 3E03          954         MVI A,03H
 05C8 84            955         ADD H
 05C9 67            956         MOV H,A
 05CA 7C            957 BC5:    MOV A,H          ;CHECK FOURTH DECADE
 05CB E6F0          958         ANI 0F0H
 05CD FE50          959         CPI 50H
 05CF DA9405        960         JC BC2           ;C IMPLIES DECADE <5, SKIP
 05D2 3E30          961         MVI A,30H
 05D4 84            962         ADD H
 05D5 67            963         MOV H,A
 05D6 C39405        964         JMP BC2
 05D9 F1            965 BC6:    POP PSW
 05DA 87            966         ADD A
 05DB C8            967         RZ               ;Z IMPLIES MANTISSA, RETURN WITH BCD
                    968                          ;IN H-L
 05DC 79            969         MOV A,C
 05DD 85            970         ADD L
 05DE 4F            971         MOV C,A
 05DF C9            972         RET
                    973 ;SUBROUTINE DROT: THIS SUBROUTINE ROTATES THE D-E REGISTER PAIR LEFT
 05E0 37            974 DROT:   STC
 05E1 3F            975         CMC
 05E2 7B            976         MOV A,E
 05E3 17            977         RAL
 05E4 5F            978         MOV E,A
 05E5 7A            979         MOV A,D
 05E6 17            980         RAL
 05E7 57            981         MOV D,A
 05E8 C9            982         RET
                    983 ;SUBROUTINE HROT: THIS SUBROUTINE ROTATES THE H-L REGISTER PAIR
                    984 ;LEFT, PUTTING THE CARRY INTO THE LSB OF L
 05E9 7D            985 HROT:   MOV A,L
 05EA 17            986         RAL
 05EB 6F            987         MOV L,A
 05EC 7C            988         MOV A,H
 05ED 17            989         RAL
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   19

 LOC  OBJ          LINE        SOURCE STATEMENT

 05EE 67            990         MOV H,A
 05EF C9            991         RET
                    992 ;SUBROUTINES ZRO AND ZRDE: THESE SUBROUTINES CHECK FOR A
                    993 ;ZERO VALUE FOR A FLOATING POINT NUMBER IN C-D-E (ZRO) OR
                    994 ;ZERO D-E ONLY (ZRDE).  IF THE RESULT WAS ZERO, THE RETURN
                    995 ;IS MADE WITH A ZERO A REGISTER.  OTHERWISE, A EQU 40H UPON RETURN.
 05F0 3E40          996 ZRO:    MVI A,40H
 05F2 B9            997         CMP C               ;CHECK FOR EXPONENT CORRESPONDING TO 0
 05F3 C0            998         RNZ                 ;NZ IMPLIES NUMBER IS NOT ZERO
 05F4 7B            999 ZRDE:   MOV A,E             ;TEST D-E
 05F5 82           1000         ADD D
 05F6 DAFE05       1001         JC ZR1              ;C  EQU  1 IMPLIES NUMBER IS NOT 0
 05F9 C2FE05       1002         JNZ ZR1             ;NZ IMPLIES NUMBER IS NOT ZERO
 05FC AF           1003         XRA A
 05FD C9           1004         RET                 ;RESULT WAS ZERO, RETURN WITH ZERO A
 05FE 3E40         1005 ZR1:    MVI A,40H
 0600 C9           1006         RET                 ;RESULT NOT ZERO, RETURN WITH A EQU 40H
                   1007 ;
                   1008 ;
                   1009 ;
                   1010 ;
                   1011 ;RAM AREA HERE
 3C00              1012         ORG     3C00H
 3C00 00           1013 S1:     DB      0
 3C01 00           1014 S2:     DB      0
 3C02 00           1015 S3:     DB      0
 3C03 00           1016 S4:     DB      0
 3C04 00           1017 S5:     DB      0
 3C05 00           1018 ME:     DB      0
 3C06 00           1019 MH:     DB      0
 3C07 00           1020 ML:     DB      0
 3C08 00           1021 MX:     DB      0
                   1022         ;
                   1023 ;
 3C09 00           1024 OPSTR:  DB      0
 3C0A 0000         1025 HLSTR:  DW      0
                   1026 ;
                   1027 ;
 3C0C 00           1028 ADJFAC: DB   0,0,0
 3C0D 00
 3C0E 00
 3C0F 00           1029 STP5:   DB   0,0,0
 3C10 00
 3C11 00
 3C12 00           1030 WT5:    DB      0,0,0
 3C13 00
 3C14 00
 3C15 00           1031 COUNT:  DB      0,0
 3C16 00
 3C17 00           1032 TIME5:  DB      0,0
 3C18 00
 3C19 00           1033 TIME6:  DB      0,0
 3C1A 00
 3C1B 00           1034 WT6:    DB      0,0,0
 3C1C 00
```

Figure C-1. (Continued)

```
ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0        MODULE   PAGE   20

  LOC  OBJ          LINE        SOURCE STATEMENT

  3C1D 00
  3C1E 00           1035 STP6:  DB      0,0,0
  3C1F 00
  3C20 00
  3C21 00           1036 FLAG5: DB      0
  3C22 00           1037 FLAG6: DB      0
  3C23 00           1038 FLAGA: DB      0
  3C24 00           1039 FLAGB: DB      0
  3C25 00           1040 OUTPUT:DB      0
                    1041 ;
                    1042        END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
ADCC    A F719    ADCD    A F71A    ADCM    A F718    ADFAC   A 000C    ADJFAC A 3C0C    BC1     A 0592    BC2     A 0594
BC3     A 05AE    BC4     A 05BE    BC5     A 05CA    BC6     A 05D9    BCD     A 0580    CALC5   A 0159    CALC6   A 0221
CH5     A 00B0    CH5A    A 00C9    CH6     A 00E0    CH6A    A 00F9    CHKA0   A 017E    CHKA1   A 015E    CHKA2   A 0246
CHKA3   A 0226    COUNT   A 3C15    DEL2    A 02EB    DELAY   A 02E9    DFTM5   A 0199    DFTM6   A 0261    DISC5   A 0119
DISC6   A 012C    DROT    A 05E0    FA00    A 02F8    FA01    A 0327    FA02    A 0329    FA03    A 0343    FA04    A 0351
FA05    A 0359    FA06    A 035E    FA07    A 0361    FA08    A 036E    FA09    A 0373    FA10    A 037E    FA11    A 038C
FADD    A 02F4    FB1     A 04FD    FB2     A 0515    FB3     A 0527    FB3A    A 0536    FB4     A 0549    FB5     A 0567
FB6     A 0568    FBCD    A 04F0    FD00    A 03E7    FD01    A 040B    FD02    A 0411    FD03    A 042B    FD04    A 043F
FD05    A 0442    FD06    A 0448    FDIV    A 03E0    FIXX    A 0392    FL1     A 03C0    FL2     A 03C7    FL3     A 03D3
FLAG5   A 3C21    FLAG6   A 3C22    FLAGA   A 3C23    FLAGB   A 3C24    FLOT    A 03BE    FM1     A 0491    FM2     A 0497
FM3     A 04B1    FM4     A 04C9    FM5     A 04DB    FM6     A 04EE    FMPY    A 0458    FN1     A 03DB    FNEG    A 03D6
FSUB    A 02F1    FX1     A 03A6    FX2     A 03B3    FX3     A 03B6    FX4     A 03BB    HLSTR   A 3C0A    HNZ     A 0142
HROT    A 05E9    INCTO   A 0036    INSP5   A 0179    INSP6   A 0241    LNZ     A 014C    ME      A 3C05    MH      A 3C06
ML      A 3C07    MX      A 3C08    NAMIN   A 00E4    NAMOUT  A 00E6    OPSTR   A 3C09    OUTPUT  A 3C25    OUTTR   A 0110
RSTS6   A 008D    RSTS62  A 0094    S1      A 3C00    S2      A 3C01    S3      A 3C02    S4      A 3C03    S5      A 3C04
STCNT   A 0042    STNAT   A 0069    STNAT2  A 0071    STOCT   A 0066    STP5    A 3C0F    STP6    A 3C1E    STR56   A 002B
TIME5   A 3C17    TIME6   A 3C19    TIMLP   A 013F    TROT    A 056F    TRT1    A 0572    WD15    A 01BB    WD15A   A 01CB
WD16    A 01DA    WD16A   A 01EA    WD17    A 01F3    WD17A   A 0203    WD18    A 020F    WD18A   A 0216    WD19    A 0283
WD19A   A 0293    WD20    A 02A2    WD20A   A 02B2    WD21    A 02BB    WD21A   A 02CB    WD22    A 02D7    WD22A   A 02DE
WT5     A 3C12    WT6     A 3C1B    ZR1     A 05FE    ZRDE    A 05F4    ZRO     A 05F0

ASSEMBLY COMPLETE,  NO ERRORS
```

Figure C-1. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0001

0000 C       TITLE:REV1.FTN
0000 C       AUTHOR:EVERETTE M. ELDRIDGE
0000 C       DATE:6-5-81
0000 C       THIS PROGRAM OPERATES TWO SCALE SYSTEMS SIMULTANEOUSLY.
0000 C       IT IS WRITTEN FOR AN SBC 80/10A SINGLE BOARD COMPUTER.
0000 C       THE A/D BOARD USED IS AN ADAC 735-SBC
0000 C       IT ADJUSTS THE CYCLE TIME OF EACH SCALE ACCORDING
0000 C       TO WHAT WEIGHT WAS RECEIVED.  ALSO, IT TURNS ON ALARMS
0000 C       FOR EXCESSIVE WEIGHT INACCURACIES AND SLOW ROLL SPEEDS.
0000 C ****SET START ADDRESS OF CODE****
0000 C       COMPILER(1)=00H
0000 C ****INITIALIZE THE A/D CONVERTER BOARD
0000 C       ADCM = 0H  STP5 SCALE A SETPOINT
0000 C       ADCM = 1H  WT5 SCALE A WEIGHT
0000 C       ADCM = 2H  STP6 SCALE B SETPOINT
0000 C       ADCM = 3H  WT6 SCALE B WEIGHT
0000 C       ADCC = 018H CONTROL FORMAT
0000 C       ADCD IS DATA LOCATION
0000         COMPILER(3)=0F718H
0000         INTEGER*1 ADCM
0000         COMPILER(3)=0F719H
0000         INTEGER*1 ADCC
0000         COMPILER(3)=0F71BH
0000         INTEGER*2 ADCD
0000 C ****SET TOP ADDRESS OF RAM MEMORY****
0000         COMPILER(3)=03FFFH
0000 C ****DECLARE VARIABLES USED****
0000         INTEGER*1 SYST5,ENDCOM5,PORTB,NOROLL5,RESTART5
0000         INTEGER*2 COUNT5,TIME5,ENDCALC5
0000         REAL*4 WT5,STP5
0000         INTEGER*1 SYST6,ENDCOM6,PORTC,NOROLL6,RESTART6
0000         INTEGER*2 COUNT6,TIME6,ENDCALC6
0000         REAL*4 WT6,STP6
0000         INTEGER*1 PORTA
0000         COMPILER(2)=040H
0040 C ****INTERRUPT SERVICE ROUTINE****
0040 C       THIS ROUTINE INCREMENTS COUNT, RESETS COUNT TO LOWER
0040 C       ZERO TO AVOID OVERFLOW, AND CHECKS TO SEE IF SCALE
0040 C       IS READY TO BE STARTED.
0040         INTERRUPT 7
0040         COUNT5=COUNT5+1
004A         COUNT6=COUNT6+1
0054    10   IF(SYST5) 11,11,20
0058    11   IF(COUNT5.LT.TIME5) GO TO 20
006C         COUNT5=0
0072         NOROLL5=0
0077         PORTB=((PORTB.OR.011H).AND.01FH)
007E         OUTPUT(229)=PORTB
0081         SYST5=1
0085    20   IF(SYST6) 21,21,30
0089    21   IF(COUNT6.LT.TIME6) GO TO 30
009D         COUNT6=0
```

Figure C-2. Program Listing for First Revision

```
UTI FORT//80 COMPILER V3.3C                          PAGE 0003

00A3          NOROLL6=0
00A8          PORTC=((PORTC.OR.011H).AND.01FH)
00AF          OUTPUT(230)=PORTC
00B2          SYST6=1
00B6     30   ENABLE
00B7          RETURN
00BD          END


PROGRAM STORAGE= 0132

VARIABLE STORAGE= 0000


SYMBOL TABLE

00B6 30
008F 21
0085 20
005E 11
0054 10
```

Figure C-2. (Continued)

```
UTI FORT/80 COMPILER V3.30                    PAGE 0003

00BD C ****MEANING OF SYSTEM PROGRAM VARIABLES****
00BD C       COUNT5= SCALE A STARTS WEIGHING WHEN COUNT5=COUNT
00BD C       COUNT6= SCALE B STARTS WEIGHING WHEN COUNT5=COUNT
00BD C       ENDCALC5= START CALCULATION OF NEXT COUNT5 WHEN =COUNT
00BD C       ENDCALC6= START CALCULATION OF NEXT COUNT6 WHEN =COUNT
00BD C       ENDCOM5= 1 WHEN SCALE A HAS REACHED SETPOINT
00BD C       ENDCOM6= 1 WHEN SCALE B HAS REACHED SETPOINT
00BD C       NOROLL5= 1 WHEN SCALE A IS WAITING FOR NEXT START
00BD C       NOROLL6= 1 WHEN SCALE B IS WAITING FOR NEXT START
00BD C       PORTA= INPUT PORT 228 STORAGE FOR BOTH SCALES
00BD C       PORTB= OUTPUT PORT 229 STORAGE FOR SCALE A
00BD C       PORTC= OUTPUT PORT 230 STORAGE FOR SCALE B
00BD C       RESTART5= 1 CAUSES COUNT5=COUNT WHEN SCALE A TURNED ON
00BD C       RESTART6= 1 CAUSES COUNT6=COUNT WHEN SCALE B TURNED ON
00BD C       STP5= VALUE OF SETPOINT FOR SCALE A
00BD C       STP6= VALUE OF SETPOINT FOR SCALE B
00BD C       SYST5= 1 WHEN PROGRAM CALCULATING NEW COUNT5
00BD C       SYST6= 1 WHEN PROGRAM CALCULATING NEW COUNT6
00BD C       TIME5= CALCULATED TIME NEEDED FOR WEIGHT RECEIVED SCALE A
00BD C       TIME6= CALCULATED TIME NEEDED FOR WEIGHT RECEIVED SCALE B
00BD C       WT5= VALUE OF WEIGHT ON SCALE A
00BD C       WT6= VALUE OF WEIGHT ON SCALE B
00BD C ****PARALLEL INPUT/OUTPUT PORTS**********************************
00BD C       PORT 228    INPUT    BIT # 0= SCALE A ON/OFF SWITCH
00BD C                                 1= NOT USED
00BD C                                 2= NOT USED
00BD C                                 3= NOT USED
00BD C                                 4= SCALE B ON/OFF SWITCH
00BD C                                 5= NOT USED
00BD C                                 6= NOT USED.
00BD C                                 7= NOT USED
00BD C       PORT 229    OUTPUT   BIT # 0= SCALE A ON/OFF OR RESTART
00BD C                                 1= SCALE A HI WEIGHT ALARM
00BD C                                 2= SCALE A LO WEIGHT ALARM
00BD C                                 3= SCALE A MALFUNCTION ALARM
00BD C                                 4= SCALE A ROLLS TOO SLOW
00BD C                                 5= SCALE A BULK SPEED
00BD C                                 6= SCALE A DRIBBLE SPEED
00BD C                                 7= NOT USED
00BD C       PORT 230    OUTPUT   BIT # (SAME AS PORT 229 EXCEPT FOR B)
00BD C
00BD C
00BD C****INITIALIZE CONSTANTS****
00BD         DISABLE
00BE         OUTPUT(231)=090H
00C2         ADCC=018H
00C7         COUNT5=0
00CD         COUNT6=0
00D0         TIME5=0
00D3         TIME6=0
00D6         RESTART5=0
00DB         RESTART6=0
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0004

00DF          PORTA=0
00E3          PORTB=0
00E7          PORTC=0
00EB          ENDCOM5=0
00EF          ENDCOM6=0
00F3          SYST5=0
00F7          SYST6=0
00FB          OUTPUT(229)=0
00FF          OUTPUT(230)=0
0103          ENABLE
0104 C
0104 C
0104 C        IF CYCLE TOO LONG, OUTPUT ROLLS TOO SLOW ALARM
0104      40  IF(COUNT5.LT.300)GO TO 45
0110          PORTB=(PORTB.AND.0EFH)
0118          OUTPUT(229)=PORTB
011D C        IF SCALE TURNED OFF, SET RESTART5 AND CHECK B
011D      45  IF(RESTART5.EQ.0)GO TO 46
0125          COUNT5=0
012B          TIME5=0
012E          RESTART5=0
0133          NOROLL5=1
0136          ENDCOM5=0
013A          PORTB=(PORTB.AND.01EH)
013F          SYST5=0
0143      46  IF(NOROLL5.NE.0)GO TO 410
014B C        IF ENDCOM5 SET, SKIP ROLL SPEED CHECK
014B      47  IF(ENDCOM5.NE.0)GO TO 100
0153 C        INPUT WEIGHT
0153          ADCM=1H
0158      50  CONTINUE
0158          IF((ADCC.AND.80H).EQ.0)GO TO 50
0162      55  WT5=ADCD
016C C        INPUT SETPOINT
016C          ADCM=0H
0172      60  CONTINUE
0172          IF((ADCC.AND.80H).EQ.0)GO TO 60
017C      65  STP5=ADCD
0186          IF(WT5-0.7*STP5)70,80,80
0192 C        TURN ON BULK
0192      70  PORTB=(PORTB.OR.020H)
019D          OUTPUT(229)=PORTB
01A2          GO TO 410
01A5 C        TURN OFF BULK AND TURN ON DRIBBLE
01A5      80  PORTB=((PORTB.OR.040H).AND.0DFH)
01AF          OUTPUT(229)=PORTB
01B4          IF(WT5-0.9*STP5)410,90,90
01C3 C        TURN OFF DRIBBLE AND RESET WEIGHT ALARMS
01C3      90  PORTB=((PORTB.AND.01FH).OR.0EH)
01D0          OUTPUT(229)=PORTB
01D5 C        SET UP DELAY FOR CALCULATIONS
01D5          ENDCALC5=COUNT5+50
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0005

01DF C       SET FLAG TO SKIP ROLL SPEED CHECKS
01DF         ENDCOM5=1
01E4 C       CHECK IF TIME FOR CALCULATIONS
01E4   100   IF(ENDCALC5.GT.COUNT5)GO TO 410
01F2 C
01F2 C
01F2 C       THIS ROUTINE CALCULATES CYCLE TIME
01F2 C       AND SETS WEIGHT ALARMS
01F2   105   ENDCOM5=0
01F7         NOROLL5=1
01FB C       INPUT WEIGHT
01FB         ADCM=1H
0200   110   CONTINUE
0200         IF((ADCC.AND.80H).EQ.0)GO TO 110
020A   120   WT5=ADCD
0214 C       INPUT SETPOINT
0214         ADCM=0H
021A   130   CONTINUE
021A         IF((ADCC.AND.80H).EQ.0)GO TO 130
0224   140   STP5=ADCD
022E C       TURN OFF SCALE A
022E         PORTB=(PORTB.AND.0FEH)
0237         OUTPUT(229)=PORTB
023C C       CALCULATE THE TIME
023C         TIME5=((WT5/STP5)*450.0)
024E C       IF LESS THAN 40 SECONDS, SET LOW ALARM
024E         IF(TIME5.GT.400)GO TO 150
025A         PORTB=(PORTB.AND.0FBH)
0262         OUTPUT(229)=PORTB
0267         GO TO 170
026A C       IF MORE THAN 50 SECONDS, SET HIGH ALARM
026A   150   IF(TIME5.LT.500)GO TO 170
0276         PORTB=(PORTB.AND.0FDH)
027E         OUTPUT(229)=PORTB
0283 C       IF MORE THAN 90 SECONDS, SET MALFUNCTION
0283 C       AND SET TIME5=450
0283         IF(TIME5.LT.900)GO TO 170
028F         PORTB=(PORTB.AND.0F7H)
0297         OUTPUT(229)=PORTB
029C         TIME5=450
02A2   170   SYST5=0
02A7         GO TO 410
02AA C       SET RESTART5 IF SCALE TURNED OFF
02AA   400   RESTART5=1
02AF         SYST5=1
02B3 C
02B3 C
02B3 C
02B3 C       SCALE B CHECK
02B3 C       IF CYCLE TOO LONG, TURN ON ROLLS TOO SLOW ALARM
02B3   410   IF(COUNT6.LT.300)GO TO 415
02BF         PORTC=(PORTC.AND.0EFH)
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0006

02C7          OUTPUT(230)=PORTC
02CC C        IF SCALE B OFF, SET RESTART6 AND CHECK SCALE A
02CC    415   IF((INPUT(228).AND.010H).EQ.0)GO TO 580
02D5          IF(RESTART6.EQ.0)GO TO 416
02DD          TIME6=0
02E3          COUNT6=0
02E6          RESTART6=0
02EB          PORTC=(PORTC.AND.01EH)
02F1          SYST6=0
02F5    416   IF(NOROLL6.NE.0)GO TO 40
02FD C        IF ENDCOM6 SET, SKIP ROLL SPEED CHECK
02FD    417   IF(ENDCOM6.NE.0)GO TO 490
0305 C        INPUT WEIGHT
0305          ADCM=3H
030A    420   CONTINUE
030A          IF((ADCC.AND.80H).EQ.0)GO TO 420
0314    430   WT6=ADCD
031E C        INPUT SETPOINT
031E    440   CONTINUE
031F          IF((ADCC.AND.80H).EQ.0)GO TO 440
0329    450   STP6=ADCD
0333          IF(WT6-0.7*STP6)460,470,470
033F C        TURN ON BULK
033F    460   PORTC=(PORTC.OR.20H)
034A          OUTPUT(230)=PORTC
034F          GO TO 40
0352 C        TURN OFF BULK AND TURN ON DRIBBLE
0352    470   PORTC=((PORTC.OR.40H).AND.0DFH)
035C          OUTPUT(230)=PORTC
0361          IF(WT6-0.9*STP6)40,480,480
0370 C        TURN OFF DRIBBLE AND WEIGHT ALARMS
0370    480   PORTC=((PORTC.AND.01FH).OR.0EH)
037D          OUTPUT(230)=PORTC
0382 C        SET UP DELAY FOR CALCULATIONS
0382          ENDCALC6=COUNT6+50
038C C        SET FLAG TO SKIP ROLL SPEED CHECK
038C          ENDCOM6=1
0391    490   IF(ENDCALC6.GT.COUNT6)GO TO 40
039F C        THIS ROUTINE CALCULATES TIME NEEDED
039F C        SETS ALARMS
039F    500   ENDCOM6=0
03A4          NOROLL6=1
03A8 C        INPUT WEIGHT
03A8          ADCM=3H
03AD    510   CONTINUE
03AD          IF((ADCC.AND.80H).EQ.0)GO TO 510
03B7    520   WT6=ADCD
03C1 C        INPUT SETPOINT
03C1    530   CONTINUE
03C2          IF((ADCC.AND.80H).EQ.0)GO TO 530
03CC    540   STP6=ADCD
03D6 C        TURN OFF SCALE B
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0007

03D6          PORTC=(PORTC.AND.0FDH)
03DF          OUTPUT(230)=PORTC
03E4 C        CALCULATE TIME6
03E4          TIME6=((WT6/STP6)*450.0)
03F6 C        CHECK FOR WEIGHT ALARMS
03F6          IF(TIME6.GT.400)GO TO 550
0402 C        IF TIME6 LESS THAN 40 SECONDS, TURN ON LOW ALARM
0402          PORTC=(PORTC.AND.0FBH)
040A          OUTPUT(230)=PORTC
040F          GO TO 570
0412    550   IF(TIME6.LT.500)GO TO 570
041E C        IF TIMES MORE THAN 50 SECONDS, TURN ON HIGH ALARM
041E          PORTC=(PORTC.AND.0FDH)
0426          OUTPUT(230)=PORTC
042B    560   IF(TIME6.LT.900)GO TO 570
0437 C        IF TIME6 MORE THAN 90 SECONDS, TURN ON MALFUNCTION ALARM
0437 C        AND SET TIME6=450
0437          PORTC=(PORTC.AND.0F7H)
043F          OUTPUT(230)=PORTC
0444          TIME6=450
044A    570   SYST6=0
044F          GO TO 40
0452    580   RESTART6=1
0457          SYST6=1
045B          GO TO 40
045E          STOP
0461          END

PROGRAM STORAGE= 0932

VARIABLE STORAGE= 0000



TOTAL PROGRAM STORAGE= 2789

TOTAL VARIABLE STORAGE= 0043


SYMBOL TABLE

042B 560
044A 570
0412 550
03CC 540
03C2 530
03B7 520
03AD 510
```

Figure C-2. (Continued)

```
UTI FORT//20 COMPILER V3.3C                    PAGE 0008

039F 500
0373 480
0352 470
0342 460
0329 450
031F 440
0314 430
030A 420
0391 490
02FD 417
02F5 416
0452 580
02CC 415
02AA 400
02A2 170
026A 150
0B1A 450.0
0224 140
021A 130
020A 120
0200 110
01F2 105
01C6 90
0B16 0.9
01A5 80
0195 70
0B12 0.7
017C 65
0172 60
0162 55
0158 50
01E4 100
014B 47
02B3 410
0143 46
011D 45
0104 40
3FD9 PORTA
3FDA STP6
3FDE WT6
3FE2 ENDCALC6
3FE4 TIME6
3FE6 COUNT6
3FE8 RESTART6
3FE9 NOROLL6
3FEA PORTC
3FEB ENDCOM6
3FEC SYST6
3FED STP5
3FF1 WT5
3FF5 ENDCALC5
3FF7 TIME5
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0009

3FF9 COUNTS
3FFB RESTARTS
3FFC NOROLLS
3FFD PORTB
3FFE ENDCOMS
3FFF SYSTS
F71A ADCD
F719 ADCC
F718 ADCM
04F6 FLOAT DEBUG
```

Figure C-2. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0001

0000 C      TITLE: REV2.FTN
0000 C      AUTHOR: EVERETTE M. ELDRIDGE
0000 C      DATE: 6-5-81
0000 C      LAST REVISION DATE: 7-13-81
0000 C      REVISION: REMOVE DIVIDE BY SIX IN INTERRUPT ROUTINE
0000 C
0000 C      THIS PROGRAM OPERATES TWO BATCH SCALES.  THE
0000 C      OPERATOR INTERFACE WITH THE MICROCOMPUTER
0000 C      CONSISTS OF THE FOLLOWING:
0000 C      (1) TTY  THIS DEVICE IS USED TO CHANGE CONSTANTS,
0000 C          SUCH AS SETPOINTS, CYCLE TIME, ROLL SPEED
0000 C          SETPOINT(BDB),AND ROLL STOP SETPOINT(MIS).
0000 C          IT ALSO INDICATES ALARM MESSAGES AND CONSTANT
0000 C          VALUES ON REQUEST.
0000 C      (2) ALARM HORN  ONE FOR EACH SCALE.  THIS DEVICE
0000 C          NOTIFIES WHEN AN ALARM HAS BEEN SENSED.
0000 C      (3) SCALE ON/OFF SWITCH  ONE FOR EACH SCALE.  WHEN
0000 C          IN THE 'ON' POSITION, THIS DEVICE PUTS THE
0000 C      (4) POWER ON/OFF SWITCH  THIS DEVICE TURNS POWER
0000 C          ON THE MICROCOMPUTER.
0000 C
0000 C      SUBROUTINES AND INTERRUPTS
0000 C      INTERRUPT 7: CLOCK INPUT
0000 C      COUT: OUTPUT TO TTY
0000 C      CINP: INPUT FROM TTY
0000 C      INITIALIZE: INITIALIZES I/O AND CONSTANTS
0000 C      ARUN: RUN SCALE A
0000 C      BRUN: RUN SCALE B
0000 C      CHOOSE: FIND WHICH SCALE IS CHOSEN
0000 C      PRLIST: LIST VALUE OF CONSTANTS ON TTY
0000 C      YN: CHECK IF CONSTANT IS NECESSARY
0000 C      VALIN: GET NEW VALUE FROM TTY
0000 C      FACTOR: COMPUTE NEW CONSTANT VALUE
0000 C      WHAT: DECIDES WHAT FUNCTION TO DO
0000 C      HORNA: ALARMS FOR SCALE A
0000 C      HORNB: ALARMS FOR SCALE B
0000 C      CALIBRATE: WRITES VALUE SCALE'S A/D TO TTY
0000 C
0000 C      TTY INPUT COMMANDS
0000 C      KEY PRESSED    ASCII VALUE      FUNCTION
0000 C      CONTROL E          05H          MIS INPUT
0000 C      CONTROL D          04H          BDS INPUT
0000 C      CONTROL T          14H          CYCLE TIME INPUT
0000 C      CONTROL S          13H          SETPOINT INPUT
0000 C      CONTROL A          01H          SCALE A CHOSEN
0000 C      CONTROL B          02H          SCALE B CHOSEN
0000 C      CONTROL L          0CH          LIST CONSTANTS
0000 C      CONTROL H          08H          LIST ALARMS
0000 C      CONTROL Z          1AH          CALIBRATE
0000 C
0000 C      SET STARTING ADDRESS OF EPROM
0000        COMPILER(1)=00H
```

Figure C-3. Program Listing for Second Revision

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0002

0006 C
0006 C      INITIALIZE THE A/D BOARD MEMORY MAPPED LOCATION
0006        COMPILER(3)=0F719H
0006        INTEGER*1 ADCC,ADCM
0006        COMPILER(3)=0F71BH
0006        INTEGER*2 ADCD
0006 C
0006 C      SET LAST ADDRESS OF RAM
0006        COMPILER(3)=03FFFH
0006 C
0006 C      DECLARE GLOBAL VARIABLES
0006 C      MISCELLANEOUS VARIABLES
0006 C      EXPLANATION OF VARIABLES
0006 C      IBLANK-(NOT USED)=1,WILL CAUSE TTY IN TO SKIP SCALE RUN
0006 C      OBLANK-(NOT USED)=1, WILL CAUSE TTY OUT TO SKIP SCALE RUN
0006 C      CHIN(1)-CHARACTER INPUT FROM TTY
0006 C      SCALE-SCALE REQUESTED FROM TTY
0006 C      WHICH-FUNCTION OR CONSTANT REQUESTED FROM TTY
0006 C      SKIPWHAT-IF=1, CONSTANT HAS BEEN CHOSEN, SKIP SUB WHAT
0006 C      SKIPYN-IF=1, YES HAS BEEN RECEIVED, SKIP SUB YN
0006 C      STAT-STATUS FROM TTY PORT
0006 C      PORTA-PARALLEL INPUT PORT STORAGE
0006 C      DUMMY-CHARACTER TO BE OUTPUT ON TTY
0006 C      CO-STORES LOCATION OF TTY OUTPUT ROUTINE COUT
0006 C      CI-STORES LOCATION OF TTY INPUT ROUTINE CINP
0006 C      NEWVAL-NEW CONSTANT VALUE
0006 C      LETTER-VALUE OF CONSTANT OUTPUT IN WRITE STATEMENT
0006 C      RENEW-FLOATING POINT VALUE OF NEWVAL
0006        INTEGER*1 IBLANK,OBLANK,CHIN(1),SCALE
0006        INTEGER*1 WHICH,SKIPWHAT,SKIPYN,STAT,PORTA
0006        INTEGER*2 DUMMY,CO,CI,NEWVAL,LETTER
0006        REAL*4 RENEW
0006 C      VARIABLE DESCRIPTION FOR EACH SCALE ARE SIMILAR.
0006 C      REPLACE * WITH SCALE LETTER ('A' OR'B')
0006 C      SYST*-IF =1,SKIP START CHECK
0006 C      NOROLL*-IF =1, SKIP ROLL SPEED CHECK
0006 C      PORTB-PARALLEL OUTPUT FOR SCALE A
0006 C      PORTC-PARALLEL OUTPUT FOR SCALE B
0006 C      SRA*-IF =1,SKIP SLOW ROLL CHECK
0006 C      ALARM*-STORAGE WHICH TELLS ALARMS DETECTED
0006 C      RESTART*-IF =1,RESTART IS NEEDED
0006 C      ENDCOM*-COMPARISON FOR ROLL SPEED COMPLETED
0006 C      M*-INTEGER STORAGE FOR MIS CONSTANT
0006 C      B*-INTEGER STORAGE FOR BDS CONSTANT
0006 C      TIME*-CYCLE TIME CALCULATED
0006 C      SCNT*-SYSTEM COUNT
0006 C      ENDCALC*-WHEN NEW CALCULATION FOR SCALE CAN BEGIN
0006 C      C*-INTEGER STORAGE FOR CYCLE TIME CONSTANT
0006 C      S*-INTEGER STORAGE FOR SETPOINT
0006 C      WT*-REAL VALUE OF WEIGHT RECEIVED
0006 C      WTI*-INTEGER VALUE OF WEIGHT RECEIVED
0006 C      BD*-REAL STORAGE OF BDS CONSTANT
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0003

0006 C       MIS*-REAL STORAGE OF MIS CONSTANT
0006 C       BDI*-INTEGER STORAGE OF BDS COMPARISON AMOUNT
0006 C       MISI*-INTEGER STORAGE OF MIS COMPARISON AMOUNT
0006 C       COUNT*-REAL STORAGE OF CYCLE TIME CONSTANT
0006 C       STP*-REAL STORAGE OF SETPOINT
0006 C       LO*-LOW ALARM SETPOINT
0006 C       HI*-HIGH ALARM SETPOINT
0006 C       HIHI*-MALFUNCTION ALARM SETPOINT
0006 C       ROLL*-ROLL SPEED ALARM SETPOINT
0006 C       THESE VARIABLES ARE FOR SCALE A
0006         INTEGER*1 SYSTA,NOROLLA,PORTB,SRAA,ALARMA
0006         INTEGER*1 RESTARTA,ENDCOMA,MA,BA
0006         INTEGER*2 TIMEA,SCNTA,ENDCALCA,CA,SA,MISIA,BDIA,WTIA
0006         INTEGER*2 LOA,HIA,HIHIA,ROLLA
0006         REAL*4 WTA,BDA,MISA,COUNTA,STPA
0006 C       THESE VARIABLES ARE FOR SCALE B
0006         INTEGER*1 SYSTB,NOROLLB,PORTC,SRAB,ALARMB
0006         INTEGER*1 RESTARTB,ENDCOMB,MB,BB
0006         INTEGER*2 TIMEB,SCNTB,ENDCALCB,CB,SB,MISIB,BDIB,WTIB
0006         INTEGER*2 LOB,HIB,HIHIB,ROLLB
0006         REAL*4 WTB,BDB,MISB,COUNTB,STPB
0006 C
0006 C       SET STARTING ADDRESS FOR PROGRAM CODE
0006         COMPILER(2)=040H
0040 C       INTERRUPT SERVICE ROUTINE:  THIS ROUTINE
0040 C       INCREMENTS SYSTEM COUNT FOR EACH SCALE
0040 C       (SCNTA AND SCNTB).  IT ALSO STARTS NEXT
0040 C       SCALE CYCLE AND RESETS SCALE COUNTS TO 0.
0040 C       IT IS INITIALIZED BY A LOW PULSE ON PIN J1-49.
0040         INTERRUPT 7
0040         SCNTA=SCNTA+1
004A         SCNTB=SCNTB+1
0054 C       CHECK FOR SLOW ROLL SPEED
0054       1 IF(NOROLLA.NE.0)GO TO 5
005C         IF(SRAA.NE.0)GO TO 5
0064         IF(SCNTA.LT.ROLLA)GO TO 5
0072 C       SET ALARM BIT AND TURN ON HORN
0072         ALARMA=ALARMA.OR.01H
007A         PORTB=PORTB.OR.02H
0082         OUTPUT(229)=PORTB
0087         SRAA=1
008C       5 IF(NOROLLB.NE.0)GO TO 10
0094         IF(SRAB.NE.0)GO TO 10
009C         IF(SCNTB.LT.ROLLB)GO TO 10
00AA         ALARMB=ALARMB.OR.01H
00B2         PORTC=PORTC.OR.02H
00BA         OUTPUT(230)=PORTC
00BF         SRAB=1
00C4      10 IF(SYSTA)11,11,20
00C8      11 IF(SCNTA.LT.TIMEA)GO TO 20
00DC         SCNTA=0
00E2         NOROLLA=0
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                          PAGE 0004

00E7 C        TURN ON SCALE A
00E7          PORTB=(PORTB.OR.01H)
00EC          OUTPUT(229)=PORTB
00EF          SYSTA=1
00F3     20   IF(SYSTB)21,21,30
00F7     21   IF(SCNTB.LT.TIMEB)GO TO 30
010B          SCNTB=0
0111          NOROLLB=0
0116 C        TURN ON SCALE B
0116          PORTC=(PORTC.OR.01H)
011B          OUTPUT(230)=PORTC
011E          SYSTB=1
0122     30   ENABLE
0123          RETURN
0129          END

PROGRAM STORAGE= 0240

VARIABLE STORAGE= 0000


SYMBOL TABLE

0122 30
00FD 21
00F3 20
00CE 11
00C4 10
009C 5
0054 1
```

Figure C-3. (Continued)

```
MTI FORT//80 COMPILER V3.3C                    PAGE 0005

0129 C
0129          SUBROUTINE ARUN
0129          INLINE/0C5H,0D5H,0E5H,0F5H/
012D C        IF SCALE A TURNED OFF, SET RESTART=1,
012D C        SYSTA=1, AND RETURN
012D    300   IF((INPUT(228).AND.01H).EQ.0)GO TO 490
0136          IF(RESTARTA.EQ.0)GO TO 346
013E          ENDCOMA=0
0143          SCNTA=0
0149          TIMEA=0
014C          RESTARTA=0
0151          NOROLLA=1
0155          SYSTA=0
0158    346   IF(NOROLLA.NE.0)GO TO 495
0160 C        IF FEED COMPLETE, SKIP ROLL RUN PART
0160          IF(ENDCOMA.NE.0)GO TO 400
0168 C        INPUT WEIGHT OF A
0168          ADCM=1
016D    350   CONTINUE
016D          IF((ADCC.AND.80H).EQ.0)GO TO 350
0177          WTIA=ADCD
017D C        IF WTIA<BDIA, TURN ON BULK
017D          IF(WTIA-BDIA)370,380,380
018C    370   PORTB=PORTB.OR.20H
0197          OUTPUT(229)=PORTB
019C          GO TO 495
019F C        IF WTIA<MISIA, TURN ON DRIBBLE AND TURN OFF BULK
019F    380   IF(WTIA-MISIA)385,390,390
01AE    385   PORTB=(PORTB.OR.40H).AND.0DFH
01BB          OUTPUT(229)=PORTB
01C0          GO TO 495
01C3    390   PORTB=PORTB.AND.01FH
01CB          OUTPUT(229)=PORTB
01D0 C        SET UP 5 SECOND DELAY BEFORE CALCULATIONS
01D0          ENDCALCA=SCNTA+50
01DA          ENDCOMA=1
01DF C        CHECK IF TIME TO CALCULATE
01DF    400   IF(ENDCALCA.GT.SCNTA)GO TO 495
01ED C        THIS PORTION CALCULATES TIME NEEDED AND SETS
01ED C        ALARMS
01ED          SYSTA=1
01F2          ENDCOMA=0
01F6          NOROLLA=1
01FA C        INPUT WEIGHT VALUE
01FA          ADCM=1
01FF    410   CONTINUE
01FF          IF((ADCC.AND.80H).EQ.0)GO TO 410
0209          WTA=ADCD
0213 C        TURN OFF SCALE A TO ALLOW IT TO DUMP
0213          PORTB=PORTB.AND.0FEH
021C          OUTPUT(229)=PORTB
0221 C        CALCULATE THE TIME NEEDED
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0006

0221          TIMEA=((WTA/STPA)*COUNTA)
0233 C        CHECK FOR ALARMS
0233          IF(TIMEA.LT.LOA)ALARMA=ALARMA.OR.02H
0249          IF(TIMEA.GT.HIA)ALARMA=ALARMA.OR.04H
025F          IF(TIMEA.LT.HIHIA)GO TO 470
026D          ALARMA=ALARMA.OR.08H
0275          TIMEA=0A
027B    470   IF(ALARMA.EQ.0)GO TO 480
0283          PORTB=PORTB.OR.02H
028B          OUTPUT(229)=PORTB
0290    480   SYSTA=0
0295          GO TO 495
0298    490   PORTB=PORTB.AND.02H
02A0          OUTPUT(229)=PORTB
02A5          NOROLLA=0
02AA          RESTARTA=1
02AE          SYSTA=1
02B2    495   INLINE/0F1H,0E1H,0D1H,0C1H/
02B6          RETURN
02B7          END

PROGRAM STORAGE= 0398

VARIABLE STORAGE= 0000


SYMBOL TABLE

0290 480
027B 470
01FF 410
01C3 390
01B1 385
019F 380
018F 370
016D 350
01DF 400
02B2 495
0158 346
0298 490
012D 300
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0007


02B7 C
02B7 C
02B7          SUBROUTINE BRUN
02B7          INLINE/0C5H,0D5H,0E5H,0F5H/
02BB C        IF SCALE B TURNED OFF, SET RESTARTB AND SYSTB
02BB C        TO 1 AND RETURN
02BB    600   IF((INPUT(228).AND.10H).EQ.0H)GO TO 790
02C4          IF(RESTARTB.EQ.0)GO TO 646
02CC          ENDCOMB=0
02D1          SCNTB=0
02D7          TIMEB=0
02DA          RESTARTB=0
02DF          NOROLLB=1
02E3          SYSTB=0
02E6 C        IF WAITING FOR SCALE START, RETURN
02E6    646   IF(NOROLLB.NE.0)GO TO 795
02EE C        IF FEED COMPLETE, SKIP SCALE FEED PART
02EE          IF(ENDCOMB.NE.0)GO TO 700
02F6 C        INPUT WEIGHT FOR B
02F6          ADCM=3
02FB    650   CONTINUE
02FB          IF((ADCC.AND.80H).EQ.0)GO TO 650
0305          WTIB=ADCD
030B C        IF WTIB<BDIB, TURN ON BULK
030B          IF(WTIB-BDIB)670,680,680
031A    670   PORTC=PORTC.OR.20H
0325          OUTPUT(230)=PORTC
032A          GO TO 795
032D C        IF WTIB<MISIB, TURN ON DRIBBLE AND TURN OFF BULK
032D    680   IF(WTIB-MISIB)685,690,690
033C    685   PORTC=((PORTC.OR.40H).AND.0DFH)
0349          OUTPUT(230)=PORTC
034E          GO TO 795
0351 C        IF WTIB> OR =MISIB, TURN OFF DRIBBLE
0351    690   PORTC=PORTC.AND.01FH
0359          OUTPUT(230)=PORTC
035E C        SET UP 5 SECOND TIME DELAY BEFORE CALCULATIONS
035E          ENDCALCB=SCNTB+50
0368          ENDCOMB=1
036D C        IS IT READY FOR CALCULATIONS
036D    700   IF(ENDCALCB.GT.SCNTB)GO TO 795
037B C        THIS PORTION CALCULATES TIME NEEDED AND SETS
037B C        ALARMS
037B          SYSTB=1
0380          ENDCOMB=0
0384          NOROLLB=1
0388 C        INPUT WEIGHT OF B
0388          ADCM=3
038D    710   CONTINUE
038D          IF((ADCC.AND.80H).EQ.0)GO TO 710
0397          WTB=ADCD
03A1 C        TURN OFF SCALE B TO ALLOW DISCHARGE
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0008

03A1          PORTC=PORTC.AND.0FEH
03AA          OUTPUT(230)=PORTC
03AF  C       CALCULATE TIME REQUIRED
03AF          TIMEB=((WTB/STPB)*COUNTB)
03C1  C       CHECK FOR ALARMS
03C1          IF(TIMEB.LT.LOB)ALARMB=ALARMB.OR.02H
03D7          IF(TIMEB.GT.HIB)ALARMB=ALARMB.OR.04H
03ED          IF(TIMEB.LT.HIHIB)GO TO 770
03FB          ALARMB=ALARMB.OR.08H
0403          TIMEB=0B
0409    770   IF(ALARMB.EQ.0)GO TO 780
0411          PORTC=PORTC.OR.02H
0419          OUTPUT(230)=PORTC
041E    780   SYSTB=0
0423          GO TO 795
0426    790   PORTC=PORTC.AND.02H
042E          OUTPUT(230)=PORTC
0433          NOROLLB=0
0438          RESTARTB=1
043C          SYSTB=1
0440    795   INLINE/0F1H,0E1H,0D1H,0C1H/
0444          RETURN
0445          END


PROGRAM STORAGE= 0398

VARIABLE STORAGE= 0000


SYMBOL TABLE

041E 780
0409 770
038D 710
0351 690
033F 685
032D 680
031D 670
02FB 650
036D 700
0440 795
02E6 646
0426 790
02BB 600
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0009

0445 C
0445 C
0445          SUBROUTINE COUT
0445 C        THIS ROUTINE TAKES VARIABLE LOCATED IN THE
0445 C        C REGISTER AND PLACES IT TO THE TTY. IF THE
0445 C        STATUS BIT IS NOT SET THE ROUTINE WILL CALL
0445 C        THE SCALE RUN PROGRAMS WHILE WAITING.
0445 C        CONTENTS OF A AND C REGISTER IS DESTROYED.
0445 C
0445    100   INLINE/0F5H/
0446    110   IF((INPUT(0EDH).AND.01H).NE.0)GO TO 120
044F          IF(OBLANK.NE.0)GO TO 115
0457          CALL ARUN
045A          CALL BRUN
045D    115   GO TO 110
0460    120   INLINE/79H,32H,ADDRESS(DUMMY)/
0464          OUTPUT(0ECH)=DUMMY
0469          INLINE/0F1H/
046A          RETURN
046B          END

PROGRAM STORAGE= 0038

VARIABLE STORAGE= 0000


SYMBOL TABLE

045D 115
0460 120
0446 110
0445 100
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0010

046B C
046B          SUBROUTINE CINP
046B C        THIS ROUTINE TAKES KEYPRESSED FROM TTY AND
046B C        INPUTS INTO BUFFER IF CARRD.  CONTENTS OF
046B C        A REGISTER DESTROYED.
046B    125   INLINE/0C5H/
046C          IF(CARRYOFF)GO TO 130
046F          CALL COUT
0472          GO TO 145
0475    130   IF((INPUT(0EDH).AND.02H).NE.0)GO TO 140
047E          IF(IBLANK.NE.0)GO TO 135
0486          CALL ARUN
0489          CALL BRUN
048C    135   GO TO 130
048F    140   INLINE/0DBH,0ECH,0E6H,07FH/
0493    145   INLINE/0C1H/
0494          RETURN
0495          END

PROGRAM STORAGE= 0042

VARIABLE STORAGE= 0000


SYMBOL TABLE

048C 135
048F 140
0493 145
0475 130
046B 125
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0011

0495 C
0495 C
0495          SUBROUTINE INITIALIZE
0495 C        THIS ROUTINE INITIALIZES ALL THE PARALLEL
0495 C        I/O, ANALOG I/O, SERIAL I/O, AND SYSTEM CONSTANTS.
0495 C
0495    200   DISABLE
0496 C        INITIALIZE SERIAL PORT
0496 C        20 MA CURRENT LOOP, 300 BAUD, PARITY OFF,
0496 C        # OF BITS=8, #OF STOP BITS=1
0496          OUTPUT(0EDH)=0CFH
049A          OUTPUT(0EDH)=025H
049E          INLINE/21H,ADDRESS(COUT),22H,ADDRESS(CO)/
04A4          INLINE/21H,ADDRESS(CINP),22H,ADDRESS(CI)/
04AA C        INITIALIZE PARALLEL I/O PORTS
04AA C        PORT 0E4H  INPUT    BIT #0 = SCALE A ON/OFF OR RESTART
04AA C             (228)                1 = NOT USED
04AA C                                  2 = NOT USED
04AA C                                  3 = NOT USED
04AA C                                  4 = SCALE B ON/OFF OR RESTART
04AA C                                  5 = NOT USED
04AA C                                  6 = NOT USED
04AA C                                  7 = NOT USED
04AA C        SCALE A
04AA C        PORT 0E5H  OUTPUT   BIT #0 = SCALE A ON/OFF
04AA C             (229)                1 = ALARM HORN
04AA C        SCALE B                   2 = NOT USED
04AA C        PORT 0E6H                 3 = NOT USED
04AA C             (230)                4 = NOT USED
04AA C                                  5 = BULK SPEED
04AA C                                  6 = DRIBBLE SPEED
04AA C                                  7 = NOT USED
04AA C
04AA          OUTPUT(231)=090H
04AE          OUTPUT(229)=0
04B2          OUTPUT(230)=0
04B6 C
04B6 C        INITIALIZE A/D
04B6 C        THIS INITIALIZES BOARD SO THAT WHEN:
04B6 C        ADCM=1, THEN ADCD=WEIGHT OF SCALE A
04B6 C        ADCM=3, THEN ADCD=WEIGHT OF SCALE B
04B6 C
04B6          ADCC=018H
04BB C
04BB C        INITIALIZE CONSTANTS
04BB          DUMMY=0
04C1          IBLANK=0
04C6          OBLANK=0
04C9          STAT=0
04CD          SKIPWHAT=0
04D1          SKIPYN=0
04D4          CHIN(1)=0
```

Figure C-3. (Continued)

```
04D8          SCALE=0
04DB          WHICH=0
04DE          LETTER=0
04E4          NEWVAL=0
04E7          MA=90
04EC          BA=70
04EF          CA=450
04F5          SA=0
04FB          SCNTA=0
04FE          COUNTA=450.0
0508          TIMEA=0
050F          SYSTA=0
0514          NOROLLA=0
0517          RESTARTA=0
051B          ENDCOMA=0
051E          SRAA=0
0522          PORTB=0
0525          ALARMA=0
0529          ENDCALCA=0
052F          MISA=0.0
0539          BDA=0.0
0540          STPA=0.1
0547          MISIA=0
054E          BDIA=0
0551          LOA=400
0557          HIA=500
055C          HIHIA=900
0562          ROLLA=300
0568          MB=90
056D          BB=70
0570          CB=450
0576          SB=0
057C          SCNTB=0
057F          COUNTB=450.0
0589          SYSTB=0
058F          NOROLLB=0
0592          RESTARTB=0
0596          ENDCOMB=0
0599          SRAB=0
059D          PORTC=0
05A0          ALARMB=0
05A4          ENDCALCB=0
05AA          MISB=0.0
05B4          BDB=0.0
05DB          STPB=0.1
05C2          MISIB=0
05C9          BDIB=0
05CC          LOB=400
05D2          HIB=500
05D7          HIHIB=900
05DD          ROLLB=300
05E3          ENABLE
```

Figure C-3. (Continued)

```
UTJ FORT//80 COMPILER V3.3C                    PAGE 0013

05E4          RETURN
05E5          END

PROGRAM STORAGE= 0336

VARIABLE STORAGE= 0000


SYMBOL TABLE

1A70 0.1
1A6C 0.0
1A68 450.0
0495 200
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0014

05E5 C
05E5 C
05E5          SUBROUTINE PRLIST
05E5 C        OUTPUTS VALUES OF CONSTANTS FOR SCALE CHOSEN
05E5 C        TO TTY
05E5   1200   IF(SCALE.EQ.041H)LETTER=MA
05F7          IF(SCALE.EQ.042H)LETTER=MB
0609          WRITE(CO,1220)LETTER
0617   1220   FORMAT(' MATERIAL-IN-SUSPENSION = ',I2,'%.')
0617          IF(SCALE.EQ.041H)LETTER=BA
0629          IF(SCALE.EQ.042H)LETTER=BB
063B          WRITE(CO,1230)LETTER
0649   1230   FORMAT(' BULK/DRIBBLE = ',I2,'%.')
0649          IF(SCALE.EQ.041H)LETTER=CA
0657          IF(SCALE.EQ.042H)LETTER=CB
0665          WRITE(CO,1240)LETTER
0673   1240   FORMAT(' CYCLE TIME = ',I3,' TENTHS OF SECONDS.')
0673          IF(SCALE.EQ.041H)LETTER=SA
0681          IF(SCALE.EQ.042H)LETTER=SB
068F          WRITE(CO,1250)LETTER
069D   1250   FORMAT(' SETPOINT = ',I3,' POUNDS.')
069D          RETURN
069E          END

PROGRAM STORAGE= 0324

VARIABLE STORAGE= 0000


SYMBOL TABLE

05E5 1200
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0015

069E C
069E C
069E          SUBROUTINE HORNA
069E C        THIS ROUTINE OUTPUTS ALARMS FOR SCALE A
069E C        ALARMA          BIT # 0 = 1, SLOW ROLL SPEED
069E C                                1 = 1, LOW WEIGHT
069E C                                2 = 1, HIGH WEIGHT
069E C                                3 = 1, MALFUNCTION
069E  1300    IF(ALARMA.NE.0)GO TO 1304
06A6          WRITE(CO,1301)
06AE  1301    FORMAT(' NO ALARMS')
06BE          GO TO 1340
06B1  1304    IF((ALARMA.AND.01H).EQ.0)GO TO 1310
06BB          WRITE(CO,1305)
06C3  1305    FORMAT(' ROLL SPEED TOO SLOW.')
06C3  1310    IF((ALARMA.AND.02H).EQ.0)GO TO 1320
06CD          LETTER=(WTA/4095.0)*200.0
06DF          WRITE(CO,1315)LETTER
06ED  1315    FORMAT(' LOW WEIGHT = ',I3,' LBS.')
06ED  1320    IF((ALARMA.AND.04H).EQ.0)GO TO 1330
06F7          LETTER=(WTA/4095.0)*200.0
0709          WRITE(CO,1325)LETTER
0717  1325    FORMAT(' HIGH WEIGHT = ',I3,' LBS.')
0717  1330    IF((ALARMA.AND.08H).EQ.0)GO TO 1340
0721          WRITE(CO,1335)
0729  1335    FORMAT(' SCALE MALFUNCTION.')
0729  1340    PORTB=PORTB.AND.0FDH
0731          OUTPUT(229)=PORTB
0736          SRAA=0
073B          ALARMA=0
073E          RETURN
073F          END

PROGRAM STORAGE= 0288

VARIABLE STORAGE= 0000


SYMBOL TABLE

0717 1330
1B2E 200.0
1B2A 4095.0
06ED 1320
06C3 1310
0729 1340
06B1 1304
069E 1300
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0016

073F C
073F C
073F          SUBROUTINE HORNB
073F C        THIS ROUTINE OUTPUTS ALARMS FOR SCALE B TO TTY
073F C        ALARMB      BIT # 0 = 1, ROLLS TOO SLOW
073F C                            1 = 1, LOW WEIGHT
073F C                            2 = 1, HIGH WEIGHT
073F C                            3 = 1, MALFUNCTION
073F  1400    IF(ALARMB.NE.0)GO TO 1404
0747          WRITE(CO,1401)
074F  1401    FORMAT(' NO ALARMS')
074F          GO TO 1440
0752  1404    IF((ALARMB.AND.01H).EQ.0)GO TO 1410
075C          WRITE(CO,1405)
0764  1405    FORMAT(' ROLL SPEED TOO SLOW.')
0764  1410    IF((ALARMB.AND.02H).EQ.0)GO TO 1420
076E          LETTER=(WTB/4095.0)*200.0
0780          WRITE(CO,1415)LETTER
078E  1415    FORMAT(' LOW WEIGHT = ',I3,' LBS.')
078E  1420    IF((ALARMB.AND.04H).EQ.0)GO TO 1430
0798          LETTER=(WTB/4095.0)*200.0
07AA          WRITE(CO,1425)LETTER
07B8  1425    FORMAT(' HIGH WEIGHT = ',I3,' LBS.')
07B8  1430    IF((ALARMB.AND.08H).EQ.0)GO TO 1440
07C2          WRITE(CO,1435)
07CA  1435    FORMAT(' SCALE MALFUNCTION')
07CA  1440    PORTC=PORTC.AND.0FDH
07D2          OUTPUT(230)=PORTC
07D7          GRAB=0
07DC          ALARMB=0
07DF          RETURN
07E0          END

PROGRAM STORAGE= 0287

VARIABLE STORAGE= 0000


SYMBOL TABLE

07B8 1430
078E 1420
0764 1410
07CA 1440
0752 1404
073F 1400
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0017

07E0 C
07E0 C
07E0          SUBROUTINE CHOOSE
07E0 C        THIS ROUTINE DECIDES THE SCALE WHICH WAS
07E0 C        WAS REQUESTED FROM THE TTY
07E0  1000    READ(CI,ERR=1025)STRING(CHIN,1)
07F3 C        IF CHARACTER IS CONTROL A, SCALE A CHOSEN
07F3          IF(CHIN(1).NE.01H)GO TO 1020
07FC          SCALE=41H
07FF  1005    WRITE(CO,1010)SCALE
080D  1010    FORMAT(' SCALE ',A1,' CHOSEN.')
080D  1015    RETURN
080E  1020    IF(CHIN(1).NE.02H)GO TO 1015
0817          SCALE=042H
081A          GO TO 1005
081D          RETURN
081E  1025    WRITE(CO,1030)
0826  1030    FORMAT(' INCORRECT ENTRY.  ENTER AGAIN.  ')
0826          GO TO 1000
0829          END

PROGRAM STORAGE= 0137

VARIABLE STORAGE= 0000


SYMBOL TABLE

080D 1015
07FF 1005
080E 1020
081E 1025
07E0 1000
```

Figure C-3. (Continued)

```
TI FORT/99 COMPILER V3.30                    PAGE 0018

0829 C
0829 C
0829          SUBROUTINE CALIBRATE
0829 C        THIS ROUTINE INPUTS THE PRESENT VALUE OF CHOSEN
0829 C        SCALE'S ANALOG INPUT AND WRITES IT TO TTY.
0829    100   IF(SCALE.EQ.42H)GO TO 120
0831          IF(SCALE.NE.41H)GO TO 150
0839          ADCM=1
083E    110   CONTINUE
083E          IF((ADCC.AND.80H).EQ.0)GO TO 110
0848          GO TO 140
084B    120   ADCM=3
0850    130   CONTINUE
0850          IF((ADCC.AND.80H).EQ.0)GO TO 130
085A    140   LETTER=ADCD
0860          WRITE(CO,145)LETTER
086E    145   FORMAT(' ANALOG INPUT= ',I5)
086E    150   RETURN
086F          END

PROGRAM STORAGE= 0093

VARIABLE STORAGE= 0000


SYMBOL TABLE

0850 130
085A 140
083E 110
086E 150
084B 120
0829 100
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0019

086F C
086F C
086F          SUBROUTINE WHAT
086F C        THIS ROUTINE DECIDES WHAT FUNCTION THE OPERATOR
086F C        HAS REQUESTED FROM THE TTY
086F  1025    IF(SCALE.EQ.0)GO TO 1198
0877 C        CHECK FOR MIS
0877  1026    READ(CI,ERR=1096)STRING(CHIN,1)
088A  1030    IF(CHIN(1).NE.05H)GO TO 1050
0893          WHICH=CHIN(1)
0897          IF(SCALE.EQ.41H)LETTER=MA
08A8          IF(SCALE.EQ.42H)LETTER=MB
08BA  1035    WRITE(CO,1040)LETTER
08C9  1040    FORMAT(' THE PRESENT VALUE IS ',I2,'%.',/,
08C8        . ' DO YOU WANT TO CHANGE IT? ')
08C8          GO TO 1095
08CB C        CHECK FOR BDS
08CB  1050    IF(CHIN(1).NE.04H)GO TO 1060
08D4          WHICH=CHIN(1)
08D8          IF(SCALE.EQ.41H)LETTER=BA
08E9          IF(SCALE.EQ.42H)LETTER=BB
08FB          GO TO 1035
08FE C        CHECK FOR CYCLE TIME
08FE  1060    IF(CHIN(1).NE.14H)GO TO 1080
0907          WHICH=CHIN(1)
090B          IF(SCALE.EQ.41H)LETTER=CA
0918          IF(SCALE.EQ.42H)LETTER=CB
0926  1065    WRITE(CO,1070)LETTER
0934  1070    FORMAT(' THE PRESENT VALUE IS ',I3,' TENTHS ',
0934        . 'OF SECONDS.',/,' DO YOU WANT TO CHANGE IT? ')
0934          GO TO 1095
0937 C        CHECK FOR SETPOINT
0937  1080    IF(CHIN(1).NE.13H)GO TO 1100
0940          WHICH=CHIN(1)
0944          IF(SCALE.EQ.41H)LETTER=SA
0951          IF(SCALE.EQ.42H)LETTER=SB
095F  1085    WRITE(CO,1090)LETTER
096D  1090    FORMAT(' THE PRESENT VALUE IS ',I3,' POUNDS.',/,
096D        . ' DO YOU WANT TO CHANGE IT? ')
096D  1095    SKIPWHAT=1
0972          RETURN
0973  1096    WRITE(CO,1097)
097B  1097    FORMAT(' INCORRECT ENTRY.  ENTER AGAIN.  ')
097B          GO TO 1026
097E C        CHECK FOR LIST
097E  1100    IF(CHIN(1).NE.0CH)GO TO 1110
0987          CALL PRLIST
098A          SCALE=0
098F          RETURN
0990 C        THIS ROUTINE OUTPUTS ALARMS
0990  1110    IF(CHIN(1).NE.08H)GO TO 1130
0999          IF(SCALE.NE.42H)GO TO 1120
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0020

09A0          CALL HORNB
09A3          SCALE=0
09A8          RETURN
09A9   1120   IF(SCALE.NE.41H)GO TO 1130
09B1          CALL HORNA
09B4          SCALE=0
09B9          RETURN
09BA C        CHECK FOR REQUEST TO CALIBRATE.
09BA   1130   IF(CHIN(1).NE.01AH)GO TO 1195
09C3          WRITE(CO,1135)
09CB   1135   FORMAT(' DO YOU WANT TO CHECK CALIBRATION?')
09CB   1136   READ(CI,ERR=1140)STRING(CHIN,1)
09DE          IF(CHIN(1).EQ.4EH)GO TO 1150
09E7          IF(CHIN(1).NE.59H)GO TO 1140
09ED          CALL CALIBRATE
09F0          GO TO 1160
09F3   1140   WRITE(CO,1145)
09FB   1145   FORMAT(' INCORRECT ENTRY.  ANSWER AGAIN.')
09FB          GO TO 1136
09FE   1150   WRITE(CO,1155)
0A06   1155   FORMAT(' GOODBYE.')
0A06   1160   SCALE=0
0A0B          RETURN
0A0C   1195   WRITE(CO,1196)
0A14   1196   FORMAT(' ENTER COMMAND AGAIN, PLEASE.')
0A14   1198   RETURN
0A15          END

PROGRAM STORAGE= 0806

VARIABLE STORAGE= 0000


SYMBOL TABLE

0A06 1160
09FE 1150
09F3 1140
09CB 1136
0A0C 1195
09A9 1120
09BA 1130
0990 1110
095F 1085
097E 1100
0926 1065
0937 1080
08FE 1060
096D 1095
08BA 1035
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0021

08CB 1050
088A 1030
0973 1096
0877 1026
0A14 1198
086F 1025
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0022

0A15 C
0A15 C
0A15          SUBROUTINE YN
0A15 C        THIS ROUTINE CHECKS ANSWER OF QUESTION.
0A15   2000   READ(CI:ERR=2065)STRING(CHIN,1)
0A28          IF(CHIN(1).NE.04EH)GO TO 2020
0A31          SKIPWHAT=0
0A35          SCALE=0
0A39          WHICH=0
0A3C          WRITE(CO,2010)
0A44   2010   FORMAT(' GOODBYE.')
0A44          RETURN
0A45   2020   IF(CHIN(1).EQ.059H)GO TO 2040
0A4E          WRITE(CO,2030)
0A56   2030   FORMAT(' ANSWER AGAIN, PLEASE.')
0A56          RETURN
0A57   2040   WRITE(CO,2050)
0A5F   2050   FORMAT(' ENTER NEW VALUE OF CONSTANT.  ')
0A5F   2060   SKIPYN=1
0A64          RETURN
0A65   2065   WRITE(CO,2066)
0A6D   2066   FORMAT(' INCORRECT ENTRY.  ENTER AGAIN.  ')
0A6D          GO TO 2000
0A70          END

PROGRAM STORAGE= 0210

VARIABLE STORAGE= 0000


SYMBOL TABLE

0A5F 2060
0A57 2040
0A45 2020
0A65 2065
0A15 2000
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.30                    PAGE 0023

0A70 C
0A70 C
0A70          SUBROUTINE FACTOR
0A70 C        THIS ROUTINE COMPUTES THE NEW VALUE
0A70  4000    RENEW=NEWVAL
0A7A          IF(SCALE.EQ.42H)GO TO 4050
0A83          IF(SCALE.NE.41H)RETURN
0A89          IF(WHICH.NE.05H)GO TO 4010
0A91          MISIA=(RENEW/100.0)*STPA
0AA3          MA=NEWVAL
0AA9          RETURN
0AAA  4010    IF(WHICH.NE.04H)GO TO 4020
0AB2          BDIA=(RENEW/100.0)*STPA
0AC4          BA=NEWVAL
0ACA          RETURN
0ACB  4020    IF(WHICH.NE.14H)GO TO 4030
0AD3          COUNTA=RENEW
0ADD          LOA=0.90*RENEW
0AE8          HIA=1.10*RENEW
0AF6          HIHIA=2.0*RENEW
0B04          ROLLA=0.75*RENEW
0B12          CA=NEWVAL
0B18          RETURN
0B19  4030    IF(WHICH.NE.13H)GO TO 4040
0B21          STPA=(RENEW/200.0)*4095.0
0B33          MISA=MA
0B42          MISIA=(MISA/100.0)*STPA
0B51          BDA=BA
0B5F          BDIA=(BDA/100.0)*STPA
0B6E          SA=NEWVAL
0B74  4040    RETURN
0B75  4050    IF(WHICH.NE.05H)GO TO 4060
0B7D          MISIB=(RENEW/100.0)*STPB
0B8F          MB=NEWVAL
0B95          RETURN
0B96  4060    IF(WHICH.NE.04H)GO TO 4070
0B9E          BDIB=(RENEW/100.0)*STPB
0BB0          BB=NEWVAL
0BB6          RETURN
0BB7  4070    IF(WHICH.NE.14H)GO TO 4080
0BBF          COUNTB=RENEW
0BC9          LOB=0.9*RENEW
0BD4          HIB=1.1*RENEW
0BE2          HIHIB=2.0*RENEW
0BF0          ROLLB=0.75*RENEW
0BFE          CB=NEWVAL
0C04          RETURN
0C05  4080    IF(WHICH.NE.13H)GO TO 4090
0C0D          STPB=(RENEW/200.0)*4095.0
0C1F          MISB=MB
0C2E          MISIB=(MISB/100.0)*STPB
0C3D          BDB=BB
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0024

0C4B            BDIB=(BDB/100.0)*STPB
0C5A            SB=NEWVAL
0C60   4090     RETURN
0C61            END

PROGRAM STORAGE= 0497

VARIABLE STORAGE= 0000


SYMBOL TABLE

0C60 4090
1E6A 1.1
1E66 0.9
0C05 4080
0BB7 4070
0B96 4060
0B74 4040
1E62 0.75
1E5E 2.0
1E5A 1.10
1E56 0.90
0B19 4030
0ACB 4020
1E52 100.0
0AAA 4010
0B75 4050
0A70 4000
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0025

0C61 C
0C61 C
0C61          SUBROUTINE VALIN
0C61 C        THIS ROUTINE GETS NEW VALUE OF CONSTANT
0C61   3000   READ(CI,ERR=3050)NEWVAL
0C72   3005   WRITE(CO,3010)NEWVAL
0C80   3010   FORMAT(' NEW VALUE IS ',I5,'.  IS THIS CORRECT? ')
0C80   3011   READ(CI,ERR=3060)STRING(CHIN,1)
0C93          IF(CHIN(1).NE.59H)GO TO 3030
0C9C          CALL FACTOR
0C9F          RETURN
0CA0   3030   WRITE(CO,3040)
0CA8   3040   FORMAT(' REENTER NEW VALUE, PLEASE.  ')
0CA8          GO TO 3000
0CAB   3050   WRITE(CO,3055)
0CB3   3055   FORMAT(' INCORRECT ENTRY.  ENTER AGAIN.  ')
0CB3          GO TO 3000
0CB6   3060   WRITE(CO,3065)
0CBE   3065   FORMAT(' INCORRECT ENTRY.  ENTER AGAIN.  ')
0CBE          GO TO 3011
0CC1          END

PROGRAM STORAGE= 0253

VARIABLE STORAGE= 0000


SYMBOL TABLE

0CA0 3030
0CB6 3060
0C80 3011
0C72 3005
0CAB 3050
0C61 3000
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0026

0CC1 C
0CC1 C
0CC1 C       MAIN PROGRAM
0CC1  5000   CALL INITIALIZE
0CC4  5010   CALL ARUN
0CC7         CALL BRUN
0CCA         STAT=(INPUT(0E0H).AND.02H)
0CD1         IF(STAT.EQ.0)GO TO 5010
0CD9         IF(SCALE.NE.0)GO TO 5050
0CE1         CALL CHOOSE
0CE4         GO TO 5010
0CE7  5050   IF(SKIPWHAT.NE.0)GO TO 5100
0CEF         CALL WHAT
0CF2         GO TO 5010
0CF5  5100   IF(SKIPYN.NE.0)GO TO 5200
0CFD         CALL YN
0D00         GO TO 5010
0D03  5200   CALL VALIN
0D06         CALL PRLIST
0D09         IBLANK=0
0D0E         OBLANK=0
0D11         SKIPWHAT=0
0D15         SKIPYN=0
0D18         CHIN(1)=0
0D1C         SCALE=0
0D1F         WHICH=0
0D22         LETTER=0
0D28         NEWVAL=0
0D2B         GO TO 5010
0D2E         STOP
0D31         END

PROGRAM STORAGE= 0112

VARIABLE STORAGE= 0000



TOTAL PROGRAM STORAGE= 7896

TOTAL VARIABLE STORAGE= 0274


SYMBOL TABLE
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0027

0D03 5200
0CF5 5100
0CE7 5050
0CC4 5010
0CC1 5000
0C61 VALIN
1E6A 1.1
1E66 0.9
1E62 0.75
1E5E 2.0
1E5A 1.10
1E56 0.90
1E52 100.0
0A70 FACTOR
0A15 YN
086F WHAT
0829 CALIBRATE
07E0 CHOOSE
073F HORNB
1B2E 200.0
1B2A 4095.0
069E HORNA
05E5 PRLIST
1A70 0.1
1A6C 0.0
1A68 450.0
0495 INITIALIZE
046B CINP
0445 COUT
02B7 BRUN
0129 ARUN
3F7F STPB
3F83 COUNTB
3F87 MISB
3F8B BDB
3F8F WTB
3F93 ROLLB
3F95 HIHIB
3F97 HIB
3F99 LOB
3F9B WTIB
3F9D BDIB
3F9F MISIB
3FA1 SB
3FA3 CB
3FA5 ENDCALCB
3FA7 SCNTB
3FA9 TIMEB
3FAB BB
3FAC MB
3FAD ENDCOMB
3FAE RESTARTB
```

Figure C-3. (Continued)

```
UTI FORT//80 COMPILER V3.3C                    PAGE 0028

3FAF ALARMB
3FB0 SRAB
3FB1 PORTC
3FB2 NOROLLB
3FB3 SYSTB
3FB4 STPA
3FB8 COUNTA
3FBC MISA
3FC0 BDA
3FC4 WTA
3FC8 ROLLA
3FCA HIHIA
3FCC HIA
3FCE LOA
3FD0 WTIA
3FD2 BDIA
3FD4 MISIA
3FD6 SA
3FD8 CA
3FDA ENDCALCA
3FDC SCNTA
3FDE TIMEA
3FE0 BA
3FE1 MA
3FE2 ENDCOMA
3FE3 RESTARTA
3FE4 ALARMA
3FE5 SRAA
3FE6 PORTB
3FE7 NOROLLA
3FE8 SYSTA
3FE9 RENEW
3FED LETTER
3FEF NEWVAL
3FF1 CI
3FF3 CO
3FF5 DUMMY
3FF7 PORTA
3FF8 STAT
3FF9 SKIPYN
3FFA SKIPWHAT
3FFB WHICH
3FFC SCALE
3FFD CHIN
3FFE OBLANK
3FFF IBLANK
F71A ADCD
F718 ADCM
F719 ADCC
119D FLOAT DEBUG
```

Figure C-3. (Continued)

VITA

Everette Millard Eldridge was born in Memphis, Tennessee on June 5, 1951. He attended elementary schools in that city and graduated from Watkins Overton High School in May 1969. The following September, he entered Memphis State University, and in May 1974, he received a Bachelor of Science degree in Electrical Engineering.

In June 1974, he accepted a job with Tennessee Eastman Company, where he is presently employed. In September 1977, he entered Graduate School at The University of Tennessee, Knoxville. He received the Master of Science degree with a major in Electrical Engineering in December 1981.

The author is a Registered Engineer in the State of Tennessee. He is a member of the Tennessee Society of Professional Engineers, the Institute of Electrical and Electronics Engineers, and the Instrument Society of America. He is a Ruling Elder at Preston Hills Presbyterian Church in Kingsport, Tennessee.

He is married to the former Gail Miller of Kingsport. They have one child named Jennifer.