A BASIC DRIVEN MICROPROCESSOR BASED EXPERIMENTATION SYSTEM

A Thesis

Presented to

The Faculty of the Department of Electrical Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

---

by

Tai-Loy Kong

December, 1978

A BASIC DRIVEN MICROPROCESSOR BASED EXPERIMENTATION SYSTEM

An Abstract of a Thesis

Presented to

The Faculty of the Department of Electrical Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

---

by

Tai-Loy Kong

December, 1978

ABSTRACT

This thesis describes a set of programs designed to extend the capability of the BASIC language to allow periodic acquisition of signals and control of devices. These programs allow a system user to define sets of inputs and sets of outputs from a group of signal lines. Output data are converted into the data formats of external devices and input data are converted into a data format compatible with programs written in BASIC. The extensions to BASIC described in this thesis provide a casual programmer with the means to perform data acquisition and control with minimum knowledge of system software.

An application of the software described in this thesis to a system that will test student's proficiency in the use of laboratory equipment is presented and evaluated.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background

This thesis describes the design and the implementation of a microprocessor based digital system that supports the periodic outputing and inputing of data using the BASIC language and provides flexible and simple hardware interfaces which can be configured through program control.

This system may be useful in small scale applications in which the periodic output of control signals and the periodic measurement of responses are required. Such applications may be found in areas such as process control, automated experimentation and automatic testing.

General purpose computer systems using large scale or mini sized computers exist for process control(1), automatic testing(2) and automated experimentation in the laboratory(3,4).

A typical example of a general purpose digital control system is the Texas Instrument model 960 minicomputer with the PCLA(5) software. The PCLA software provides control of hardware lines through a FORTRAN-like high level language which provides statements for outputing and inputing binary data through sets of lines. Another example of a control language is the CONTROL language(6). A process control program written in CONTROL consists of control sets. Each control set tests the values of the data on certain hardware lines and depending on these values performs specified actions. In this system each set is executed when the execution of the previous set is completed.

Computer systems are also used in the implementation of automatic testing equipment. Such systems use programmable digital instruments(7), standard instrumentation interfaces (8) and high level testing languages(9) as basic elements. One example of a testing language is ATLAS(10) which is used for the testing of avionics electronic equipment. Some automatic testing equipment also use modified versions of standard high level languages such as BASIC(11) and FORTRAN (12) as testing languages.

The computer systems mentioned in the previous discussion are too complex and costly for use in small applications. Microprocessors are suitable for small scale applications in such areas such as process control(13,14), automated experimentation(15,16) and automatic testing(17). Since microprocessors are relatively new(introduced in 1971), general microcomputer systems for these applications are not generally available. Individual dedicated systems have to be developed for each specific application. In particular, most high level languages available for microprocessors presently do not provide support to peripheral devices other than standard I/O devices such as teletypes and line printers. A user must write an application program in machine language to control and monitor any other devices. This approach requires the user to do the following:

1. Learn machine language to write the application program.
2. Learn the operation of the hardware interface controllers so as to interface them to his devices.
3. Learn to program in real-time if the output and input are time dependent.

The above efforts are quite time consuming if the user is not familiar with machine language. From the point

of view of the user it would be far better to be able to program the algorithms for generating the control signals and reducing the data obtained from the devices in a high level language.

The system described in this thesis provides a user with a set of hardware lines that can provide digital outputs and receive digital inputs. These lines can be interfaced with external devices such as sensors and actuators. Control of these lines is performed by a program written in BASIC by the user. The system also provides the user with a limited real-time(periodic) input and output capability.

BASIC(18) is chosen as the system language because it is available for most microprocessors, popular among scientific researchers, and is easy to learn and use. BASIC is an interpretive language which simplifies the development and testing of programs. A program can be easily modified and executed without a compilation process. In developing a program, it is possible to stop execution anywhere by inserting a STOP statement in the appropriate location.

1.2 System Features

A typical use of the system described in this thesis is shown in figure 1.1. The system may be interfaced to basic devices such as switches, digital to analog converters and analog to digital converters(19,20) through its parallel I/O lines. For example, if the system is interfaced to an analog to digital converter, some lines of the system would be connected to the 'channel select' and 'start conversion' inputs of the A/D and other lines would be connected to the digital outputs of the A/D. Through program control the user can control these devices and monitor their operation periodically. Controls are applied through output lines and measurements are made by reading data on the input lines.

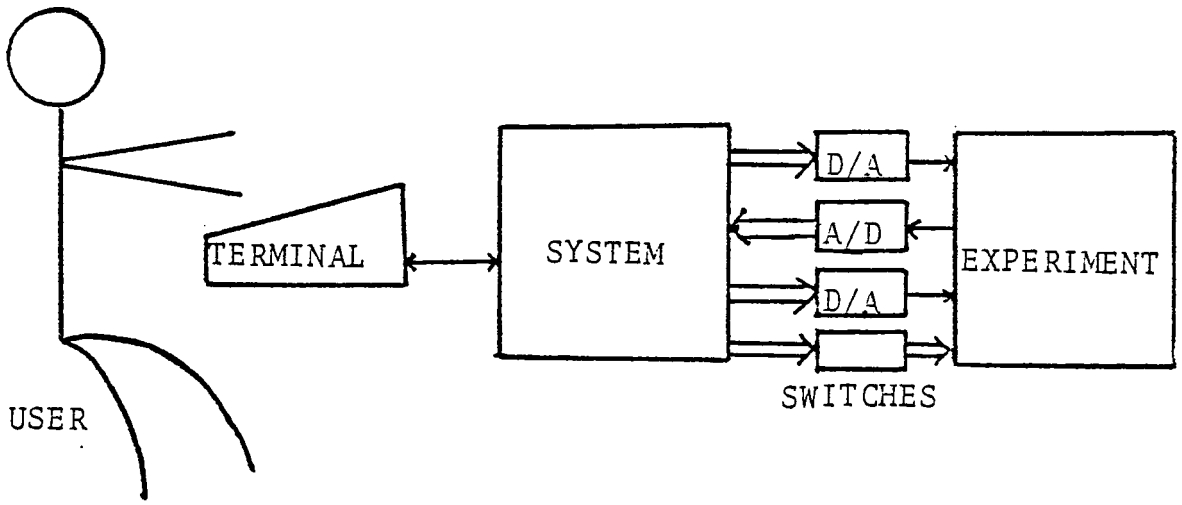


Figure 1.1: Typical Use Of The System

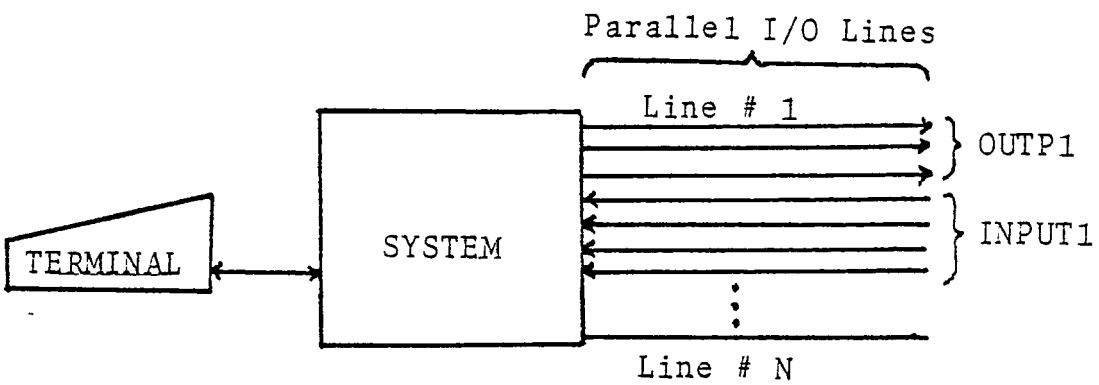


Figure 1.2: Configuration Of I/O Ports

The control and monitoring functions are performed through the use of statements written in BASIC.

The BASIC interpreter used in this system(21) satisfies the minimum standards set for BASIC by ANSI Committee X3J2.

The parallel I/O lines of the system have the following features:

1. The BASIC program controls all I/O functions associated with these lines by executing BASIC statements. There is no need for programming at the machine language level.
2. Output data to be sent to these lines are stored as BASIC array variables and the input data is stored as a BASIC array variable for subsequent use in a program. The system programs perform all the necessary conversion between the hardware formats and the BASIC interpreter format.
3. The user can partition the I/O lines of the system into mutually exclusive subsets,as shown in figure 1.2, to serve as output and input ports. Each of the ports is given a name. All references to a port are made in terms of its name.
4. A port can have one of the following data formats:
   a. Binary using from one to sixteen lines.
   b. Unsigned BCD using from one to sixteen lines.
   c. Signed BCD using from one to seventeen lines.
5. The user can output to or input from these ports periodically. The period may be adjusted from 10 ms to 24 hours in 10 ms increments.
6. The system can perform periodic I/O for a maximum of 6 ports simultaneously.

1.3 Thesis Organization

Chapters 2 and 3 provide information on the use of the system. Chapter 2 provides an overall description of the system operation while chapter 3 describes the specific functions associated with the parallel I/O commands and how they are called through BASIC.

In addition to reading chapters 2 and 3 a user should be familiar with BASIC language by reading a manual(21), and understand the system monitor of the microcomputer in use(22). Chapters 4 through 7 describe the details of how the system is implemented. Appendix A of this thesis describes an application of the system. The system is interfaced to a test box which is capable of generating and receiving electrical signals. This system is used to evaluate the proficiency of students in the use of electronic instruments at the University of Houston. Appendix B presents a listing of the machine language programs described in this thesis.

CHAPTER 2

SYSTEM REQUIREMENTS AND MODES OF OPERATION

This chapter presents the hardware and software blocks of the system. The modes of operation of the system are also described.

2.1 Hardware Requirements

The hardware components of the system are illustrated in figure 2.1. They consist of:

1. Microprocessor
2. A terminal with keyboard entry, display, program storage and program loading capabilities.
3. Serial data interface controller which interfaces the terminal to the microprocessor.
4. Parallel data interface controllers. The data lines of these controllers are programmable as inputs or outputs. The lines of these controllers are used to interface with the user's devices.
5. A timer that generates pulses at 10 ms intervals.
6. ROM(read only memories) for system programs.
7. RAM(random access memories) used as scratch by the system programs and for the BASIC program of the user.

2.2 Software Requirements

The system software all reside in ROM. Three machine language programs are required. The first two programs are standard programs available from the microprocessor family. The third program is written in this thesis to add I/O

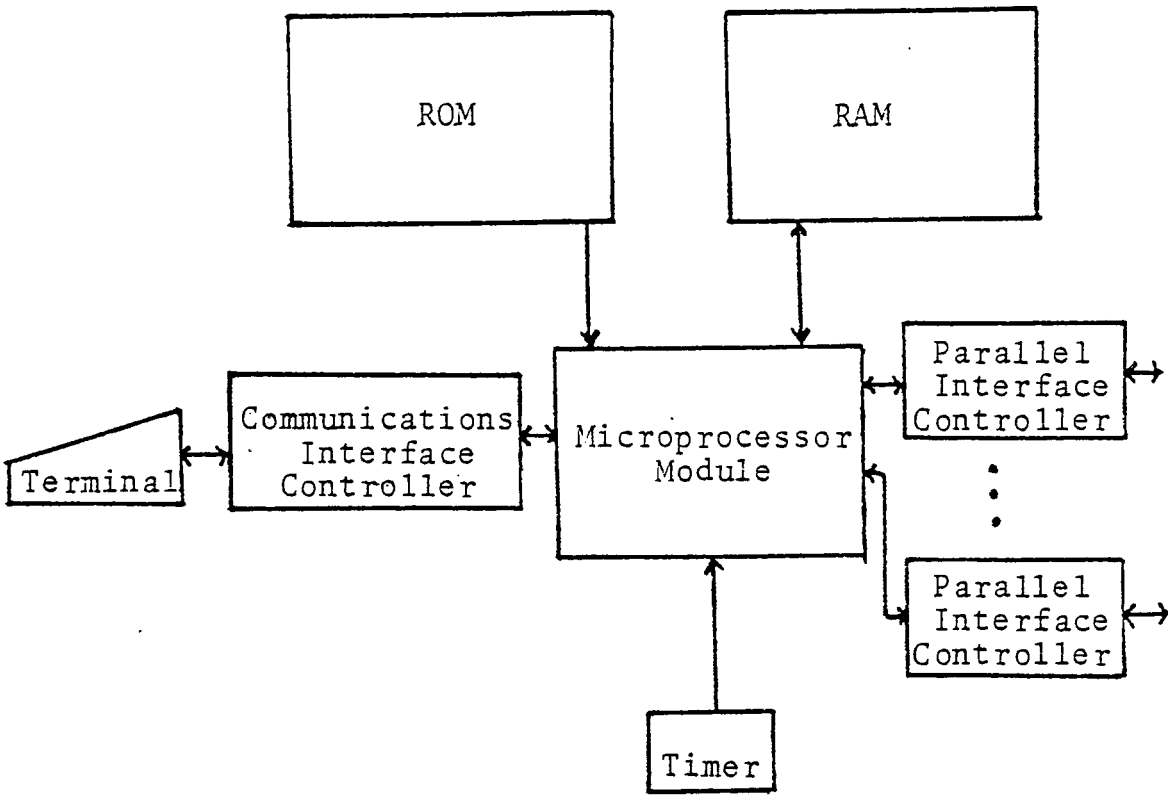


Figure 2.1: Hardware Configuration

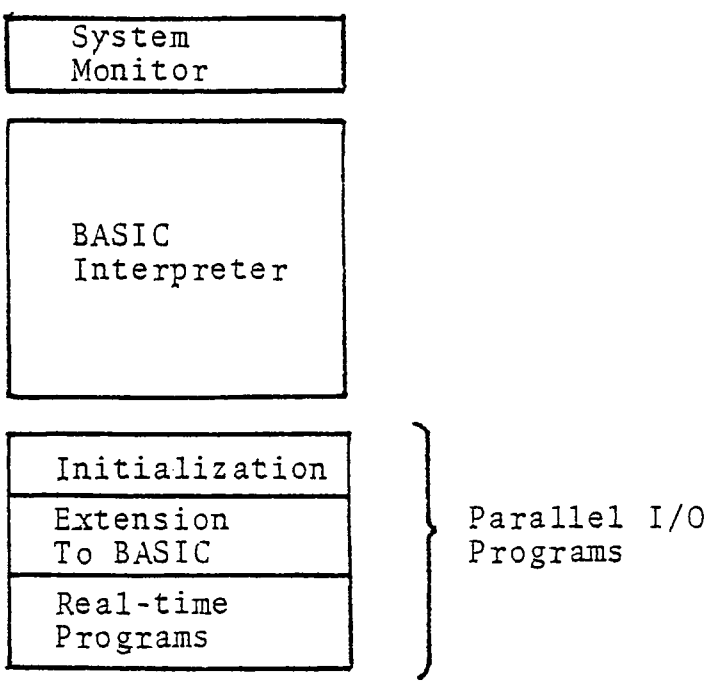


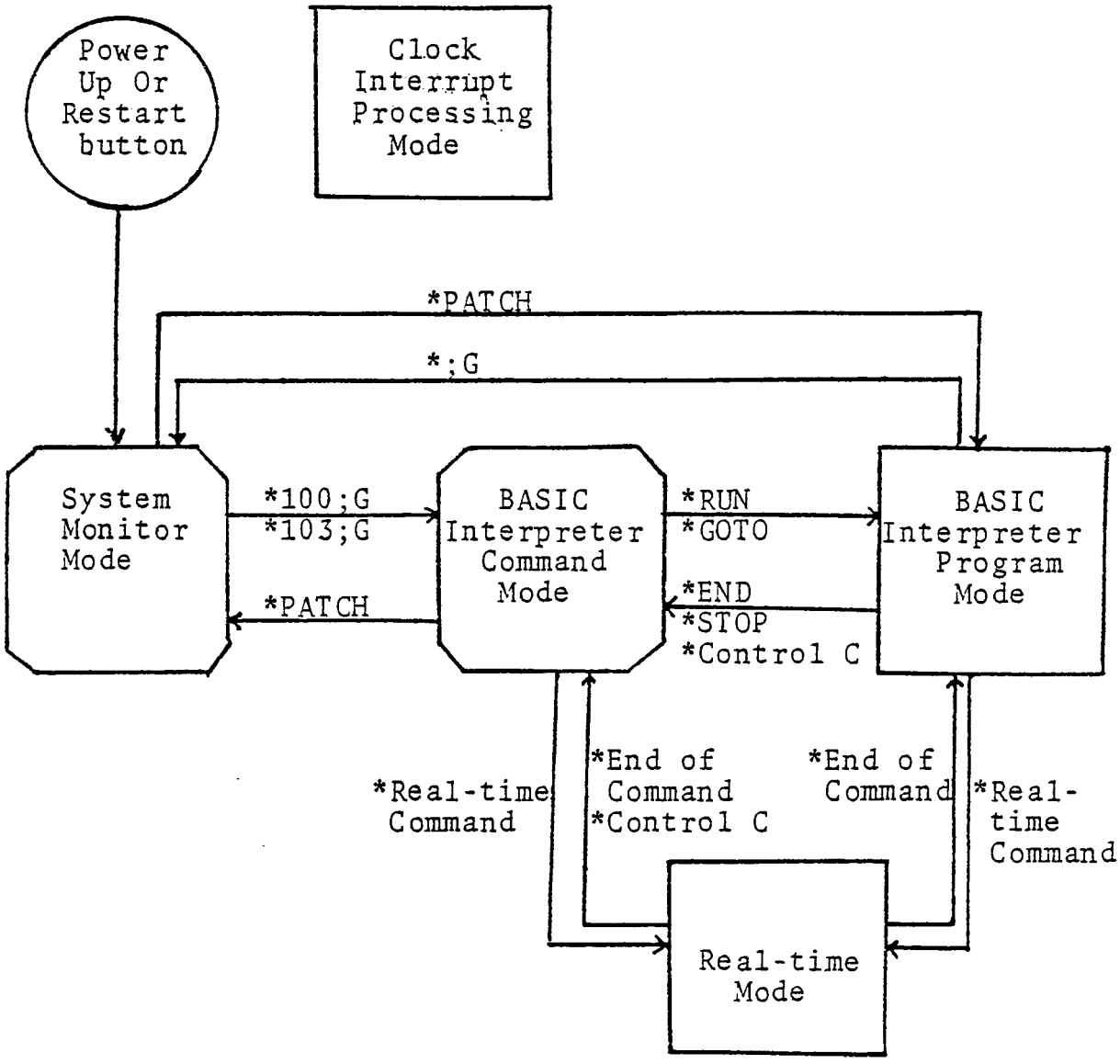
Figure 2.2: System Programs

capabilities to BASIC. The three programs are:

1. System monitor- This program is used to load and punch binary object programs, execute machine language programs, change register contents and change memory contents.
2. BASIC interpreter- This program executes a high level language program written in BASIC.
3. Parallel I/O program- This program consists of three parts:
   a. Initialization program- This program initializes the system hardware and software to a predetermined state.
   b. Extension to BASIC- This program performs the non real-time parallel I/O commands. Non real-time commands are those which do not require co-ordination with the clock interrupt driven UPDATE program for execution.
   c. Real-time programs- These programs include the interrupt driven UPDATE program and the real-time command modules. UPDATE maintains time and time related variables of the system. The real-time command modules consist of the real-time I/O task and the WAIT command module. They require co-ordination with the UPDATE program for execution.

2.3 System Modes Of Operation

When the system is running the processor can execute any of the system programs. It is convenient to consider the processor to be in different modes of operation. The functions of each of these modes and the ways they are entered or exited are illustrated in figure 2.3. The user



denotes the modes in which the user can communicate with the system.

Figure 2.3: System Modes Of Operation

can communicate with the system while it is in the system monitor mode or in the BASIC command mode. The system modes are described below:

1. System monitor mode- In this mode the user can load or save binary object programs through the terminal, execute programs stored in memory, change memory contents, and change CPU contents.

   This mode is entered:

   a. Upon power up or by depressing the restart button.
   b. From the BASIC interpreter command mode when user types PATCH.
   c. From the BASIC program mode when the statement PATCH in a BASIC program is executed.

   From this mode one may enter the BASIC command mode by typing '100;GC/R' or '103;GC/R' where C/R denotes the carriage return key. If '100 ;GC/R' is entered, the system is initialized before the command mode is entered.

2. BASIC interpreter command mode- In this mode, a BASIC program can be loaded or saved. The program can be edited by entering labelled BASIC statements. BASIC statements, including the parallel I/O commands of this system, will be executed directly if they are entered without labels. These unlabelled statements will not be edited as part of user's BASIC program.

   This mode is entered:

a. From the system monitor mode when '100;GC/R' or '103;GC/R' is typed.
b. From the BASIC program mode when the statement STOP or END in a BASIC program is executed, when an error occurs in it or if the keys CONTROL C are depressed.
c. From the real-time mode when the real-time command has been executed or when the keys CONTROL C are depressed.

From this mode one may enter the following modes:

a. BASIC program mode by typing 'RUN' or 'GOTO label' where 'label' is the label of the statement to be executed.
b. System monitor mode by typing 'PATCH'.
c. Real-time mode through the use of a real-time command.

3. BASIC program mode- In this mode the BASIC interpreter executes BASIC statements, including the non real-time parallel I/O commands provided by the system.

This mode may be entered from the BASIC command mode by typing 'RUN' or 'GOTO label' statements.

From this mode one may enter the following modes:

a. BASIC command mode through the execution of a STOP or END statement, if an error is detected or when the keys CONTROL C are depressed.

b. System monitor mode if a PATCH statement is executed in the BASIC program.
c. Real-time mode if a real-time command is executed in the BASIC program.

4. Real-time mode- The system enters this mode when it is performing either a "RTX' command or a 'WAI' command.

This mode may be entered:

a. From the BASIC interpreter command mode upon execution of an unlabelled real-time command.
b. From the BASIC program mode upon execution of a labelled real-time command in the BASIC program.

From this mode one may enter the following modes:

a. The BASIC command mode after the execution of a real-time command given in the BASIC command mode.
b. The BASIC program mode after the execution of a real-time command given in the BASIC program.
c. The BASIC command mode through the depression of the keys CONTROL C.

5. Clock interrupt processing mode- This mode is transparent to the user. When in this mode the system executes the clock interrupt driven UPDATE program. The system time and time related variables are updated.

The clock interrupt can be serviced by the processor in all modes of operation except

during initialization. At the end of an interrupt processing call, control is returned to the interrupted mode.

The remainder of this thesis will describe only the commands that have been added to the BASIC set of commands. For details on the monitor commands(system monitor mode) and standard BASIC commands and statements(BASIC command and program modes) the reader is refered to references (21) and (22).

CHAPTER 3

USER MANUAL

This chapter describes the restrictions that the system imposes on the user. The expanded BASIC commands are described and the error messages associated with these commands are explained.

3.1 User Restrictions

a. The command programs use the BASIC variables Q0 to Q9. These variables should be the first variables referenced and they should be referenced in the order listed before the expanded commands are used. The BASIC variables are used as follows:

1. Q0- The OUT command outputs the number in Q0 to an output port.
2. Q1- The INP command inputs data from an input port and stores it in Q1.
3. Q2- This is a vector of 3 elements. The TMR command stores the time-of-day into Q2.
4. Q3- This is a vector used to pass data to the command programs when periodic output is requested for an output port by the POR command. The dimension of Q3 should be greater than or equal to the number of words requested. This vector may be omitted if the POR and PIR commands are not used.
5. Q4 to Q9- These are vectors used to store data acquired periodically from input ports. Q4 stores the data for the input port specified in the first periodic input request made by the PIR command, Q5 stores the data for the input port specified in the second periodic input request made by the command, and

so on. The dimension of each of these vectors must be greater than or equal to the number of input words requested for the port using that vector. These vectors may be omitted if the PIR command is not used.

b. The BASIC interpreter used in this system provides eight standard I/O handlers for outputing to and inputing from I/O devices such as teletypes and line printers. One of these I/O handlers, output handler #5, has been modified as part of the command programs. This output handler should not be used except when using the expanded commands.

3.2 Commands

This section lists all the commands that are added to the BASIC interpreter. The purpose of each command, the format in which they should be written and the parameters associated with each command are given. Examples for each command are given when appropriate.

Every expanded BASIC command is called by a BASIC statement of the form

```
PRINT #5,"command"
```

where command is a string of characters specifying the command. In general, a command is specified by a three letter mnemonic followed by parameters. If the command requires no parameters, the command is specified only by the mnemonic. If a command requires parameters, the mnemonic is separated from the parameters by a blank and the parameters are separated from each other by a comma.

In the following presentation of command specifications, parameters enclosed in parenthesis() are identified as optional.

3.2.1 CON(Configure Port)

This command configures the hardware and software of the system so that a port is configured in the system. In a port, the lines with larger line numbers are the most significant data bit lines.

a. Command format:
   Print #5, "CON name, port type, first line number (, last line number)"

b. Command parameters:
   1. name--user defined name for the port. It must be a letter followed by one to five letters or digits.
   2. port type--specified by two letters. The first letter denotes input or output port. If it is 'I', the port is an input port. If it is 'O', it is an output port. The second letter defines the data format of the port. The letter 'B' defines the port to be binary, the letter 'D' defines it to be unsigned BCD and the letter 'S' defines it to be signed BCD.
   3. first line number--it is the numeric label of the first line (least significant bit of the port) assigned to the port. 'First line number' must be a positive integer not greater than the total number of lines that exist in the system, otherwise an error message will be displayed. The total number of lines is 8 times the number of PIC's.
   4. last line number--it is the numeric label of the last line assigned to the port. 'Last line

number' must be a positive integer not greater than the total number of lines that exist in the system and it must be greater than 'first line number'. A port with binary or decimal data format can be one to 16 lines wide. A port with signed BCD format can be one to seventeen lines wide, with the last line used for the sign. If the port has only one line, this parameter is omitted.

c. Example:

Print #5, "CON OUT,OS,16,32"

This command configures an output port called OUT. The data format of the port is signed BCD and it comprises lines 16 through 32.

3.2.2 DPT(Delete Port)

This command deletes a port previously configured by the user through the use of a CON command.

a. Command format:

```
PRINT #5,"DPT name"
```

b. Command parameters:

1. name- name of a port that has been configured by the user.

3.2.3 OUT(Output To Port)

This command outputs a value to a port. If an integer is specified in the command, the integer is outputed to the port. If it is not specified, the data in the BASIC variable Q0 is outputed. In either case, the data is converted to the port's data format prior to being outputed.

a. Command format:
   PRINT #5,"OUT name(,integer)"

b. Command parameters:
   1. name- name of a port that has been configured by the user.
   2. integer- a decimal integer not greater than the maximum value or less than the minimum value that can be outputed to a port, otherwise, an error message will be displayed. This parameter is optional.

c. Examples:

   PRINT #5,"OUT OPORT,255"

   In this example, the integer 255 is outputed to the port called OPORT. The integer is either converted to a binary number, a BCD number or a signed BCD number, depending on the data format of OPORT. The converted data is outputed to OPORT.

   PRINT#5,"OUT OPORT"

   This example is the same as above except that the output data is retrieved from the BASIC variable Q0.

3.2.4 INP(Input From Port)

This command inputs the decimal equivalent of the data of an input port into the variable Q1. The data at the port is read and interpreted according to the data format of the port. It is converted into a decimal integer and then stored into the BASIC variable Q1.

a. Command format:
   PRINT #5,"INP name"

b. Command parameter:
   name- name of a port that has been configured by the user.

3.2.5 PIR(Periodic Input Request)

This command sets up the tables so that periodic input will be performed for the input port specified when the RTX command is given. A maximum of six ports can be set up for periodic input or output. A maximum of 255 words can be requested for an input port. For each input port requested, a BASIC vector should have been dimensioned earlier for the storing of input data, as described in section 3.1.

a. Command format:
   PRINT #5,"PIR name(,start time),no. of words,period"

b. Command parameters:
   1. name- name of an input port that has been configured by the user.
   2. no. of words- number of words to be acquired. It must be a positive integer not greater than 255.
   3. start time- the time when real-time input from the input port will begin after the RTX command has been executed. This parameter is specified as

      wHxMySzT

      where wH denotes the no. of hours, xM the no. of minutes, yS the no. of sec. and zT the no. of ten ms. 'w' can be any integer from 0 to 24, 'x' from 0 to 59, 'y' from 0 to 59 and z from 0 to 99. The integer and the letter denoting the unit of time can be omitted if it is 0. 'start time' may be selected to be 0 to 24 hours, in intervals of 10 ms. This parameter is optional. If it is omitted, it is assumed to be 0.

4. period- the time between consecutive input of words from an input port. The format of this parameter is the same as 'start time'. 'period' can be selected to be 10 ms to 24 hours, in intervals of 10 ms.

c. Example:

```
PRINT#5,"PIR INPORT,10M10T,100,1T"
```

This command instructs the system to start inputing from the input port called INPORT at 10 minutes and 100 ms after the system has entered the real-time I/O mode(when the command RTX is executed). 100 words should be inputed and the time between input from this port is 10 ms.

3.2.6 POR(Periodic Output Request)

This command sets up the tables containing the data which will be periodically outputed to a specified output port when the RTX command is executed. A maximum of six ports can be set up for periodic input and output. The data to be outputed should already have been stored in the BASIC vector Q3. The command checks the data, converts the data into the data format of the output port and stores them into an internal buffer from which it will be outputed.

a. Command format:
   PRINT #5,"POR name(,start time),no. of words, period(,total time)"

b. Command parameters:
   1. name- name of an output port that has been configured by the CON command.
   2. start time- the time at which periodic output to the output port should begin after the RTX command is executed. 'start time' can be selected to be 0 to 24 hours, in intervals of 10 ms. This parameter is optional. If omitted, it is assumed to be 0.
   3. no. of words- number of words to be outputed. It must be a positive integer not greater than 255.
   4. period- the time interval between consecutive output words to the port. It can be selected to be 10 ms to 24 hours, in intervals of 10 ms.
   5. total time- Two modes of output can be specified. In the one-shot mode, 'total time' is omitted. In this case, output for the port is considered

finished after the last word of the requested number of words has been outputed. In the circular mode of output, 'total time' has to be specified. 'total time' denotes the length of time that the circular mode is to last. In this case, user's data will be outputed for a time equal to 'total time'. If the last word of user's data has been outputed and the total time has not elapsed, the first word of user's data will be outputed again. 'total time' can be selected to be 10 ms to 24 hours, in intervals of 10 ms.

c. Example:

```
PRINT #5,"POR OPORT,1M,50,1S,5M"
```

This command causes tables to be prepared so that data will be outputed to the port called OPORT at 1 second intervals 1 minute after the RTX command has been executed. The output function will last for 5 minutes. The first word to be outputed is the first word of the 50 words supplied by the user in the BASIC vector Q3. If the 50th word has been outputed but the time elapsed is not yet equal to 'total time', the system will scan the output buffer from the beginning as many times as required to fullfill the 'total time'.

3.2.7 DRQ(Delete I/O Request)

This command deletes the last one or more periodic I/O requests made by the PIR and POR commands. It is useful when the user finds that he has made an error in a request.

a. Command format:
   PRINT #5,"DRQ no. of request"

b. Command parameter:
   no. of requests- specifies the last one or more requests to be deleted. It must be a positive integer not greater than 6.

3.2.8 RTX(Periodic Input/Output Execute)

This command causes the system to enter the real-time I/O mode and to execute the periodic input and output requests made by the POR and PIR commands.

a. Command format:

```
PRINT #5,"RTX"
```

3.2.9 TMS(Set System Clock)

This command sets the system's time-of-day clock.

a. Command format:
   PRINT #5,"TMS time"

b. Command parameter:
   time- the hours, minutes and seconds to which the system clock is to be set. The format of this parameter is wHxMzS. The value of this parameter can be from 0 seconds to 23 hours 59 minutes and 59 seconds, in intervals of 1 second.

3.2.10 TMR(Read System Clock)

This command is used to read the time-of-day. The clock is read and the hours, minutes and seconds are stored in the BASIC vector Q2. The time can also be displayed on the system console.

a. Command format:
   PRINT #5,"TMR display"

b. Command parameter:
   display- can be the letter 'N' or 'D'. If it is 'D', the time is stored into Q2 as well as displayed on the system console. If it is 'N', the time is not displayed on the console.

3.2.11 WAI(Wait)

This command causes the BASIC program to wait a certain period of time prior to executing the next statement.

a. Command format:
   PRINT #5,"WAI time"

b. Command parameter:
   time- 'time' is specified as wHxMzS. It can be selected to be 1 second to 24 hours, in intervals of 1 second.

3.2.12 CONTROL C Keys

When the system is performing periodic I/O or waiting, it is possible to change its mode to the BASIC command mode by pressing the CONTROL C keys on the keyboard.

If the RTX command is given later, periodic I/O will resume at the point where it was stopped.

3.3 Error Messages

The following error message are provided to assist the interpretation of errors in command execution:

29 Illegal mnemonic for command.
30 Illegal parameters for command.
31 Duplicate port name.
32 Lines already used by one of the ports configured.
33 Port specified in command does not exist.
34 Error in data to be outputed.
35 The BASIC variable for the command does not exist.
36 Internal data buffer overflow.
37 Ports tables full(more than 20 ports configured).
38 Periodic I/O tables full.

CHAPTER 4

PROGRAM STRUCTURE, PROGRAM PARAMETERS AND VARIABLES

The structure of the extension to BASIC programs and the real-time programs is illustrated in figure 4.1. The extension to BASIC programs consist of the command interpreter and the command modules which do not require co-ordination with UPDATE for execution. The real-time programs are the clock driven UPDATE program and the command modules which require co-ordination with UPDATE for execution.

When any of the expanded BASIC commands is called, program control is passed to the command interpreter. The command interpreter identifies the command, checks the parameters of the command for validity and also converts the parameters when necessary(e.g. convert an ASCII coded number into BCD). If the command involves a BASIC extension command module, control is passed to that module which executes the command and returns to the main program. If the command involves a real-time command module, control is passed to that command module. The real-time command module then co-ordinates with UPDATE to execute the command. Control is then returned to the main program.

4.2 Program Parameters

The programs use the following parameters stored in ROM:

1. NUMPIC- This one byte parameter specifies the number of parallel interface controllers in the system. In the present system, it is equal to 4.

2. PICADR- 8 bytes. This table contains the addresses

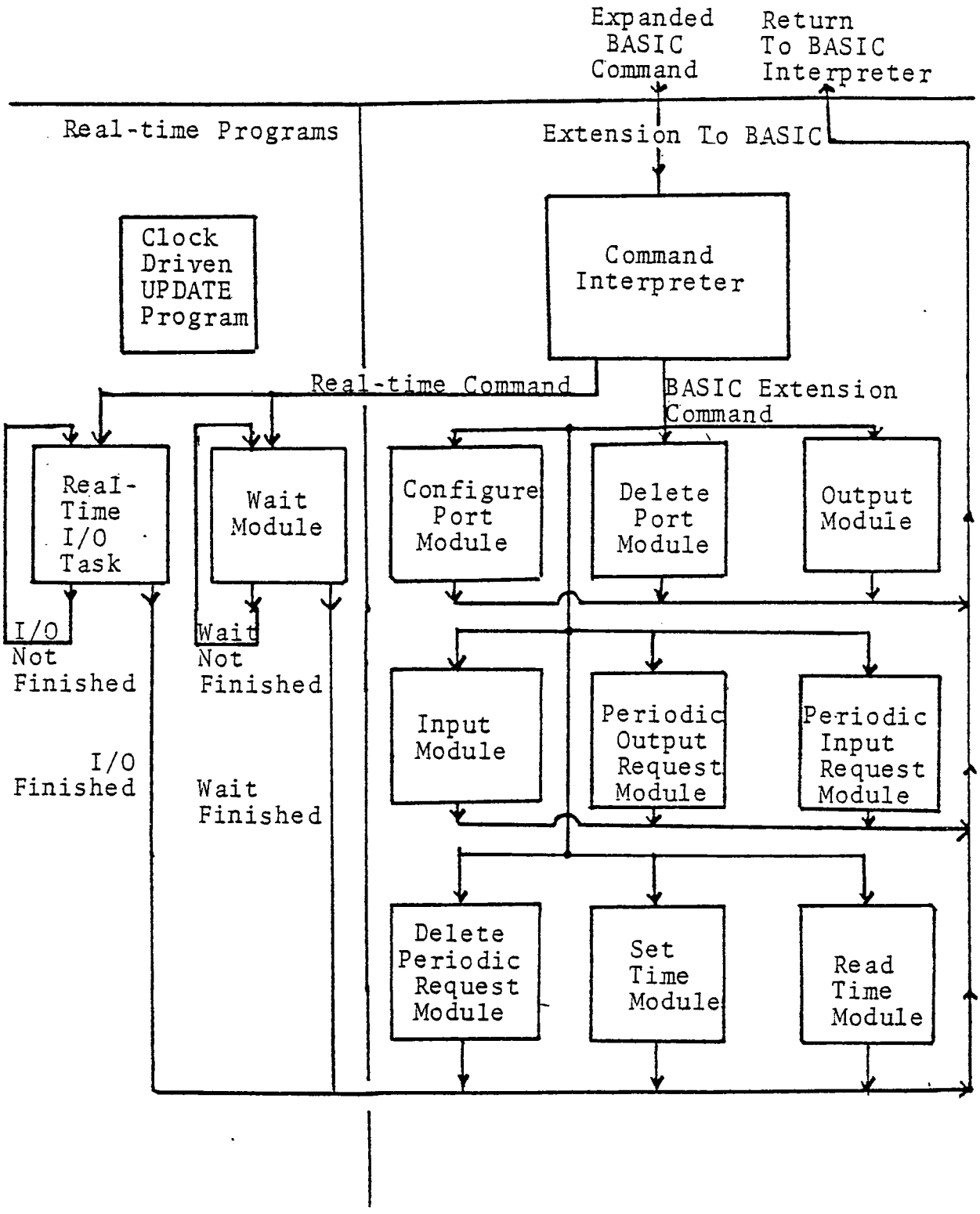


Figure 4.1: Program Structure

of all the parallel interface controllers in the system. The address stored in bytes 1 and 2 of this table is the address of the controller which interfaces lines 1 through 8 of the system, the address stored in bytes 3 and 4 is the address of the controller which serves as lines 9 through 16, and so on. At the present time, the contents of this table are 8400_{16},8402_{16},8404_{16} and 8406_{16}.

4.3 Program Variables

The variables of the command programs are illustrated in figure 4.2. These variables are described below:

1. TIME(system clock)- 4 bytes. This is the system clock. The first byte contains the hours of the day, the second the minutes, the third the seconds and the fourth the tens of ms. The contents of TIME is continuosly updated by the clock interrupt driven UPDATE program.

2. NPORTS(number of ports)- 1 byte. This varaible contains the number of ports that the user has configured in the system.

3. PORTST(ports table)- 240 bytes. This table contains a description of all the user configured ports in the system. Each port defined by the user requires 12 bytes in this table, so that a maximum of 20 ports may be stored in this table. The information regarding each port in this table is illustrated in figure 4.3. The first six bytes in a port table

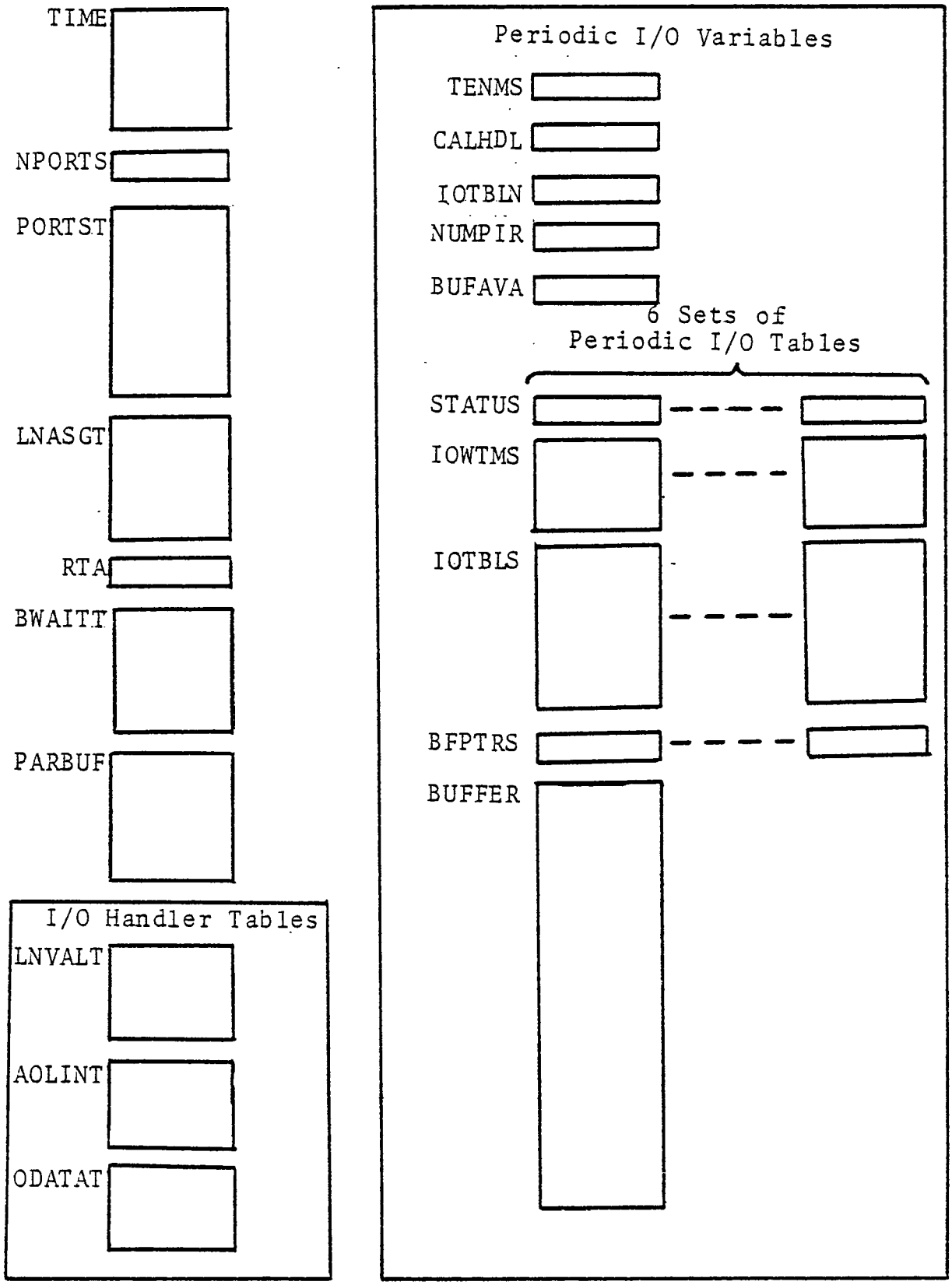


Figure 4.2: Program Variables

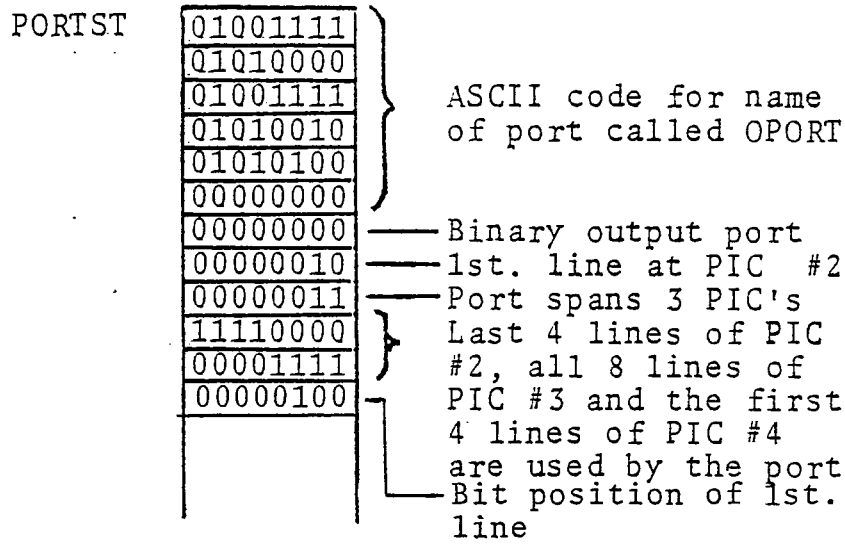


Figure 4.3: The PORTST Table

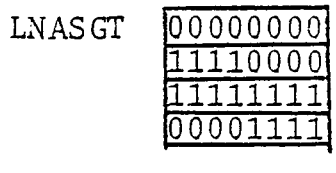


Figure 4.4: The LNASGT Table

contain the ASCII coded name of the port. If the name contains less than 6 characters, the unused bytes contain 0's. The seventh byte specifies the port type. When bit 0 of this byte is 1, it is an input port, otherwise it is an output port. Bits 1 and 2 of byte 7 denote the data format of the port. If the number represented by these 2 bits is equal to 0, the data format is binary, if it is equal to 1, it is unsigned BCD, if it is equal to 2, it is signed BCD. Bytes 8 to 12 specify the location of the port. Byte 8 contains the numeric label of the PIC(parallel interface controller) containing the first line of the port(The numeric label of a PIC is determined by its location in the ROM table PICADR.). Byte 9 denotes how many PIC's the port spans. Byte 10 and 11 contain bit maps of the lines used by the port in the first and last PIC's which the port spans. Byte 12 contains the bit position of the first line in the PIC where it is located. In figure 4.3, the first port in PORTST is called OPORT and it is a binary output port. OPORT uses the last 4 lines of PIC no. 2, all 8 lines of PIC no. 3 and the first 4 lines of PIC no. 4. Hence it is 16 lines wide.

4. LNASGT(lines assignment table)- 4 bytes. This table contains a bit map of all the used I/O lines of the system. Byte 1 represents the lines of PIC no. 1, byte 2 represents PIC no. 2, and so on. If a bit in this table is set, the line corresponding to that bit has been selected by the user in one of his ports.

If a bit is 0, the line represented by that bit is not used. Figure 4.4 shows the appearance of LNASGT if the port shown in figure 4.3 is the only port configured.

5. RTA(real-time command module active)- This one byte variable is used by the real-time command modules to inform the clock interrupt driven UPDATE program of which real-time command module is active so that UPDATE will co-ordinate the activities of that command module. If RTA is equal to 0, no real-time command module is active. If RTA is 1, the real-time I/O task is performing periodic I/O. If RTA is 2, the WAI command module is active.

6. BWAITT(BASIC program waiting time)- 4 bytes. These bytes contain the time that the BASIC program has to wait before execution of the next BASIC statement. The first byte contains the hours, the second the minutes, the third the seconds and the fourth the tens of ms.

7 PARBUF(parameters buffer)- 20 bytes. This buffer is used by the command interpreter to pass formatted parameters of the command to the various command modules.

The tables listed in 7 to 10 are used by the I/O handler and are called collectively as the I/O handler tables.

7. LNVALT(lines values table)- 4 bytes. Whenever the I/O handler is called, it inputs the values of all the lines and stores them into this

table. The values of the lines of PIC no. 1 are stored in byte 1, those of PIC no. 2 in byte 2, and so on. After inputing the values of all the lines, the I/O handler uses the data in the other 2 handler tables to modify LNVALT. The modified data in LNVALT is then outputed to the lines.

9. AOLINT(active output lines table)- 4 bytes. This table contains a bit map of the output lines whose values are to be outputed by the I/O handler. If a bit is 0, the output line represented by that bit should be outputed. If it is 1, the output line should not be changed.

10. ODATAT(output data table)- 4 bytes. In this table, the bits that correspond to the output lines are set to 1 or 0, according to the values of the data to be outputed. This is done before the I/O handler is called.

The variables and tables described in 11 to 20 are used for the purpose of periodic output and input and are collectively called Periodic I/O varaibles. Periodic I/O variables are used by the POR and PIR command modules, the real-time I/O task and the UPDATE program to perform periodic I/O.

11. TENMS(ten ms)- This one byte flag is set by the UPDATE program to inform the real-time I/O task that 10 ms have elapsed. The I/O task continuosly checks TENMS to see if it is set. When it is set, the I/O task clears it and executes the I/O task once.

12. CALHDL(call I/O handler)- 1 byte. This flag is set by the UPDATE program to inform the I/O task to call the I/O handler when one or more of the input and output ports need to be read or written into.

13. IOTBLN(periodic I/O table no.)- 1 byte. This variable contains the number of periodic I/O requests that have been made by the POR and PIR commands. A maximum of 6 requests can be made because only 6 sets of tables are provided.

14. NUMPIR(number of input requests)- 1 byte. This variable contains the number of periodic input requests that have been made by the PIR command.

15. STATUS(status of ports)- 6 bytes, 1 byte for each port involved in periodic I/O to a maximum of 6 ports. If a port's entry in STATUS is equal to 0, it denotes that the periodic I/O for the port is finished. If it is equal to 1, the port is active, meaning that it requires input or output. If it is equal to 2, the port will be active when the next clock interrupt occurs. If it is 3, the port will require more than 10 ms to become active.

16. IOWTMS(I/O wait times of ports)- 24 bytes, 4 bytes for each port involved in periodic I/O to a maximum of 6 ports. Each group of 4 bytes contains the time the port has to wait before it will be active. The first

byte contains hours, the second minutes, the third seconds and the fourth tens of ms.

17. IOTBLS(periodic I/O tables)- 78 bytes, 13 bytes for each port involved in periodic I/O to a maximum of 6 ports. The information stored in an I/O table is as illustrated in figure 4.5. The contents of the first 2 bytes of an I/O table points to the location in PORTST where information about the port assigned to the I/O table is stored. The third byte contains the number of words that is requested for the port. Bytes 4 to 7 contain the period between input or output in hours, minutes, seconds and tens of ms. Byte 8 is used only with output operations. If it is equal to 1, the output mode is circular. If it is 0, the mode is single-shot. Bytes 9 to 12 are used in circular mode output only. They contain the time that the circular mode is to last in hours, minutes, seconds and tens of ms. Byte 13 is used as a counter for the number of words that has been outputed or inputed.

18. BFPTRS(buffer pointers)- 12 bytes, 2 bytes for each port involved in periodic I/O to a maximum of 6 ports. The contents of these 2 bytes points to the location in the data buffer where the I/O task is to obtain output data for an output port or to store input data for an input port.

19. BUFAVA(available buffer space)- 2 bytes. It

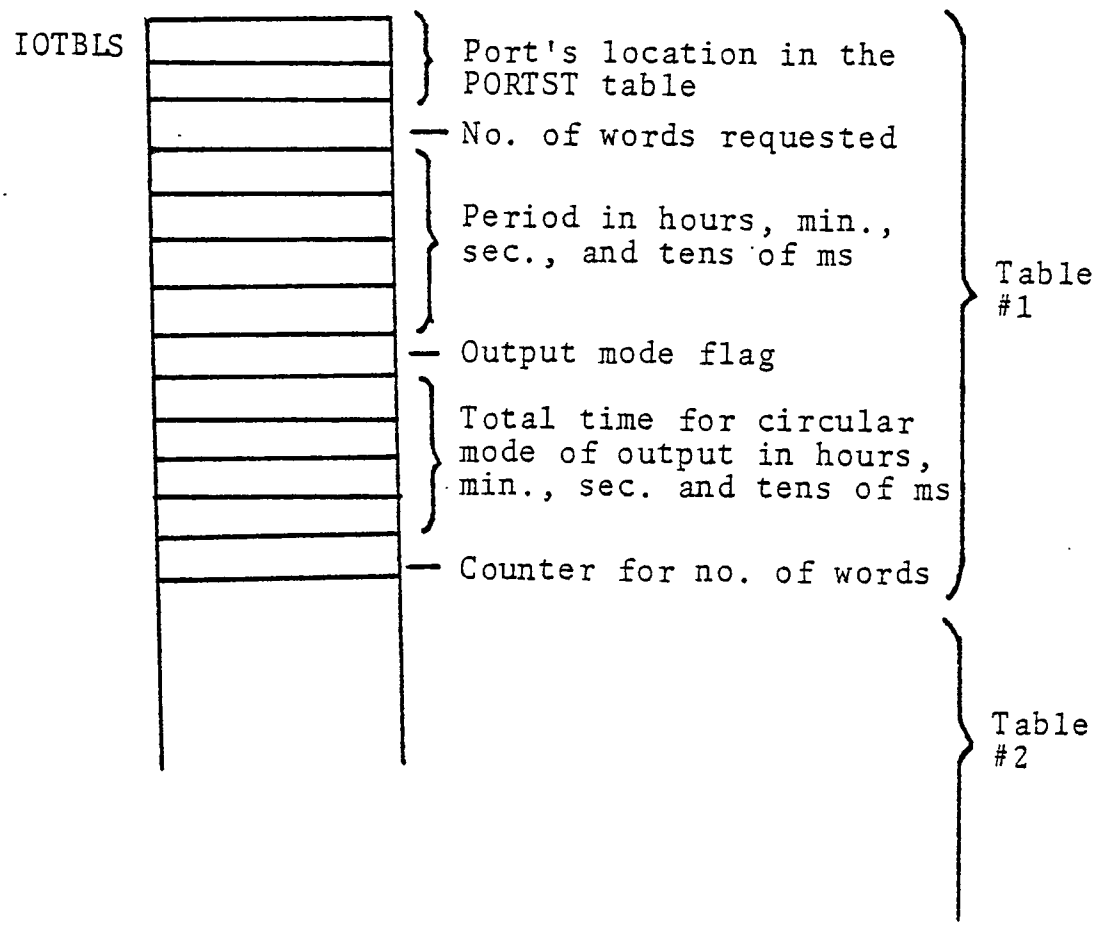


Figure 4.5: The IOTBLS Tables

contains the size of the buffer space that is left to be requested by the PIR or POR commands for periodic I/O.

20. BUFFER- 1 K bytes. This buffer is used to store the converted output data for periodic output to output ports and to store the data inputed periodically from input ports.

CHAPTER 5

SOFTWARE INTERFACE BETWEEN BASIC AND COMMAND PROGRAMS

5.1 Transfer Of Command

The transfer of program control and commands is achieved by modifying an I/O handler of the standard BASIC interpreter. The standard BASIC interpreter used in this system provides 8 I/O handlers for outputing and inputing of ASCII coded characters to and from standard I/O devices such as teletypes and line printers. Each of these I/O handlers outputs or inputs a character when called and returns to the calling program by executing a 'return from subroutine' instruction.

Output handler no. 5 of these 7 I/O handlers has been modified. Its flowchart is shown in figure 5.1. Instead of outputing the character it receives, the modified handler tests the character to see if it is a carriage return. If it is not, the character is stored in a buffer called COMAND. If it is, program control is passed to the command interpreter.

The general format of a command is:

```
PRINT #5,"command"
```

where command is a string of characters specifying the command. Whenever such a statement is executed by the BASIC interpreter, the modified output handler passes the command and program control to the command interpreter. The command interpreter identifies the command, formats the parameters of the command and directs program control to the corresponding command module. The command module executes the command and returns control to the BASIC interpreter by executing a 'return from subroutine' instruction.

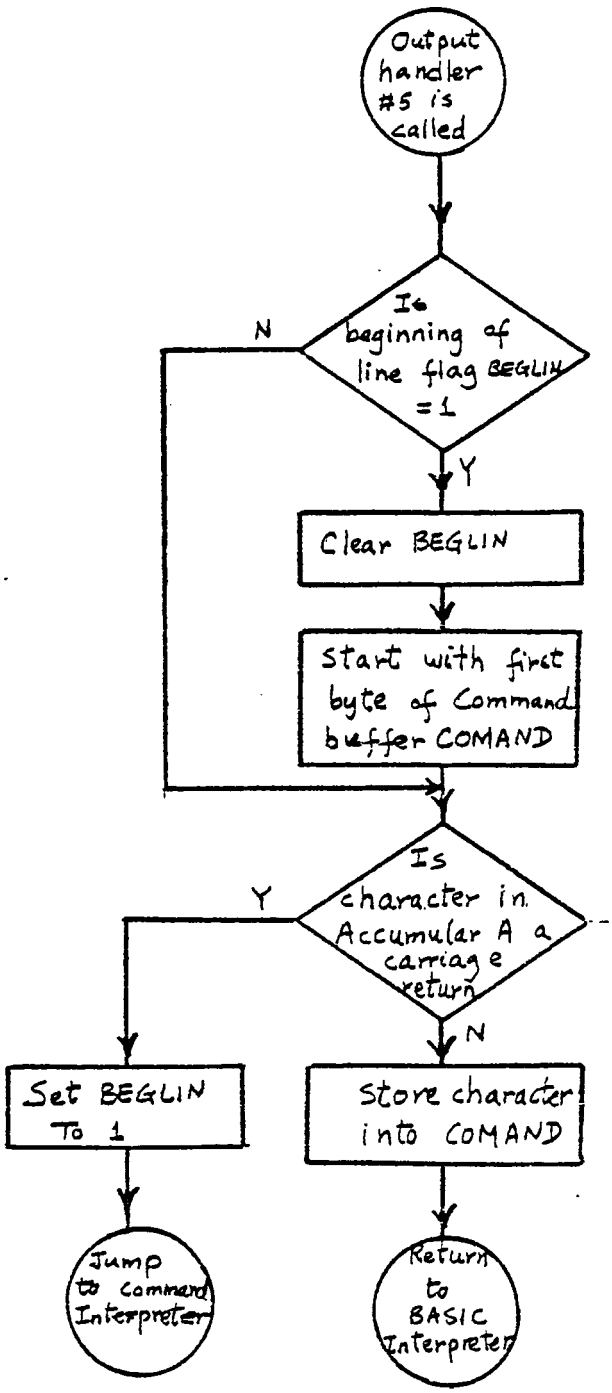


Figure 5.1: Modified Output Handler Number 5

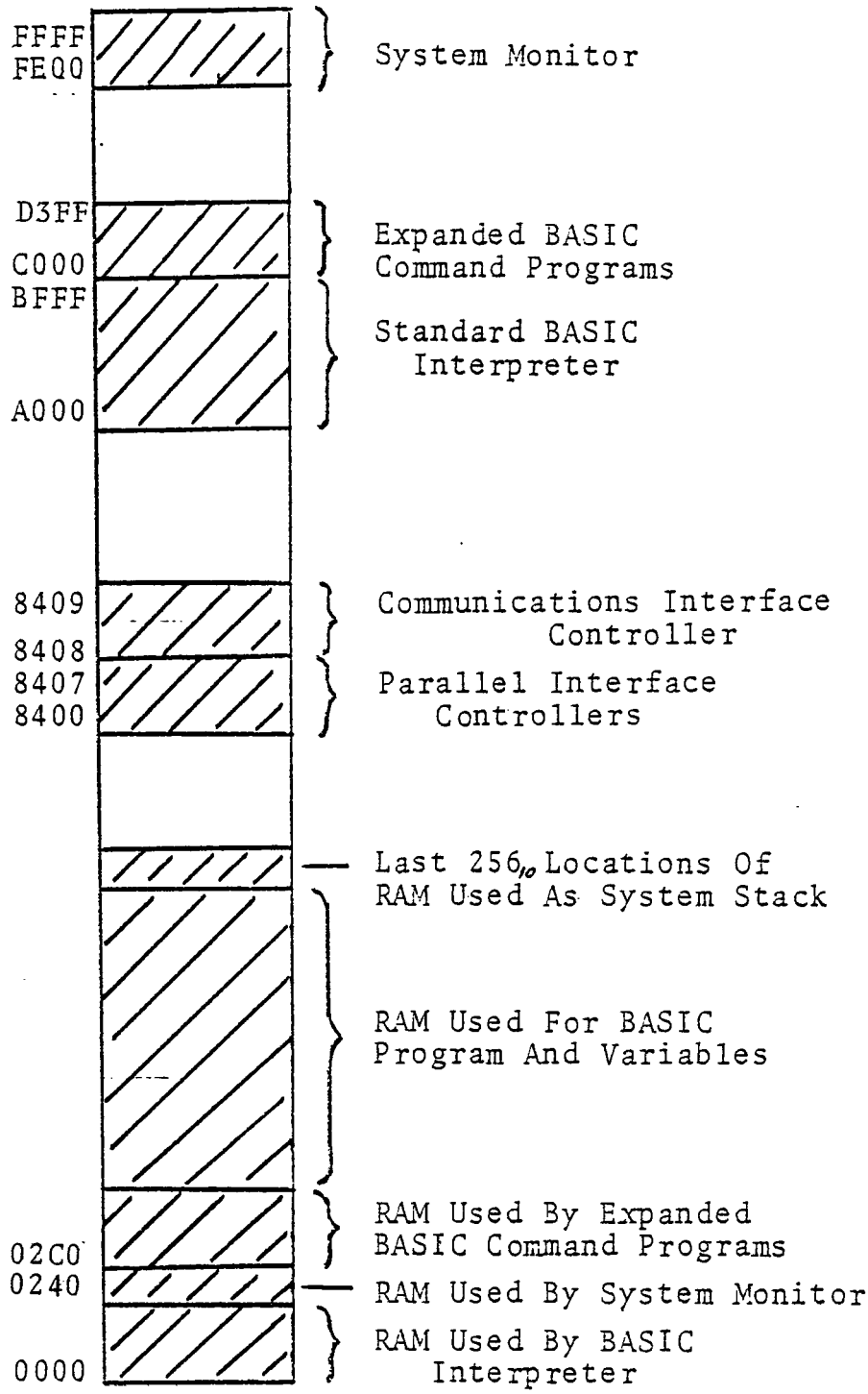
5.2 System Memory Organization

The memory organization of the system is as shown in figure 5.2. The standard BASIC interpreter is located in ROM at $A000_{16}$ to $BFFF_{16}$. The machine language programs that provide the expanded BASIC commands are appended at the end of the interpreter and are stored in ROM located at $C000_{16}$ to $D3FF_{16}$. The system monitor is located at $FE00_{16}$ to $FFFF_{16}$. The RAM of the system starts at address 0. The standard BASIC interpreter uses the locations 0 to $23F_{16}$. The system monitor uses the locations 240_{16} to $2BF_{16}$. The RAM used by the expanded BASIC command programs starts at $2C0_{16}$. The last 256_{10} locations of the RAM is used as the stack of the system. The serial interface controller used to interface the terminal with the processor is located at 8408_{16} to 8409_{16}. The four parallel interface controllers that serve as the parallel I/O lines of the system are located at 8400_{16} to 8407_{16}.

The BASIC interpreter allocates the RAM after the locations used by the expanded BASIC command programs for the BASIC source program. The BASIC interpreter has a pointer called EBSCPG which points to the end of the user BASIC program. It tells the interpreter where to start allocating memory for the variables referenced in the BASIC program. The contents of EBSCPG is changed by the BASIC interpreter when the BASIC program of the user is being edited.

5.3 Transfer Of Data

Certain commands require data to be transferred between the BASIC program and the command programs. The BASIC variables Q0 and Q1 and the BASIC vectors Q2 to Q9 are used for this purpose. The command programs assume that these variables are located right after the BASIC program and in the order listed, as shown in figure 5.3. Hence these variables have to be referenced first and in the order listed.



Note: All numbers all in hexadecimal unless indicated otherwise.

Figure 5.2: System Memory Organization

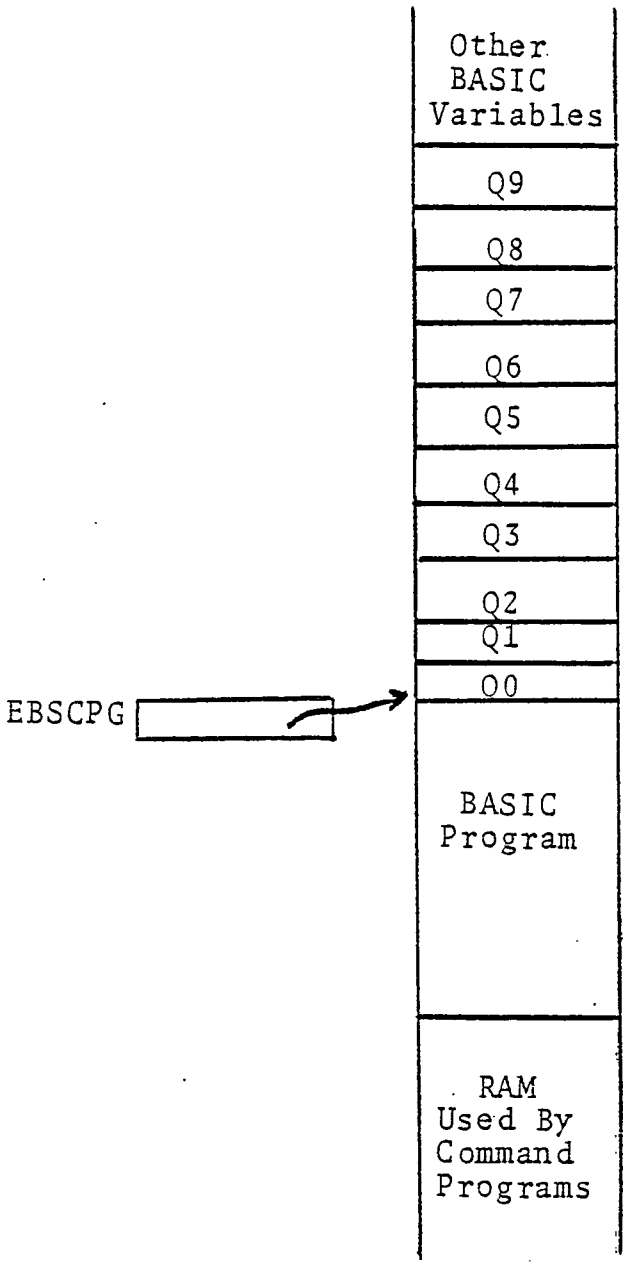


Figure 5.3: Transfer Of Data

Whenever the command programs need to use these variables, they read the contents of the pointer EBSCPG to determine the location of these variables. If data are needed from these variables, one or more of these variables is read and the data obtained is converted to a form suitable for use by the command programs. If data is to be transferred from the command programs to the BASIC program, the data is converted into the BASIC interpreter's data format and then stored into one or more of these variables.

CHAPTER 6

INITIALIZATION PROGRAM AND EXTENSION TO BASIC

This chapter describes the initialization part and the extension to BASIC part of the command programs. The real-time programs are described in the next chapter.

6.1 Initialization Program

The initialization program of the command programs is executed when '100;GC/R' is typed in the system monitor mode. It initializes the command programs and the system PIC's. At its completion, it proceeds to the BASIC interpreter's initialization program which initializes the BASIC interpreter and enters the BASIC command mode.

The initialization program of the command programs performs the following functions:

1. Initializes all the hardware lines of the system as inputs.
2. Clears the variable NPORTS and the table LNASGT.
3. Clears the variables RTA and PIOTBN.
4. Sets the system clock to 0, initializes the interrupt request vector and enables the clock interrupt.

6.2 Extension To BASIC

As defined earlier, extension to BASIC consists of the command interpreter and the non real-time command modules which do not need co-ordination with the real-time UPDATE program to execute. The global variables listed below are described in section 4.3.

6.2.1 Command Interpreter

a. Global variables used:
   PARBUF

b. Program operation:

The program structure and the flowchart of the command interpreter are shown in figures 6.1 and 6.2.

The command interpreter consists of a subprogram that identifies the command and subprograms that checks and formats the parameters associated with each command. The command and parameters are passed from the BASIC interpreter to the command interpreter by the modified output handler #5. The command is compared with a table of allowed commands. When a match is found, program control is passed to one of the parameters formatting subprograms which corresponds to that command. The subprogram checks and formats(e.g. convert from ASCII to BCD) the parameters. If the parameters are correct, they are passed to the parameters buffer PARBUF for the command module. Program control is then passed to the corresponding command module which will execute the command.

If either the command or the parameters are incorrect, an error message is displayed and program control is passed to the BASIC command mode.

6.2.2 CON Command Module

a. Global variables used:
   PARBUF, NPORTS, PORTST, LNASGT

b. Program operation:

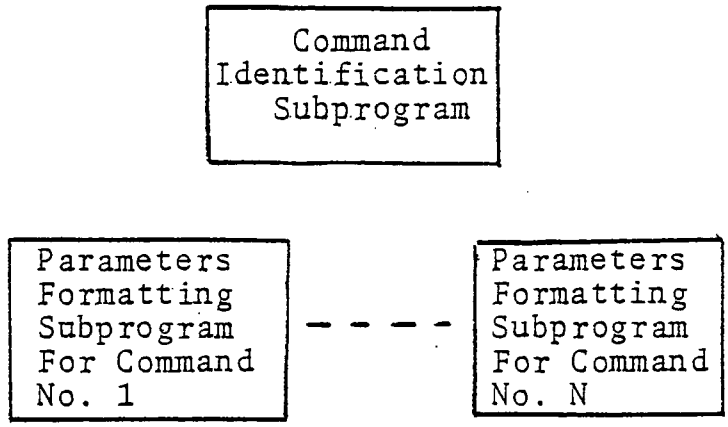


Figure 6.1: Command Interpreter Program Structure

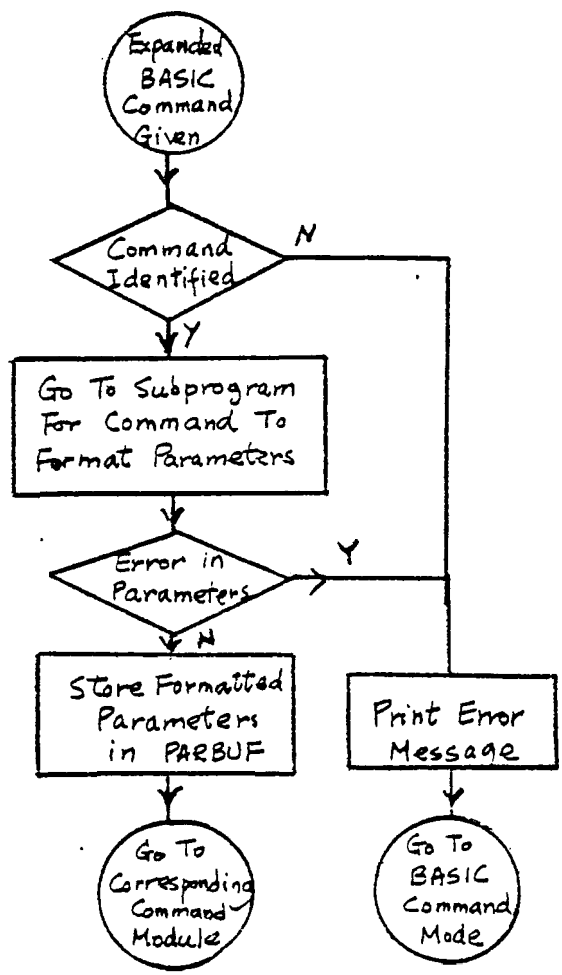


Figure 6.2 Command Interpreter Flowchart

The flowchart of this module is shown in figure 6.3.

This command module executes the CON command. The module first checks if the name of the port defined by the user already exists in PORTST. If it does, there is an error. If it does not, it checks if the lines to be used by the port defined by the user are already used by checking the LNASGT table. If they are, an error exists. Finally, the module checks if PORTST is full. If it is, an error exists. When there is an error, an error message is displayed and program control is passed to the BASIC command mode.

If no error exists, the name of the port is entered into the next 6 available locations of PORTST. The data format and whether an input port or output port are entered into the 7th byte. The data format of a port can be binary, unsigned BCD or signed BCD. From the 'first line number' and 'last line number' the numeric label of the PIC containing the first line of the port and the number of PIC's the port spans are calculated. These two numbers are entered into bytes 8 and 9 of the entry for the port in PORTST. The last two bytes of an entry in PORTST for a port contain the bit maps of the lines used in the first and the last PIC that the port spans. The command module sets the bits in these two bytes accordingly.

After the entry for the port is entered into PORTST, NPORTS is increased by one. Also, the command module sets the bits in LNASGT that correspond to the lines used by the port. The lines used by the port are programmed as output or input lines, according to whether the port is defined to be an output or input port. Program control is then returned to the BASIC interpreter.

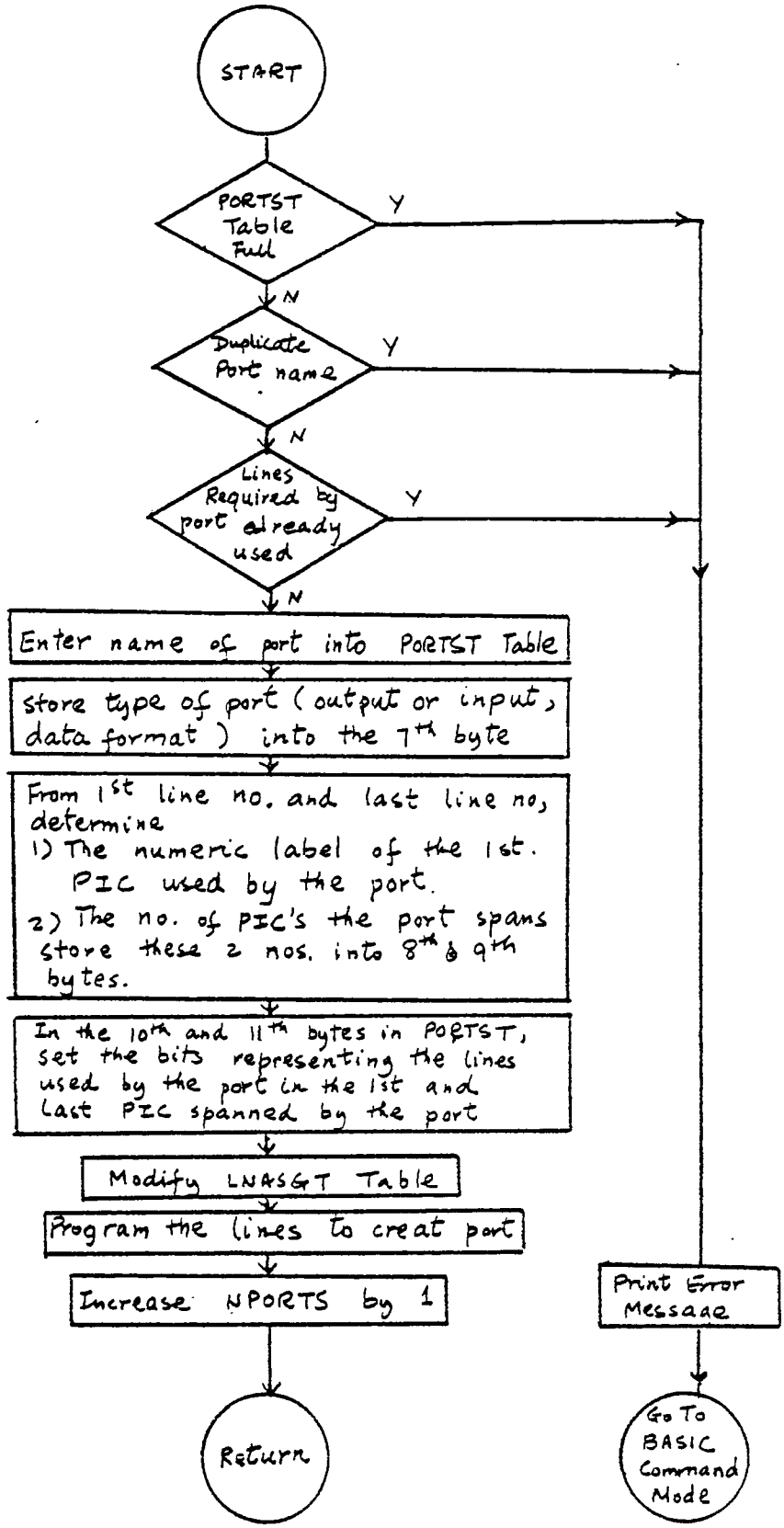


Figure 6.3 CON Command Module Flowchart

6.2.3 DPT Command Module

a. Global variables used:
   PARBUF, NPORTS, PORTST, LNASGT

b. Program operation:

The flowchart of the command module is shown in figure 6.4. The module compares the name specified with the names of the ports contained in the PORTST table. If a match is not found, an error message is displayed and program control is passed to the BASIC command mode. If a match is found, the bits in LNASGT that correspond to the lines of the specified port are cleared to indicate that those lines are no longer used. The hardware lines of the specified port are programmed as inputs(unused lines are programmed as inputs). The entry of the port in PORTST is deleted and the entries below the deleted entry are shifted one entry upward in PORTST. NPORTS is decremented by one. Program control is then returned to the BASIC interpreter.

6.2.4 OUT Command Module

a. Global variables used:
   PARBUF, NPORTS, PORTST, I/O handler tables

b. BASIC variable used:
   Q0

c. Program operation:

The flowchart of the program is shown in figure 6.5. The module searches PORTST for the port specified. If the name does not exist or if the port is not an output port, an error message is displayed and program control is passed to the BASIC command mode. If a match is found, the module determines if the data to be outputed is stored in the BASIC variable Q0 or supplied as a parameter with the command. In the first case,

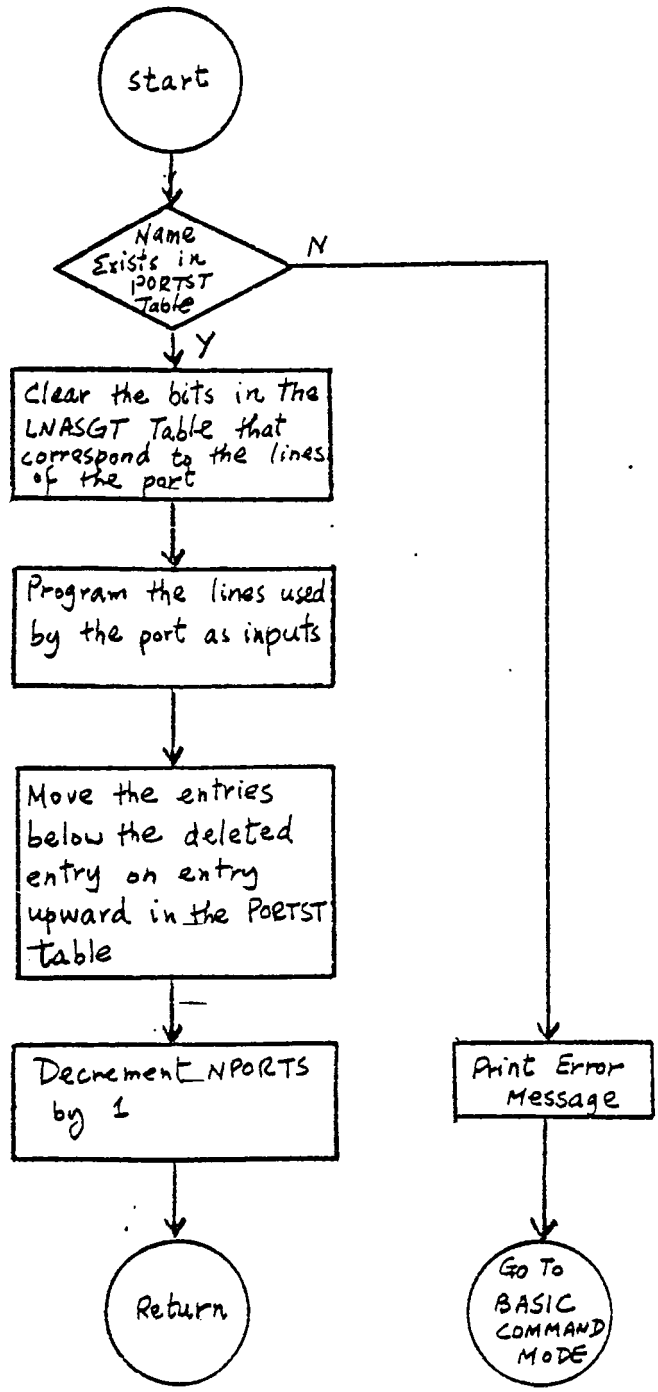


Figure 6.4: DPT Command Module Flowchart

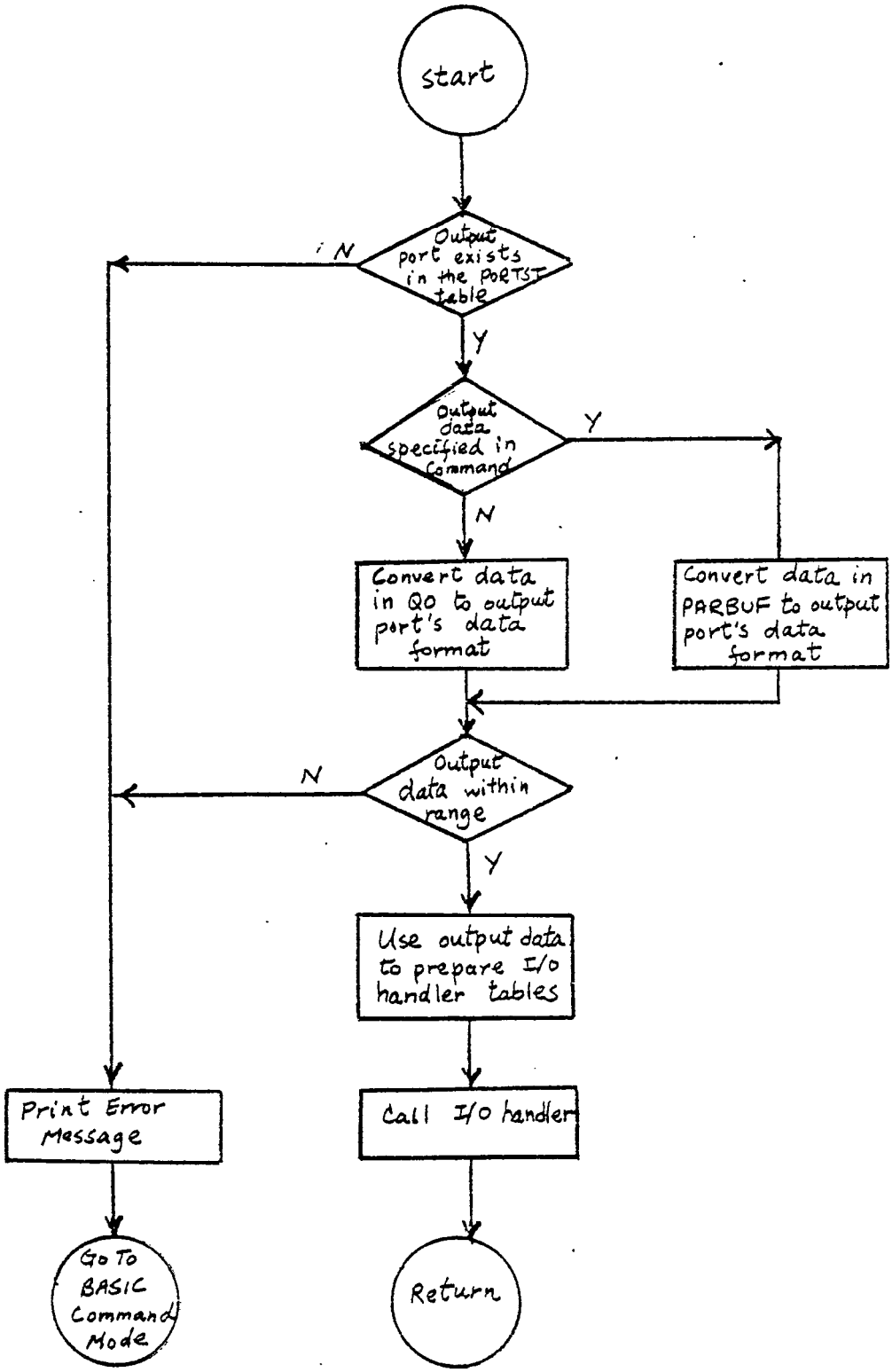


Figure 6.5: OUT Command Module Flowchart

data is read from the BASIC variable Q0. In the second case, data is read from PARBUF. In either case, the data read is converted to the data format of the output port. The converted data is compared with the maximum and minimum values that can be outputed to the port specified. If the converted output data is out of range, an error message is displayed and program control is passed to the BASIC interpreter command mode. If the data is within range, the I/O handler tables are prepared and the I/O handler is called to perform output to the port. Program control is then returned to the BASIC interpreter.

6.2.5 INP Command Module

a. Global variables used:
   PARBUF, NPORTS, PORTST, I/O handler tables

b. BASIC variable used:
   Q1

c. Program operation:

The program inputs from an input port. It first searches for the input port in PORTST. If a match is not found, an error message is displayed and program control is passed to the BASIC command mode. If a match is found, the I/O handler is called to input the values of all the lines into LNVALT. The module then reads PORTST to determine the location of the lines of the specified port and retrieves the relevant data from the LNVALT table. The data obtained is interpreted according to the data format of the input port and then converted into BASIC interpreter's data format. The converted data is stored into the BASIC variable Q1. Program control is then returned to the BASIC interpreter.

6.2.6 POR Command Module

a. Global variables used:
   PARBUF, Periodic I/O variables, NPORTS, PORTST

b. BASIC variables used:
   Vector Q3

c. Program operation:

The flowchart of this module is shown in figure 6.6.

This module executes the POR command to set up tables so that periodic output will be performed to a port when the RTX command is executed. The module first checks if the variable IOTBLN(periodic I/O table number) is 0. If it is, the module initializes BUFAVA(data buffer space available) to 400_{16} bytes and NUMIRQ(number of input requests) to 0.

The program checks whether IOTBLN is equal to 6. If it is, all the 6 Periodic I/O tables have been assigned, an error message is displayed and program control is transfered to the BASIC command mode. If an unassigned table exists, the command module checks PORTST for the specified port. If it does not exist, an error message is given and program control is passed to the BASIC command mode. If it exists, the output port's location in PORTST is stored into the assigned Periodic I/O table.

The command module checks whether the number of bytes required to store the requested number of words to be outputed is greater than the contents of BUFAVA. In this system, 2 bytes are used to store a word for a port with binary or unsigned BCD data format. For signed BCD port, 3 bytes are used for each word. If the required number of bytes is greater, the remaining buffer area is not large

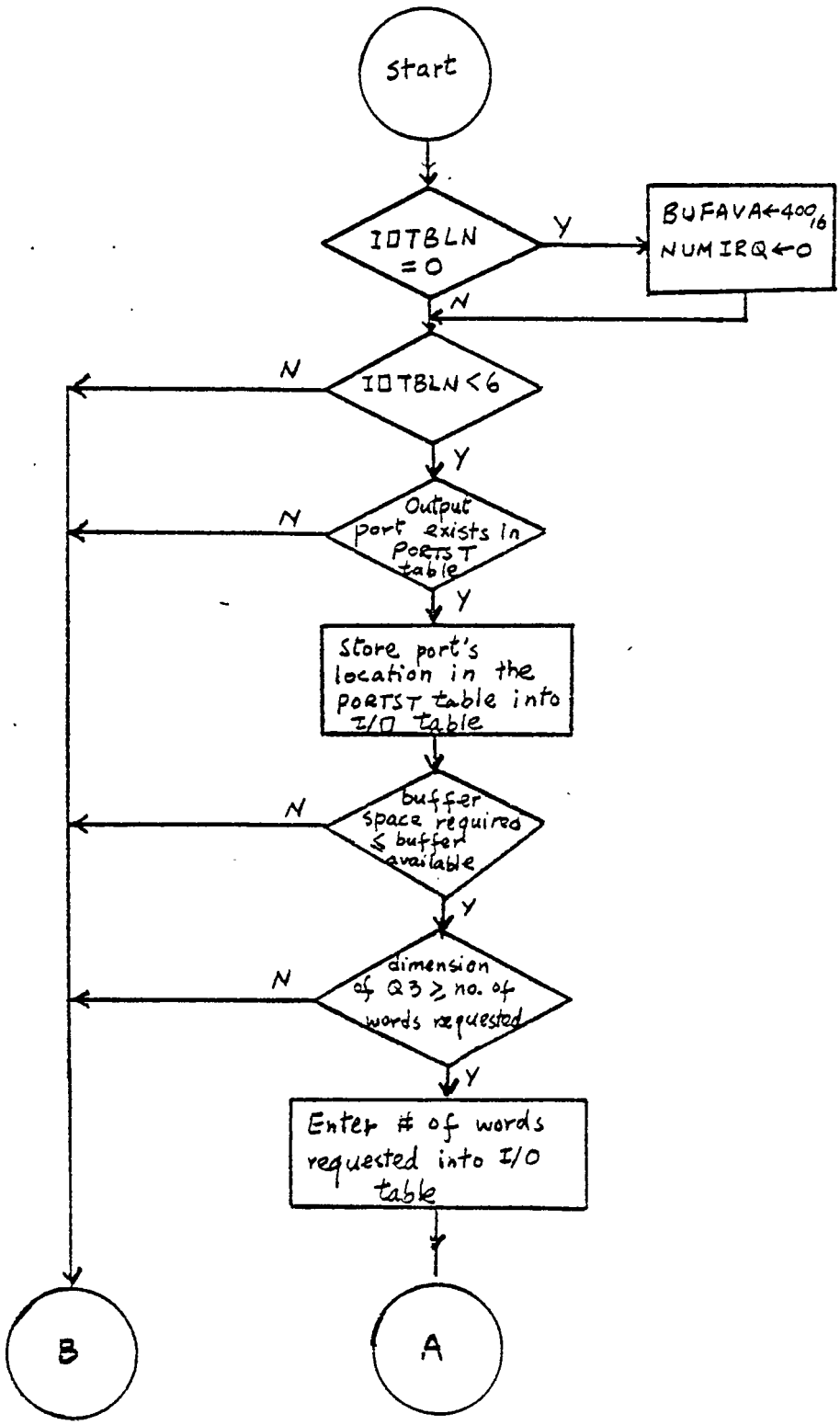


Figure 6.6: POR Command Module Flowchart

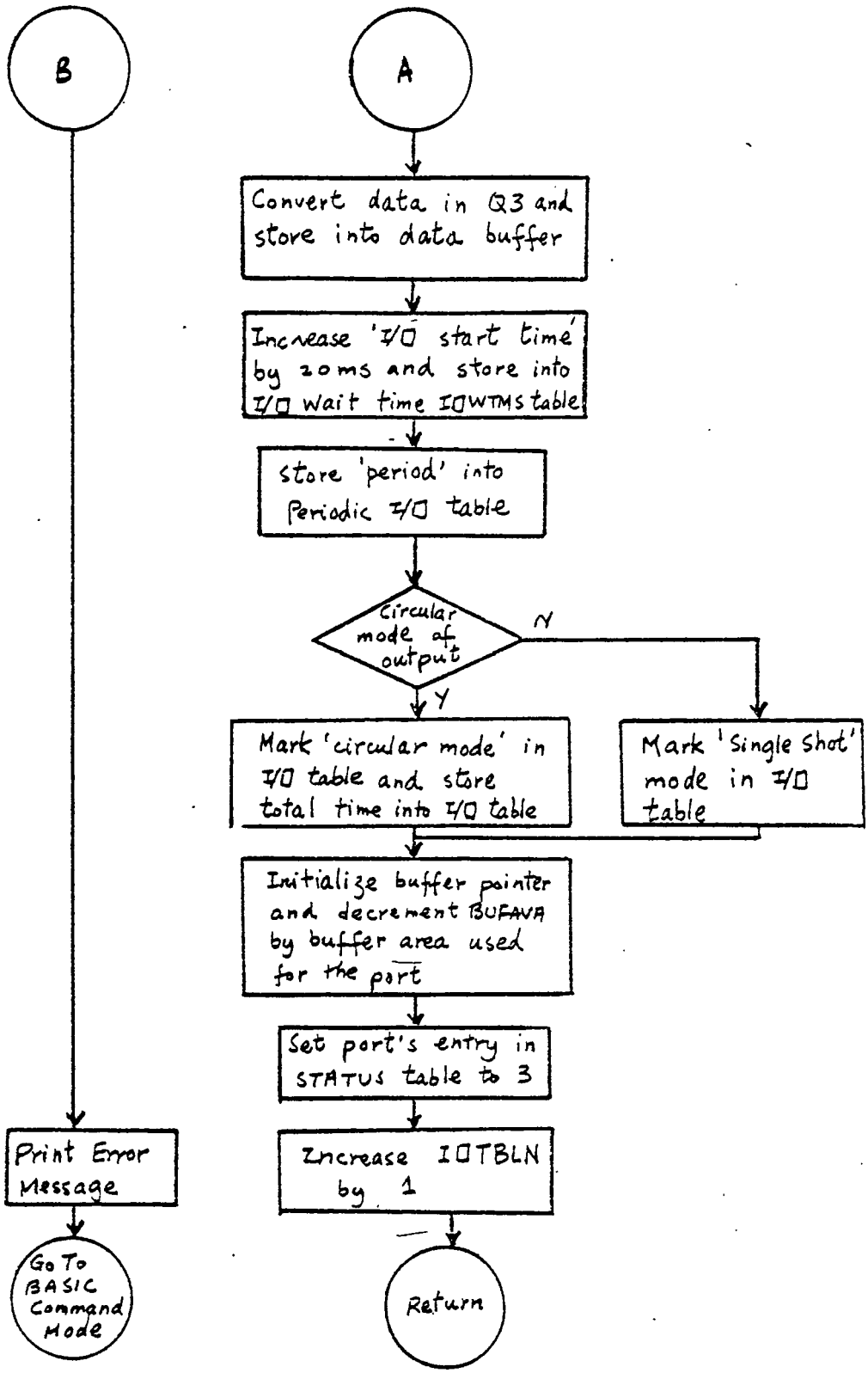


Figure 6.6(Continued): POR Command Module Flowchart

enough to hold the output data for the port. An error message is given and program control is transferred to the BASIC command mode.

If there is enough buffer space, the module checks if the BASIC vector Q3 is located at the proper location in memory and that its dimension is greater than or equal to the number of words requested. If it is not, an error exists and program control is transferred to the BASIC command mode. If no error exists, the module enters the 'number of words' requested into bytes 3 and 13 of the Periodic I/O table assigned to the port. The output data stored in the BASIC vector Q3 is then converted into the data format of the output port and stored into the buffer space reserved for the output port. Each time a number in the BASIC vector is converted, it is checked for being within the range that can be outputed to the specified port (e.g. the maximum number that can be outputed to a port with 8 lines using the binary data format is 255). If it is not, an error message is given and program control is passed to the BASIC command mode.

Next the command module increases the 'start time' specified in the parameters of the command by 20 ms. This is done so that all the ports involved in periodic I/O will not be active in at least 20 ms when the RTX command is executed. The increased 'start time' is then stored into the 4 bytes in the IOWTMS(I/O wait times) table that correspond to the assigned Periodic I/O table. The 'period' of output is stored into bytes 4 to 7 of the assigned I/O table.

If the command specifies circular mode of output, the 8th byte in the I/O table is set to 1 and the 'total time' for circular mode is stored into bytes 9 to 12 of that table. If the command indicates single-shot mode of output, byte 8 of that table is cleared.

The module adjusts the buffer pointer for the port in the array BFPTRS(buffer pointers) so that it points to the location in the data buffer where the first output word for the port is stored. BUFAVA is then decremented by the number of bytes that has been reserved in the data buffer for the output port.

The entry in the STATUS table corresponding to the output port is set to 3. IOTBLN is incremented by one and program control is returned to the BASIC interpreter.

6.2.7 PIR Command Module

a. Global variables used:
PARBUF, Periodic I/O variables, NPORTS, PORTST

b. BASIC variables used:
The BASIC vectors Q4 to Q9, one vector for each input request.

c. Program operation:

This module is similar to the Periodic Output Request module except in the following details:

1. It does not have to convert output data although an input buffer area still has to be reserved for the port.
2. It checks if a BASIC vector has been dimensioned for the port and that its size is large enough to store the requested number of words to be inputed.
3. There is no circular mode in input, hence bytes 8 to 12 of a Periodic I/O table assigned to an input port are not used.

6.2.8 DRQ Command Module

a. Global variables used:

PARBUF, IOTBLN, BUFAVA, NUMPIR

b. Program operation:

For each of the requests to be deleted, the module adds the size of the reserved buffer area for the request to the contents of BUFAVA. If the deleted request is an input request, NUMPIR is decreased by one. IOTBLN is then decremented by the number of requests deleted. If the number of requests to be deleted is equal to or greater than the number of requests that have been made, IOTBLN is cleared.

6.2.9 TMS Command Module

a. Global variables used:
PARBUF, TIME

b. Program operation:

The command module first clears the fourth byte of the system time-of-day clock TIME. The fourth byte contains the tens of ms. Then the first three bytes of TIME are set to the hours, minutes and seconds specified by the TMS command.

6.2.10 TMR Command Module

a. Global variables used:
PARBUF, TIME

b. BASIC variables used:
The vector Q0

c. Program operation:

This command module executes the TMR command to read the system clock. The module sets the interrupt mask of the processor. The system clock TIME is read and the hours, minutes and seconds are stored into three scratch bytes.

Interrupt mask is then enabled. The time stored in the scratch bytes is converted into BASIC's data format and stored into the BASIC vector Q2 which has three elements. If the letter 'D' is specified in the command, the time is also displayed on the system console. Program control is then returned to the BASIC interpreter.

CHAPTER 7

REAL-TIME PROGRAMS

This chapter describes the real-time programs of the system. The real-time programs consist of the clock interrupt driven UPDATE program and the real-time command modules which require co-ordination with UPDATE for execution. These modules comprise the real-time I/O task and the wait command module.

7.1.1 UPDATE Program

The 100 hz pulses generated by the timer can interrupt the processor at all times except when the system is being initialized. The interrupt causes the UPDATE program to be executed.

The general flowchart of UPDATE is shown in figure 7.1. UPDATE always maintains the system time-of-day clock. Then it checks the global variable RTA(real-time active) to see if any of the real-time command modules is active. If any is active, UPDATE updates the time dependent variables of the command module and then returns from interrupt. If no real-time command module is active, UPDATE returns from interrupt.

7.1.2 Activation Of I/O Task By UPDATE

The manner in which the real-time I/O task is executed is illustrated in figure 7.2. When the command RTX is given, program control is passed to the I/O task. The I/O task clears the variable TENMS(ten ms). It sets RTA to 1 to inform UPDATE that the I/O task is active. It then enters a loop to test when TENMS is equal to 1.

After this, UPDATE and the I/O task are executed alternately.

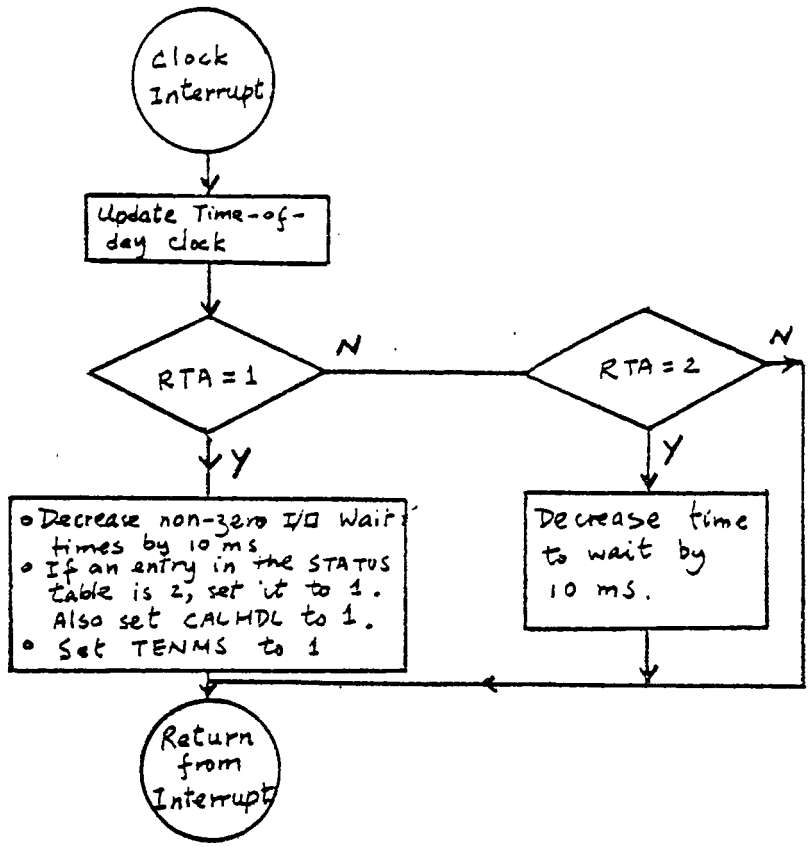


Figure 7.1: UPDATE Program Flowchart

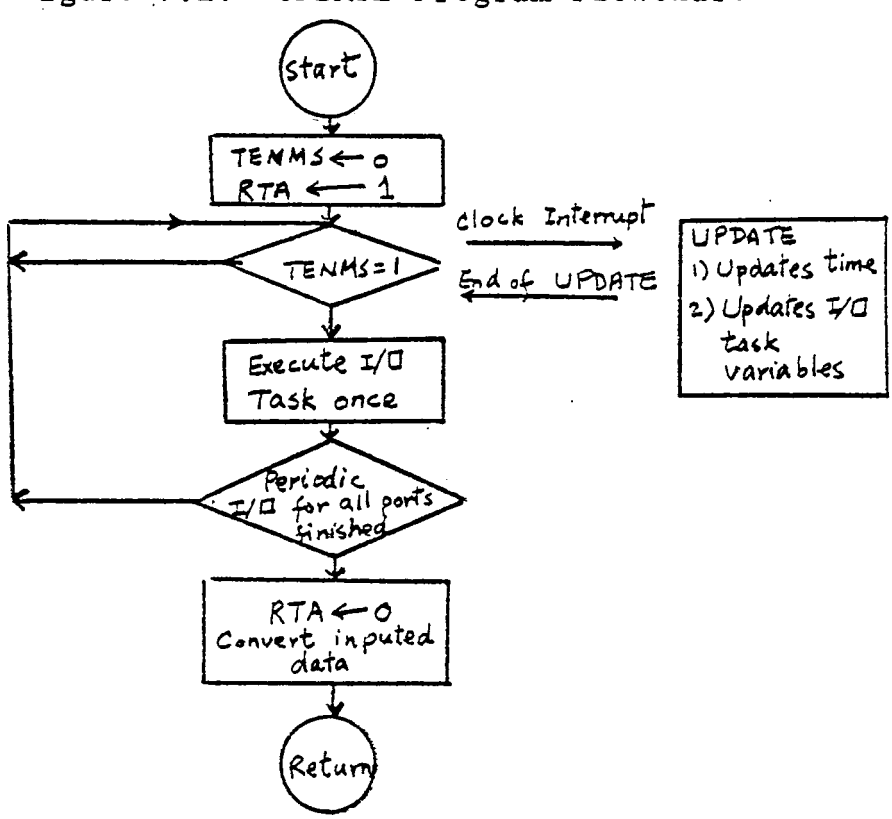


Figure 7.2: Simplified I/O Task Flowchart

When a clock interrupt occurs, UPDATE updates time as well as the variables of the I/O task. It also sets TENMS to 1 and then returns from interrupt. The I/O task detects that TENMS is equal to 1 and leaves the loop. It clears TENMS and executes the I/O task once. At the end of its cycle of execution, it enters a wait loop monitoring TENMS to be set by UPDATE.

The above procedure continues until the I/O task senses that all I/O has been completed. It clears RTA to inform UPDATE not to update its variables. The I/O task then converts all the input data and returns to the BASIC interpreter.

7.1.3 Activation Of I/O Ports By UPDATE

UPDATE activates the I/O ports by modifying the following variables of the I/O task:

1. CALHDL(call I/O handler)
2. STATUS(status of ports)
3. IOWTMS(I/O wait times of ports)

When a clock interrupt occurs and if RTA is equal to 1, UPDATE checks the STATUS table. If an entry is equal to 3, it means the I/O wait time of the port is at least 20 ms. UPDATE decrements the I/O wait time of the port by 10 ms. The I/O task later checks if the I/O wait time of the port has decreased to 10 ms and sets the entry of the port in STATUS to 2 if it has. If the entry of the port in STATUS is equal to 2, it means that the I/O wait time of the port is 10 ms. UPDATE clears the I/O wait time of the port and sets its entry in STATUS to 1 to signal the I/O task that the port is active. If an entry in STATUS is 0, periodic I/O for the port has finished. If any port becomes active, UPDATE sets CALHDL to 1.

7.2 Real-time I/O Task

This section describes the real-time I/O task which is executed by the RTX command. It is convenient to describe the major functions of the I/O task before describing its overall operation.

7.2.1 I/O Handler

The I/O handler performs the actual input and output of data from and to the lines of the system. It is called by the I/O task whenever any port becomes active. The I/O task detects this by reading CALHDL which is set to 1 by the UPDATE program whenever a port becomes active.

Whenever the I/O handler is called, it uses the parameters NUMPIC(number of parallel interface controllers) and PICADR(parallel interface controllers addresses) to read the values of all the system lines and store them into the I/O handler table LNVALT. The logical 'AND' of the contents of the I/O handler tables LNVALT and AOLINT is formed and stored in LNVALT. Then the logical 'OR' of the I/O handler table ODATAT and the modified LNVALT is formed and stored in LNVALT. The resulting values in LNVALT are then outputed to the system lines.

If only input is required, the bits of the AOLINT table should all be set to 1 and the bits of ODATAT should all be cleared so that the output lines of the system would not be affected.

7.2.2 Output Of Data

Output of data to output ports by the I/O task is done in two stages:

a. The I/O task prepares the AOLINT and ODATAT tables

of the I/O handler tables for all the output ports which will be active when the next clock interrupt occurs. These ports are characterised by a 2 in their STATUS bytes. An output port's STATUS byte will be equal to 2 if either the period of output is 10 ms or if the period is greater than 10 ms and UPDATE has decremented it to 10 ms. To prepare the tables, the I/O task first sets all the bits in AOLINT to 1. The PORTST table is read to determine the locations of the lines of the output ports. The bits in AOLINT that correspond to the lines of these output ports are cleared. Next, all the bits of ODATAT are cleared. Then, for each of the output ports the output data is read from its buffer area in the data buffer and then stored into the bits in ODATAT that correspond to the lines of the port.

b. When the next clock interrupt comes, UPDATE detects that these ports are active and it sets these ports' entries in the STATUS table to 1. CALHDL is set to 1 to inform the I/O task that output is required. The I/O task calls the I/O handler which uses the I/O handler tables prepared earlier by the I/O task to output to the lines of these output ports.

7.2.3 Input Of Data

When any input port becomes active, UPDATE informs the I/O task to call the I/O handler by setting CALHDL to 1. The I/O handler inputs the values of all the lines into LNVALT. The I/O task now checks the STATUS table to see which input ports are active. For each of these active input ports, the I/O task determines the locations of the lines of the

port from PORTST and selects the relevant data from LNVALT. The selected data is then stored into the buffer area reserved for the input port.

7.2.4 Conversion Of Input Data

When periodic I/O is finished for all the ports, the I/O task needs to convert all the data inputed. For each of the port performing periodic input, the I/O task reads its Periodic I/O table to see how many input words were inputed. The buffer pointer for the input port is re-adjusted to point to the first word inputed for the port. The inputed data for the port is then interpreted according to the data format of the input port and converted into BASIC interpreter's data format. The converted data is then stored into the BASIC vector reserved for the input port. The order of storage of the inputed data of the input ports in the BASIC vectors Q4 to Q9 is in the order that the periodic requests were made for these input ports.

7.2.5 Stopping The I/O Task

Each time the I/O task is executed and the periodic I/O is not finished, the I/O task checks the keyboard to see if the keys CONTROL C have been depressed. If they have, the I/O task stops and program control is passed to the BASIC interpreter command mode. Otherwise, the I/O task waits for the next clock interrupt to occur.

7.2.6 Overall Operation

The flowchart of the I/O task is as shown in figure 7.4.

When the RTX(real-time execute) command is given, program control is passed to the real-time I/O task. The I/O task clears TENMS and CALHDL and sets RTA to 1. It then

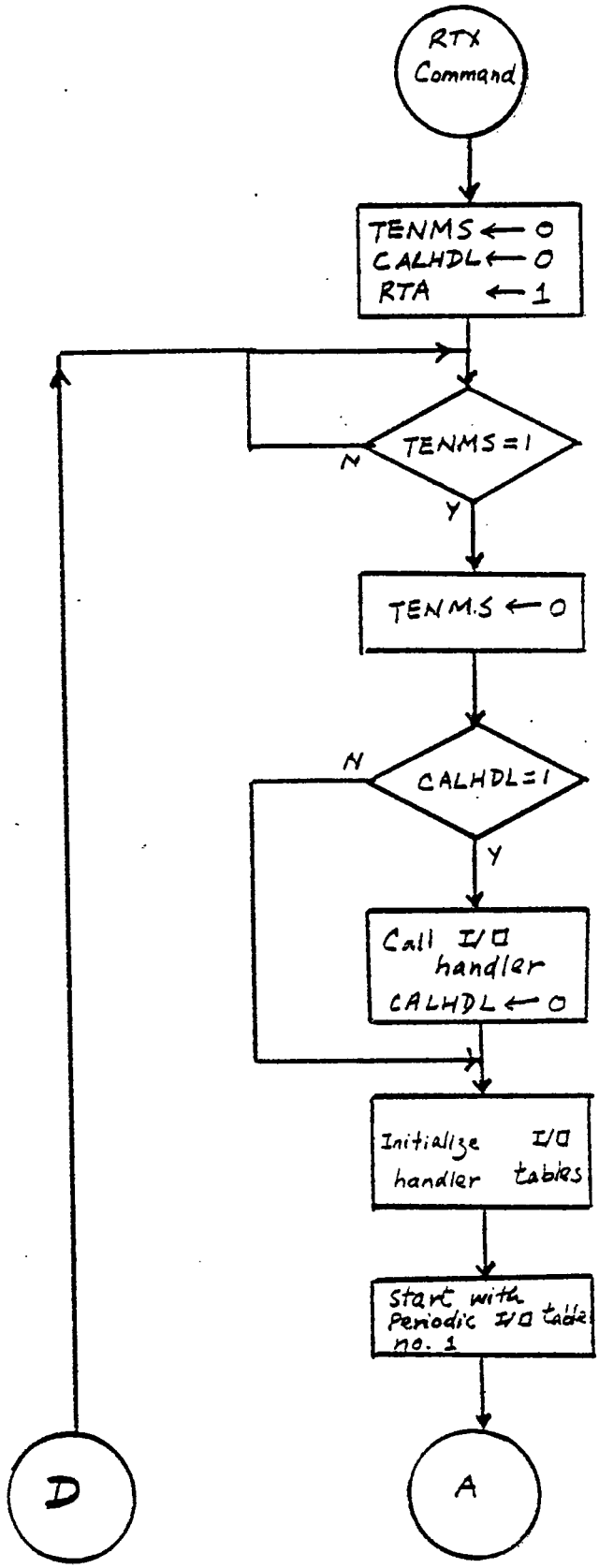


Figure 7.3: I/O Task Flowchart

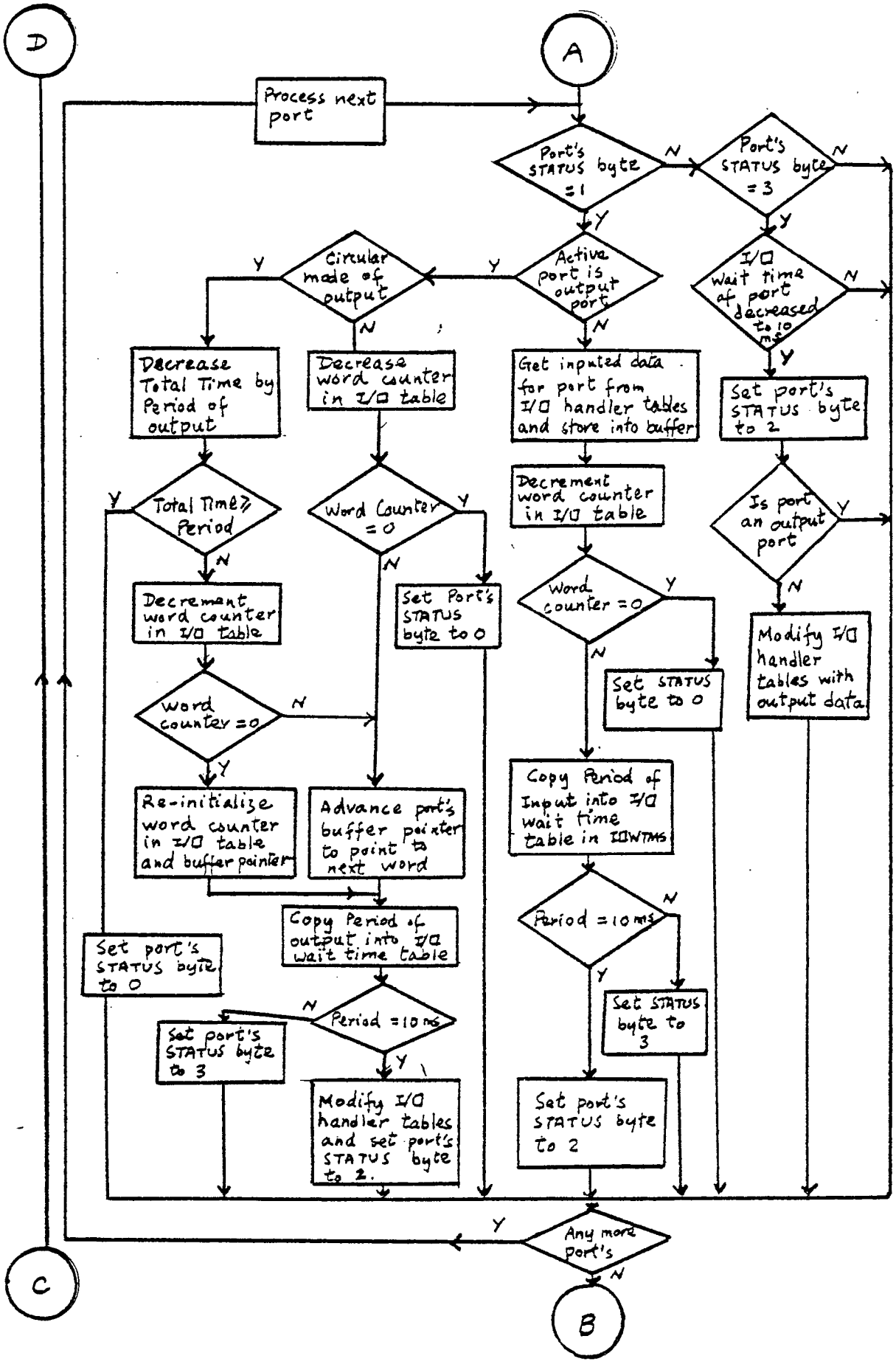


Figure 7.3(Continued): I/O Task Flowchart

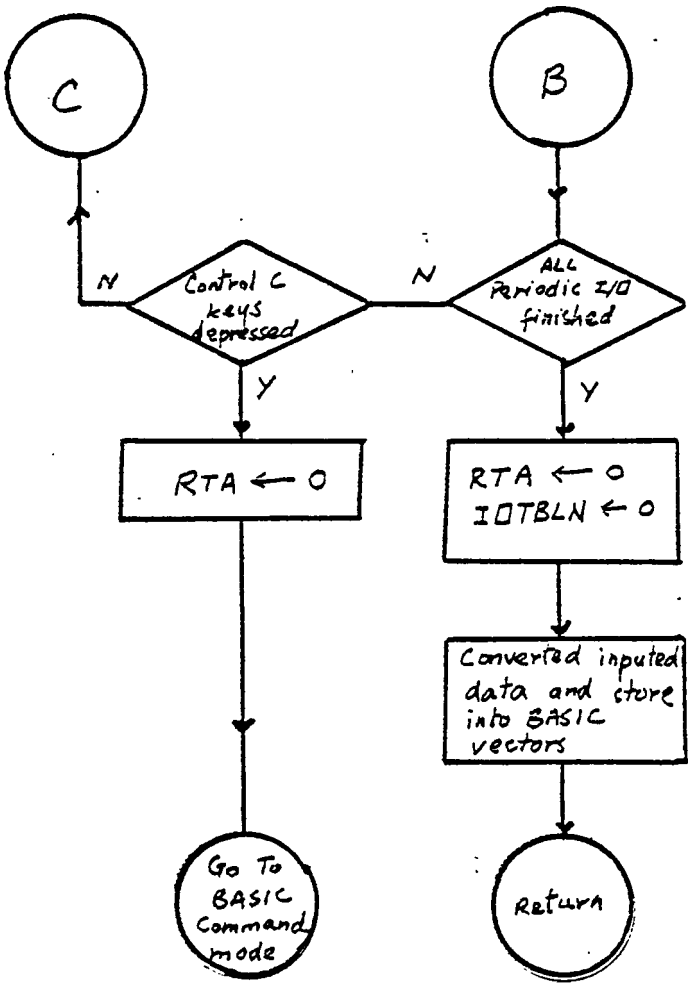


Figure 7.3(Continued): I/O Task Flowchart

enters a loop to check if TENMS is equal to 1. After this, the I/O task is executed whenever a clock interrupt occurs.

After a clock interrupt has occured, the I/O task clears TENMS and checks CALHDL to see if the I/O handler needs to be called. If CALHDL is equal to 1, the I/O task clears CALHDL and calls the I/O handler to read the values of the lines of the system into LNVALT and to output to the lines of the active output ports.

The I/O task then initializes the I/O handler tables by setting all the bits of AOLINT to 1 and clearing the bits of ODATAT. After this, the I/O task checks every port involved in periodic I/O to see what needs to be done for each of them. The number of ports involved in periodic I/O is indicated in the variable IOTBLN which contains the number of Periodic I/O tables assigned. Depending on the value of a port's entry in the STATUS table and whether it is an output or input port, different actions are performed for the port. The possible actions are listed below:

1. STATUS byte is equal to 3
   a. Output port:

      The I/O task checks if the I/O wait time of this port has been decremented to 10 ms by the UPDATE program. If it has, the port will be active at the next clock interrupt and the STATUS byte of the port is set to 2. Output data is read from the port's buffer area and the I/O handler tables are modified.
   b. Input port:

      Same as output port above except that the I/O handler tables need not be modified.

      The case that the STATUS byte is equal to 2 at this point of the I/O task does not exist because the byte would have been set to 1 by

UPDATE during the last clock interrupt.

2. STATUS byte is equal to 1

   a. Output port:

      The output port was outputed when the I/O handler was called at the beginning of the I/O task. The output mode of the port can be circular or single-shot, as indicated by byte 8 of the Periodic I/O table for the port.

      1. Single-shot mode- The I/O task decrements the counter for number of words(byte 13 of the I/O table). If it is not zero, output is not finished for the port and the buffer pointer is advanced to point to the next word to be outputed.

      2. Circular mode- The I/O task decreases the 'total time' of output by the amount 'period' of output. If the time left is greater than or equal to the 'period' of output, output for the port is not finished. In this case, the word counter is decremented. If the resulting contents of the word counter is not zero, the buffer pointer is advanced to point to the next word to be outputed. If the counter is zero, it is re-initialized to the 'number of words' requested and the buffer pointer is changed to point to the first word to be outputed. If output for the port is not finished, the 'period' of output stored in bytes 4 to 7 of the I/O table is copied into the 'I/O wait time' of the port. The STATUS byte for the port is set to 2 if the 'period' is equal to 10 ms. Otherwise, it is set to 3. If

output for the port is finished, the STATUS byte is cleared.

b. Input port:

In this case, the input port is active. The I/O task reads PORTST and retrieves the relevant data for the input port from the LNVALT table. The data is then stored into the buffer area reserved for the input port. The modification of the Periodic I/O table, the buffer pointer, the STATUS byte and the I/O wait time is similar to that of single-shot mode of output described above.

3. STATUS byte is equal to 0

In this case, periodic I/O for the output or input port is finished and no further action is necessary.

After the I/O task has processed all the ports, it checks if the periodic I/O for all ports is finished. If periodic I/O is not finished, it checks the keyboard to see if the keys CONTROL C have been depressed. If they have, the I/O task clears RTA and program control is passed to the BASIC command mode. If not, the I/O task waits for the next clock interrupt.

If all periodic I/O are finished, the I/O task clears RTA and converts all the inputed data. The I/O task then clears IOTBLN to inform the POR and PIR command modules that all the Periodic I/O tables are available again. Program control is then returned to the BASIC interpreter.

7.3 WAI Command Module

This module executes the WAI command to cause the BASIC program to stop its execution for a certain period

of time before continuing.

Execution of the WAI command causes program control to go to this module. The 'time to wait' in hours, minutes and seconds specified in the command is decremented by 1 second and then stored into the first 3 bytes of BWAITT(BASIC program wait time). The fourth byte of BWAITT, which contains the tens of ms is set to 100. The module then sets RTA to 2 to inform UPDATE that it is active and enters a loop to check if the fourth byte of BWAITT is 0.

Whenever a clock interrupt comes, UPDATE detects that RTA is equal to 2 and it decrements the fourth byte of BWAITT by 1. It then returns from interrupt.

When the module senses that the fourth byte of BWAITT is 0, it checks if the other 3 bytes are 0. If they are not, the time is decremented by 1 sec. and the fourth byte is set to 100. Then it enters the loop again. If all the bytes of BWAITT are 0, the module clears RTA and program control is returned to the BASIC interpreter.

CHAPTER 8

CONCLUSIONS

In this thesis, the design and implementation of a microprocessor based digital system which combines the power of the BASIC language with the capabilities of periodic parallel I/O have been presented. The system has been implemented and tested and it meets all the design specifications.

The system has been implemented with the Motorola 6800 microprocessor. It consists of a Micromodule 1A monoboard microcomputer, the MIKBUG system monitor, the SWTPC BASIC interpreter and the command programs described in this thesis.

The system has been applied to the automatic testing of student electronic laboratory practice at the University of Houston, as described in Appendix A. Without the system, the testing program would probably have been written in machine language. This approach would have required considerable programming effort. In addition, the testing program would have been difficult to change in the future when the tests for the students are changed. The use of this system provides an easy solution. The testing program has been written in BASIC without any code in machine language. All testing is performed by the BASIC program with 400 statements.

The command programs that provide the additional parallel I/O capabilities have been interfaced to the standard BASIC interpreter by modifying one of its output handlers. Since most BASIC interpreters have several output handlers, the command programs can also be added to other

BASIC interpreters. Similar systems using other microprocessors can be implemented by writing similar command programs in the machine languages of those microprocessors and modifying the output handlers of the BASIC interpreters for those microprocessors.

The command programs described in this thesis are modular in structure. Additional commands can be added to the system easily.

Use of the system has demonstrated that some improvements should be considered in future design revisions. The following two modifications are recommended:

1. This system does not support real-time data acquisition or control because it does not provide a frame work for real-time computations. In real-time data acquisition and control systems the values of control signals are often computed from the values of acquired data. Additional work should be done to allow BASIC programs to be configured as real-time tasks and an executive program to be incorporated in the system to schedule the task execution.

2. The BASIC interpreter in this system does not have file management capabilities. It is desirable to use an interpreter with such capabilities for the storing of output data and input data.

REFERENCES

1. 'Digital Computer Applications to Process Control', Proceedings of the 5th. International Conference of IFAC/IFIP, 1977.

2. Marce Eleccion, 'Automatic Test Equipment: Hardware and Software', Spectrum, June 1976.

3. John F. Barkley and Peter S. Shoenfeld, 'A Central Laboratory Automation Facility', American Laboratory, February 1975.

4. John S. Fok and Earl A. Abrahamson, 'A Large Computer System for On-line Data Acquisition and Analysis of Data from Analytical Instruments', American Laboratory, June 1975.

5. Texas Instruments, 'Process Control Language PCLA', Manual no. 955381-9701, 1976.

6. R. C. Ruhl, 'The Control Language Challenges High Level Compilers and Programmable Controllers', Control Engineering, November 1976.

7. H. A. Dorey, 'Digital Instrumentation', IEE Conference on Digital Instrumentation, November 1973.

8. 'IEEE Standard Digital Interface for Programmable Instrumentation', IEEE Standard 488-1975.

9. F. Liguori, 'The Test Language Dilemma', Proceedings of The Association for Computing Machinery National Convention, August 1971.

10. T. A. Ellison, 'ATLAS-Abbreviated Test Language for Avionics Systems', WESCON Convention Record, August 1971.

11. R. A. Grimm, 'An Automatic Test System Utilizing Standard Instruments and Abbreviated English Language', IEEE Automatic Support Systems Symposium Record, November 1968.

12. H. L. Fischer, A. G. Vallance, 'A Fortran Solution to Higher Order Test Language Requirement', IEEE Automatic Support Systems Symposium Record, November 1972.

13. C. E. Jones, 'Microprocessors-Impact on Controls', IEEE Transactions on Industrial Electronics and Control Instrumentation, August 1975.

14. 'Industrial Applications of Micropricessors', Proceedings of the Annual Conference of the IEEE Industrial Electronics and Control Instrumentation Group, 1977.

15. Thomas A. Seim, 'Microprocessors Aid Experimentation in Scientific Laboratory', Computer Design, September 1976.

16. F. J. Taylor, P. Mohan, P. Joseph, T. H. Pries, 'An All Digital Automated Wind Measurement System', IEEE Transactions on Instrumentation and Measurement, June 1977.

17. W. E. Paulsen, 'Common GSE under Microprocessor Control', IEEE AUTOTESTCON 1976.

18. General Electric, Mark III Foreground Reference Manual BASIC Language, Revision F, July 1973.

19. Eric R. Garen, 'Monolithic Data Conversion Devices, Part II: Analog to Digital Converters', Computer Design, April 1978.

20. Eric R. Garen, Danel M. Forsyth and Manolito E. Adan,

'Monolithic Data Conversion Devices, Part I: Digital to Analog Converters', Computer Design, March 1978.

21. Southwest Technical Products Corporation, 'SWTPC 8K BASIC VERSION 2.0 USER GUIDE', 1976.

22. Motorola Semiconductor Products Inc., 'MCM6830L7 MIKBUG/ MINIBUG ROM'.

23. Dong S. Kim, 'Panel for Automatic Testing of Students Laboratory Practice', MEE Project, University of Houston, May 1978.

APPENDIX A

AUTOMATIC TESTING OF STUDENTS' ELECTRONIC LABORATORY PRACTICE

This appendix describes an example in which the system is interfaced to a test box designed as a computer peripheral device for the automatic testing of a student's practical knowledge in the use of electronic instrumentation(23).

The test box provides different electrical signals for the student to measure depending on the control signals sent to it by the computer. Automatic testing of students is carried out by an interactive software program that provides communication between the system and the student without requiring instructor assistance. The software program monitors each testing sequence and a table of acceptable answers for each test.

A.1 Test Box Interface

A parallel digital interface is used between the test box and the computer. To the computer, the interface consists of four output ports and two input ports, as shown in figure A.1. The ports are listed below:

Output ports:

1. CIRCUT-4 lines. The output to this port selects a circuit out of 11 different possible circuits. This circuit will provide the electrical signal for the student to measure.
2. PARAM1-4 lines. The output to this port selects a value of a parameter of the circuit.
3. PARAM2-4 lines. The output to this port selects a value of another parameter of the circuit.

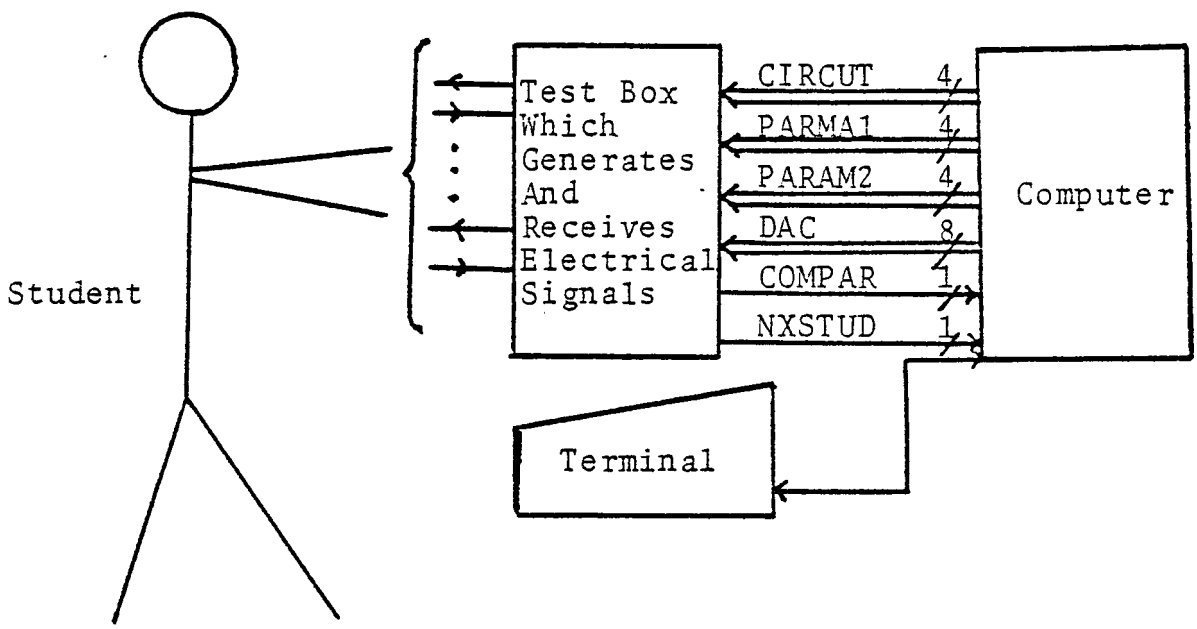


Figure A.1: Student Testing System

4. DAC -8 lines. The output to this port controls the voltage output of a DAC(digital to analog converter). The output is used by the circuit selected.

Input ports:

1. COMPAR-1 line. This is the input of a comparator which compares an unknown voltage with the voltage of the DAC. If the voltage of the DAC is greater than the unknown voltage, this input is 1. If it is smaller, the input is 0. COMPAR is used by the computer to determine a student's applied voltage using a successive approximation routine.
2. NXSTUD-1 line. This input is connected to a lock-switch. When the switch is turned, this input is 1. Otherwise it is 0. When the BASIC testing program finishes, it reads this input and waits for it to go to 1. When it goes to 1, the BASIC program jumps to the beginning of the program to test the next student.

A.2 Electronic Laboratory Practice Tests

There are three tests. Test 1 is DC testing, test 2 is AC testing and test 3 is Use Of Oscilloscope testing. The tests are summarized in tables A.1, A.2 and A.3. The test panel to which the electrical signals of the circuits are connected is shown in figure A.2. In the tables, the column called 'Output At CIRCUT' specifies the data that needs to be sent to CIRCUT in order to activate the particular circuit for the question that the student is answering. The columns 'Output At PARAM1', 'Output At PARAM2' and 'Output At DAC' specify the data that need to be sent to these ports for that

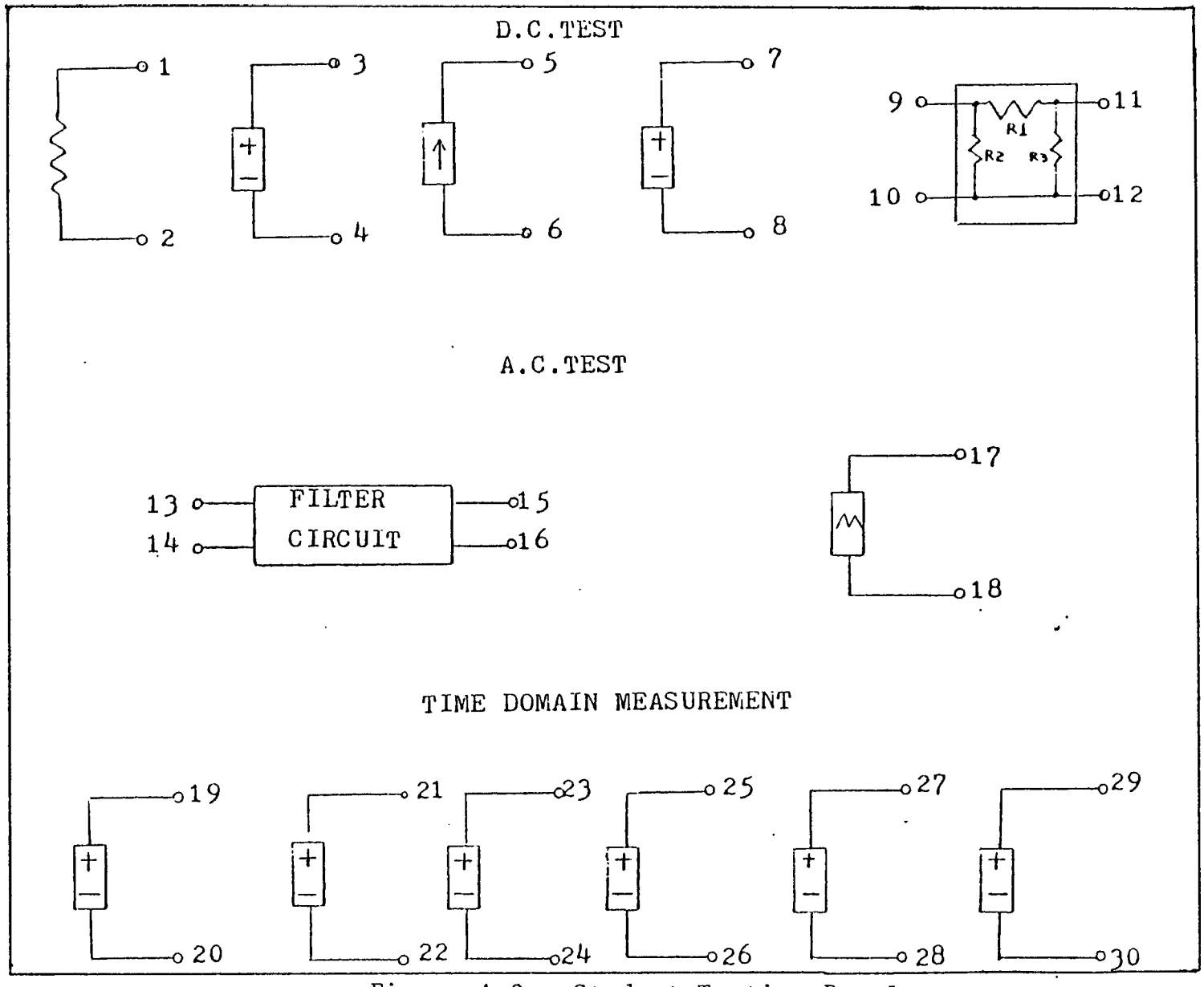


Figure A.2: Student Testing Panel

No.	Description	Output To CIRCUT	Possible Output To PARAM1	Possible Output To PARAM2	Possible Output To DAC	No. Of Possible Answers
1.1	Measure resistance between terminals 1 and 2.	0	0 to 15	——	——	16
1.2	Measure voltage between terminals 3 and 4.	1	——	——	0,10,20, ...,240	16
1.3	Measure current between terminals 5 and 6.	2	——	——	0,10,20, ...,240	16
1.4	Apply specified voltage between 7 and 8.	3	——	——	——	16
1.5	Find the value of the resistor R1 in the two port network shown at terminals 9, 10, 11 and 12.	4	0 to 15	0 to 15	——	R1 is 500 ohms
1.6	Find the value of the resistor R2 in the two port network.	4	0 to 15	0 to 15	——	16
1.7	Find the value of the resistor R3 in the two port network.	4	0 to 15	0 to 15	——	16

Table A.1: Summary Of Questions Of Test 1: DC Testing

No.	Description	Output To CIRCUT	Possible Output To PARAM1	Possible Output To PARAM2	Possible Output To DAC	No. Of Possible Answers
2.1	Apply 1 K hz sine wave at terminals 13 and 14 and measure gain at 15 and 16.	5	15	0,4,8, 12,15	——	5
2.2	Apply 6 K hz sine wave at terminals 13 and 14 and measure the phase difference between the output at 15, 16 and the input.	5	0,4,8, 12,15	15	——	5
2.3	Apply input at 13 and 14 and measure the cut-off frequency of the low pass filter at 13, 14, 15 and 16.	5	0,4,8, 12,15	15	——	5
2.4	Measure the RMS value of the waveform at terminals 17 and 18.	6	0 to 15	0	——	16
2.5	Measure the average value of the waveform at terminals 17 and 18.	6	0 to 15	0	——	16

Table A.2: Summary Of Questions Of Test 2:AC Testing

No.	Description	Output To CIRCUT	Possible Output To PARAM1	Possible Output To PARAM2	Possible Output To DAC	No. Of Possible Answers
3.1	Measure the DC voltage between the terminals 19 and 20.	1	——	——	0,10,20, ...,240	16
3.2	Measure the time difference the waveform at 23, 24 and the waveform at 21, 22.	7	0 to 15	0 to 15	——	16
3.3	Measure the time difference between the waveform at 25, 26 and the one at 21, 22.	7	0 to 15	0 to 15	——	16
3.4	Measure the peak to peak voltage of the waveform at 27 and 28.	8	0 to 15	0	——	16
3.5	Measure the rise time of the waveform at 29 and 30.	9	0 to 15	0	——	16

Table A.3: Summary Of Questions Of Test 3: Use Of Oscilloscope

question. Not all ports are used with each question. In these cases, a horizontal bar denotes the unused port. The column 'Number Of Possible Answers' denotes the number of different values of the electrical variable which the student may be measuring in that question.

In all of these questions except the 'Supply Voltage' question in test 1, the student types the measured value at the keyboard. In the 'Supply Voltage' question, the computer measures the supplied voltage by varying the output to the port DAC and reading the port COMPAR.

The possible values of the electrical variables that are generated in the questions of the tests are documented in (23).

A.3 Rules For Taking Tests

1. A student can choose to take test 1, 2 or 3.
2. If a student passes test 1 or 2, he can choose to proceed to the next test. If he fails a test, the system informs him so and halts, waiting for a lock-switch to be turned(by the instructor).
3. A student is allowed 4 mistakes in each of the tests. If a student makes a first mistake on a question and the allowable number of mistakes is not exceeded, he is required to try it again.

A.4 Instructor Procedure

The instructor turns on the power of the system and loads in the BASIC testing program. Then he sets the system time and types 'RUN'. The system is now ready for the student to take the tests. After the student finishes the test or tests or if he fails, the system halts and waits for a

lock-switch to be turned(by the instructor). By turning the switch, the system will be ready to test the next student again.

A.5 BASIC Test Program

This section describes the overall operation of the BASIC testing program that performs the automatic testing of students. The details of how a test is given are described in the next section. The general flowchart of the testing program is as shown in figure A.3.

The variables Q0, Q1 and the vector Q2 are first referenced in the program. Then the lines of the computer are configured to the way that the test box is connected to the computer system. In the configuration as indicated in the program, lines 1 to 4 of the system are used for the port CIRCUT, lines 5 to 8 for PARAM1, lines 9 to 12 for PARAM2, lines 13 to 20 for DAC, line 21 for COMPAR and line 22 for NXSTUD.

The program prints 'ELECTRONIC LABORATORY PRACTICE' and asks the student to type his name and student number. After the student has given this information, the program prints the time of the day and the rules for taking the tests. It then asks the student for the test he wishes to take. After the student types the number of the test that he wishes to take, the program starts giving the student that test.

In taking a test, a student may fail or pass it. If he fails, the program informs him so and waits for the lock-switch to be turned. If he passes test 1 or test 2, the program asks him if he wants to continue with the next test. If he types in 'Y', the next test is given. If a 'N' is entered, the program halts and waits for the lock-switch to

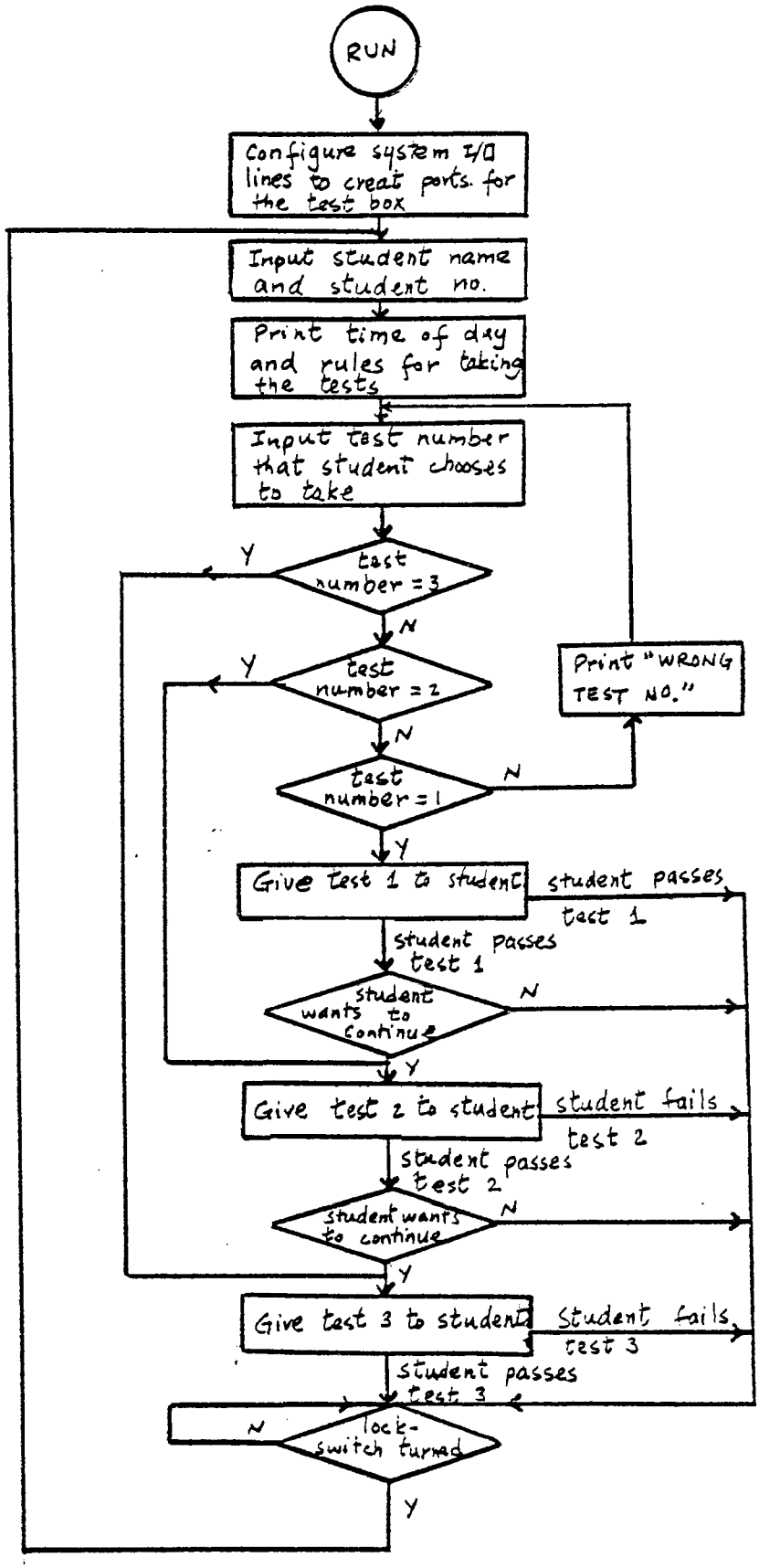


Figure A.3: BASIC Testing Program Flowchart

be turned. If a student passes test 3, there is no next test and the program waits for the lock-switch to be turned. When the lock-switch is turned, the program jumps back to the point where it prints 'ELECTRONIC LABORATORY PRACTICE' so that another student can be tested.

The BASIC testing program is listed in figure A.5 and a print-out of a student who took the tests is shown in figure A.6.

A.6 Giving A Test

The flowchart for giving a test is shown in figure A.4.

The questions in a test are given to the student one after the other. The program first sends data to CIRCUT to activate the electronic circuit for the question. If parameters and/or voltage are required for the electronic circuit, data are generated and then sent to the ports PARAM1, PARAM2 and DAC. The program next prints a message on the console to inform the student which question he is answering and what he has to do in answering that question. It then waits for the answer. After the answer is obtained, 4 things may happen:

1. The answer is correct. In this case, the program goes on to the next question.
2. The answer is the first wrong answer for the question and the student has not exceeded the number of mistakes allowed in the test. In this case, the program instructs the student to try the question again.
3. The answer is the second wrong answer and the student has not exceeded the total number of mistakes allowed. In this case, the program prints a code which represents the correct answer for

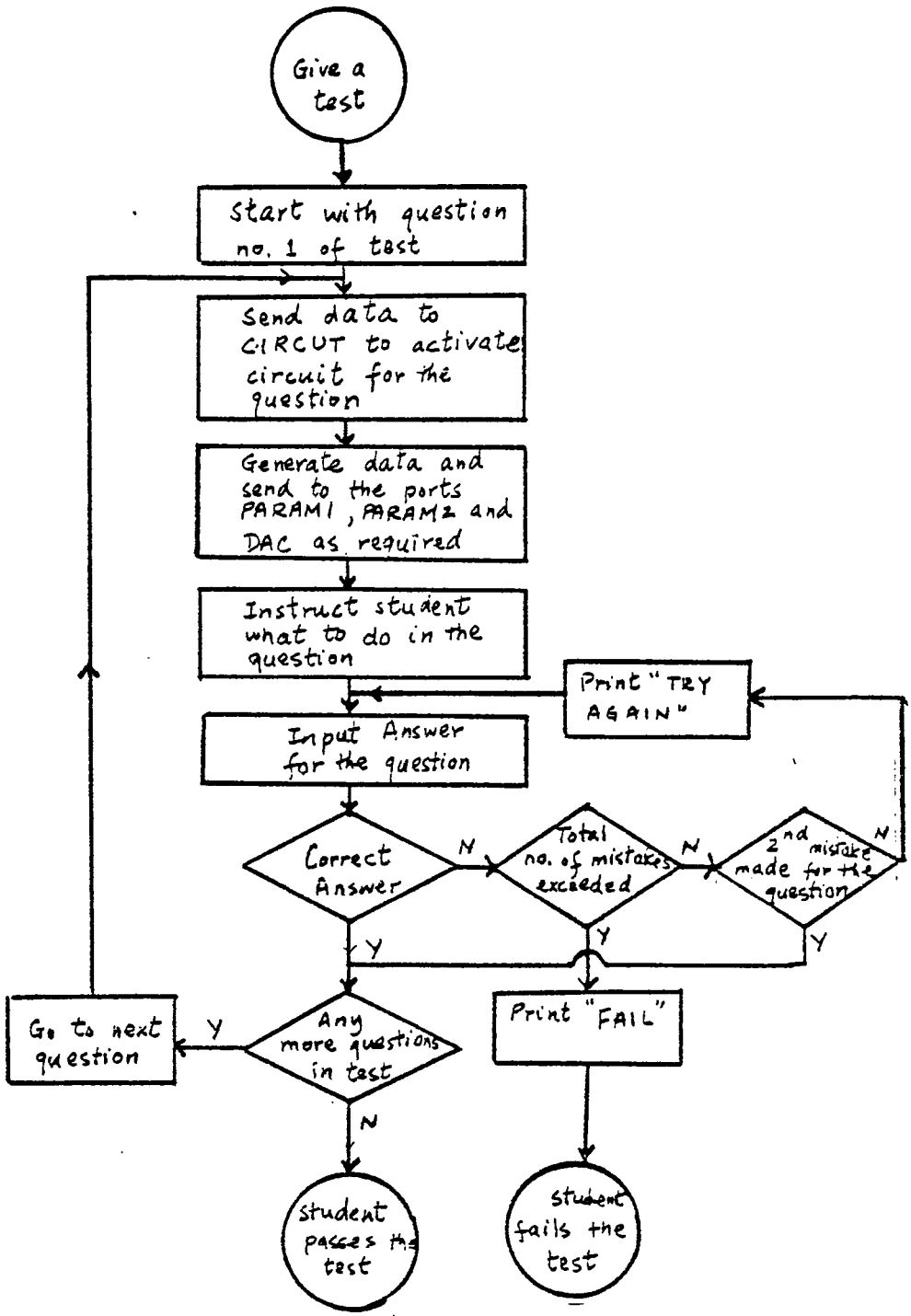


Figure A.4: Giving A Test Flowchart

the instructor's use. Then it goes on to give the next question.

4. The answer is wrong and the student has exceeded the maximum allowable number of mistakes. In this case, the program prints the code for the correct answer, informs the student that he fails the test and waits for the lock-switch to be turned.

When there are no additional questions to be given for the test, the student passes the test.

```
0005 REM ***CØNFIGURE SYSTEM LINES TØ INTERFACE TEST BØX***
0010 PRINT #5,"CØN DIAGNØ,ØB,1,4"
0020 PRINT #5,"CØN CIRCUT,ØB,5,8"
0030 PRINT #5,"CØN PARAM1,ØB,9,12"
0035 PRINT #5,"CØN PARAM2,ØB,13,16"
0040 PRINT #5,"CØN DAC,ØB,17,24"
0045 PRINT #5,"CØN NXSTUD,IB,31"
0050 PRINT #5,"CØN CØMPAR,IB,32"
0060 REM ***DIAGNØ IS FØR DIAGNØSTICSØFTHE TEST BØX
0061 REM ***DIAGNØ SHØULD BE HIGH WHEN A STUDENT IS TESTED
0070 PRINT #5,"ØUT DIAGNØ,15"
0071 REM ***REFERENCE THE VARIABLES Q'S FIRST***
0072 LET Q0=Q1
0073 DIM Q2(3),F(16)
0074 PRINT "ELECTRØNIC INSTRUMENTATIØN TESTS"
0075 PRINT "ENTER NAME AND STUDENT NØ."
0076 REM ***INPUT STUDENT NAME AND STUDENT NØ.***
0077 INPUT A$
0078 REM ***READ AND PRINT TIME*
0079 PRINT #5,"TMR N"
0080 PRINT "TIME: ";Q2(1);"HØUR ";Q2(2);"MIN ";Q2(3)
0082 REM ***PRINT TEST RULES***
0083 PRINT "*THERE ARE 3 TESTS"
0085 PRINT "*TEST NØ. 1: DC TESTING, 7 QUESTIØNS"
0090 PRINT "*TEST NØ. 2: AC TESTING, 5 QUESTIØNS"
0095 PRINT "*TEST NØ. 3: USE ØF ØSCILLØSCØPE, 5 QUESTIØNS"
0098 PRINT "*YØU CAN START WITH TEST 1, 2 ØR 3"
0100 PRINT "*YØU CAN MAKE AT MØST 4 MISTAKES ØN A TEST"
0105 PRINT "*IF YØU MAKE A FIRST MISTAKE ØN A QUESTIØN AND"
0106 PRINT " YØU HAVE NØT EXCEEDED 4 MISTAKES IN THE TEST,"
0107 PRINT " YØU ARE REQUIRED TØTRY IT AGAIN"
0108 PRINT "*PRESS CØNTRØL C TØ DELETE A WRØNG ENTRY"
0110 PRINT "*DØ NØT ENTER UNITS WHEN GIVING ANSWERS"
0111 REM "**RESTØRE DATA PØINTER***
0112 RESTØRE
0114 PRINT "ENTER THE NØ. ØF TEST YØU WANT TØ TAKE"
0116 INPUT A
0117 REM ***GIVE TEST 1***
0118 IF A=1GØTØ140
0119 REM ***GIVE TEST 2***
0120 IF A=2GØTØ1180
0121 REM ***GIVE TEST 3***
0122 IF A=3GØTØ1685
0124 PRINT "WRØNG TEST NØ."
0125 GØTØ 116
0126 REM ***IN THE FØLLØWING PRØGRAM, WHEN THE SUBRØUTINE***
0127 REM ***AT 4000 IS CALLED, A, B, C AND D CØNTAIN THE***
0128 REM ***DATA TØ BE SENT TØ CIRCUT,PARAM1,PARAM2 AND DAC**
```

Figure A.5: Student Testing Program

```
0129 REM ***RESPECTIVELY. P CØNTAINS THE NØ. ØF PØSSIBLE***
0130 REM ***ANSWERS AND R PØINTS TØ THE CØRRECT ANSWER***
0137 REM ***TEST 1: DC TESTING***
0137 PRINT "TEST 1: DC TESTING"
0139 REM ***M IS TØTAL NØ. ØF MISTAKES.G=1 STUDENT FAILS***
0140 LET M=0
0145 LET G=0
0150 REM ***1.1: MEASURE RESISTANCE***
0170 LET A=0
0180 LET B=INT(RND*16)
0190 LET C=0
0200 LET D=128
0205 LET P=16
0207 LET R=B+1
0210 PRINT "1.1:MEASURE R BETWEEN 1 & 2 IN K ØHMS"
0220 DATA 8.19,7.71,7.15,6.67,6.09,5.6,5.05,4.57
0230 DATA 4.11,3.62,3.08,2.59,2.01,1.52,.98,.49
0260 GØSUB 4000
0265 IF G=1GØTØ2400
0270 REM ***1.2 MEASURE VØLTAGE***
0280 LET A=1
0290 LET R=INT(RND*16)+1
0300 LET D=(R-1)*16
0310 PRINT "1.2:MEASURE VØLTAGE BETWEEN 3 & 4 IN VØLTS"
0320 DATA -10,-8.76,-7.52,-6.62,-5,-3.74,-2.49,-1.23
0330 DATA .06,1.32,2.57,3.83,5.1,6.36,7.61,8.87
0360 GØSUB 4000
0365 IF G=1GØTØ2400
0400 REM ***1.3 MEASURE CURRENT***
0410 LET A=2
0420 LET R=INT(RND*16)+1
0430 LET D=(R-1)*16
0450 PRINT "1.3:MEASURE CURRENT BETWEEN 5 & 6 IN MA."
0460 DATA -10,-8.75,-7.5,-6.24,-4.98,-3.72,-2.47,-1.22
0470 DATA .06,1.31,2.56,3.8,5.06,6.32,7.57,8.83
0480 GØSUB 4000
0490 IF G=1GØTØ2400
0500 REM ***1.4 SUPPLY VØLTAGE***
0501 REM ***SET NØ. ØF ATTEMPTS TØ 0***
0502 LET N=0
0505 PRINT #5,"ØUT CIRCUT,3"
0510 LET V=2*INT(RND*10)-9
0520 PRINT "1.4: SUPPLY ";V;"VØLTS BETWEEN 7 AND 8"
0530 PRINT "      ENTER R C/R WHEN READY"
0540 INPUT A$
0560 PRINT #5,"ØUT DAC,255"
0570 PRINT #5,"INP CØMPAR"
0575 REM ***Q1=0 MEANS VØLTAGE>10 VØLTS***
```

Figure A.5(Continued): Student Testing Program

```
0580 IF Q1=1GØTØ590
0582 PRINT "SUPPLIED VØLTAGE GREATER THAN 10 VØLTS."
0584 GØTØ 790
0590 PRINT #5,"ØUT DAC,0"
0600 PRINT #5,"INP CØMPAR"
0605 REM ***Q1=1 MEANS VØLTAGE < -10 VØLTS***
0610 IF Q1=0GØTØ630
0612 PRINT "SUPPLLIED VØLTAGE LESS THAN 10 VØLTS"
0614 GØTØ 790
0620 REM ***SUCCESSIVE APPRØXIMATIØN TØ MEASURE PPLIED***
0621 REM ***VØLTAGE***
0630 LET Q0=128
0635 LET A=64
0640 PRINT #5,"ØUT DAC"
0660 PRINT 5,"INP CØMPAR"
0670 IF Q1=1GØTØ700
0680 LET Q0=Q0+A
0690 GØTØ 710
0700 LET Q0=Q0-A
0710 IF A=1GØTØ740
0720 LET A=A/2
0725 GØTØ 640
0730 REM **Q0 CØNTAINS DIGITAL EQUIVALENT ØF APPLIED VØLTAGE*
0735 REM ***CØNVERT Q0 TØ ANALØG EQUIVALENT***
0740 LET A=20*Q0/255-10
0742 REM ***PRINT 2 DIGITS RIGHT ØF DECIMAL PØINT***
0743 DIGITS= 2
0745 PRINT A;"VØLTS IS SUPPLIED"
0746 DIGITS= 0
0747 REM ***10 % ERRØR ALLØWED***
0748 IF V<0GØTØ765
0750 IF A>V*1.1GØTØ790
0760 IF A<V*.9GØTØ790
0762 GØTØ 770
0765 IF A>V*.9GØTØ790
0767 IF A<V*1.1GØTØ790
0770 PRINT "CØRRECT"
0780 GØTØ 1000
0790 PRINT "INCØRRECT"
0800 LET M=M+1
0810 IF M<5GØTØ820
0815 PRINT "5 MISTAKES MADE, FAIL"
0817 GØTØ 2400
0820 LET N=N+1
0830 IF N=2GØTØ1000
0840 PRINT "TRY AGAIN"
0850 GØTØ 540
1000 REM ***1.5 MEASURE R1 ØF TWØ PØRT NETWØRK***
```

Figure A.5(Continued): Student Testing Program

```
1010 LET A=4
1020 LET R1=INT(RND*16)+1
1030 LET R2=INT(RND*16)+1
1040 LET B=R1-1
1042 LET C=R2-1
1044 LET P=1
1045 LET R=1
1046 PRINT "1.5:WHAT IS R1 IN 2 PØRT NETWØRK AT 9,10,11&12?"
1047 DATA .51
1048 GØSUB 4000
1049 IF G=1GØTØ2400
1050 REM ***1.5 MEASURE R2***
1055 LET P=16
1060 LET R=R1
1070 PRINT "1.6:WHAT IS R2 IN SAME NETWØRK IN K ØHMS?"
1080 DATA 7.92,7.44,6.9,6.46,5.81,5.33,4.84,4.35
1010 DATA 4.05,3.57,3.08,2.59,1.94,1.46,.97,.48
1100 GØSUB 4000
1112 IF G=1GØTØ2400
1114 REM ***1.7 MEASURE R3***
1115 LET R=R2
1120 PRINT "1.7:WHAT IS R3 IN THE SAME NETWØRK IN K ØHMS?"
1130 DATA 8.11,7.68,7.13,6.66,6.02,5.56,5,4.51
1140 DATA 4.24,3.75,3.25,2.73,2.05,1.55,1.03,.52
1150 GØSUB 4000
1152 IF G=1GØTØ2400
1154 REM ***END ØF TEST 1***
1155 PRINT "YØU PASS TEST 1. DØ YØU WANT TØ CØNTINUE TEST 2?"
1156 PRINT "ENTER 'Y' ØR 'N'"
1160 INPUT A$
1164 REM ***IF 'Y' IS ENTERED, GØTØ TEST 2***
1165 IF A$="Y"GØTØ1192
1169 REM ***WAIT FØR SWITCH TØ BE TURNED***
1170 GØTØ 2400
1175 REM ***TEST 2: AC TESTING***
1176 REM ***MØVE DATA PØINTER TØTEST 2 DATA***
1180 FØR N=1TØ81
1185 READ A
1190 NEXT N
1192 PRINT "TEST 2: AC TESTING"
1195 LET M=0
1196 LET G=0
1198 REM ***2.1 MEASURE GAIN***
1199 REM ***SELECT 1 ØUT ØF 5 ØUTPUTS TØ PARAM1 RANDØMLY***
1200 LET A=5
1210 DATA 0,4,8,12,15
1220 FØR P=1 TØ 5
1230 READ F(P)
```

Figure A.5(Continued): Student Testing Program

```
1240 NEXT P
1250 LET R=INT(RND*5)+1
1260 LET C=F(R)
1265 LET B=15
1270 LET P=5
1282 PRINT "2.1:APPLY 1 K HZ SINE WAVE AT 13 AND 14 AND"
1283 PRINT "    MESURE GAIN AT ØUTPUT AT 15, 16 IN DB."
1290 DATA 16.69,13.39,9.95,7.43,4.45
1300 GØSUB 4000
1305 IF G=1GØTØ2400
1310 REM ***2.2 MEASURE PHASE***
1315 LET C=15
1320 DATA 0,4,8,12,15
1330 FØR P=1TØ5
1340 READ F(P)
1350 NEPT P
1352 LET R=INT(RND*5)+1
1353 LET B=F(R)
1354 LET P=5
1355 PRINT "2.2:IN SAME CIRCUIT,APPLY 6KHZ SINE WAVE AT INPUT"
1356 PRINT "    AND MEASURE PHASE DIFFERENCE BETWEEN INPUT"
1357 PRINT "    AND ØUTPUT IN DEGREES"
1358 DATA 70,62,48,28,11
1360 GØSUB 4000
1370 IF G=1GØTØ2400
1375 REM ***2.3 MEASURE CUT-ØFF FREQUENCY***
1380 DATA 0,4,8,12,15
1390 FØR P=1TØ5
1400 READ F(P)
1410 NEXT P
1420 LET R=INT(RND*5)+1
1430 LET B=F(R)
1440 LET P=5
1450 PRINT "2.3:IN SAME CIRCUIT,MEASURE CUTØFF FREQUENCY IN"
1460 PRINT "    KHZ."
1470 DATA 2.4,3.7,5.3,8.5,17.2
1480 GØSUB 4000
1490 IF G=1GØTØ2400
1500 REM ***2.4 MEASURE RMS VØLTAGE***
1510 LET A=6
1520 LET B=INT(RND*16)
1525 LET C=0
1530 LET P=16
1540 LET R=B+1
1550 PRINT "2.4 WHAT IS THE RMS VALUE ØF THE WAVEFØRM AT 17"
1555 PRINT "    & 18?"
1560 DATA 2.18,2.05,1.9,1.77,1.61,1.48,1.33,1.19
1570 DATA 1.07,.94,.79,.66,.5,.37,.23,.11
```

Figure A.5(Continued): Student Testing Program

```
1580 GØSUB 4000
1585 IF G=1GØTØ2400
1590 REM ***2.5 MEASURE AVERAGE VØLTAGE***
1610 PRINT "2.5:WHAT IS THE AVERAGE VALUE ØF THE SAME "
1611 PRINT "    WAVEFØRM IN VØLTS?"
1620 DATA .66,.63,.58,.53,.49,.44,.40,.35
1630 DATA .33,.29,.25,.20,.16,.12,.08,.03
1640 GØSUB 4000
@1642 IF G=1GØTØ2400
1643 REM ***END ØF TEST 2***
1645 PRINT "YØU PASS DEST 2. DØ YØU WANT TØ GØ TØ TEST 3?"
1646 PRINT "ENTER 'Y' ØR 'N'"
1650 INPUT A$
1660 IF A$="Y"GØTØ1720
1665 GØTØ 2400
1680 REM ***TEST 3: USE ØF ØSCILLØSCØPE***
1684 REM ***MØVE DATA PØINTER TØ TEST 3 DATA***
1685 FØR N=1TØ143
1690 READ A
1695 NEXT N
1720 PRINT "PART 3: USE ØF ØSCILLØSCØPE"
1725 LET M=0
1726 LET G=0
1728 REM ***3.1 MEASURE VØLTAGE***
1730 LET A=1
1735 LET B=0
1740 LET C=0
1745 LET R=INT(RND*16)+1
1750 LET D=(R-1)*16
1760 LET P=16
1770 PRINT "3.1: USE ØSCILLØSCØPE TØ MEASURE VØLTAGE AT 19 &"
1771 PRINT "    20 AND GIVE ANSWER IN VØLTS."
1780 DATA -10,-8.76,-7.52,-6.62,-5,-3.74,-2.49,-1.23
1790 DATA .06,1.32,2.57,3.83,5.1,6.36,7.61,8.87
1800 GØSUB 4000
1805 IF G=1GØTØ2400
1810 REM ***3.2 MEASURE TIME DIFFERENCE T1***
1820 LET A=8
1830 LET R1=INT(RND*16)+1
1840 LET R2=INT(RND*16)+1
1850 LET B=R1-1
1855 LET C=R2-1
1860 LET P=16
1870 LET R=R1
1880 PRINT 3.2: WHAT IS THE TIME DIFFERENCE BETWEEN THE"
1881 PRINT "    WAVEFØRMS AT 21,22 AND 23,24 IN MS?"
1900 DATA 0,.11,.22,.33,.44,.55,.66,.77
1910 DATA .88,.99,1.1,1.21,1.32,1.43,1.54,1.65
```

Figure A.5(Continued): Student Testing Program

```
1920 GØSUB 4000
1925 IF G=1GØTØ2400
1928 REM ***3.3 MEASURE TIME DIFFERENCE T2***
1930 LET R=R2
1940 PRINT "3.3: WHAT IS THE TIME DIFFERENCE BETWEEN THE"
1950 PRINT "     WAVEFØRMS AT 21,22 AND 26,27 IN MS?"
1960 DATA 0,.11,.22,.3#,.44,.55,.66,.77
1970 DATA .88,.99,1.1,1.21,1.32,1.43,1.54,1.65
1980 GØSUB 4000
1990 IF G=1GØTØ2400
2100 REM ***3.4 MEASURE PEAK TØ PEAK VØLTAGE***
2110 LET A=7
2120 LET B=INT(RND*16)
2125 LET C=0
2130 LET P=16
2140 LET R=B+1
2150 PRINT "3.4: WHAT IS THE PEAK TØ PEAK VØLTAGE ØF"
2151 PRINT "     THE WAVEFØRM AT 27,28 IN VØLTS?"
@2160 DATA 7.8,7.2,6.8,6.4,5.6,5.2,4.7,4.2,3.8
2170 DATA 3.4,2.8,2.4,1.9,1.4,.9,.46
2180 GØSUB 4000
2190 IF G=1GØTØ2400
2200 REM ***3.5 MEASURE RISETIME***
2210 LET A=9
2220 LET B=INT(RND*16)
2230 LET P=16
2240 LET R=B+1
2250 PRINT "3.5: WHAT IS THE RISETIME ØF THE WAVEFØRM AT"
2251 PRINT "     29,30 IN MICRØSEC?"
2260 DATA 175,165,154,142,130,122,114,105
2270 DATA 95,85,75,63,50,36,23,10
2280 GØSUB 4000
2285 IF G=1GØTØ2400
2288 REM ***END ØF TEST 3***
2290 PRINT "YØU PASS TET 3"
2400 PRINT #5,"TMR N"
2402 PRINT "TIME: ";Q2(1);"HØUR ";Q2(2);"MIN ";Q2(3);"SEC"
2406 PRINT "ASK INSTRUCTØR TØ RE-START SYSTEM"
2410 REM ***WAIT FØR INSTRUCTØR TØ TURN SWITCH***
2420 PRINT #5,"INP NXSTUD"
2430 IF Q1=0GØTØ2420
2440 GØTØ 74
3995 REM ***SUBRØUTINE TØ ØUTPUT A, B, C, C TØ CIRCUT,***
3996 REM ***PARAM1, PARAM2, DAC. ALSØ TØ INPUT ANSWER AND***
3997 REM ***CHECK IF ANSWER IS CØRRECT***
3999 REM ***ØUTPUT DATA***
4000 LET Q0=A
4010 PRINT #5,"ØUT CIRCUT"
```

Figure A.5(Continued): Student Testing Program

```
4020 LET Q0=B
4030 PRINT #5,"ØUT PARAM1"
4040 LET Q0=C
4042 PRINT #5,"ØUT ØARAM2"
4045 LET Q0=D
4050 PRINT #5,"ØUT DAC"
4055 REM ***SET NØ. ØF ATTEMPTS TØ 0***
4060 LET N=0
4065 REM ***READ PØSSIBLE ANSWERS***
4070 FØR T=1TØP
4080 READ F(T)
4090 NEXT T
4095 PRINT F(R)
4097 REM ***READ ANSWER FRØM KEYBØARD***
4098 INPUT S
4099 REM **CHECK IF ANSWER IS CØRRECT.10 % ERRØR ALLØWED***
4100 REM ***IF CØRRECT ANSWER IS PØSITIE ØR NEGATIVE***
4101 IF F(R)>0GØTØ4110
4102 REM NEGATIVE ANSWER
4103 IF F(R)=0GØTØ4125
4104 IF S<F(R)*1.1GØTØ4150
4105 IF S>F(R)*.9GØTØ4150
4106 GØTØ 4130
4110 IF S>F(R)*1.1GØTØ4150
4120 IF S<F(R)*.9GØTØ4150
4122 GØTØ 4130
4123 REM ***NO. BETWEEN .1 AND -.1 IS CØNSIDERED***
4124 REM ***CØRRECT IF CØRRECT ANSWER IS 0.***
4125 IF S>.1GØTØ4150
4126 IF S<-.1GØTØ4150
4130 PRINT "CØRRECT"
4140 RETURN
4150 PRINT "INCØRRECT"
4160 LET M=M+1
4165 REM ***IF MADE 5 MISTAKES, FAIL***
4170 IF M=5GØTØ4240
4180 LET N=N+1
4185 REM ***IF 2ND ATTEMPT, GØ TØ NEXT QUESTIØN***
4190 IF N=2GØTØ4220
4195 REM ***IF 1ST ATTEMPT, TRY AGAIN***
4200 PRINT "TRY AGAIN"
4210 GØTØ 4098
4220 PRINT "CØRRECT ANSWER CØDE: ";R
4230 RETURN
4240 PRINT "CØRRECT ANSWAR CØDE: ";R
4242 PRINT "5 MISTAKES MADE, FAIL"
4245 REM ***INDICATE STUDENT FAIL***
4250 LET G=1
4260 RETURN
```

Figure A.5(Continued): Student Testing Program

```
ELECTRØNIC INSTRUMENTATIØN TESTS
ENTER NAME AND STUDENT NØ.
? TAI-LØY KØNG, 218735
TIME:  10 HØUR  30 MIN  29 SEC
*THERE ARE 3 TESTS
*TEST NØ. 1: DC TESTING, 7 QUESTIØNS
*TEST NØ. 2: AC TESTING, 5 QUESTIØNS
*TEST NØ. 3: USE ØF ØSCILLØSCØPE, 5 QUESTIØNS
*YØU CAN START WITH TEST 1, 2 ØR 3
*YØU CAN MAKE AT MØST 4 MISTAKES ØN A TEST
*IF YØU MAKE A FIRST MISTAKE ØN A QUESTIØN AND
 YØU HAVE NØT EXCEEDED 4 MISTAKES IN THE TEST,
 YØU ARE REQUIRED TØ TRY IT AGAIN
*PRESS CØNTRØL C TØ DELETE A WRØNG ENTRY
*DØ NØT ENTER UNITS WHEN GIVING ANSWERS
ENTER THE NØ. ØF TEST YØU WANT TØ TAKE
? 4
WRØNG TEST NØ.
? 1
1.1:MEASURE R BETWEEN 1 & 2 IN K ØHMS

? 3
INCØRRECT
TRY AGAIN
? 2.6
CØRRECT
1.2:MEASURE VØLTAGE BETWEEN 3 & 4 IN VØLTS

? 5
CØRRECT
1.3:MEASURE CURRENT BETWEEN 5 & 6 IN MA.

? -3.6
CØRRECT
1.4: SUPPLY 7 VØLTS BETWEEN 7 AND 8
     ENTER R C/R WHEN READY
? R
SUPPLIED VØLTAGE GREATER THAN 10 VØLTS.
INCØRRECT
TRY AGAIN
? R
7.12 VØLTS IS SUPPLIED
CØRRECT
1.5:WHAT IS R1 IN 2 PØRT NETWØRK AT 9,10,11&12?

? .5
CØRRECT
```

Figure A.6: Printout Of A Student Taking The Tests

```
1.6:WHAT IS R2 IN SAME NETWØRK IN K ØHMS?

? 2.6
CØRRECT
1.7:WHAT IS R3 IN THE SAME NETWØRK IN K ØHMS?

? .5
CØRRECT
YØU PASS TEST 1. DØ YØU WANT TØ CØNTINUE TEST 2?
ENTER 'Y' ØR 'N'
? Y
TEST 2: AC TESTING
2.1:APPLY 1 K HZ SINE WAVE AT 13 AND 14 AND
    MEASURE GAIN AT ØUTPUT AT 15, 16 IN DB.

? 17
CØRRECT
2.2:IN SAME CIRCUIT,APPLY 6KHZ SINE WAVE AT INPUT
    AND MEASURE PHASE DIFFERENCE BETWEEN INPUT
    AND ØUTPUT IN DEGREES

? 80
INCØRRECT
TRY AGAIN
? 90
INCØRRECT
CØRRECT ANSWER CØDE: 1
2.3:IN SAME CIRCUIT,MEASURE CUTØFF FREQUENCY IN
    KHZ.

? 7
INCØRRECT
TRY AGAIN
? 6
INCØRRECT
CØRRECT ANSWER CØDE: 3
2.4 WHAT IS THE RMS VALUE ØF THE WAVEFØRM AT 17
    & 18?
1.61
? 3
INCØRRECT
CØRRECT ANSWER CØDE: 5
5 MISTAKES MADE, FAIL
TIME: 11 HØUR 29 MIN 20 SEC
ASK INSTRUCTØR TØ RE-START SYSTEM
```

Figure A.6(Continued): Printout Of A Student Taking The Tests

APPENDIX B

PROGRAM LISTING

This appendix contains the listing of the assembly language programs written in this thesis to provide the expanded commands to BASIC in this system. The programs consist of a data section residing in RAM and a program section residing in ROM. The data section consists of the global variables, four scratch pads called SCRAT1, SCRAT2, SCRAT3 and SCRAT4 and an area used by the clock interrupt servicing UPDATE program only. Global variables are locations used by more than one program module. A program module is any of the main programs or subroutines in the listing.

The program section consists of the global parameters, the main programs, level 1 subroutines, level 2 subroutines and level 3 subroutines. Global parameters are tables in ROM used by more than one program module. The main programs consist of the initialization program, the UPDATE program, the command interpreter, the error printing program and the various command modules. Each of the main programs except the UPDATE program uses the same scratch pad SCRAT1. The UPDATE program uses a separate scratch pad of its own. The common subroutines of the programs used by more than one program module have been grouped into level 1, level 2 and level 3 subroutines. A subroutine of a particular level is called by at least two program modules of a higher level(a main program or a subroutine with a smaller level number). Level 1, level 2 and level 3 subroutines use the scratch pads SCRAT1, SCRAT2 and SCRAT3 respectively.

B.1 Main Programs

B.2 Level 1 Subroutines

B.3 Level 2 Subroutines

B.4 Level 3 Subroutines

```
                NAM     GLØBAL
                ØPT     REL,PAGE=47
          *GLØBAL PARAMETERS AND VARIABLES
                XDEF    BSCVBL,BSCPTR,CMPØRT,BSCHSA
                XDEF    NUMPIC,PICADT,PWRS10,BUFSIZ
                XDEF    TIME,NPØRTS,PØRTST,LNASGT,RTA
                XDEF    BWAITT,PARBUF,LNVALT,AØLINT,ØDATAT
                XDEF    TENMS,CALHDL,IØTBLN,NUMPIR,BUFAVA
                XDEF    STATUS,IØWTMS,IØTBLS,BFPTRS,BUFFER
                XDEF    BEGLIN,CRRECE
0103   A BCMMØD EQU     $ABAB     BASIC CØMMAND ENTRY
002A   A BSCVBL EQU     $2A       BASIC VARIABLES PØINTER
005B   A BSCPTR EQU     $5B       BASIC STATEMENT PØINTER
0096   A CMPØRT EQU     $96       BASIC CØMMAND PØRT NØ.
0106   A BSCHSA EQU     $A106     BASIC I/Ø HANDLERS ADR.
          *GLØBAL PARAMETERS
                PSCT
04     A NUMPIC FCB     4         NUMBER ØF PIC'S
8400   A PICADT FDB     $8400     PIC ADDRESSES TABLE
8402   A        FDB     $8402,$8404,$8406
000A   A PWRS10 FDB     10        HEX EQUIV. ØF PWR'S ØF 10
0064   A        FDB     100,1000,10000
0400   A BUFSIZ FDB     $400      DATA BUFFER SIZE
          *GLØBAL VARIABLES 1. THESE VARIABLES ARE USED
          *TØ STØRE INFØRMATIØN FØR THE CØMMAND PRØGRAMS
                DSCT
0004   A TIME   RMB     4         SYSTEM TIME-ØF  -DAY
0001   A NPØRTS RMB     1         NUMBER ØF PØRTS
00F0   A PØRTST RMB     240       PØRTS TABLE
0004   A LNASGT RMB     4         LINES ASSIGNMENT TABLE
0001   A RTA    RMB     1         REAL TIME ACTIVE FLAG
0004   A BWAITT RMB     4         BASIC PRØGRAM WAIT TIME
001E   A PARBUF RMB     30        CØMMAND PARAMETERS BUFFER
00     A BEGLIN FCB     0         CØMMAND 1ST CHARAC. FLAG
00     A CRRECE FCB     0         CØMMAND LAST CHARAC. FLAG
0004   A LNVALT RMB     4         I/Ø HANDLER TABLES LNVALT,
0004   A AØLINT RMB     4         AØLINT AND ØDATAT
0004   A ØDATAT RMB     4
          *PERIØDIC I/Ø VARIABLES
0001   A TENMS  RMB     1         TEN MS FLAG
0001   A CALHDL RMB     1         CALL I/Ø HANDLER FLAG
0001   A IØTBLN RMB     1         PERIØDIC I/Ø TABLE NØ.
0001   A NUMPIR RMB     1         NØ. ØF INPUT REQUESTS
0002   A BUFAVA RMB     2         AVAILABLE BUFFER SPACE
0005   A STATUS RMB     5         STATUS ØF PØRTS
0018   A IØWTMS RMB     24        I/Ø WAIT TIMES
004E   A IØTBLS RMB     78        PERIØDIC I/Ø TABLES
```

```
000C  A BFPTRS RMB    12        BUFFER PØINTERS
0400  A BUFFER RMB    $400
        *GLØBAL VARIABLES 2. THESE VARIABLES ARE USED
        *TØ TRANSFER DATA TØAND FRØM SUBRØUTINES WHEN
        *THEY ARE CALLED.
               XDEF   SIGN,BCDNUM,HEXNUM,ØDATA,INDATA
               XDEF   ØPØRTL,ØDATAL,IPØRTL,INDATL(IBSCQL
               XDEF   BSCQL,SØURCE,ADDEND,MINUEN
               XDEF   HBYTES,LBYTES,NLINES
0001  A SIGN   RMB    1         SIGN ØF BCD NØ.
0005  A BCDNUM RMB    5         BCD NUMBER, 5 DIGITS
0002  A HEXNUM RMB    2         16 BIT HEX NØ.
0003  A ØDATA  RMB    3         DATA TØ BE ØUTPUTED
0003  A INDATA RMB    3         PØRT'S INPUTED DATA
0002  A ØPØRTL RMB    2         ØUTPUT PØRT LØCATIØN
0002  A ØDATAL RMB    2         ØUTPUT DATA LØCATIØN
0002  A IPØRTL RMB    2         INPUT PØRT LØCATIØN
0002  A INDATL RMB    2         INPUT STØRING PØINTER
0002  A IBSCQL RMB    2         INPUT VARIABLE LØCATIØN
0002  A BSCQL  RMB    2         BASIC VARIABLE LØCATIØN
0002  A SØURCE RMB    2         LØCATIØN TØ BE CØPIED FRØM
0002  A ADDEND RMB    2         16 BIT NØ. TØ BE ADDED
0002  A MINUEN RMB    2         16 BIT NØ TØ BE SUBT.
0002  A HBYTES RMB    2         LØCATIØN ØF BIGGER NØ.
0002  A LBYTES RMB    2         LØCATIØN ØF LESSER NØ.
0001  A NLINES RMB    1         NØ. ØF DATA LINES
               END
```

INITIA

```
                        NAM     INITIA
                        ØPT     REL,PAGE=49
                *INITIALIZATIØN PRØGRAM FØR EXPANDED CØMMANDS
                        XREF    BCMMØD,NUMPIC,PICADT,NPØRTS,LNASGT
                        XREF    RTA,TIME,IØTBLN,BEGLIN,CRRECE
                        XREF    UPDATE,RPCPIC,RPGPIC
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRATI  CØMM    DSCT
   0002   A     PICATL  RMB     2          PICADT TABLE PØINTER
                        PSCT
0F              INITIA  SEI                *SET INTERRUPT MASK
4F                      CLRA               *PRØGRAM CLØCK PIC
BD 0000   A             JSR     RPCPIC     **AS INPUTS
F6 0000   A             LDAB    NU
5A                      DECB
27 14 001F              BEQ     CLRNPT     *SYSTEM HAS 1 PIC
CE 0002   A             LDX     #PICADT+2 *PRØGRAM ØTHER PIC'S
FF 0000   N     NEXPIC  STX     PICATL     **AS INPUTS
4F                      CLRA
BD 0000   A             JSR     RPGPIC
5A                      DECB
27 07 001F              BEQ     CLRNPT
FE 0000   N             LDX     PICATL
08                      INX
08                     INX
20 EF 000E              BRA     NEXPIC
7F 0000   A     CLRNPT  CLR     NPØRTS     *CLEAR NØ. ØF PØRTS
B6 0000   A             LDAA    NUMPIC     *CLEAR LNASGT TABLE
CE 0000   A             LDX     #LNASGT
6F 00     A     CLRLAT  CLR     X
4A                      DECA
27 03 0030              BEQ     CLRRTA
08                      INX
20 F8 0028              BRA     CLRLAT
7F 0000   A     CLRRTA  CLR     RTA        *CLEAR RTA
7F 0000   A             CLR     IØTBLN     *CLEAR IØTBLN
7F 0000   A             CLR     BEGLIN     *CØMMAND 1ST CHARAC. FLAG
7F 0000   A             CLR     CRRECE     *CØMMAND LAST CHARAC. FLAG
7F 0000   A             CLR     TIME       *CLEAR CLØCK
7F 0001   A             CLR     TIME+1
7F 0002   A             CLR     TIME+2
7F 0003   A             CLR     TIME+3
CE 0000   A             LDX     #UPDATE    *INITIALIZE INTERRUPT
FF FFF8   A             STX     $FFF8      **REQUEST VECTØR
0E                      CLI                *ENABLE INTERRUPT
7E 0000   A             JMP     BCMMØD     *BASIC CØMMAND MØDE
                        END
```

UPDATE

```
                        NAM     UPDATE
                        ØPT     REL,PAGE=49
                *CLØCK INTERRUPT REQUEST SERVICE PRØGRAM
                        XREF    PICADT,TIME,RTA,TENMS,CALHDL
                        XREF    IØTBLN,STATUS,IØWTMS,BWAITT
                        XDEF    UPDATE
                *SCRATCH FØR UPDATE
                        DSCT
   0002   A IØWTTL RMB     2          IØWTMS TABLE PØINTER
   0002   A STATTL RMB     2          STATUS TABLE PØINTER
   0001   A IØTCTR RMB     1          PERIØDIC I/Ø CØUNTER
                        PSCT
                *CLØCK INTERRUPT DRIVEN TIME UPDATE PRØGRAM
FE 0000   A UPDATE LDX     PICADT     *READ CLØCK PIC TØ
A6 00     A        LDAA    X          **CLEAR INTERRUPT
7C 0003   A        INC     TIME+3     *INCREMENT TENS ØF MS
86 64     A        LDAA    #100       *100 TENS ØF MS?
B1 0003   A        CMPA    TIME+3
26 2A 0039         BNE     CHKRTA     *NØ
7F 0003   A        CLR     TIME+3     *CLEAR TENS ØF MS
7C 0002   A        INC     TIME+2     *INCREMENT SEC
86 3C     A        LDAA    #60        *60 SEC?
B1 0002   A        CMPA    TIME+2
26 1D 0039         BNE     CHKRTA     *NØ
7C 0001   A        INC     TIME+1     *YES, INCREMENT MIN.
7F 0002   A        CLR     TIME+2     *CLEAR SEC
B1 0001   A        CMPA    TIME+1     *60 MIN.?
26 12 0039         BNE     CHKRTA     *NØ
7F 0001   A        CLR     TIME+1     *YES, CLEAR MIN
86 17     A        LDAA    #23        *23 HØURS?
B1 0000   A        CMPA    TIME
26 05 0036         BNE     INHØUR     *NØ
7F 0000   A        CLR     TIME       *CLEAR HØURS
20 03 0039         BRA     CHKRTA
7C 0000   A INHØUR INC     TIME       *INCREMENT HØUR
B6 0000   A CHKRTA LDAA    RTA        *PERIØDIC I/Ø ACTIVE?
81 01     A        CMPA    #1
27 05 0045         BEQ     UPRTIØ     *YES
81 02     A        CMPA    #2         *WAIT MØDULE ACTIVE?
27 48 008C         BEQ     UPWAIT     *YES
3B                 RTI                *NØ, RETURN FRØM INT.
                *UPDATE REAL TIME I/Ø TASK
CE 0000   A UPRTIØ LDX     #IØWTMS    *UPDATE PERIØDIC I/Ø
08                 INX                **VARIABLES
08                 INX
08                 INX
FF 0000   D        STX     IØWTTL
CE 0000   A        LDX     #STATUS
FF 0002   D        STX     STATTL
```

UPDATE

```
F6 0000  A         LDAB   IØTBLN
F7 0004  D         STAB   IØTCTR
A6 00    A NXSTAT  LDAA   X         *GET STATUS BYTE
27 10 006E         BEQ    DECNWT    *PØRT I/Ø IS FINISHED
81 02    A         CMPA   #2        *PØRT SHØULD BE ACTIVE?
26 07 0069         BNE    DECMS     *NØ
86 01    A         LDAA   #1        *YES
A7 00    A         STAA   X         *ACTIVATE JØB
B7 0000  A         STAA   CALHDL    *TELL I/Ø TASK CALL HDL.
FE 0000  D DECMS   LDX    IØWTTL    *DECREASE I/Ø WAIT
6A 00    A         DEC    X         **TIME BY 10 MS
7A 0004  D DECNWT  DEC    IØTCTR    *ALL PØRTS UPDATED?
27 13 0086         BEQ    SETTMS    *YES
FE 0000  D         LDX    IØWTTL    *NØ, CØNTINUE WITH NEXT
08                 INX              **NEXT I/Ø WAIT TIME
08                 INX
08                 INX
08                 INX
FF 0000  D         STX    IØWTTL
FE 0002  D         LDX    STATTL    *NEXT STATUS BYTE
08                 INX
FF 0002  D         STX    STATTL
20 D4 005A         BRA    NXSTAT
86 01    A SETTMS  LDAA   #1        *TELL I/Ø TASK TEN MS
B7 0000  A         STAA   TENMS     **HAS PASSED
3B                 RTI              *RETUPN FRØM INTERRUPT
            *UPDATE WAI(WAIT) CØMMAND MØDULE
7A 0003  A UPWAIT  DEC    BWAITT+3  *DECREASE WAIT TIME
3B                 RTI              **BY 10 MS
                   END
```

```
                NAM     CMDINT
                ØPT     REL,PAGE=47
        *CØMMAND INTERPRETER FØR EXPANDED BASIC
        *CØMMANDS
                XREF    NUMPIC,PARBUF,CRRECE,BEGLIN
                XREF    SØURCE,HEXNUM,HBYTES,LBYTES
                XREF    SIGN,BCDNUM
                XREF    CØNFIG,DLPØRT,ØUTPUT,INPUT,RTIØRQ
                XREF    DELREQ,RTIØTK,TIMSET,TMREAD,WAIT
                XREF    IØER29,IØER30,BCDHEX,TSFBYT,CPBYTS
                XREF    BLDADR
                XDEF    PØRTS
        *SCRATCH PAD FØR MAIN PRØGRAMS
        SCRAT1  CØMM    DSCT
0001    A CMDCTR RMB    1          CØMAND CØUNTER
0002    A CMDCHL RMB    2          CØMMAND TABLE LØCATIØN
0001    A CMDCHC RMB    1          CØMMAND CHRACTER CØUNTER
0002    A CØMANL RMB    2          CØMMAND LØCATIØN
0002    A BSCMDL RMB    2          BASIC CØMMAND LØCATIØN
0002    A BSCMDP RMB    2          BASIC CØMMAND PØINTER
0001    A DELIM  RMB    1          DELIMETER
0001    A NNAMEC RMB    1          NUMBER ØF NAME CHARACTERS
0001    A LSTLNN RMB    1          LAST LINE NUMBER
0002    A XR     RMB    2          X REGISTER SAVED
0001    A NUMDIG RMB    1          NUMBER ØF DIGITS
0001    A DIGCTR RMB    1          CØUNTER FØR NØ. ØF DIGITS
0002    A LGTMTL RMB    2          TIME LIMIT TABLE LØC.
0002    A TIMEL  RMB    2          LØCATIØN IN TIME
0004    A TIME   RMB    4          STØRES CØNVERTED TIME
0001    A TIMIND RMB    1          TIME PARAMETER FLAG
0001    A UNTCTR RMB    1          TIME UNIT CØUNTER
0001    A CMPTDL RMB    1          TIME DELIMITER FLAG
0001    A TIMDLM RMB    1          STØRES TIME DELIMITER
0002    A ASCIIL RMB    2          ASCII NØ. LØCATIØN
0002    A ASCIIP RMB    2          ASCII DIGIT PØINTER
0001    A ASCNDL RMB    1          ASCII NØ. DELIMITER
0001    A CMPDLM RMB    1          CØMPARE NØ. DELIM. FLAG
0002    A BCDDGL RMB    2          BCD DIGIT LØCATIØN
01      A TMSWAI RMB    1        TMS ØR WAI FLAG
0001    A NUCHRC RMB    1         UNUSED CHARACTER CØUNTER
0001    A RB     RMB    1          SAVE B
0048    A BSCCMD RMB    72         EXPANDED BASIC CØMMAND
                PSCT
0B      A NUMCMD FCB    11         NUMBER ØF CØMMANDS
        *TABLE ØF CØMMAND MNEMØNICS
43      A CMDTBL FCC    'CØN DPT ØUT INP PØR PIR '
44      A        FCC    'DRQ RTX'
```

CMDINT

```
   00     A         FCB     0
   54     A         FCC     'TMS TMR WAI '
   48     A LGTMUN  FCC     'H'        ALLØWABLE DIGITS AND
   18     A         FCB     24         UNITS FØR TIME PARAMETER
   4D     A         FCC     'M'
   3B     A         FCB     59
   53     A         FCC     'S'
   3B     A         FCB     59
   54     A         FCC     'T'
   63     A         FCB     99
   18 E   A TIME24  FCB     24,0,0,0 TIME LIMIT 24 HOURS
   17     A TIME23  FCB     23,59,59,0 TIME LIMIT 23H59M59S
            *ADDRESSES ØF PARAMETERS FØRMATTING SUBPRØGBAMS
   0120   P PARSPG  FDB     CØNPAR,DELPAR,ØUTPAR,INPPAR
   02B0   P         FDB     RTØPAR,RTIPAR,DRQPAR,RTIØTK,TMSPAR,
   03D6   P         FDB     WAIPAR
            *MØDIFIED BASIC INTERPRETER ØUTPUT HANDLER
            *NØ. 5 FØR EXPANDED BASIC CØMMANDS
F7 0027   N PØRT5   STAB    RB         *SAVE A AND X
FF 000D   N         STX     XR
F6 0000   A         LDAB    BEGLIN     *BEGINNING ØF LINE?
26 15 0073          BNE     CRRECQ     *NØ
CE 0028   N         LDX     #BSCCMD    *YES, CLEAR BSCCMD
FF 0008   N         STX     BSCMDP
C6 48     A         LDAB    #72
6F 00     A CLRBCM  CLR     X
5A                  DECB
27 03 006E          BEQ     SETBGL
08                  INX
20 F8 0066          BRA     CLRBCM
C6 01     A SETBGL  LDAB    #1         *INDICATE NØT BEGINNING
F7 0000   A         STAB    BEGLIN     **ØF LINE
F6 0000   A CRRECQ  LDAB    CRRECE     *C/R RECEIVED?
26 1B 0093          BNE     DECCTB     *YES
81 0D     A         CMPA    #$D        *NØ, IS THE CHR. C/R?
27 0B 0087          BEQ     SETCRR     *YES
FE 0008   N         LDX     BSCMDP     *STØRE CHARACTER ØF
A7 00     A         STAA    X          **EXPANDED CØMMAND
08                  INX
FF 0008   N         STX     BSCMDP
20 1A 00A1          BRA     PRTRTN     *RETURN TØ BASIC
C6 01     A SETCRR  LDAB    #1         *INDICATE C/R RECEIVED
F7 0000   A         STAB    CRRECE
C6 06     A         LDAB    #6
F7 0026   N         STAB    NUCHRC
20 0E 00A1          BRA     PRTRTN
7A 0026   N DECCTR  DEC     NUCHRC     *IGNØRE CHR. AFTER C/R
```

CMDINT

```
26 09 00A1          BNE    PRTRTN
7F 0000  A          CLR    BEGLIN    *INDICATE NEW LINE
7F 0000  A          CLR    CRRECE    *CLEAR C/R RECEIVED FLAG
7E 00A8  P          JMP    CMDINT    *GØ TØ CØMMAND PRØGRAMS
FE 000D  N PRTRTN   LDX    XR        *RESTØRE B AND X
F6 0027  N          LDAB   RB
39                  RTS              *RETURN TØ BASIC
           *CØMMAND INTERPRETER
B6 0000  P CMDINT   LDAA   NUMCMD    *CØMPARE WITH TABLE
B7 0000  N          STAA   CMDCTR    **ØF CØMMAND MNEMØNICS
CE 0001  P          LDX    #CMDTBL
FF 0001  N          STX    CMDCHL
FF 0004  N          STX    CØMANL
CE 0028  N          LDX    #BSCCMD
FF 0008  N          STX    BSCMDP
86 04    A CMPCMD   LDAA   #4        *CØMPARE WITH A CØMMAND
B7 0003  N          STAA   CMDCHC    **MNEMØNIC
A6 00    A CPCMDC   LDAA   X
FE  001  N          LDX    CMDCHL
A1 00    A          CMPA   X
26 12 00DD          BNE    NXCMDQ
7A 0003  N          DEC    CMDCHC
27 2B 00FB          BEQ    CMDMAT
08                  INX
FF 0001  N          STX    CMDCHL
FE 0008  N          LDX    BSCMDP
08                  INX
FF 0008  N          STX    BSCMDP
20 E5 00C2          BRA    CPCMDC
7A 0000  N NXCMDQ   DEC    CMDCTR    *LAST CØMMAND MNEMØNIC?
26 03 00E5          BNE    NEXCMD    *NØ, CØMPARE WITH NEXT ØNE
7E 0000  A          JMP    IØER29    *CØMMAND NØT MATCH
C6 04    A NEXCMD   LDAB   #4        *CØMPARE NEXT MNEMØNIC
FE 0004  N          LDX    CØMANL
BD 0000  A          JSR    BLDADR
FF 0004  N          STX    CØMANL
FF 0001  N          STX    CMDCHL
CE 0028  N          LDX    #BSCCMD
FF 0008  N          STX    BSCMDP
20 C2 00BD          BRA    CMPCMD
FE 0008  N CMDMAT   LDX    BSCMDP    *CØMMAND MATCH
08                  INX
FF 0006  N          STX    BSCMDL
C6 1E    A          LDAB   #30       *CLEAR PARBUF
CE 0000  A          LDX    #PARBUF
6F 00    A CLRPBF   CLR    X
5A                  DECB
```

```
27 03 010F          BEQ    DETSPG
08                  INX
20 F8 0107          BRA    CLRPBF
F6 0000   P DETSPG  LDAB   NUMCMD    *GØ TØ CØMMAND PARAMETERS
F0 0000   N         SUBB   CMDCTR    **CHECKING SUBPRØGRAM
58                  ASLB
CE 003D   P         LDX   PARSPG
BD 0000   A         JSR    BLDADR
EE 00     A         LDX    X
6E 00     A         JMP    X
            * CØN CØMMAND PARAMETERS CHECK SUBPRØGRAM
FE 0006   N CØNPAR  LDX    BSCMDL    *NAME FØRMAT CØRRECT?
86 2C     A         LDAA   #',
BD 057F   P         JSR    LGNAMQ
26 03 012D          BNE    LGNAM
7E 0000   A         JMP    IØER30    *NAME FØRMAT WRØNG
FE 0006   N LGNAM   LDX    BSCMDL    *PASS NAME TØ PARBUF
FF 0000   A         STX    SØURCE
CE 0000   A         LDX    #PARBUF
BD 0000   A         JSR    TSFBYT
FE 0000   A         LDX    SØURCE
08                  INX
FF 0006   N         STX    BSCMDL
86 49     A         LDAA   #'I       *INPUT PØRT?
A1 00     A         CMPA   X
26 05 014B          BNE    ØPØRTQ    *NØ
7C 0006   A         INC    PARBUF+6  *INDICATE INPUT
20 09 0154          BRA    PTTYPE
86 4F     A ØPØRTQ  LDAA   #'Ø       *ØUTPUT PØRT?
A1 00     A         CMPA   X
27 03 0154          BEQ    PTTYPE
7E 0000   A         JMP    IØER30    *NØT 'I' ØR 'Ø'
A6 01     A PTTYPE  LDAA   1,X       *DATA FØRMAT ØF PØRT
81 42     A         CMPA   #'B       **BINARY?
26 04 015E          BNE    BCDPTQ    *NØ
86 02     A         LDAA   #2        *INDICATE BINARY
20 11 016F          BRA    STPTYP
81 44     A BCDPTQ  CMPA   #'D       *BCD?
26 04 0166          BNE    SBCDPT    *NØ
86 04     A         LDAA   #4        *INDICATE BCD
20 09 016F          BRA    STPTYP
81 53     A SBCDPT  CMPA   #'S       *SIGNED BCD?
27 03 016D          BEQ    SBCDP     *YES
7E 0000   A         JMP    IØER30    *NØT B,D ØR S
86 06     A SBCDP   LDAA   #6        *INDICATE SIGNED BCD
BA 0006   A STPTYP  ØRAA   PARBUF+6  *STØRE DATA FØRMAT
B7 0006   A         STAA   PARBUF+6
```

CMDINT

```
86 2C     A CHKIØD LDAA   #',
A1 02    A        CMPA   2,X
27 03 017E         BEQ    IØDCØR
7E 0000   A        JMP    IØER30
08          IØDCØR INX              *GET FIRST LINE NØ.
08                 INX
08                 INX
FF 0006   N        STX    BSCMDL
86 2C     A        LDAA   #',       *IS FIRST LINE NØ.
BD 04F0   P        JSR    ASCBCD    **FØLLØWED BY ','?
26 0A 0195         BNE    CHKFLN    *MØRE THAN 1 LINE
86 00    A        LDAA   #0        *IS FIBST LINE NØ.
BD 04F0   P        JSR    ASCBCD    **FØLLØWED BY '0'?
26 03 0195         BNE    CHKFLN
7E 0000   A        JMP    IØER30    *WRØNG DELIMITER
FF 0006   N CHKFLN STX    BSCMDL
BD 0000   A        JSR    BCDHEX    *CØNVERT TØ HEX
26 03 01A0         BNE    FLNCRQ
7E 0000   A        JMP    IØER30    *FIRST LINE NØ. TØØ BIG
B6 0000   A FLNCRQ LDAA   HEXNUM
27 03 01A8         BEQ    FLNCQ2
7E 0000   A        JMP    IØER30    *FIRST LINE NØ. TØØ BIG
B6 0000   A FLNCQ2 LDAA   NUMPIC    *FIND TØTAL NØ. LINES
48                 ASLA              **THE SYSTEM HAS
48                 ASLA
48                 ASLA
B7 000C   N        STAA   LSTLNN    *STØRE TØTAL NØ.
B1 0001   A        CMPA   HEXNUM+1  *IS TØTAL NØ. > FIRST
22 05 01BB         BHI    FLNEOQ    **LINE NØ.?
27 03 01BB         BEQ    FLNEOQ
7E 0000   A        JMP    IØER30    *YES
B6 0001   A FLNEOQ LDAA   HEXNUM+1  *FIRST LINE NØ. IS 0?
26 03 01C3         BNE    FLNNEO    *NØ
7E 0000   A        JMP    IØER30    *YES
B7 0007   A FLNNEO STAA   PARBUF+7  *STØRE FIRST LINE NØ.
FE 0006   N        LDX    BSCMDL
09                 DEX
A6 00     A        LDAA   X         *JUST ØNE LINE?
26 05 01D3         BNE    MØR1LN    *MØRE THAN 1 LINE
7C 0008   A        INC    PARBUF+8  INDICATE 1 LINE
20 3A 020D         BRA    JCØNMØ    GØ TØ CØN MØDULE
86 00     A MØR1LN LDAA   #0        MØRE THAN 1 LINE
FE 0006   N        LDX    BSCMDL    *GET LAST LINE NØ.
BD 04F0   P        JSR    ASCBCD
27 33 0210         BEQ    ILGPAR    *LAST LINE NØ. WRØNG
BD 0000   A        JSR    BCDHEX    *CØNVERT TØ HEX
27 2E 0210         BEQ    ILGPAR    *LAST LINE NØ. TØØ BIG
```

CMDINT

```
B6 0000  A          LDAA   HEXNUM
26 29 0210          BNE    ILGPAR    *LAST LN. NØ. TØØ BIG
B6 0001  A          LDAA   HEXNUM+1
B1 000C  N          CMPA   LSTLNN    *CØMPARE WITH TØTAL NØ.
22 21 0210          BHI    ILGPAR    *LAST LINE NØ. >  TØTAL
B1 0007  A          CMPA   PARBUF+7 *CØMPARE WITH F. LN. NØ.
23 1C 0210          BLS    ILGPAR    *LAST NØ. <= FIRST NØ.
B0 0007  A          SUBA   PARBUF+7 *GET NØ. ØF LINES
F6 0006  A          LDAB   PARBUF+6 *GET DATA FØRMAT
54                  LSRB
C1 03    A          CMPB   #3        *SIGNED BCD?
27 06 0205          BEQ    MLNS17    *YES,MAX 17 LINES
81 0F    A          CMPA   #15       *NØ,MAX 16 LINES
2E 0D 0210          BGT    ILGPAR    *NØ. ØF LINES>16
20 04 0209          BRA    INLINS    *INCREASE NØ. ØF LINES 1
81 10    A MLNS17   CMPA   #16
2E 07 0210          BGT    ILGPAR    NØ. ØF LINES>17
4C          INLINS INCA              *INCREASE NØ. ØF LINES BY
B7 0008  A          STAA   PARBUF+8
7E 0000  A JCØNMØ   JMP    CØNFIG    *GØ TØ CØN MØDULE
7E 0000  A ILGPAR   JMP    IØER30    *PARAMETERS ERRØR
           *DELETE PØRT CØMMAND PARAMETERS SUBPRØGRAM
86 00    A DELPAR   LDAA   #0        *CØRRECT NAMEFØRMAT?
CE 002C  N          LDX    #BSCCMD+4
BD 057F  P          JSR    LGNAMQ
26 0A 0227          BNE    TSFDLP    *YES
B6 002C  N          LDAA   BSCCMD+4 *DELETE ALL PØRTS?
81 24    A          CMPA   #'$
26 0F 0233          BNE    ILGDPP    *WRØNG PARAMETERS
'E 0000  A          JMP    DLPØRT    *DELETE ALL PØRTS
FF 0000  A TSFDLP   STX    SØURCE    *TRANSFER NAME TØ
CE 0000  A          LDX    #PARBUF   **PARBUF
BD 0000  A          JSR    TSFBYT
7E 0000  A          JMP    DLPØRT    *GØ TØ DLPØRT MØDULE
7E 0000  A ILGDPP   JMP    IØER30    *WRØNG DLPØRT PARAMETERS
           *ØUTPUT PARAMETERS SUBPRØGRAM
86 2C    A ØUTPAR   LDAA   #',       *ØUTPUT CØNSTANT?
FE 0006  N      LDX     BSCMDL
BD 057F  P          JSR    LGNAMQ
261B 025B       BNE   ØCØNST   *YES
86 00    A          LDAA   #0        *ØUTPET VARIABLE QO?
BD 057F  P          JSR    LGNAMQ *NAME CØRRECT FØRMAT?
26 03 024A          BNE    LGØPN     *YES
7E 0000  A          JMP    IØER30    *NØ, ERRØR
FE 0006  N LGØPN    LDX    BSCMDL    *TRANSFER NAME ØF PØRT
FF 0000  A          STX    SØURCE
CE 0000  A        LDX    #PARBUF
```

CMDINT

```
BD 0000  A          JSR    TSFBYT
7F 0006  A          CLR    PARBUF+6
20 2F 028A          BRA    JØUTMØ    *GØ TØ ØUTPUT MØDULE
FE 0006  N ØCØNST LDX    BSCMDL    *ØUTPUT CØNSTANT, TRANSFER
FF 0000  A          STX    SØURCE    **NAME TØ PARBUF
CE 0000  A          LDX    #PARBUF
BD 0000  A          JSR    TSFBYT
7C 0006  A          INC    PARBUF+6 *INDICATE ØUTPUT CØNSTANT
FE 0000  A          LDX    SØERCE
08                  INX
FF 0006  N          STX    BSCMDH
A6 00    A          LDAA   X
81 2D    A          CMPA   #'-       *NEGATIVE CØNSTANT?
26 09 0280          BNE    ASCCPQ    *NØ
86 01    A          LDAA   #1        *INDICATE NEGATIVE
B7 0000  A          STAA   SIGN      **CØNSTANT
08                  INX
FF 0006  N          STX    BSCMDL
86 00    A ASCCPQ LDAA   #0        *CØNVERT ASCII TØ BCD
BD 04F0  P          JSR    ASCBCD
26 03 028A          BNE    JØUTMØ    *CØNVERSIØN CØRRECT
7E 0000  A          JMP    IØER30    *CØNVERSIØN INCØRRECT
7E 0000  A JØUTMØ JMP    ØUTPT    *GØ TØ ØUTPUT MØDULE
           *INPUTCØMMAND PARAMETERS SUBPRØGRAM
86 00    A INPPAR LDAA   #0        *NAME FØRMAT CØRRECT?
FE 0006  N          LDX    BSCMDL
BD 057F  P          JSR    LGNAMQ
26 03 029A          BNE    LGIPNA    *YES
7E 0000  A          JMP    IØER30    *NØ, ERRØR
FE 0006  N LGIPNA LDX    BSCMDL    *TRANSFER NAME TØ
FF 0000  A          STX    SØURCE    **PARBUF
CE 0000  A          LDX    #PARBUF
BD 0000  A          JSR    TSFBYT
7E 0000  A          JMP    INPUT     *GØ TØ INPUT MØDULE
           *PERIØDIC ØUTPUT AND INPUT REQUEST CØMMANDS
           *PARAMETERS SUBPRØGRAM
86 01    A RTIPAR LDAA   #1        *INDICATE INPUT  REQUEST
B7 0000  A          STAA   PARBUF
20 00 02B0          BRA    RTØPAR
FE 0006  N RTØPAR LDX    BSCMDL    *NAME FØRMAT CØRRECT?
86 2C    A          LDAA   #',
D 057F  P         JSR   LGNAMQ
26 03 02BD          BNE    LGIØPN    *YES
7E 0000  A          JMP    IØER30    *NØ, ERRØR
FF 0000  A LGIØPN STX    SØURCE    *TRANSFER NAME TØ
CE 0001  A          LDX    #PARBUF+1 **PARBUF
BD 0000  A          JSR    TSFBYT
```

CMDINT

```
FE 0000  A         LDX    SØURCE    *START TIME SPECIFIED?
08                 INX
FF 0006  N         STX    BSCMDL
86 2C    A         LDAA   #',
BD 042E  P         JSR    CØNTMD
27 25 02F9         BEQ    NUMWDQ    *NØ
FF 0006  N STØSTM  STX    BSCMDL    *START TIME LESS
CE 0015  N         LDX    #TIME     **THAN ØR EQUAL TØ
FF 0000  A         STX    LBYTES    **24 HØURS?
CE 0035  P         LDX    #TIME24
FF 0000  A         STX    HBYTES
BD 0000  A         JSR    CPBYTS
26 03 02EB         BNE    LGSTTM    *YES
7E 0000  A         JMP    IØER30    *NØ, ERRØR
CE 0015  N LGSTTM  LDX    #TIME     *STØRE START TIME
FF 0000  A         STX    SØURCE    **INTØ PARBUF
C6 04    A         LDAB   #4
CE 0007  A         LDX    #PARBUF+7
BD 0000  A         JSR    TSFBYT
FE 0006  N NUMWDQ  LDX    BSCMDL    *CØNVERT # ØF WØRDS
86 2C    A         LDAA   #',       **FRØM ASCII TØ BCD
BD 04F0  P         JSR    ASCBCD
26 03 0306         BNE    LGNUMW    *CØNVERSIØN CØRRECT
7E 0000  A         JMP    IØER30    *CØNVERSIØN INCØRRECT
FF 0006  N LGNUMW  STX    BSCMDL
BD 0000  A         JSR    BCDHEX    *CØNVERT BCD TØ HEX
26 03 0311         BNE    LGNMWD    *CØNVERSIØN CØRRECT
7E 0000  A         JMP    IØER30    *CØNVERSIØN INCØRRECT
B6 0000  A LGNMWD  LDAA   HEXNUM
27 03 0319         BEQ    LGNWØR
7E 0000  A         JMP    IØER30    *# ØF WØRD>255
B6 0001  A LGNWØR  LDAA   HEXNUM+1  *STØRE # ØF WØRDS
B7 000B  A         STAA   PARBUF+11 **INTØ PARBUF
BD 0438  P         JSR    CØNTM     *GET PERIØD ØF I/Ø
C1 02    A         CMPB   #2        *NØN-ZERØ PERIØD?
27 03 0329         BEQ    TMBWNO    *YES
7E 0000  A         JMP    IØER30    *NØ, ERRØR
FF 0006  N TMBWNO  STX    BSCMDL
CE 0015  N         LDX    #TIME     *PERIØD<ØR=24 HØURS?
FF 0000  A         STX    LBYTES
CE 0035  P         LDX    #TIME24
FF 0000  A         STX    HBYTES
BD 0000  A         JSR    CPBYTS
26 03 0340         BNE    LGTMBW    *YES
7E 0000  A         JMP    IØER30    *NØ, ERRØR
CE 0015  N LGTMBW  HDX    #TIME     *TRANSFER PERIØD
FF 0000  A         STX    SØURCE    **TØ PARBUF
```

```
CE 000C  A          LDX    #PARBUF+12
C6 04    A          LDAB   #4
BD 0000  A          JSR    TSFBYT
FE 0006  N          LDX    BSCMDL
B6 0000  A          LDAA   PARBUF
27 07 035D         BEQ    CØMMAO    *ØUTPUT REQUEST
A6 00    A          LDAA   X         *INPUT REQUEST. CØMMAND
27 50 03AA          BEQ    IØRMØD    **TERMINATE WITH 0?
7E 0000  A          JMP    IØER30    *NØ, ERRØR
A6 00    A CØMMAO   LDAA   X         *ØUTPUT REQUEST, CØMMAND
27 49 03AA          BEQ    IØRMØD    **TERMINATE WITH 0?
81 2C    A          CMPA   #',       * NØ. CIRCULAR ØUTPUT?
27 03 0368          BEQ    CIRDLM    *YES
7E 0000  A          JMP    IØER30    *NØ, ERRØR
A6 01    A CIRDLM   LDAA   1,X       *'C' DENØTING CIRCULAR
81 43    A          CMPA   #'C       **ØUTPUT?
27 03 0371          BEQ    CIRREQ    *YES
7E 0000  A          JMP    IØER30    *NØ, ERRØR
08          CIRREQ  INX              *GET TØTAL TIME
08                 INX
08                  INX
FF 0006  N          STX    BSCMDL
86 00    A          LDAA   #0
BD 042E  P          JSR    CØNTMD
C1 02    A          CMPB   #2        *NØN-ZERØ TØTAL TIME?
27 03 0383          BEQ    TØTMNO    *YES
7E 0000  A          JMP    IØER30    *NØ,ERRØR
CE 0015  N TØTMNO   LDX    #TIME     *CØMPARE TØTAL TIME
FF 0000  A          STX    HBYTES    **WIDH PERIØD
CE 000C  A          LDX    #PARBUF+12
FF 0000  A          STX    LBYTES
BD 0000  A          JSR    CPBYTS
26 03 0397          BNE    TTMGTB    *TØTAL>ØR=PERIØD
7E 0000  A          JMP    IØER30    *TØTAL<PERIØD, ERRØR
CE 0015  N TTMGTB   LDX    #TIME     *TRANSFER TØTAL TIME
FF 0000  A          STX    SØURCE    **PARBUF
86 01    A          LDAA   #1
B7 0010  A          STAA   PARBUF+16
CE 0011  A          LDX    #PARBUF+17
C6 04    A          LDAB   #4
BD 0000  A          JSR    TSFBYT
7E 0000  A IØRMØD   JMP    RTIØRQ    *GØ TØ REQUEST MØDULE
           *DELETE REQUEST CØMMAND PARAMETERS SUBPRØGRAM
86 00    A DRQPAR   LDAA   #0        *CØNVERT # ØF REQUESTS
CE 002C  N          LDX    #BSCCMD+4 **TØ BE DELETED FRØM
BD 04F0  P          JSR    ASCBCD    **ASCII TØ BCD
27 17 03CE          BEQ    ILGDRP    *CØNVERSIØN INCØRRECT
```

```
BD 0000  A          JSR    BCDHEX    *CØNVERT TØ HEX
27 12 03CE          BEQ    ILGDRP    *CØNVERSIØN INCØRRECT
B6 0000  A          LDAA   HEXNUM
26 0D 03CE          BNE    ILGDRP    *NØ.>255, INCØRRECT
B6 0001  A          LDAA   HEXNUM+1
81 06    A          CMPA   #6
22 06 03CE          BHI    ILGDRP    *NØ.>6, INCØRRECT
B7 0000  A          STAA   PARBUF    *STØRE # INTØ PARBUF
7E 0000  A          JMP    DELREQ    *GØ TØ DEL. REQ. MØDULE
7E 0000  A ILGDRP JMP    IØER30
           *TMS AND WAI CØMMANDS PARAMETERS SUBPRØGRAM
7F 0025  N TMSPAR CLR    TMSWAI    *INDICATE TIME SET
20 05 03DB          BRA    CNTSTW
86 01    A WAIPAR LDAA   #1
B7 0025  N          STAA   TMSWAI
CE 002C  N CNTSTW LDX    #BSCCMD+4 *CØNVERT TIME FRØM
86 00    A          LDAA   #0        **ASCII TØ HEX
BD 042E  P          JSR    CØNTMD
26 03 03E8          BNE    CPTSWQ
7E 0000  A          JMP    IØER30
B6 0025  N CPTSQ LDAA   TMSWAI    *TMS ØR WAI CØMMAND?
27 05 03F2          BEQ    LDTM23    *TMS CØMMAND
CE 0035  P          LDX    #TIME24   *CØMPARE WITH 24 HØURS
20 03 03F5          BRA    CPTSTW
CE 0039  P LDTM23 LDX    #TIME23   *CØMPARE WITH 23H59M59S
FF 0000  A CPTSTW STX    HBYTES    *CØMPARE TIMES
CE 0015  N          LDX    #TIME
FF 0000  A          STX    LBYTES
BD 0000  A          JSR    CPBYTS
26 03 0406          BNE    TSFSWT
7E 0000  A          JMP    IØER30    *TIME ØUT ØF RANGE
CE 0015N TSFSWT LDX    #TIME     *TRANSFER TIME TØ
FF 0000  A          STX    SØURCE    **PARBUF
CE 0000  A          LDX    #PARBUF
C6 04    A          LDAB   #4
BD 0000  A          JSR    TSFBYT
B6 0025  N          LDAA   TMSWAI
27 03 041C          BEQ    JMPTMS
7E 0000  A          JMP    WAIT      *GØ TØ WAIT MØDULE
7E 0000  A JMPTMS JMP    TIMSET    *GØ TØ TMS MØDULE
           *TMR CØMMAND PARAMETERS SUBPRØGRAM
B6 002C  N TMRPAR LDAA   BSCCMD+4
81 44    A          CMPA   #'D       *DISPLAY TIME TØØ?
26 05 042B          BNE    JMPTMR
86 01    A          LDAA   #1        *INDICATE DISPLAY
B7 0000  A          STAA   PARBUF
7E 0000  A JMPTMR JMP    TMREAD
```

```
                *SUBRØUTINE TØ CØNVERT TIME.
                *TIME SPECIFIED IN THE FØRMAT WHXMYSZT WHERE
                *H,M,S AND T DENØTD HØURS, MIN., SEC AND 10 MS.
                *LØCATIØN ØF THE PARAMETERS IS SPECIFIED IN
                *BSCMDL. THE ENTRY PØINT CØNTMD CØNVERTS TIME
                *AND ALSØ CHECKS IF IT IS FØLLØWED BY THE
                *DELIMITER STØRED IN ACC. A. THE ENTRY PØINT
                *CØNTM JUST CØNVERTS THE PARAMETER. IF THERE
                *IS AN ERRØR, ACC. B IS CLEARED. IF THE PARAMETER
                *IS 0, ACC. B IS SET TØ 1. IF THE PARAMETER IS
                *NØN-ZERØ, ACC. B IS SETTED TØ 2. X PØINTS TØ
                *THE LØCATIØN AFTER THE PARAMETER ØR DELIMITER.
                *IF CØNVERSIØN IS CØRRECT. THE CØNVERTED TIME
                *IS STØRED IN TIME.
B7 001C  N CØNTMD STAA   TIMDLM   *STØRE DELIMITER
86 01    A        LDAA   #1       *SET CØMPARE DELIM.
B7 001B  N        STAA   CMPTDL   **INDICATØR
20 03 043B        BRA    INITIM
7F 001B  N CØNTM  CLR    CMPTDL
7F 0015  N INITIM CLR    TIME     *CLEAR TIME
7F 0016  N        CLR    TIME+1
7F 0017  N        CLR    TIME+2
7F 0018  N        CLR    TIME+3
FE 0006  N        LDX    BSCMDL   *INITIALIZE BSCMDP
FF 0008  N        STX    BSCMDP
7F 0019  N        CLR    TIMIND   *CLEAR TIME INDICATIØR
86 04    A        LDAA   #4       *4 TIME UNITS
B7 001A  N        STAA   UNTCTR   **TØ BE CØNVERTED
CE 0015  N        LDX    #TIME
FF 0013  N        STX    TIMEL
CE 002D  P        LDX    #LGTMUN  *LGTMUN CØNTAINS RANGE
FF 0011  N        STX    LGTMTL   **AND MNEMØNIC ØF UNITS
A6 00    A TIMUNT LDAA   X        *GET UNIT MNEMØNIC
FE 0008  N        LDX    BSCMDP
BD 04F0  P        JSR    ASCBCD   *CØNVERT  A UNIT
26 02 046D        BNE    STIMI1   *UNIT EXISTS
20 23 049         BRA    NXTUNQ   *NEXT UNIT?
86 01    A STIMI1 LDAA   #1       *INDICATE UNIT EXISTS
FF 0008  N        STX    BSCMDP   *ADVANCE BSCMDP
B7 0019  N        STAA   TIMIND
BD 0000  A        JSR    BCDHEX   *CØNVERT TØ HEX
26 02 047C        BNE    LGTMN1   *CØNVERSIØN CØRRECT
20 62 04DE        BRA    NTIMPA   *CØNVERSIØN INCØRRECT
B6 0000  A LGTMN1 LDAA   HEXNUM
26 5D 04DE        BNE    NTIMPA   *NØ. >255,ERRØR
B6 0001  A        LDAA   HEXNUM+1
FE 0011  N LGLNQ  LDX    LGTMTL   *GET CØRRECT RANGE
```

```
A1 01     A          CMPA   1,X       *CØMPARE WITH RANGE
2E 53 04DE           BGT    NTIMPA    *NØ. > RANGE
FE 0013   N          LDX    TIMEL     *# WITHIN RANGE
A7 00     A          STAA   X         *STØRE INTØ TIME
7A 001A   N NXTUNQ   DEC    UNTCTR    *CØNVERT NEXT UNIT?
27 11 04A6           BEQ    TIMPAQ    *NØ
FE 0013   N          LDX    TIMEL     *YES
08                   INX
FF 0013   N          STX    TIMEL
FE 0011   N          LDX    LGTMTL
08                   INX
08                   INX
FF 0011   N          STX    LGTMTL
20 BB 0461           BRA    TIMUNT
B6 0019   N TIMPAQ   LDAA   TIMIND    *PARAMETER EXISTS?
27 33 04DE           BEQ    NTIMPA    *NØ
B6 001B   N          LDAA   CMPTDL    *CØMPARE DELIMITER?
27 0E 04BE           BEQ    TIMEOQ    *NØ
FE 0008   N          LDX    BSCMDP    *YES
A6 00     A          LDAA   X         *GET DELIMITER
B1 001C   N          CMPA   TIMDLM
26 24 04DE           BNE    NTIMPA    *DELIM. NØT MATCH
08                   INX
FF 0008   N          STX    BSCMDP
F6 0015   N TIMEOQ   LDAB   TIME      *IS TIME = 0
26 15 04D8           BNE    SETB2
F6 0016   N          LDAB   TIME+1
26 10 04D8           BNE    SETB2
F6 0017   N          LDAB   TIME+2
26 0B 04D8           BNE    SETB2
F6 0018   N          LDAB   TIME+3
26  6 04D8           BNE    SETB2
FE 0008   N          LDX    BSCMDP
C6 01     A          LDAB   #1        *INDICATE 0 TIME
29                   RTS
FE 0008   N SETB2    LDX    BSCMDP    *INDICATE NØN-ZERØ
C6 02     A          LDAB   #2        **TIME
39                RTS
FE 0006   N NTIMPA LDX  BSCMDL    *INDICATE PARAMETER
5F                   CLRB             **DØES NØT EXIST
39                   RTS
            *SUBRØUTINE TØ FIND IF THE CHARACTER IN
            *ACC. A IS AN ASCII LETTER. IF IT IS, ACC.
            *B IS SET TØ 1, ØTHERWISE IT IS CLEARED.
81 41     A LETTEQ CMPA   #'A
2D 07 04EE           BLT    NØTLET
81 5A     A          CMPA   #'Z
```

```
2E 03 04EE          BGT    NØTLET
C6 01     A         LDAB   #1
39                  RTS
5F          NØTLET CLRB
39                 RTS
            *SUBRØUTINE TØ CØNVERT AN ASCII NØ.
            *TØ BCD. REG. X PØINTS TØ ASCII STRING.
            *ASCII NØ. MAXIMUM ØF 5 DIGITS. THE ENTRY
            *PØINT ASCBCD WILL CHECK IF THE NØ.
            *IS FØLLØWED BY THE HEX NØ. IN ACC. A ALSØ.
            *ASCHEX JUST CØNVERTS THE ASCII NØ.
            *CØNVERTED NØ. IS STØRED IN BCDNUM. ACC. B
            *IS SETTED TØ 1 IF ASCII NØ. EXIST, CLEARED
            *ØTHERWISE. REG. X PØINTS TØ CHARACTER
            *FØLLØWING NØ. ØR DELIMITER IF NØ. EXISTS,
            *UNCHANGED ØTHERWISE.
B7 0021   N ASCBCD STAA   ASCNDL    *STØRE DELIMITER
86 01     A         LDAA   #1       *INDICATE CØMPARE
B7 0022   N         STAA   CMPDLM   **DELIMITER
20 03 04FD     BRA    CØUNTD
7F 0022   N ASCHX  CLR    CMPDLM    *NØ DELIMITEB
7F 000F   N CØUNTD CLR    NUMDIG    *CLEAR DIGIT CØUNTER
FF 001D   N         STX    ASCIIL   *SAVE X
A6 00     A CØUNT  LDAA   X         *GET CHARACTER
BD 0572   P         JSR    DIGITQ   *ASCII DIGIT?
27 0D 0517          BEQ    CMPDLQ   *NØ
7C 000F   N         INC    NUMDIG *INCREMENT CØUNTER
F6 000F   N         LDAB   NUMDIG   *NØ. ØF DIGITS>5?
C1 05     A         CMPB   #5
2E 59 056D          BGT    NØASCN   *YES
08                  INX
20 EC 0503          BRA    CØUNT
F6 000F   N CMPDLQ LDAB   NUMDIG    *ANY ASCII DIGITS?
27 51 056D          BEQ    NØASCN   *NØ
7D 0022   N         TST    CMPDLM   *CØMPARE DELIMITER?
26 09 052A          BNE    CMPDL    *YES
FF 001D   N         STX    ASCIIL   *NØ, SAVE X
09                  DEX
FF 001F   N       STX    ASCIIP
20 0E 0538          BRA    ASCBD
B1 0021   N CMPDL  CMPA   ASCNDL    *CØMPARE DELIMITER
26 3E 056D          BNE    NØASCN   *DELIMITER INCØRRECT
08                  INX
FF 001D   N         STX    ASCIIL
09                  DEX
09                  DEX
FF 001F   N         STX    ASCIIP
```

```
CE 0000   A ASCBD  LDX    #BCDNUM  *CLEAR BCDNUM
86 06     A        LDAA   #6
6F 00     A CLRBCD CLR    X
4A                 DECA
27 03 0545         BEQ    ASCBC
08                 INX
20 F8 053D         BRA    CLRBCD
CE 0004   A ASCBC  LDX    #BCDNUM+4
FF 0023   N        STX    BCDDGL
FE 001F   N        LDX    ASCIIP
A6 00     A LASCDG LDAA   X        *LØAD ASCII DIGIT
80 30     A        SUBA   #$30     *CØNVERT TØ BCD
FE 0023   N        LDX    BCDDGL   *STØRE BCD DIGIT
A7 00     A        STAA   X
5A                 DECB             *FINISHED?
27 0D 0567         BEQ    ASCBDF   *YES
09                 DEX              *GØ TØ NEXT DIGIT
FF 0023   N        STX    BCDDGL
FE 001F   N        LDX    ASCIIP
09                 DEX
FF 001F   N        STX    ASCIIP
20 E7 054E         BRA    LASCDG
FE 001D   N ASCBDF LDX    ASCIIL   *MØDIFY X REG.
C6 01     A        LDAB   #1       *INDICATE ASCII #
39                 RTS              **EXISTS
FE 001D   N NØASCN LDX    ASCIIL   *RESTØRE X REG.
5F                 CLRB             *INDICATE ASCII #
39                 RTS              **DØES NØT XIST
            *SUBRØUTINETØ FIND IF CHARACTER IN
            *ACC. A IS AN ASCII DIGIT. ACC. B IS
            *SETTED TØ 1 IF IT IS, CLEARED ØTHERWISE.
81 30     A DIGITQ CMPA   #'0
2D 07 057D         BLT    NØTDIG
81 39     A        CMPA   #'9
2E 03 057D         BGT    NØTDIG
C6 01     A        LDAB   #1
39                 RTS
5F          NØTDIG CLRB
39                 RTS
            *SUBRØUTINE TØ CHECK NAME FØRMAT
B7 000A   N LGNAMQ STAA   DELIM    *STØRE DELIMITER
FF 000D   N        STX    XR
A6 00     A        LDAA   X        *1ST CHR. A LETTER?
BD 04E3   P        JSR    LETTEQ
27 2D 05B9         BEQ    ILGNAM   *NØ
86 01     A        LDAA   #1
B7 000B   N        STAA   NNAMEC
```

```
08                    INX
A6 00     A LETDIG LDAA    X
BD 04E3   P        JSR     LETTEQ    *NEXT CHR. A LETTER?
26 07 05A0         BNE    NXLDQ     *YES
BD 0572   P        JSR     DIGITQ    *NEXT CHR. A DIGIT?
26 02 05A0         BNE     NXLDQ     *YES
20 0D 05AD         BRA     NAMDLQ    *NØT LETTER ØR DIGIT
7C 000B   N NXLDQ  INC     NNAMEC    *INCREASE NØ. ØF CHR.
8606      A        LDAA   #6        *NØ.ØF CHR. > 6?
B1 000B   N        CMPA    NNAMEC
2D 0F 05B9         BLT     ILGNAM    *YES
08                 INX               *NØ
20 E5 0592         BRA     LETDIG
B1 000A   N NAMDLQ CMPA    DELIM     *IS DELIMITER CØRRECT?
26 07 05B9         BNE     ILGNAM    *NØ
FE 000D   N LGNAME LDX     XR        *NAME FØRMAT CØRRECT
F6 000B   N        LDAB    NNAMEC
39                 RTS
FE 000D   N ILGNAM LDX     XR        *NAME FØRMAT WRØNG
5F                 CLRB
39                 RTS
                   END
; 00000
```

ERRMSG

```
                    NAM     ERRMSG
                    ØPT     REL
              *SCRATCH PAD FØR MAIN PRØGRAMS
                    XREF    BSCPTR,BCMMD
                    XREF    ØUTSTR,ØUTBCD,ØUTBDB,ØUTBDC
                    XDEF    IØER29,IØER30,IØER31,IØER32,IØER33
                    XDEF    IØER34,IØER35,IØER36,IØER37,IØER38
                    XDEF    IØER39,IØER40
              *PRØGRAM TØ PRINT ERRØR FØR EXPANDED CØMMANDS
              SCRATI CØMM   DSCT
   0001     A ERRCØD RMB    1          SAVE ERRØR CØDE
                    PSCT
   45       A ERRØR  FCC    'ERRØR ' *ERRØR MESSAGE
   04       A        FCB    4
   20       A LINENØ FCC    ' IN LINE NØ. ' *LINE NØ. MSG.
   04       A        FCB    4
86 29       A IØER29 LDAA   #$29       *ILLEGAL CØMMAND
20 2C 0045           BRA    ERRMSG
86 30       A IØER30 LDAA   #$30       *ILLEGAL PARAMETERS FØR
20 28 0045           BRA    ERRMSG     **CØMMAND
86 31       A IØER31 LDAA   #$31       *DUPLICATE NAME FØR PØRT
20 24 0045           BRA    ERRMSG
86 32       A IØER32 LDAA   #$32       *LINES ALREADY USED BY
20 20 0045           BRA    ERRMSG     **ØTHER PØRTS
86 33       A IØER33 LDAA   #$33       *PØRT DØES NØT EXIST
20 1C 0045           BRA    ERRMSG
86 34       A IØER34 LDAA   #$34       *ØUTPUT DATA ERRØR
20 18 0045           BRA    ERRMSG
86 35       A IØER35 LDAA   #$35       *BASIC VARIABLES ØR
20 14 0045           BRA    ERRMSG     **VECTØRS NØT REFERENCED
86 36       A IØER36 LDAA   #$36       *INTERNAL BUFFER ØVERFLØW
20 10 0045           BRA    ERRMSG
86 37       A IØER37 LDAA   #$37       *PØRTS TABLE FULL
20 0C 0045           BRA    ERMSG
86 38       A IØER38 LDAA   #$38       *PERIØDIC I/Ø TABLES FULL
20 08 0045           BRA    ERRMSG
86 27       A IØER39 LDAA   #39        *NØT USED
20 04 0045           BRA    ERRMSG
86 28       A IØER40 LDAA   #40        *NØT USED
20 00 0045           BRA    ERRMSG
B7 0000     N ERRMSG STAA   ERRCØD     *SAVE ERRØR CØDE
CE 0000     P        LDX    #ERRØR     *PRINT 'ERRØR' ØN CØMMAND
BD 0000     A        JSR    ØUTSTR     **PØRT
B6 0000     N        LDAA   ERRCØD     *GET ERRØR CØDE
BD 0000     A        JSR    ØUTBDB     *ØUTPUT CØDE AND BLANK
CE 0007     P        LDX    #LINENØ    *PRINT 'LINE NØ.'
BD 0000     A        JSR    ØUTSTR
FE 0000     A        LDX    BSCPTR     *GET BASIC STATEMENT
A6 00       A        LDAA   X          **LABEL
```

```
BD 0000  A        JSR    ØUTBCD    *ØUTPUT 1ST 2 DIGITS
FE 0000  A        LDX    BSCPTR    *ØUTPUT LAST 2 DIGITS
A6 01    A        LDAA   1,X       **AND CARRIAGE RETURN
BD 0000  A        JSR    ØUTBDC
31               INS              *ADJUST STACK PØINTER
08                INX
7E 0000  A        JMP    BCMMØD    *GØ TØ BASIC CØMMAND
                  END
```

CØNFIG

```
                    NAM     CØNFIG
                    ØPT     REL,PAGE=49
              *CØN(CØNFIGURE PØRT) CØMMAND MØDULE
                    XREF    PICADT,PARBUF,LNASGT,NPØRTS,PØRTST
                    XREF    SØURCE
                    XREF    IØER31,IØER32,IØER37
                    XREF    SRPØRT,TSFBYT,MULPLY,BLDADR
                    XDEF    CØNFIG
              *SCRATCH PAD FØR MAIN PRØGRAMS
              SCRATI CØMM   DSCT
   0002   A PØRTTL RMB     2          PØRTST TABLE PØINTER
   0002   A PØRTEN RMB     2          NEXT PØRTST ENTRY ADR.
   0002   A LASGTL RMB     2          LNASGT TABLE PØINTER
   0002   A PICATL RMB     2          PICADT TABLE PØINTER
   0002   A PARBFL RMB     2          PARBUF PØINTER
   0001   A NUMLIN RMB     1          NØ. ØF LINES IN PØRT
   0001   A AVALIN RMB     1          AVAILABLE NØ. ØF LINES
   0001   A PPICCN RMB     1          PØRT PIC CØUNTER
   0001   A SHFCNT RMB     1          NØ. ØF SHIFTS CØUNTER
   0001   A TEMPXY RMB     1          TEMPØRARY STØRAGE
                PSCT
B6 0000   A CØNFIG LDAA    NPØRTS     *PØRTST TABLE FULL?
81 14     A        CMPA    #20
26 03 000A         BNE     PRTTNF     *NØ
7E 0000   A        JMP     IØER37     *YES,ERRØR
CE 0000   A PRTTNF LDX     #PARBUF    *PØRT NAME EXISTS?
BD 0000   A        JSR     SRPØRT
27 03 0015         BEQ     PNEXIS     *NØ
7E 0000   A        JMP     IØER31     *YES,ERRØR
B6 0000   A PNEXIS LDAA    NPØRTS     *GET NEXT AVAILABLE
C6 0C     A        LDAB    #12        **ENTRY LØCATIØN
BD 0000   A        JSR     MULPLY
CE 0000   A        LDX     #PØRTST
BD 0000   A        JSR     BLDADR
FF 0002   N        STX     PØRTEN
FF 0000   N        STX     PØRTTL
CE 0000   A        LDX     #PARBUF
FF 0008   N        STX     PARBFL
C6 06     A        LDAB    #6         *TRANSFER NAME TØ
CE 0000   A        LDX     #PARBUF    STØRE NAME **PØRTST
FF 0000   A        STX     SØURCE
FE 0000   N        LDX     PØRTTL
BD 0000   A        JSR     TSFBYT
FF 0000   N        STX     PØRTTL
B6 0006   A        LDAA    PARBUF+6   *INDICATE I ØR Ø
A7 00     A        STAA    X          **IN PØRTST
B6 0007   A PPICLB LDAA    PARBUF+7   *GET FIRST LINE NØ.
4A                 DECA
16                TAB
```

CØNFIG

```
44                  LSRA                *DIVIDE BY 8
44                  LSRA
44                  LSRA
FE 0000   N         LDX     PØRTTL      *STØRE PIC NØ.  ØF
A7 01     A         STAA    1,X         **1ST. LINE ØF PØRT
17                  TBA
84 07     A         ANDA    #7
A7 05     A         STAA    5,X         STØRE CHIFT CØUNT
B7 000D   N         STAA    SHFCNT
B6 0008   A         LDAA    PARBUF+8    *GET # ØF LINES
B7 000A   N         STAA    NUMLIN
86 08     A         LDAA    #8          *FIND # ØF LINES
B0 000D   N         SUBA    SHFCNT      **USED IN 1ST. PIC
B7 000B   N         STAA    AVALIN      **FØR PØRT
4F                  CLRA
0D          FØRMFM  SEC                 *SHIFT AS MANY ØNES
49                  RØLA                **INTØ A AS THE NØ.
7A 000A   N         DEC     NUMLIN      **ØF LINES USED IN THE
27 05 0075          BEQ     SHFFMK      **1ST PIC FØR THE PØRT
7A 000B   N         DEC     AVALIN
26 F4 0069          BNE     FØRMFM
F6 000D   N SHFFMK  LDAB    SHFCNT      *SHIFT 0'S INTO A
27 04 007E          BEQ     STØFMK      **TØ GET BIT MAP ØF
48          SHIFT1  ASLA                **LINES USED IN 1ST
5A                  DECB                **PIC FØR THE PØRT
26 FC 007A          BNE     SHIFT1
FE 0000   N STØFMK  LDX     PØRTTL      *STØRE FIRST BIT MAP
A7 03     A         STAA    3,X
B6 000A   N         LDAA    NUMLIN      *MØRE THAN 1 PIC?
26 08 0090          BNE     TWPICQ      *YES
86 01     A         LDAA    #1          *1 PIC USED
A7 02     A         STAA    2,X
6F 04     A         CLR     4,X         *CLEAR 2ND BIT MAP
20 1D 00AD          BRA     ØVERLQ
81 08     A TWPICQ  CMPA    #8          *2 PIC'S USED?
2E 06 009A          BGT     THRPIC      *NØ
86 02     A         LDAA    #2          *2 PIC'S USED
A7 02     A         STAA    2,X
20 09 00A3          BRA     FØRMLM
C6 03     A THRPIC  LDAB    #3          *3 PIC'S USED
E7 02     A         STAB    2,X
80 08     A         SUBA    #8
B7 000A   N         STAA    NUMLIN
4F          FØRMLM  CLRA                *FØRM 2ND BIT MAP
0D          LASMSK  SEC                 *SHIFT AS MANY 1'S
49                  RØLA                **AS THE NØ. ØF LINES
7A 000A   N         DEC     NUMLIN      **USED IN LAST PIC
26 F9 00A4          BNE     LASMSK      **USED BY PØRT
A7 04     A         STAA    4,X         *STØRE 2ND BIT MAP
```

CØNFIG

```
A6 02     A ØVERLQ LDAA   2,X       *LINES ALREADY USED?
B7 000C   N        STAA   PPICCN
A6 03     A        LDAA   3,X       *GET 1ST BIT MAP
E6 01     A        LDAB   1,X       *GET LØCATIØN IN
CE 0000   A        LDX    #LNASGT   **LINE ASSIGNMENT
BD 0000   A        JSR    BLDADR    **TABLE LNASGT
FF 0004   N        STX    LASGTL
FF 000E   N        STX    TEMPXY
A4 00     A        ANDA   X         *AND BIT MAP & LNASGT
27 03 00C9         BEQ    NØVLP1    *1 PIC CØRRECT
7E 0000   A        JMP    IØER32    *LINES ALREADY USED
7A 000C   N NØVLP1 DEC    PPICCN    *JUST 1 PIC?
27 23 00F1         BEQ    MDLASG    *YES
86 01     A        LDAA   #1        *2 PIC'S?
B1 000C   N        CMPA   PPICCN
27 0D 00E2         BEQ    LPPØVQ    *LAST PØRT PIC ØVERLAP?
86 FF     A        LDAA   #$FF      *LINES IN 2ND PIC
A4 01     A        ANDA   1,X       **USED
27 03 00DE         BEQ    NØVLP2    *NØ
7E 0000   A        JMP    IØER32    *YES,ERRØR
08           NØVLP2 INX
FF 000E   N        STX    TEMPXY
FE 0000   N LPPØVQ LDX    PØRTTL    *LINES IN LAST PIC
A6 04     A        LDAA   4,X       **USED?GET 2ND MAP
FE 000E   N        LDX    TEMPXY
A4 01     A        ANDA   1,X
27 03 00F1         BEQ    MDLASG    *NØT USED
7E 0000   A        JMP    IØER32    *YES,ERRØR
FE 0000   N MDLASG LDX    PØRTTL   *MØDIFY LNASGT TABLE
A6 02     A        LDAA   2,X       **AND PRØGRAM HARDWARE
B7 000C   N        STAA   PPICCN
A6 03     A        LDAA   3,X
FE 0004   N        LDX    LASGTL
16                 TAB
AA 00     A        ØRAA   X
A7 00     A        STAA   X         *MØDIFY 1ST BYTE
FE 0000   N        LDX    PØRTTL
B6 0006   A        LDAA   PARBUF+6 *INPUT PØRT?
85 01     A        BITA   #1
26 22 012F         BNE    MLASG2    *YES
17                 TBA              *ØUTPUT PØRT
E6 01     A        LDAB   1,X       *1ST PIC USED BY PØRT
27 0E 0120         BEQ    PGCPIC    **IS #1? YES
CE 0000   A        LDX    #PICADT   *PIC IS NØT THE ØNE
58                 ASLB             **USED BY THE CLØCK
BD 0000   A        JSR    BLDADR    **INTERRUPT
FF 0006   N        STX    PICATL
8D 53 0171         BSR    PGPIC
20 0F 012F         BRA    MLASG2
```

CØNFIG

```
C6 03     A PGCPIC LDAB   #3         *PIC IS PIC #1 WHICH
FE 0000   A        LDX    PICADT     **IS USED BY CLØCK
E7 01     A        STAB   1,X        **INTERRUPT
AA 00     A        ØRAA   X
A7 00     A        STAA   X
C6 07     A        LDAB   #7
E7 01     A        STAB   1,X
7A 000C   N MLASG2 DEC    PPICCN     *MØDIFY NEXT BYTE
27 4A 017E         BEQ    ICNPRT     **AND PIC? NØ
86 01     A        LDAA   #1         *2 PIC'S USED?
B1 000C   N        CMPA   PPICCN
27 1A 0155         BEQ    MLASG3     *YES
FE 0004   N        LDX    LASGTL     *3 PIC'S USED
08                 INX               *MØDIFY 2ND BYTE
FF 0004   N        STX    LASGTL     **ØF LNASGT
86 FF     A        LDAA   #$FF
A7 00     A        STAA   X
F6 000&   A        LDAB   PARBUF+6 *INPUT PØRT?
26 0A 0155         BNE    MLASG3     *YES
FE 0006   N        LDX    PICATL     *PRØGRAM 2ND PIC
08                 INX
08                 INX
FF 0006   N        STX    PICATL
8D 1C 0171         BSR    PGPIC
FE 0000   N MLASG3 LDX    PØRTTL     *MØDIFY BYTE IN
A6 04     A        LDAA   4,X        **LNASGT FØR LAST
16                 TAB               **PIC USED
FE 0004   N        LDX    LASGTL
AA 01     A        ØRAA   1,X
A7 01     A        STAA   1,X
B6 0006   A        LDAA   PARBUF+6 *INPUT PØRT?
26 17 017E         BNE    ICNPRT     *YES
FE 0006   N        LDX    PICATL     *PRØGRAM LAST PIC
08                 INX               **USED BY THE PØRT
08                 INX
17                 TBA
8D 02 0171         BSB    PGPIC
20 0D 017E         BRA    ICNPRT
            *SUBRØUTINE TØ PRØGRAM LINES ØF PIC
            *(EXCEPT PIC #1) AS ØUTPUTS, ADDRESS
            *ØF PIC STØRED IN X, ØUTPUT LINES DENØTED BY 1'S
            *IN ACC. A. ØRIGINAL ØUTPUT LINES ØF PIC REMAIN
            *AS ØUTPUT LINES.
EE 00     A PGPIC  LDX    X
6F 01     A        CLR    1,X
AA 00     A        ØRAA   X
A7 00     A        STAA   X
86 04     A        LDAA   #4
A7 01     A        STAA   1,X
```

```
CØNFIG

39                  RTS
7C 0000   A ICNPRT  INC     NPØRTS    *INCREASE NPØRTS 1
39                  RTS
                    END
 00000
```

DLPØRT

```
                        NAM     DLPØRT
                        ØPT     REL,PAGE=49
                *DPT(DELETE PØRT) CØMMAND MØDULE
                        XREF    NUMPIC,PICADT,PARBUF,LNASGT,NPØRTS
                        XREF    PØRTST,SØURCE,MINUEN
                        XREF    IØER33,SRPØRT,RPCPIC,RPGPIC,BLDADR
                        XREF    MULPLY,TSFBYT,SUBTRA
                        XDEF    DLPØRT
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRATI  CØMM    DSCT
   0001   A SHFCTR RMB     1          NØ. ØF PØRTS TØ SHIFT
   0002   A PØRTTL RMB     2          PØRTST TABLE PØINTER
   0001   A PØRTCN RMB     1          PØRT ENTRY NØ. IN PØRTST
   0001   A PICCTR RMB     1          PIC CØUNTER
   0002   A LASGTL RMB     2          LNASGT TABLE PØINTER
   0002   A PICATL RMB     2          PICADT TABLE PØINTER
                        PSCT
B6 0000   A DLPØRT LDAA    PARBUF     *DELETE ALL PØRTS?
26 3F 0044         BNE     DL1PØR     *DELETE 1 PØRT
CE 0000   A        LDX     #PICADT    *YES, PRØGRAM PIC
EE 00     A        LDX     X          **#1 AS INPUTS
4F                 CLRA
BD 0000   A        JSR     RPCPIC
CE 0000   A        LDX     #LNASGT    *CLEAR 1ST. BYTE ØF
6F 00     A        CLR     X          **LNASGT
08                 INX
FF 0005   N        STX     LASGTL
CE 0002   A        LDX     #PICADT+2
FF 0007   N        STX     PICATL
B6 0000   A        LDAA    NUMPIC
4A                 DECA
B7 0004   N        STAA    PICCTR
4F          DALLPT CLRA                *PRØGRAM REST ØF PIC'S
BD 0000   A        JSR     RPGPIC     **AS INPUTS AND CLEAR
FE 0005   N        LDX     LASGTL     **REST ØF LNASGT
6F 00     A        CLR     X
7A 0004   N        DEC     PICCTR     *FINISHED?
27 0E 0040         BEQ     CNPØRT     *YES
08                 INX                *NØ
FF 0005   N        STX     LASGTL
FE 0007   N        LDX     PICATL
08                 INX
08                 INX
FF 0007   N        STX     PICATL
20 E4 0024         BRA     DALLPT
7F 0000   A CNPØRT CLR     NPØRTS     *CLEAR NPØRTS
39                 RTS
CE 0000   A DL1PØR LDX     #PARBUF    *DELETE 1 PØRT
BD 0000   A        JSR     SRPØRT     *SEARCH FØR PØRT NAME
```

DLPØRT

```
26 03 004F          BNE    DPØRT     *PØRT NAME EXISTS
7E 0000  A          JMP    IØER33    *NAME DØES NØT EXIST
FF 0001  N DPØRT    STX    PØRTTL    *SAVE LØCATIØN ØF ENTRY
F7 0003  N          STAB   PØRTCN    *SAVE ENTRY NØ.
A6 08    A          LDAA   8,X       *GET # ØF PIC'S
B7 0004  N          STAA   PICCTR
A6 09    A          LDAA   9,X       *GET !ST BIT MAP
43                  CØMA
E6 07    A          LDAB   7,X       *GET 1ST PØRT PIC #
CE 0000  A          LDX    #LNASGT   *GØ TØ CØRRECT LØCATIØN
BD 0000  A          JSR    BLDADR    **IN LNASGT
FF 0005  N          STX    LASGTL
16                  TAB              *SAVE BIT MAP
A4 00    A          ANDA   X
A7 00    A          STAA   X         *MØDIFY LNASGT
FE 0001  N          LDX    PØRTTL
A6 06    A          LDAA   6,X       *INPUT PØRT?
85 01    A          BITA   #1
26 1D 0093          BNE    CLASG2    *INPUT PØRT
17                  TBA              *ØUTPUT PØRT
E6 07    A          LDAB   7,X       *1ST PØRT PIC IS
26 0B 0086          BNE    CPIC1     **#1? NØ
CE 0000  A          LDX    #PICADT   *PRØGRAM LINES USED BY
FF 0007  N          STX    PICATL    **PØRT IN PIC #1 AS
BD 0000  A          JSR    RPCPIC    **INPUTS
20 0D 0093          BRA    CLASG2
CE 0000  A CPIC1    LDX    #PICADT   *PRØGRAM LINES USED BY
58                  ASLB             **PØRT IN PØRT'S 1ST
BD 0000  A          JSR    BLDADR    **PIC(NØT PIC #1) AS
FF 0007  N          STX    PICATL    **INPUTS
BD 0000  A          JSR    RPGPIC
7A 0004  N CLASG2   DEC    PICCTR    *MØRE THAN 1 PIC?
27 45 00DD          BEQ    SHFPTS    *NØ, SHIFT PØRTST
86 01    A          LDAA   #1        *2 PIC'S?
B1 0004  N          CMPA   PICCTR
27 1E 00BD          BEQ    CLASG3    *YES
FE  005  N          LDX    LASGTL    *3 PIC'S USED BY PØRT
08                  INX
FF 0005  N          STX    LASGTL
6F 00    A          CLR    X
FE 0001  N          LDX    PØRTTL    *INPUT PØRT?
E6 06    A          LDAB   6,X
C5 01    A          BITB   #1
26 0C 00BD          BNE    CLASG3    *YES
FE 0007  N          LDX    PICATL    *PRØGRAM PØRT'S 2ND PIC
08                  INX              **AS INPUTS
08                  INX
FF 0007  N          STX    PICATL
4F                  CLRA
```

DLPØRT

```
BD 0000  A         JSR    RØGPIC
FE 0001  N CLASG3  LDX    PØRTTL    *MØDIFY PØRT'S LAST
A6 0A    A         LDAA   10,X      **BYTE IN LNASGT
43                 CØMA
16                 TAB
FE 0005  N         LDX    LASGTL
A4 01    A         ANDA   1,X
A7 01    A         STAA   1,X
FE 0001  N         LDX    PØRTTL    *INPUT PØRT?
E6 06    A         LDAB   6,X
C5 01    A         BITB   #1
26 09 00DD         BNE    SHFPTS    *YES
FE 0007  N         LDX    PICATL    *PRØGRAM LINES USED
08                 INX              **BY PØRT IN PØRT'S LAST
08                 INX              **PIC AS INPUTS
17                 TBA
BD 0000  A         JSR    RPGPIC
B6 0000  A SHFPTS  LDAA   NPØRTS    *SHIFT PØRTST TABLE
B0 0003  N         SUBA   PØRTCN
27 2B 0110         BEQ    DNPØRT    *DELETED PØRT IS LAST
B7 0000  N         STAA   SHFCTR    *# ØF ENTRIES TØ
F6 0003  N         LDAB   PØRTCN    **BE SHIFTED
86 0C    A         LDAA   #12
BD 0000  A         JSR    MULPLY
CE 0000  A         LDX    #PØRTST
BD 0000  A         JSR    BLDADR
FF 0000  A         STX    SØURCE
C6 0C    A         LDAB   #12
F7 0001  A         STAB   MINUEN+1
7F 0000  A         CLR    MINUEN
BD 0000  A         JSR    SUBTRA
C6 0C    A SHFPRT  LDAB   #12       *MØVE 1 ENTRY UP
BD 0000  A         JSR    TSFBYT
7A 0000  N         DEC    SHFCTR    *ALL ENTRIES SHIFTED?
27 02 0110         BEQ    DNPØRT    *YES
20 F4 0104         BRA    SHFPRT    *NØ, SHIFT NEXT ENTRY
7A 0000  A DNPØRT  DEC    NPØRTS    *DECREASE NPØRTS 1
39                 RTS
                   END
```

ØUTPUT

```
                        NAM     ØUTPUT
                        ØPT     REL
                *ØUT(ØUTPUT TØ PØRT) CØMMAND MØDULE
                        XREF    BSCVBL,PARBUF,ØPØRTL,ØDATA,ØDATAL
                        XREF    BSCQL,IØER33,IØER34,IØER35,BSCBCD
                        XREF    STØØUT,INIHTS,MØDHTS,IØHDLR,SRPØRT
                        XDEF    ØUTPUT
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRAT1  CØMM    DSCT
   0002   A PØRTTL RMB     2          PØRT LØCATIØN IN PØRTST
                        PSCT
CE 0000   A ØUTPUT LDX     #PARBUF    *PØRT NAME EXISTS?
BD 0000   A        JSR     SRPØRT
26 03 000B         BNE     ØPEXIQ     *YES
7E 0000   A        JMP     IØER33     *NØ, ERRØR
FF 0000   N ØPEXIQ STX     PØRTTL     *ØUTPUT PØRT?
A6 06     A        LDAA    6,X
85 01     A        BITA    #1
27 03 0017         BEQ     ØPEXIS     YES
7E 0000   A        JMP     IØER33     *NØ, ERRØR
B6 0006   A ØPEXIS LDAA    PARBUF+6   *ØUTPUT VARIABLE?
27 02 001E         BEQ     ØUTVAR     *ØUTPUT VARIABLE
20 1D 003B         BRA     SØDATA
FE 0000   A ØUTVAR LDX     BSCVBL     *GET QO LØCATIØN
86 51     A        LDAA    #'Q        *QO?
A1 00     A        CMPA    X
27 03 002A         BEQ     ASCII0
7E 0000   A        JMP     IØER35
86 30     A ASCII0 LDAA    #'0
A1 01     A        CMPA    1,X
27 03 0033         BEQ     SBSCAD
7E 0000   A        JMP     IØER35
08              SBSCAD INX            *GØ TØ 1ST DATA BYTE
08                     INX
FF 0000   A        STX     BSCQL      *CØNVERT TØ BCD
BD 0000   A        JSR     BSCBCD
FE 0000   N SØDATA LDX     PØRTTL     *STØRE DATA INTØ
FF 0000   A        STX     ØPØRTL     **ØDATA
A6 06     A        LDAA    6,X
44                 LSRA
81 03     A        CMPA    #3
27 08 0050         BEQ     ØSGBCD     *3 BYTE WØRD
7F 0000   A        CLR     ØDATA      *2 BYTE WØRD
CE 0001   A        LDX     #ØDATA+1
20 03 0053         BRA     SØDATL
CE 0000   A ØSGBCD LDX     #ØDATA
FF 0000   A SØDATL STX     ØDATAL
BD 0000   A        JSR     STØØUT
26 03 005E         BNE     ØUT        *ØUTPUT DATA STØRED
```

```
7E 0000  A        JMP    IØER34   *ØUTPUT DATA ERRØR
BD 0000  A ØUT    JSR    INIHTS   *INITIALIZE I/Ø TABLES
FE 0000  N        LDX    PØRTTL
BD 0000  A        JSR    MØDHTS   *MØDIFY I/Ø TABLES
BD 0000  A        JSR    IØHDLR   *CALL I/Ø HANDLER
39                RTS
                  END
```

INPUT

```
                        NAM     INPUT
                        ØPT     REL
                *INP(INPUT FRØM PØRT) CØMMAND MØDULE
                        XREF    BSCVBL,PARBUF,IPØRTL,INDATA,INDATL
                        XREF    IBSCQL,IØER33,IØER35,SRPØRT,INIHTS
                        XREF    IØHDLR,GETINP,IDABSC,BLDADR
                        XDEF    INPUT
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRAT1  CØMM    DSCT
   0002   A     PØRTTL  RMB     2          PØRT LØCATIØN IN PØRTST
                        PSCT
CE 0000   A     INPUT   LDX     #PARBUF    *PØRT NAME EXISTS?
BD 0000   A             JSR     SRPØRT
26 03 000B              BNE     IPEXIQ     *YES
7E 0000   A             JMP     IØER33     *NØ, ERRØR
A6 06     A     IPEXIQ  LDAA    6,X        *INPUT PØRT?
85 01     A             BITA    #1
26 03 0014              BNE     IPEXIS     *YES
7E 0000   A             JMP     IØER33     *NØ, ERRØR
FF 0000   N     IPEXIS  STX     PØRTTL     *INITIALIZE TABLES
BD 0000   A             JSR     INIHTS
BD 0000   A             JSR     IØHDLR     *CALL I/Ø HANDLER
FE 0000   N             LDX     PØRTTL     *GET DATA INTØ
BD 0000   A             JSR     GETINP     **INDATA
FE 0000   A             LDX     BSCVBL     *GET Q1 LØCATIØN
C6 08     A             LDAB    #8
BD 0000   A             JSR     BLDADR
A6 00     A             LDAA    X          *Q1?
81 51     A             CMPA    #'Q
27 03 0034              BEQ     ASCII1
7E 0000   A             JMP     IØER35
A6 01     A     ASCII1  LDAA    1,X
81 31     A             CMPA    #'1
27 03 003D              BEQ     IBYTE
7E 0000   A             JMP     IØER35
08                IBYTE INX                *GØ TØ 1ST DATA BYTE
08                      INX
FF 0000   A             STX     IBSCQL
FE 0000N              LDX     PØRTTL
FF 0000   A             STX     IPØRTL
A6 06     A             LDAA    6,X        *DATA FØRMAT ØF PØRT?
44                      LSRA
81 03     A             CMPA    #3
26 05 0054              BNE     LXIDA2     *2 BYTE WØRD
CE 0000   A             LDX     #INDATA    *3 BYTE WØRD
20 03 0057              BRA     SIDATL
CE 0001   A     LXIDA2  LDX     #INDATA+1
FF 0000   A     SIDATL  STX     INDATL     *STØRE DATA LØCATIØN
BD 0000   A             JSR     IDABSC     *STØRE DATA IN Q1
39                      RTS
                        END
```

RTIØRQ

```
                    NAM     RTIØRQ
                    ØPT     REL,PAGE=47
            *PØR(PERIØDIC ØUTPUT REQUEST) AND PIR(PERIØDIC
             *INPUT REQUEST) CØMMAND MØDULES
                    XREF    BSCVBL,BUFSIZ,PARBUF,IØTBLN,NUMPIR
                    XREF    STATUS,IØWTIS,IØTBLS,BUFAVA
                    XREF    BFPTRS,BUFFER
                    XREF    ØDATAL,ØPØRTL,PSCQL,HBYTES,LBYTES
                    XREF    ADDEND,MINUEN,SØURCE
                    XREF    IØER33,IØER34,IØER36,IØER37,IØER38
                    XREF    STØØUT,SRPØRT,PSCVTQ,PSCBCD,CPBYTS
                    XREF    BLDAD2,SUBTRA,MULPLY,BLDADR,TSFBYT
                    XDEF    RTIØRQ
             *SCRATCH PAD FØR MAIN PRØGRAMS
             SCRAT1 CØMM    DSCT
   0001   A PØRTTP RMB     1          *PØRT DATA FØRMAT
   0002   A BFPTTL RMB     2          *BFPTRS TABLE PØINTER
   0002   A BUFLØC RMB     2          *BUFFER LØCATIØN
   0001   A WØRDCN RMB     1          *WØRD CØUNTER
   0002   A BSCVTL RMB     2          *BASIC VECTØR LØCATIØN
   0002   A PØRTTL RMB     2          *PØRTST TABLE PØINTER
   0002   A IØTBSL RMB     2          *I/Ø TABLES PØINTER
   0002   A BUFREQ RMB     2          *BUFFER SPACE REQUESTED
   0002   A TEMPXY RMB     2          *TEMPØRARY STØRAGE
             *PERIØDIC ØUTPUT AND PERIØDIC INPUT REQUEST
             *CØMMAND MØDULES
                    PSCT
B6 0000   A RTIØRQ LDAA    IØTBLN     *ANY I/Ø TABLES USED?
26 0B 0010         BNE     NXIØTQ     *YES
FE 0000   A        LDX     BUFSIZ     *INITIALIZE AVAILABLE
FF 0000   A        STX     BUFAVA     **BUFFER SPACE
7F 0000   A        CLR     NUMPIR     *SET NØ. ØF IN@. REQ. 0
20 07 0017         BRA     PØRTEQ
81 06     A NXIØTQ CMPA    #6         *ALL I/Ø TABLES USED?
2D 03 0017         BLT     PØRTEQ     *NØ
7E 0000   A        JMP     IØER38     *YES,ERRØR
CE 0001   A PØRTEA LDX     #PARBUF+1 *SPECIFIED PØRT EXISTS
BD 0000   A        JSR     SRPØRT     **IN PØRTST TABLE?
26 03 0022         BNE     RQPEXS     *YES
7E 0000   A        JMP     IØER33     *NØ,ERRØR
A6 06     A RQPEXS LDAA    6,X        *GET TYPE ØF PØRT
7D 0000   A        TST     PARBUF     *ØUTPUT ØR INPUT REQUEST?
27 07 0030         BEQ     PØRTØQ     *ØUTPUT REQUEST
85 01     A        BITA    #1         *INPUT PØRT?
26 0A 0037         BNE     IØTADR     *YES
7E 0000   A        JMP     IØER33     *ØUTPUT PØRT, ERRØR
85 01     A PØRTØQ BITA    #1         *ØUTPUT PØRT?
```

RTIØRQ

```
27 03 0037          BEQ    IØTADR   *YES
7E 0000  A          JMP    IØER33   *INPUTPØRT, ERRØR
FF 0008  N IØTADR   STX    PØRTTL
B6 0000  A          LDAA   IØTBLN   *FIND ADDRESS ØF I/Ø
C6 0D    A          LDAB   #13      *TABLE
BD 0000  A          JSR    MULPLY
CE 0000  A          LDX    #IØTBLS
BD 0000  A          JSR    BLDADR
FF 000A  N          STX    IØTBSL   *STØRE I/Ø TABLE ADDRESS
B6 0008  N          LDAA   PØRTTL   *STØRE PØRT ADR. INTØ
A7 00    A          STAA   X        **I/Ø TABLE
B6 0009  N          LDAA   PØRTTL+1
A7 01    A          STAA   1,X
FE 0008  N          LDX    PØRTTL   *FIND PØRT DATA FØRMAT
A6 06    A          LDAA   6,X
44                  LSRA
B7 0000  N          STAA   PØRTTP
B6 000B  A          LDAA   PARBUF+11 *GET NØ. ØF WØRDS
7F 000C  N          CLR    BUFREQ   **REQUESTED
48                  ASLA            *MULTIPLY BY 2
79 000C  N          RØL    BUFREQ
B7 000D  N          STAA   BUFREQ+1
B6 0000  N          LDAA   PØRTTP
81 03    A          CMPA   #3
27 02 0074          BEQ    WØRD3R
20 0C 0080          BRA    CMPBUF   *2 BYTE WØRD
FE 000C  N WØRD3R   LDX    BUFREQ   *3 BYTES WØRD
F6 000B  A          LDAB   PARBUF+11
BD 0000  A          JSR    BLDADR
FF 000C  N          STX    BUFREQ
CE 0000  A CMPBUF   LDX    #BUFAVA  *ENØUGH BUFFER SPACE LEFT?
FF 0000  A          STX    HBYTES
CE 000C  N          LDX    #BUFREQ
FF 0000  A          STX    LBYTES
C6 02    A          LDAB   #2
BD 0000  A          JSR    CPBYTS
26 03 0096          BNE    ØBSCVQ   *YES
7E 0000  A          JMP    IØER36   *NØ,ERRØR
FE 0000  A ØBSCVQ   LDX    BSCVBL   *ØUTPUT VECTØR EXISTS?
C6 28    A          LDAB   #40      *GET VECTØR ADDRESS
BD 0000  A          JSR    BLDADR
BD 0000  A          JSR    BSCVTQ   *QX VECTØR?
26 03 00A6          BNE    IØREQQ   *YES
7E 0000  A          JMP    IØER37
B6 0000  A IØREQQ   LDAA   PARBUF   *INPUT ØR ØUTPUT
27 11 00BC          BEQ    BSCSIZ   **REQUEST?
B6 0000  A          LDAA   NUMPIR   *INPUT REQUEST
```

RTIØRQ

```
4C                  INCA
EE 04     A NXINPV LDX     4,X        *NEXT INPUT VECTØR
BD 0000 A         JSR    BSCVTQ    *A QX VECTØR?
26 03 00B9          BNE    NXINVQ    *A QX VECTØR
7E 0000   A         JMP    IØER37    *NØ VECTØR RESERVED
4A          NXINVQ DECA              *VECTØR ADR. ØBTAINED?
                    BNE    NXINPV    *NO, NEXT BASIC VECTOR
B6 000B   A BSCSIZ LDAA    PARBUF+11 *GET NØ. ØF WØRDS
A1 02     A         CMPA   2,X       **REQUESTED
23 03 00C6          BLS    STØWDC    *VECTØR BIG ENØUGH
7E 0000   A         JMP    IØER37
B6 000B   A STØWDC LDAA    PARBUF+11
FE 000A   N         LDX    IØTBSL    *STØRE NØ. ØF WØRDS
A7 02     A         STAA   2,X       **INTØ I/Ø TABLE
A7 0C     A         STAA   12,X
CE 000C   A         LDX    #PARBUF+12 *STØRE PERIØD ØF ØUTPUT
FF 0000   A         STX    SØURCE    **ØR INPUT INTØ I/Ø TABLE
FE 000A   N         LDX    IØTBSL
08                  INX
08                  INX
08                  INX
C6 04     A         LDAB   #4
BD 0000   A         JSR    TSFBYT
CE 0007   A         LDX    #PARBUF+7 *STØRE START TIME FØR
FF 0000   A         STX    SØURCE    **PERIØDIC I/Ø INTØ
B6 0000   A         LDAA   IØTBLN    **I/Ø WAIT TIME TABLE
C6 04     A         LDAB   #4
BD 0000   A         JSR    MUPLY
CE 0000   A         LDX    #IØWTMS
BD 0000   A         JSR    BLDADR
C6 04     A         LDAB   #4
BD 0000   A         JSR    TSFBYT
09                  DEX              *INCREASE START TIME BY
6C 00     A         INC    X        **20 MILLISECØNDS
6C 00     A         INC    X
F6 0000   A         LDAB   IØTBLN    *FIND LØCATIØN ØF ENTRY
CE 0000   A         LDX    #STATUC   **IN STATUS TABLE
BD 0000   A         JSR    BLDADR
86 03     A         LDAA   #3
A7 00     A         STAA   X         *SET STATUS TØ 3
FE 0000   A         LDX    BUFAVA    *FIND LØCATIØN IN BUFFER
FF 0000   A         STX    MINUEN    **WHERE BUFFER SPACE FØR
FE 0000   A         LDX    BUFSIZ    **PØRT IS RESERVED
BD 0000   A         JSR    SUBTRA
FF 0000   A         STX    ADDEND
CE 0000   A         LDX    #BUFFER
BD 0000   A         JSR    BLDAD2
```

RTIØRQ

```
FF 000E  N         STX    TEMPXY    *STØRE INTØ SCRATCH
F6 0000  A         LDAB   IØTBLN    *FIND LØCATIØN ØF BUFFER
58                 ASLB             **PØINTER
CE 0000  A         LDX    #BFPTRS
BD 0000  A         JSR    BLDADR
FF 0001  N         STX    BFPTTL
B6 000E  N         LDAA   TEMPXY    *STØRE LØCATIØN ØF BUFFER
A7 00    A         STAA   X         **SPACE RESERVED INTØ
B6 000F  N         LDAA   TEMPXY+1  **BUFFER PØINTER
A7 01    A         STAA   1,X
B6 0000  A         LDAA   PARBUF    *PERIØDIC ØUT ØR IN?
27 03 0143         BEQ    SGLCIR    *ØUT
7E 01C5  P         JMP    DECBFS    *IN
FE 000A  N SGLCIR  LDX    IØTBSL    *CIRCULAR ØR ØNE-SHØT?
B6 0010  A         LDAA   PARBUF+16
26 04 014F         BNE    CIRØUT    *CIRCULAR ØUTPUT MØDE
6F 07    A         CLR    7,X       *INDICATE ØNE SHØT MØDE
20 19 0168         BRA    CØNØDA
C6 07    A CIRØUT  LDAB   #7        *STØRE TØTAL TIME INTØ
BD 0000  A         JSR    BLDADR    **I/Ø TABLE
A7 00    A         STAA   X         *INDICATE CIRCULAR MØDE
08                 INX
FF 000E  N         STX    TEMPXY
CE 0011  A         LDX    #PARBUF+17
FF 0000  A         STX    SØURCE
FE 000E  N         LDX    TEMPXY
C6 04    A         LDAB   #4
BD 0000  A         JSR    TSFBYT
FE 0001  N CØNØDA  LDX    BFPTTL    *CØNVERT DATA IN BASIC
EE 00    A         LDX    X         **VECTØR AND STØRE INTØ
FF 0003  N         STX    BUFLØC    **BUFFER
FF 0000  A         STX    ØDATAL
FE 0008  N         LDX    PØRTTL
FF 0000  A         STX    ØPØRTL
FE 0000  A         LDX    BSCVBL
C6 2E    A         LDAB   #46
BD0000   A         JSR    BLDADR
FF 0000  A         STX    BSCQL
FE 000A  N         LDX    IØTBSL
A6 02    A         LDAA   2,X
B7 0005  N         STAA   WØRDCN
BD 0000  A VECTBF  JSR    BSCBCD    *CØNVERT 1 VARIABLE TØ BCD
BD 0000  A         JSR    STØØUT    *STØRE DATA INTØ BUFFER
26 03 0197         BNE    NXBSØQ    *DATA CØRRECT
7E 0000  A         JMP    IØER34    *WRØNG DATA
7A 0005  N NXBSØQ  DEC    WØRDCN    *ALL CØNVERSIØN FINISHED?
27 29 01C5         BEQ    DECBFS    *YES
```

RTIØRQ

```
FE 0000  A          LDX    BSCQL     *NØ, CØNTINUE WITH NEXT
C6 06    A          LDAB   #6        **VARIABLE
BD 0000  A          JSR    BLDADR
FF 0000  A          STX    BSCQL
FE 0003  N          LDX    BUFLØC
B6 0000  N          LDAA   PØRTTP    *PØRT DATA FØRMAT?
81 03    A          CMPA   #3
27 04 01B5          BEQ    WØRD3B    *SIGNED BCD
C6 02    A          LDAB   #2        *ØTHERS, 2 BYTE WØRD
20 02 01B7          BRA    ADVBFP
C6 03    A WØRD3B   LDAB   #3        *3 BYTE WØRD
BD 0000  A ADVBFP   JSR    BLDADR
FF 0003  N          STX    BUFLØC
FF 0000  A          STX    ØDATAL
20 CA 018C          BRA    VECTBF
7E 0000  A VDATER   JMP    IØER34
FE 000C  N DECBFS   LDX    BUFREQ    *DECREASE BUFFER SPACE
FF 0000  A          STX    MINUEN    **AVAILABLE
FE 0000  A          LDX    BUFAVA
BD 0000  A          JSR    SUBTRA
FE 0000  A          STX    BUFAVA
F6 0000  A          LDAB   PARBUF
27 00 01D9          BEQ    INCTBN
7C 0000  A INCTBN   INC    IØTBLN    *INCREASE I/Ø TABLE NØ.
B6 0000  A          LDAA   PARBUF    *IN ØR ØUT REQUEST?
27 03 01E4          BEQ    PIØRQF    *ØUTPUT REQUEST
7C 0000  A          INC    NUMPIR   *INCREMENT NUMPIR
39          PIØRQF  RTS
39                  RTS
                    END
```

DELREQ

```
                        NAM     DELREQ
                        ØPT     REL,PAGE=50
                *DRQ(DELETE REQUEST) CØMMAND MØDULE
                        XREF    PARBUF,IØTBLN,NUMPIR,IØTBLS,BUFAVA
                        XREF    ADDEND
                        XREF    BLDADR,BLDAD2,MULPLY
                        XDEF    DELREQ
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRAT1  CØMM    DSCT
   0002   A     IØTBSL  RMB     2
   0002   A     BUFREQ  RMB     2
                        PSCT
7D 0000   A     DELREQ  TST     IØTBLN    *ANY I/Ø TABLES ASSIGNED?
27 66 006B              BEQ     DLREQF    *NØ
B6 0000   A             LDAA    PARBUF    *DELETE 0 REQUEST?
27 61 006B              BEQ     DLREQF    *YES
B1 0000   A             CMPA    IØTBLN    *DELETE ALL REQUESTS?
2D 04 0013              BLT     DL1REQ    *NØ
7F 0000   A             CLR     IØTBLN    *YES
39                      RTS
7A 0000   A     DL1REQ  DEC     IØTBLN    *DELETE 1 REQUEST
B6 0000   A             LDAA    IØTBLN    *GET ADDRESS ØF LAST
C6 0D     A             LDAB    #13       **I/Ø TABLE ASSIGNED
BD 0000   A             JSR     MULPLY
CE 0000   A             LDX     #IØTBLS
BD 0000   A             JSR     BLDADR
FF 0000   N             STX     IØTBSL    *SAVE I/Ø TABLE ADR.
A6 02     A             LDAA    2,X       *GET NØ. ØF WØRDS
7F 0002   N             CLR     BUFREQ    **REQUESTED
48                      ASLA              *FIND NØ. ØF BYTES IN
79 0002   N             RØL     BUFREQ    **BUFFER RESERVED
B7 0003   N             STAA    BUFREQ+1
EE 00     A             LDX     X         *FIND PØRT DATA FØRMAT
A6 06     A             LDAA    6,X
44                      LSRA
81 03     A             CMPA    #3        *SIGNED BCD?
26 0D 0049              BNE     BCDBIP    *BCD ØR BINARY
FE 0000   N             LDX     IØTBSL    *SINGED BCD
E6 02     A             LDAB    2,X       *3 BYTE WØRD
FE 0002   N             LDX     BUFREQ
BD 0000   A             JSR     BLDADR
20 03 004C              BRA     INCBFA
FE 0002   N     BCDBIP  LDX     BUFREQ    *2 BYTE WØRD
FF 0000   A     INCBFA  STX     ADDEND    *INCREASE BUFAVA BY
FE 0000   A             LDX     BUFAVA    **NØ ØF BYTES RESERVED
BD 0000   A             JSR     BLDAD2
FF 0000   A             STX     BUFAVA
FE 0000   N             LDX     IØTBSL    *INPUT REQUEST DELETED?
EE 00     A             LDX     X
A6 06     A             LDAA    6,X
```

DELREQ

```
85 01     A          BITA    #1
27 03 0066           BEQ     DLNXRQ    *ØUTPUT PØRT
7A 0000   A          DEC     NUMPIR    *DECREMENT NUMPIR
7A 0000   A DLNXRQ   DEC     PARBUF    *DELETE MØRE REQUESTS?
26 A8 0013           BNE     DL1REQ    *YES
39          DLREQF   RTS               *FINISHED
                     END
```

RTIØT

```
                    NAM     RTIØT
                    ØPT     REL,PAGE=49
                *REAL-TIME I/Ø TASK FØR PERIØDIC ØUTPUT
                *AND INPUT
                    XREF    BSCVBL,HEXNUM,BSCQL,ØDATA,INDATA
                    XREF    IPØRTL,IBSCQL,INDATL,MINUEN
                    XREF    TENMS,CALHDL,RTA,IØTBLN,NUMPIR
                    XREF    STATUS,IØWTMS,IØTBLS,BFPTRS
                    XREF    IØER35,IØHDLR,INIHTS,MØDHTS,GETINP
                    XREF    IDABSC,ADJHMS,BSCVTQ,BREAK,HEXBSC
                    XREF    SUBTRA,BLDADR
                    XDEF    RTIØTK
                *SCRATCH PAD FØR MAIN PRØGRAMS
                SCRAT1 CØMM  DSCT
   0001   A PØRTTP RMB     1          *PØRT DATA FØRMAT
   0002   A BSCVTL RMB     2          *LØCATIØN IN VECTØR
   0001   A WØRDCN RMB     1          *WØRD CØUNTER
   0002   A BUFLØC RMB     2          *LØCATIØN IN BUFFER
   0001   A RTIØF  RMB     1          *I/Ø FINISH FLAG
   0001   A IØTCTR RMB     1          *I/Ø TABLE CØUNTER
   0002   A STATTL RMB     2          *STATUS TABLE PØINTER
   0002   A IØTBSL RMB     2          *I/Ø TABLES PØINTER
   0002   A IØWTSL RMB     2          *I/Ø WAIT TIMES PØINTER
   0002   A BFPTTL RMB     2          *BFPTRS TABLE PØINTER
   0001   A IØWHR  RMB     1          *HØUR ØF PERIØD
   0001   A IØWMIN RMB     1          *MINUTE ØF PERIØD
   0001   A IØWSEC RMB     1          *SEC ØF PERIØD
   0001   A BØRRØW RMB     1
   0002   A TEMPXY RMB     2          *TEMPØRARY STØRAGE
   0001   A EQTMS  RMB     1          *PERIØD = !0 MS FLAG
   0002   A BSCVBG RMB     2          *ADDRESS ØF BASIC VECTØR
   0001   A PIRCTR RMB     1          INPUT REQUEST CØUNTER
                PSCT
                *REAL-TIME I/Ø TASK
B6 0000   A RTIØTK LDAA    IØTBLN     *ANY I/Ø REQUEST?
26 03 0008         BNE     PINPQ      *YES
7E 0315   P        JMP     RTXF       *NØ
B6 0000   A PINPQ  LDAA    NUMPIR     *ANY INPUT REQUESTED?
27 59 0066         BEQ     RTIØX      *NØ, JUST ØUTPUT
B7 0019   N        STAA    PIRCTR     *YES, BASIC VECTØRS
FE 0000   A        LDX     BSCVBL     **EXIST TØ STØRE DATA
C6 28     A        LDAB    #40        **TØ BE INPUTED?
BD 0000   A        JSR     BLDADR     *GET ØUTPUT VECTØR ADR.
BD 0000   A        JSR     BSCVTQ     *A VECTØR?
26 03 0020         BNE     SBSCBG     *YES
7E 0000   A        JMP     IØER35     *NØ, ERRØR
FF 0017   N SBSCBG STX     BSCVBG     *SAVE VECTØR ADDRESS
FF 0001   N        STX     BSCVTL
CE 0000   A        LDX     #IØTBLS
```

RTIØT

```
FF 000A  N         STX    IØTBSL
EE 00    A INPTBQ  LDX    X          *INPUT TABLE?
A6 06    A         LDAA   6,X        *INPUT PØRT USING TABLE?
85 01    A         BITA   #1
26 0D 0041        BNE    INPVTQ     *YES
FE 000A  N NXIØTV  LDX    IØTBSL     *NØ, GØ TØ NEXT TABLE
C6 0D    A        LDAB   #13
BD 0000  A         JSR    BLDADR
FF 000A  N         STX    IØTBSL
20 EB 002C         BRA    INPTBQ
FE 0001  N INPVTQ  LDX    BSCVTL     *INPUT VECTØR EXISTS
EE 04    A         LDX    4,X        **FØR INPUT PØRT?
FF 0001  N         STX    BSCVTL
BD 0000  A         JSR    BSCVTQ
26 03 0051         BNE    IVECSZ     *YES
7E 0000  A         JMP    IØER35     *NØ
A6 02    A IVECSZ  LDAA   2,X        *VECTØR LARGE ENØUGH?
FE 000A  N         LDX    IØTBSL
A1 02    A         CMPA   2,X
22 05 005F         BHI    NXINVQ     *YES
27 03 005F         BEQ    NXINVQ     *YES
7E 0000  A         JMP    IØER35     *NØ
7A 0019  N NXINVQ  DEC    PIRCTR     *ANY MØRE VECTØRS TØ
27 02 0066         BEQ    RTIØX      **BE CHECKED? NØ
20 CE 0034         BRA    NXIØTV
7F 0000  A RTIØX   CLR    TENMS      *CLEAR 10 MS FLAG
86 01    A         LDAA   #1         *TELL UPDATE REAL-TIME
B7 0000  A         STAA   RTA        **I/Ø TASK IS ACTIVE
B6 0000  A LØØP    LDAA   TENMS      *WAIT FØR 10 MS
27 FB 006E         BEQ    LØØP       **CLØCK INTERUPT
7F0000   A        CLR    TENMS      *CLEAR 10 MS FLAG
B6 0000  A         LDAA   CALHDL     *CALL I/Ø HANDLER?
27 06 0081         BEQ    INHDLT     *NØ
BD 0000  A         JSR    IØHDLR     *YES
7F 0000  A         CLR    CALHDL     *CLEAR CALL HDL. FLAG
BD 0000  A INHDLT  JSR    INIHTS     *INITIAL I/Ø TABLES
86 01    A         LDAA   #1         *INITIALIZE REAL TIME
B7 0006  N         STAA   RTIØF      **I/Ø FINISH FLAG
86 01    A         LDAA   #1         *INITIALIZE VARIØUS CØUNTE
B7 0007  N         STAA   IØTCTR     **AND PØINTERS
CE 0000  A         LDX    #STATUS
FF 0008  N         STX    STATTL
CE 0000  A         LDX    #IØTBLS
FF 000A  N         STX    IØTBSL
CE 0000  A         LDX    #IØWTMS
FF 000C  N         STX    IØWTSL
CE 0000  A         LDX    #BFPTRS
FF 000E  N         STX    BFPTTL
FE 0008  N NXPØRT  LDX    STATTL     *GET PØRT STATUS BYTE
```

RTIØT

```
A6 00     A          LDAA   X
26 03 00B0           BNE    CMPST3    *PØRT'S  I/Ø NØT FINISHED
7E 021B   P          JMP    NXIØTQ    *PØRT'S  I/Ø FINISHED
FE 000A   N CMPST3   LDX    IØTBSL    *STØRE PØRT DATA FØRMAT
EE 00     A          LDX    X
E6 06     A          LDAB   6,X
54                   LSRB
F7 0000   N          STAB   PØRTTP
81 03     A          CMPA   #3        *PØRT IS NØT ACTIVE?
26 03 00C2           BNE    ACTØPQ    *PØRT IS ACTIVE
7E 01D2   P          JMP    RDYPTQ    *PØRT ACTIVE NEXT INT.?
FE 000A   N ACTØPQ   LDX    IØTBSL    *ACTIVE ØUTPUT PØRT?
EE 00     A          LDX    X
A6 06     A          LDAA   6,X
85 01     A          BITA   #1
26 5A 0127           BNE    ACTIP     *ACTIVE INPUT PØRT
FE 000A   N          LDX    IØTBSL    *ACTIVE ØUTPUT PØRT
A6 07     A          LDAA   7,X       *ØNE SHØT ØUTPUT?
27 77 014B           BEQ    ØNESHØ    *YES
FE 000A   N          LDX    IØTBSL    *CIRCULAR ØUTPUT
BD 0316   P          JSR    SUBTIM    *SUBTRACT PERIØD FRØM
26 08 00E4           BNE    CØUTNF    *TØTAL TIME. NØT FINISHED
FE 0008   N          LDX    STATTL    *FINISHED. CLEAR STATUS BY
6F 00     A          CLR    X
7E 021B   P          JMP    NXIØTQ
FE 000A   N CØUTNF   LDX    IØTBSL    *DEC. WØRD CØUNTER FØR
6A 0C     A          DEC    12,X      **CIRCULAR ØUTPUT
26 6F 015A           BNE    ADBFPT    *LAST WØRD NØT ØUTPUTTED
E6 02     A          LDAB   2,X       *INITIALIZE WØRD CØUNTER
E7 0C     A          STAB   12,X
5A                   DECB             *RE-INITIALIZE BUFFER
17                   TBA              **PØINTER
7F 0000   A          CLR    MINUEN
58                   ASLB
79 0000   A          RØL    MINUEN
F7 0001   A          STAB   MINUEN+1
F6 0000   N          LDAB   PØRTTP
C1 03     A          CMPB   #3
26 0B 010D           BNE    RINCBP    *2 BYTE WØRD
BB 0001   A          ADDA   MINUEN+1  *3 BYTE WØRD
B7 0001   A          STAA   MINUEN+1
25 03 010D           BCS    RINCBP
7C 0000   A          INC    MINUEN    *PRØPAGATE CARRY
FE 000E   N RINCBP   LDX    BFPTTL    "GET PRESENT CØNTENTS
EE 00     A          LDX    X         **ØF BUFFER PØINTER
BD 0000   A          JSR    SUBTRA    *RE-ADJUST BUFFER PØINTER
FF 0014   N          STX    TEMPXY    *STØRE LØCATIØN INTØ
FE 000E   N          LDX    BFPTTL    **BUFFER PØINTER
B6 0014   N          LDAA   TEMPXY
```

RTIØT

```
A7 00     A          STAA   X
B6 0015   N          LDAA   TEMPXY+1
A7 01     A          STAA   1,X
"0 52 0179           BRA    INIIØW
FE 000A   N ACTIP    LDX    IØTBSL    *ACTIVE PØRT IS INPUT
EE 00     A          LDX    X        **INPUT PØRT
BD 0000   A          JSR    GETINP    *GET DATA INTØ INDATA
FE 000E   N          LDX    BFPTTL    *GET BUFFER LØCATIØN
EE 00     A          LDX    X
B6 0000   N          LDAA   PØRTTP
81 03     A          CMPA   #3
26 06 0141           BNE    SIBUF2    2 BYTES WØRD
B6 0000   A          LDAA   INDATA    *3 BYTES WØRD
A7 00     A          STAA   X         *STØRE DATA INTØ BUFFER
08                   INX
B6 0001   A SIBUF2   LDAA   INDATA+1
A7 00     A          STAA   X
B6 0002   A          LDAA   INDATA+2
A7 01     A          STAA   1,X
FE 000A   N ØNESHØ   LDX    IØTBSL    *DEC. WØRD CØUNTER FØR
6A 0C     A          DEC    12,X      **ØNE SHØT ØUTPUT AND
26 08 015A           BNE    ADBFPT    **INPUT. NØT FINISHED
FE 0008   N          LDX    STATTL    *FINISHED. CLEAR STATUS
6F 00     A          CLR    X         **BYTE
7E 021B   P          JMP    NXIØTQ
FE 000E   N ADBFPT   LDX    BFPTTL    *ADVANCE BUFFER PØINTER
EE 00     A          LDX    X
B6 0000   N          LDAA   PØRTTP
81 03     A          CMPA   #3        *PØRP DATA FØRMAT?
26 01 0167           BNE    ADBFP2    ADVANCE 2 BYTES
08                   INX              *ADVANCE 3 BYTES
08          ADBFP2   INX
08                   INX
FF 0014   N          STX    TEMPXY
FE 000E   N          LDX    BFPTTL
B6 0014   N          LDAA   TEMPXY
A7 00     A          STAA   X
B6 0015   N          LDAA   TEMPXY+1
A7 01     A          STAA   1,X
7F 0006   N INIIØW   CLR    RTIØF     *CLEAR RTIØF FLAG
FE 000A   N          LDX    IØTBSL    *INITIALIZE I/Ø TIME
86 01     A          LDAA   #1        *SET EQUAL TØ 10 MS
B7 0016   N          STAA   EQTMS     **FLAG
A6 03     A          LDAA   3,X       *GET HØUR ØF PERIØD
27 03 018B           BEQ    STØRHR    *HØURS IS 0
7F 0016   N          CLR    EQTMS     *CLEAR EQTMS FLAG
B7 0010   N STØRHR   STAA   IØWHR     *STØRE HR. IN IØWHR
A6 04     A          LDAA   4,X       *GET MIN. ØF PERIØD
27 03 0195           BEQ    STØMIN    *MIN IS 0
```

RTIØT

```
7F 0016  N         CLR    EQTMS
B7 0011  N STØMIN  STAA   IØWMIN    *STØRE MIN IN IØWMIN
A6 05    A         LDAA   5,X       *GET SEC ØF PERIØD IN A
27 05 01A1         BEQ    HMSEQ0    *SEC IS 0
7F 0016  N         CLR    EQTMS
20 12 01B3         BRA    LØDTMS
F6 0016  N HMSEQ0  LDAB   EQTMS     *HØUR, MIN & SEC =0?
27 0D 01B3         BEQ    LØDTMS    *NØ, PERIØD>10MS
E6 06    A         LDAB   6,X       *YES, GET TENS ØF MS
C1 01    A         CMPB   #1        *10 MS?
26 02 01AE         BNE    CEQTMS    *NØ
20 07 01B5         BRA    CPYIØW    *YES
7F 0016  N CEQTMS  CLR    EQTMS
20 02 01B5         BRA    CPYIØW
E6 06    A LØDTMS  LDAB   6,X
FE 000C  N CPYIØW  LDX    IØWTSL    *CØPY PERIØD INTØ I/Ø WAIT
BD 036C  P         JSR    INIØWT    **TIME TABLE
F6 0016  N         LDAB   EQTMS    *PERIØD=10MS?
26 23 01E3         BNE    RDYPRT    *YES
A6 03    A         LDAA   3,X       *DECREASE UNIT ØF SEC
81  1    A         CMPA   #1        **BY 1 AND ADD 100 TØ
2E 03 01C9         BGT    SSTAT3    **UNIT ØF 10 MS IF TEN
BD 0000  A         JSR    ADJHMS    **MS ØF PERIØD<=10MS
86 03    A SSTAT3  LDAA   #3        *SET STATUS BYTE TØ
FE 0008  N        LDX    STATTL    **TØ 3
A7 00    A         STAA   X         *PØRT ACTIVE IN 10MS?
20 49 021B         BRA    NXIØTQ
7F 0006  N RDYPTQ  CLR    RTIØF
FE 000C  N         LDX    IØWTSL
A6 03    A         LDAA   3,X
81 01    A         CMPA   #1
26 3D 021B         BNE    NXIØTQ    *NØ
BD 0000  A         JSR    ADJHMS    *HØUR, MIN AND SEC=0?
26 38 021B         BNE    NXIØTQ    *NØ
86 02    A RDYPRT  LDAA   #2        *PØRT ACTIVE IN 10MS
FE 0008  N         LDX    STATTL    *SET STATUS BYTE
A7 00    A         STAA   X         **TØ 2
FE 000A  N         LDX    IØTBSL    *ØUTPUT PØRT?
EE 00    A         LDX    X
A6 06    A         LDAA   6,X
85 01    A         BITA   #1
26 26 021B         BNE    NXIØTQ   *INPUT PØRD
FE 000E  N         LDX    BFPTTL    *ØUTPUT PØRT
EE 00    A         LDX    X
81 06    A         CMPA   #6        *SIGNED BCD PØRT?
26 08 0206         BNE    PØHXBC    *NØ
A6 00    A         LDAA   X         *YES, 3 BYTE WØRD
B7 0000  A         STAA   ØDATA
08                 INX
```

```
RTIØT
20 03 0209          BRA    SØDA23   *GET ØTHER 2 BYTES
7F 0000   A PØHXBC CLR    ØDATA
A6 00     A SØDA23 LDAA   X
B7 0001   A         STAA   ØDATA+1
6 01     A         LDAA   1,X
B7 0002   A         STAA   ØDATA+2
FE 000A   N         LDX    IØTBSL   *MØDIFY I/Ø TABLES
EE 00     A         LDX    X
BD 0000   A         JSR    MØDHTS
B6 0007   N NXIØTQ LDAA   IØTCTR   *ALL PØRTS SERVICED IN
B1 0000   A         CMPA   IØTBLN   **THE CURRENT INTERRUPT?
27 2B 024E          BEQ    RTIØFQ   *YES
7C 0007   N         INC    IØTCTR   *NØ, GØ TØ NEXT PØRT
FE 0008   N         LDX    STATTL
08                  INX
FF 0008   N         STX    STATTL
FE 000C   N         LDX    IØWTSL
C6 04     A         LDAB   #4
BD 0000   A         JSR    BLDADR
FF 000C   N         STX    IØWTSL
FE 000A   N         LDX    IØTBSL
C6 0D     A         LDAB   #13
BD 0000   A         JSR    BLDADR
FF 000A   N         STX    IØTBSL
FE 000E   N         LDX    BFPTTL
08                  INX
08                  INX
FF 000E   N         STX    BFPTTL
7E 00A6   P         JMP    NXPØRT
B6 0006   N RTIØFQ LDAA   RTIØF    *PERIØDIC I/Ø FINISHED?
26 06 0259          BNE    CLRRTA   *YES
BD 0000   A         JSR    BREAK    *BREAK TEST
7E 006E   P         JMP    LØØP     *CØNTINUE PERIØDIC I/Ø
7F 0000   A CLRRTA CLR    RTA      *TELL UPDATE I/Ø FINISHED
B6 0000   A         LDAA   NUMPIR   *ANY INPUT REQUESTS?
26 03 0264          BNE    ICØNIN   *YES
7E 0312   P         JMP    CLRTBN   *NØ
CE 0000   A ICØNIN LDX    #BFPTRS  *INITIALIZATIØN
FF 000E   N         STX    BFPTTL
CE 0000   A         LDX    #IØTBLS
FF 000A   N         STX    IØTBSL
EE 00     A IØTBQC LDX    X        *INPUT ØR ØUTPUT TABLE?
A6 06     A         LDAA   6,X
85 01     A         BITA   #1
26 03 027B          BNE    CØN1PT   *INPUT TABLE
7E 02FC   P         JMP    NXIØTC   *GØ TØ NEXT I/Ø TABLE
FE 0017   N CØN1PT LDX    BSCVBG   *CØNVERSIØN FØR 1 INPUT
EE 04     A         LDX    4,X      **PØRT. GET NEXT INPUT
FF 0017   N         STX    BSCVBG   **VECTØR ADDRESS
```

RTIØT

```
C6 06     A          LDAB   #6        *GET 1ST VARIABLE
BD 0000   A          JSR    BLDADR    **LØCATIØN IN VECTØR
FF 0001   N          STX    BSCVTL
FE 000A   N          LDX    IØTBSL    *GET DATA FØRMAT
EE  0     A          LDX    X
A6 06     A          LDAA   6,X
44                   LSRA
B7 0000   N          STAA   PØRTTP
FF 0000   A          STX    IPØRTL    *STØRE INPUT PØRT ADR.
FE000A  N          LDX    IØTBSL    *GET NØ. ØF WØRDS
E6 02     A          LDAB   2,X
F7 0003   N          STAB   WØRDCN
7F 0000   A          CLR    MINUEN    *READJUST BUFFER PØINTER
5A                   DECB
17                   TBA
58                   ASLB
79 0000   A          RØL    MINUEN
F7 0001   A          STAB   MINUEN+1
F6 0000   N          LDAB   PØRTTP    *SIGNED BCD PØRT?
81 03     A          CMPA   #3
26 0A 02BE           BNE    BIBCDC    *BINARY ØR BCD FØRMAT?
16                   TAB              *SIGNED BCD PØRT
FE 0000   A          LDX    MINUEN    *3 BYTE WØRD
BD 0000   A          JSR    BLDADR
FF 0000   A          STX    MINUEN
FE 000E   N BIBCDC   LDX    BFPTTL
EE 00     A          LDX    X
BD 0000   A          JSR    SUBTRA
FF 0004   N          STX    BUFLØC    *STØRE ADJUSTED PØINTER
FF 0000   A C1INPW   STX    INDATL    *CØNVERT 1 INPET WØRD
FE 0001   N          LDX    BSCVTL
FF 0000   A          STX    IBSCQL
BD 0000   A          JSR    IDABSC    *STØRE DATA INTØ BASIC
7A 0003   N          DEC    WØRDCN    *INPUT PØRT FINISHED?
27 1D 02F7           BEQ    CNXIPQ    *YES. MØRE INPUT PØRTS?
C6 06     A          LDAB   #6       *INPUT @ØRT NØT FINISHED
FE 0001   N          LDX    BSCVTL    *CØNVERT NEXT WØRD
BD 0000   A          JSR    BLDADR
FF 0001   N          STX    BSCVTL
FE 0004   N          LDX    BUFLØC    *ADVANCE BUFFER PØINTER
B6 0000   N          LDAA   PØRTTP    *DATA FØRMAT?
81 03     A          CMPA   3
26 01 02F0           BNE    ABLØC2    *ADVANCE 2 BYTES
08                   INX              *ADVANCE 3 BYTES
08          ABLØC2   INX
08                   INX
FF 0004   N          STX    BUFLØC
20 D2 02C9           BRA    C1INPW    *CØNVERT NEXT WØRD
7A 0000   A CNXIPQ   DEC    NUMPIR    *ALL INPUTTED DATA FØR
```

RTIØT

```
27 16 0312          BEQ    CLRTBN   **PØRTS FINISHED? YES
FE 000E   N NXIØTC  LDX    BFPTTL   *NØ, GØTØ NEXT I/Ø
08                  INX              **TABLE
08                  INX
FF 000E   N         STX    BFPTTL
FE 000A   N         LDX    IØTBSL
C6 0D     A         LDAB   #13
BD 0000   A         JSR    BLDADR
FF 000A   N         STX    IØTBSL
7E 0270   P         JMP    IØTBQC   *GØ TØ NEXT TABLE
7F 0000   A CLRTBN  CLR    IØTBLN   *RELEASE I/Ø TABLES
39          RTXF    RTS             *PERIØDIC I/Ø FINISHED
            *SUBRØUTINE TØ SUBTRACT PERIØD FRØM TØTAL
            *TIME FØR CIRCULAR MØDE ØUTPUT
7F 0013   N CUBTIM  CLR    BØRRØW   *INITIAL BØRRØW TØ 0
A6 0B     A         LDAA   11,X     *TEN MS ØF TØTAL TIME
A1 06     A         CMPA   6,X      **>=TEN MS ØF PERIØD?
27 07 0326          BEQ    SUBTMS   *YES
22 05 0326          BHI    SUBTMS   *YES
8B 64     A         ADDA   #100     *NØ, BØRRØW FRØM SEC
7C 0013   N         INC    BØRRØW
A0 06     A SUBTMS  SUBA   6,X      *SUBTRACT 10 MS'S
A7 0B     A         STAA   11,X     *MØDIFY 10 MS'S ØF TØTAL
A6 0A     A         LDAA   10,X     *SEC ØF TATAL TIME>=
E6 05     A         LDAB   5,X      **SEC ØF PERIØD+BØRRØW?
FB 0013   N         ADDB   BØRRØW
7F 0013   N         CLR    BØRRØW
11                  CBA
22 07 033E          BHI    SUBSEC   *YES
27 05 033E          BEQ    SUBSEC   *YES
7C 0013   N         INC    BØRRØW   *NØ, BØRRØW FRØM MIN
8B 3C     A         ADDA   #60
10          SUBSEC  SBA             *SUBTRACT SEC'S
A7 0A     A         STAA   10,X     *CHANGE SEC ØF TØTAL TIME
A6 09     A         LDAA   9,X      *MIN ØF TØTAL TIME>=
E6 04     A         LDAB   4,X      **MIN ØF PERIØD+BØRRØW?
FB 0013   N         ADDB   BØRRØW
7F 0013   N         CLR    BØRRØW
11                  CBA
27 07 0355          BEQ    SUBMIN   *YES
22 05 0355          BHI    SUBMIN   *YES
7C 0013   N         INC    BØRRØW   *NØ, BØRRØW 1 HØUR
8B 3C     A         ADDA   #60
10          SUBMIN  SBA             *SUBTRACT MIN'S
A7 09     A         STAA   9,X      *CHANGE MIN ØF TØTAL TIME
A6 08     A         LDAA   8,X      *HØUR ØF TØTAL TIME>=
E6 03     A         LDAB   3,X      **HØUR ØF PERIØD+BØRRØW?
FB 0013   N         ADDB   BØRRØW
11                  CBA
```

RTIØT

```
27 02 0364          BEQ    SUBHR    *EQUAL
23 06 036A          BLS    CLRB     *LESS THAN
10            SUBHR SBA             *>=, SUBTRACT HØURS
A7 08      A        STAA   8,X      *CHANGE HRS. ØF TØTAL TIME
C6 01      A        LDAB   #1       *INDICATE TØTAL TIME>=
39                  RTS             **PERIØD
5F            CLRB  CLRB            *INDICATE TØTAL TIME<
39                 RTS             **PERIØD
              *SUBRØUTINE TØ CØPY PERIØD INTØ I/Ø WAIT TIME
              *LØCATIØN ØF I/Ø TABLE STØRED IN X.
E7 03      A INIØWT STAB   3,X
A7 02      A        STAA   2,X
B6 0011    N        LDAA   IØWMIN
A7 01      A        STAA   1,X
B6 0010    N        LDAA   IØWHR
A7 00      A        STAA   X
39                  RTS
                    END
 00000
```

TMREAD

```
                    NAM     TMREAD
                    ØPT     REL,PAGE=49
                    XREF    BSCVBL,TIME,PARBUF,BSCQL,HEXNUM
                    XREF    IØER35,BSCVTQ,ØUTSTR,ØUTBDB,ØUTBDC
                    XREF    HEXBSC,BLDADR
                    XDEF    TMREAD
            *TMR(READ TIME) CØMMAND MØDULE
            *SCRATCH PAD FØR MAIN PRØGRAM
            SCRAT1 CØMM     DSCT
   0001   A CØNTMC RMB      1          TIME UNIT CØUNTER
   0003   A TIMEF  RMB      3          SCRATCH TØ STØRE TIME
   0002   A TIMEFL RMB      2          LØCATIØN IN TIMEF
   0002   A TMVTLC RMB      2          *TIME VECTØR LØCATIØN
                    PSCT
   54     A TIMEMG FCC      'TIME(HØUR MIN SEC): '
   04     A        FCB      4
0F          TMREAD SEI                 *SET INTERRUPT MASK
B6 0000    A        LDAA     TIME      *READ HØUR, MINUTE
B7 0001    N        STAA     TIMEF     **AND SECØNDS AND
B6 0001    A        LDAA     TIME+1    **STØRE INTØ TIMEF
B7 0002    N        STAA     TIMEF+1
B6 0002    A        LDAA     TIME+2
B7 0003    N        STAA     TIMEF+2
0E                  CLI                *CLEAR INTERRUPT MASK
CE 0001    N        LDX      #TIMEF
FF 0004    N        STX      TIMEFL
86 03      A        LDAA     #3
B7 0000    N        STAA     CØNTMC
FE 0000    A        LDX      BSCVBL    *GET Q2 LØCATIØN
C6 10      A        LDAB     #16
BD 0000    A        JSR      BLDADR
BD 0000    A        JSR      BSCVTQ    *A NUMBERED Q VECTØR?
26 03 0044          BNE      TMVECS    *YES. VECTØR SIZE?
7E 0000    A        JMP      IØER35    *NØT Q VECTØR, ERRØR
C6 03      A TMVECS LDAB     #3        *MØRE THAN 3 ELEMENTS?
E1 02      A        CMPB     2,X
23 03 004D          BLS      DSPTMQ    *AT LEAST 3 ELEMENTS
7E 0000    A        JMP      IØER35    *LESS THAN 3 ELEMENTS
7D 0000    A DSPTMQ TST      PARBUF    *DISPLAY TIME?
27 0C 005E          BEQ      HRVARI    *NØ
FF 0006    N        STX      TMVTLC    *SAVE TIME VECTØR ADR.
CE 0000    P        LDX      #TIMEMG   *GET 'TIME' STRING
BD 0000    A        JSR      ØUTSTR    *ØUTPUT 'TIME'
FE 0006    N        LDX      TMVTLC
C6 06      A HRVARI LDAB     #6        *HØUR VARIABLE LØCATIØN
BD 0000    A CØNTIM JSR      BLDADR
FF 0000    A        STX      BSCQL     *CØNVERT A UNIT ØF
7F 0000    A        CLR      HEXNUM    **TIME AND STØRE INTØ
FE 0004    N        LDX      TIMEFL    **A BASIC VABIABLE
```

TMREAD

```
A6 00     A          LDAA   X
B7 0001   A          STAA   HEXNUM+1
BD 0000   A          JSR    HEXBSC
7D 0000   A          TST    PARBUF    *DISPLAY TIME?
27 15 008E           BEQ    CNXTMQ    *NØ
FE 0000   A          LDX    BSCQL     *YES
A6 00     A          LDAA   X         *GET 1ST DIGIT
E6 05     A          LDAB   5,X       *NØ. > 9?
C1 02     A          CMPB   #2
2D 0A 008E          BLT    CNXTMQ    *YES
E6 01     A          LDAB   1,X       *GET NEXT DIGIT
59                   RØLB             *SHIFT DIGITS INTØ A
49                   RØLA
59                   RØLB
49                   RØLA
59                   RØLB
49                   RØLA
59                   RØLB
49                   RØLA
7A 0000   N CNXTMQ   DEC    CØNTMC    *SEC CØNVERTED?
27 16 00A9           BEQ    DSSECQ    *YES
7D 0000   A          TST    PARBUF    *DISPLAY LAST UNIT?
27 03 009B           BEQ    CNNXTM    *NØ
BD 0000   A          JSR    ØUTBDB    *DISPLAY DIGITS & BLANK
FE 0004   N CNNXTM   LDX    TIMEFL    *CØNTINUE TØ CØNVERT
08                   INX              **NEXT UNIT
FF 0004   N          STX    TIMEFL
FE 0000   A          LDX    BSCQL     *NEXT BASIC VARIABLE
C6 06     A          LDAB   #6        **IN Q2 VECTØR
20 B7 0060           BRA    CØNTIM
7D 0000   A DSSECQ   TST    PARBUF    *DISPLAY TIME?
27 03 00B1           BEQ    TMREDF    *NØ
BD 0000   A          JSR    ØUTBDC    *DISPLAY SECØNDS & C/R
39          TMREDF   RTS              FINISHED
                     EN@
 00000
```

TIMSET

```
                        NAM     TIMSET
                        ØPT     REL
                *TMS(TIME SET) CØMMAND MØDULE
                        XREF    PARBUF,TIME
                        XDEF    TIMSET
                        PSCT
                *TMS(SET SYSTEM CLØCK) CØMMAND MØDULE
7F 0003   A TIMSET CLR     TIME+3    *CLR 10 MS ØF CLØCK
B6 0000   A        LDAA    PARBUF    *SET HØUR
B7 0000   A        STAA    TIME
B6 0001   A        LDAA    PARBUF+1  *SET MINUTE
B7 0001   A        STAA    TIME+1
B6 0002   A        LDAA    PARBUF+2  *SET SECØNDS
B7 0002   A        STAA    TIME+2
39                 RTS
                   END
```

WAIT

```
                        NAM     WAIT
                        ØPT     REL
                *WAIT CØMMAND MØDULE
                        XREF    PARBUF,BWAITT,RTA,ADJHMS,BREAK
                        XDEF    WAIT
                        PSCT
                *WAITCØMMAND MØDULE
B6 0000   A WAIT   LDAA    PARBUF    *STØRE TIME TØ
B7 0000   A        STAA    BWAITT    **WAIT INTØ BWAITT
B6 0001   A        LDAA    PARBUF+1
B7 0001   A        STAA    BWAITT+1
B6 0002   A        LDAA    PARBUF+2
B7 0002   A        STAA    BWAITT+2
CE 0000   A        LDX     #BWAITT   *CØNVERT 1 SEC. TØ
BD 0000   A        JSR     ADJHMS    *100 TENS ØF MS
86 02     A        LDAA    #2        *INFØRM UPDATE WAIT
B7 0000   A        STAA    RTA       **MØDULE IS ACTIVE
B6 0003   A WLØØP  LDAA    BWAITT+3  *TENS ØF MS IS 0?
26 FB 001D         BNE     WLØØP     *NØ
CE 0000   A        LDX     #BWAITT   *YES. TIME TØ WAIT
BD 0000   A        JSR     ADJHMS    **IS 0?
27 05 002F         BEQ     WAITFN    *YES
BD 0000   A        JSR     BREAK     *BREAK TEST
20 EE 001D         BRA     WLØØP     *CØNTINUE TØWAIT
7F 0000   A WAITFN CLR     RTA       *INFØRM UPDATE WAIT
39                 RTS               **IS FINISHED
                   END
```

LEVEL 1 SUBROUTINES

IØHDLR

```
                    NAM     IØHDLR
                    ØPT     REL,PAGE=49
            *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
            SCRAT2 CØMM     DSCT
   0001   A PICCTR RMB      1
    002   A PICATL RMB      2
   0002   A LNVLTL RMB      2
   0002   A AØLNTL RMB      2
   0002   A ØDATTL RMB      2
                    XREF    PICADT,LNVALT,AØLINT,ØDATAT
                    XREF    NUMPIC
                    XDEF    IØHDLR
            *I/Ø HANDLER
                    PSCT
CE 0000   A IØHDLR LDX      #PICADT   *READ ALL PIC'S
FF 0001   N         STX     PICATL
B6 0000   A         LDAA    NUMPIC
B7 0000   N         STAA    PICCTR
CE 0000   A         LDX     #LNVALT
FF 0003   N         STX     LNVLTL
FE 0001   N         LDX     PICATL
EE 00     A REDPIC  LDX     X         *READ ØNE PIC
A6 00     A         LDAA    X
FE 0003   N         LDX     LNVLTL
A7 00     A         STAA    X
7A 0000   N         DEC     PICCTR    *ANY MØRE PIC'S?
27 0E 0031          BEQ     ANDAØL    *NØ
08                  INX
FF 0003   N         STX     LNVLTL
FE 0001   N         LDX     PICATL
08                  INX
08                  INX
FF 0001   N         STX     PICATL
20 E4 0015          BRA     REDPIC
B6 0000   A ANDAØL  LDAA    NUMPIC    *'AND' LNVALT WITH
B7 0000   N         STAA    PICCTR    **AØLINT
CE 0000   A         LDX     #LNVALT
FF 0003   N         STX     LNVLTL
CE 0000   A         LDX     #AØLINT
FF 0005   N         STX     AØLNTL
A6 00     A ANDAØ1  LDAA    X         *'AND' 1 BYTE
FE 0003   N         LDX     LNVLTL
A4 00     A         ANDA    X
A7 00     A         STAA    X
7A 0000   N         DEC     PICCTR    *ALL BYTES FINISHED?
27 0D 005E          BEQ     ØRØDAT    *YES
08                  INX
FF 0003   N         STX     LNVLTL
FE 0005   N         LDX     AØLNTL
```

IØHDLR

```
08                  INX
FF 0005  N          STX     AØLNTL
20 E5 0043          BRA     ANDAØ1
B6 0000  A ØRØDAT   LDAA    NUMPIC    *'ØR' CØNTENTS ØF LNVALT
B7 0000  N          STAA    PICCTR    *AND ØDATAT
CE 0000  A          LDX     #LNVALT
FF 0003  N          STX     LNVLTL
CE 0000  A          LDX     #ØDATAT
FF 0007  N          STX     ØDATTL
A6 00    A ØRØDA1   LDAA    X         *'ØR' 1 BYTE
FE 0003  N          LDX     LNVLTL
AA 00    A          ØRAA    X
A7 00    A          STAA    X
7A 0000  N          DEC     PICCTR    *ALL BYTES 'ØR'ED?
27 0D 008B          BEQ     ØUTLVT    *YES
08                  INX
FF 0003  N          STX     LNVLTL
FE 0007  N          LDX     ØDATTL
08                  INX
FF 0007  N          STX     ØDATTL
20 E5 0070          BRA     ØRØDA1
B6 0000  A ØUTLVT   LDAA    NUMPIC    *ØUTPUT MØDIFIED LNVALT
B7 0000  N          STAA    PICCTR    **TØ PIC'S
CE 0000  A          LDX     #PICADT
FF 0001  N          STX     PICATL
CE 0000  A          LDX     #LNVALT
FF 0003  N          STX     LNVLTL
A6 00    A ØUTPIC   LDAA    X         *ØUTPUT TØ 1 PIC
FE 0001  N          LDX     PICATL
EE 00    A          LDX     X
A7 00    A          STAA    X
7A 0000  N          DEC     PICCTR    *ALL PIC'S ØUTPUTTED?
27 11 00BC          BEQ     IØHDLF    *YES
FE 0001  N          LDX     PICATL
08                  INX               *MØVE TØ ADDRESS ØF
08                  INX               **NEXT PIC
FF 0001  N          STX     PICATL
FE 0003  N          LDX     LNVLTL
08                  INX
FF 0003  N          STX     LNVLTL
20 E1 009D          BRA     ØUTPIC
39          IØHDLF  RTS               *I/Ø HANDLER FINISHED
                    END
```

MØDHTS

```
                    NAM     MØDHTS
                    ØPT     REL,PAGE=49
              *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
              SCRAT2 CØMM   DSCT
   0002   A PØRTTL RMB      2          PØRTST TABLE PØINTER
   0001   A PPICCN RMB      1          PØRT PIC CØUNTER
   0002   A AØLNTL RMB      2          AØLINT TABLE PØINTER
                    XREF    ØDATA,ØDATAT,AØLINT,BLDADR
                    XDEF    MØDHTS
              *SUBRØUTINE TØ MØDIFY IØ HANDLER'S ØUTPUT TABLES
             *LØCATIØN ØF PØRT IN PØRTST SPECIFIED IN X,
              *ØUTPUT DATA STØRED IN ØDATA
                    PSCT
FF 0000   N MØDHTS STX      PØRTTL
A6 08     A        LDAA     8,X        *GET NØ. ØF PIC'S WHICH
B7 0002   N        STAA     PPICCN     **THE PØRT SPANS
A6 09     A        LDAA     9,X        *GET 1ST BIT MAP
E6 07     A        LDAB     7,X        *GET NØ. ØF PIC'S WHICH
CE 0000   A        LDX      #AØLINT    **PØRT SPANS
BD 0000   A        JSR      BLDADR     *GET LØCATIØN IN AØLINT
43                 CØMA                **TABLE AND SET THE BITS
A4 00     A        ANDA     X          *CØRRESPØNDING TØ PØRT TØ
A7 00     A        STAA     X
7A 0002   N        DEC      PPICCN
27 1B 0037         BEQ      DETSHF    *PØRT SPANS 1 PIC
86 02     A       LDAA   #2
B1 0002   N        CMPA     PPICCN
26 03 0026         BNE      LASTMK     *PØRT SPANS 2 PIC
08                 INX                 *PØRT SPANS 3 PIC'S
6F 00     A        CLR      X          *CLEAR NEXT BYTE IN AØLINT
08          LASTMK INX                 *CLEAR BITS CØRRESPØNDING
FF 0003   N        STX      AØLNTL     **TØ LINES ØF THE LAST
FE 0000   N        LDX      PØRTTL     **PIC USED BY THE PØRT
A6 0A     A        LDAA     10,X
43                 CØMA
FE 0003   N        LDX      AØLNTL
A4 00     A        ANDA     X
A7 00     A        STAA     X
FE 0000   N LETSHF LDX      PØRTTL
A6 0B     A        LDAA     11,X
27 0E 004C         BEQ      SHFDAF
78 0002   A SHFDAT ASL      ØDATA+2    *SHIFT ØUTPUT DATA
79 0001   A        RØL      ØDATA+1
79 0000   A        RØL      ØDATA
4A                 DECA
27 02 004C         BEQ      SHFDAF
20 F2 003E         BRA      SHFDAT
FE 0000   N SHFDAF LDX      PØRTTL     *STØRE DATA INTØ ØDATAT
A6 08     A        LDAA     8,X
```

MØDHTS

```
E6 07     A          LDAB   7,X
CE 0000   A          LDX    #ØDATAT
BD 0000   A          JSR    BLDADR    *GET LØCATIØN IN ØDATAT
F6 0002   A          LDAB   ØDATA+2
EA 00     A          ØRAB   X
E7 00     A          STAB   X         *STØRE DATA
4A                   DECA
27 1B 007E           BEQ    MDHTSF    *PØRT SPANS 1 PIC
81 02     A          CMPA   #2
26 10 0077           BNE    MDHTS2    *PØRT SPANS 2 PIC'S
B6 0001   A          LDAA   ØDATA+1   *STØRE DATA INTØ 3
AA 01     A          ØRAA   1,X       **BYTES ØF ØDATAT
A7 01     A          STAA   1,X
B6 0000   A          LDAA   ØDATA
AA 02     A          ØRAA   2,X
A7 02     A          STAA   2,X
20 07 007E           BRA  MDHTSF
B6 0001   A MDHTS2   LDAA   ØDATA+1   *STØRE DATA INTØ 2
AA 01     A          ØRAA   1,X       **2 BYTES ØF ØDATAT
A7 01     A          STAA   1,X
39          MDHTSF   RTS
                     END
```

INIHTS

```
                     NAM    INIHTS
                     ØPT    REL
            *SUBRØUTINE TØ INITIALIZE I/Ø HANDLER TABLES.
            *SET ALL BITS ØF AØLINT AND CLEAR ALL BITS ØF
            *ØDATAT
            *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
            SCRAT2 CØMM   DSCT
   0002   A AØLNTL RMB    2          AØLINT TABLE PØINTER
   0002   A ØDATTL RMB    2          ØDATAT TABLE PØINTER
                     XREF   NUMPIC,AØLINT,ØDATAT
                     XDEF   INIHTS
                     PSCT
CE 0000   A INIHTS   LDX    #AØLINT   *SET ALL BITS ØF AØLINT
B6 0000   A          LDAA   NUMPIC    **TØ 1
C6 FF     A          LDAB   #$FF
E7 00     A INIAØL   STAB   X
4A                   DECA             *FINISHED?
27 03 0010           BEQ    INIØDA    *YES
08                   INX
20 F8 0008           BRA    INIAØL
CE 0000   A INIØDA   LDX    #ØDATAT   *CLEAR ALL BITS ØF
B6 0000   A          LDAA   NUMPIC    **ØDATAT
6F 00     A INIØD1   CLR    X
4A                   DECA             *FINISHED?
27 03 001E           BEQ    INIHTF    *YES
08                   INX
20 F8 0016           BRA    INIØD1
39          INIHTF   RTS
                     END
```

GETINP

```
                    NAM     GETINP
                    ØPT     REL
              *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
              SCRAT2 CØMM   DSCT
   0002   A PØRTTL RMB     2          PØRTST TABLE PØINTER
   0001   A MASK1  RMB     1          STØRES 1ST BIT MAP
   0001   A MASK2  RMB     1          STØRES 2ND BIT MAP
                    XREF    INDATA,LNVALT,BLDADR
                    XDEF    GETINP
              *SUBRØUTINE TØ GET DATA FØR PØRT FRØM LNVALT
              *LØCATIØN ØF PØRT IN PØRTST SPECIFIED IN X.
              *DATA IS STØRED IN INDATA.
                    PSCT
 7F 0000   A GETINP CLR     INDATA
 7F 0001   A        CLR     INDATA+1
 FF 0000   N        STX     PØRTTL
A6 09      A       LDAA    9,X        *GET FIRST BIT MAP
 B7 0002   N        STAA    MASK1
 A6 0A     A        LDAA    10,X       *GET 2ND BIT MAP
 B7 0003   N        STAA    MASK2
 E6 07     A        LDAB    7,X        *GET LABEL ØF PIC
 A6 08     A        LDAA    8,X        *GET # ØF PIC'S
 CE 0000   A        LDX     #LNVALT    *GET CØRRECT LØCATIØN IN
 BD 0000   A        JSR     BLDADR     **LNVALT TABLE
 E6 00     A        LDAB    X          *GET DATA
 F4 0002   N        ANDB    MASK1
 F7 0002   A        STAB    INDATA+2
 4A                 DECA
 27 18 0040         BEQ     SHIFTQ     *PØRT SPANS 1 PIC
 E6 01     A        LDAB    1,X        *GET NEXT SIGNIFICANT
 4A                 DECA               **DATA BYTE
 27 0D 003A        BEQ     IN2PIC     *PØRT SPANS 2 PIC
 F7 0001   A        STAB    INDATA+1
 E6 02     A        LDAB    2,X        *GET MØST SIG. BYTE
 F4 0003   N        ANDB    MASK2
 F7 0000   A        STAB    INDATA
 20 060040         BRA     SHIFTQ
 F4 0003   N IN2PIC ANDB    MASK2
 F7 0001   A        STAB    INDATA+1
 FE 0000   N SHIFTQ LDX     PØRTTL
 E6 0B     A        LDAB    11,X
 27 0D 0054         BEQ     GETINF
 0C          SHFIDA CLC                *SHIFT INPUT DATA
 76 0000   A        RØR     INDATA
 76 0001   A        RØR     INDATA+1
 76 0002   A        RØR     INDATA+2
 5A                 DECB
 26 F3 0047         BNE     SHFIDA
 39          GETINF RTS
                    END
```

BSCBCD

```
                        NAM     BSCBCD
                        ØPT     REL,PAGE=49
                *CØMMØN SCRATCH FØR LEVEL 1 SUBRØUTINES
                SCRAT2 CØMM     DSCT
   0002   A BCDDGL RMB     2
   0001   A BØRRØW RMB     1
   0002   A BSCQDL RMB     2
   0001   A NIBBLE RMB     1
                        XREF    HEXNUM,SIGN,BCDNUM,BSCQL
                        XREF    BLDADR
                        XDEF    BSCBCD
                        PSCT
                *SUBRØUTINE TØ CØNVERT BASIC VARIABLE TØ BCD.
                *LØCATIØN ØF VARIABLE GIVEN IN BSCQL AND BCD
                *NUMBER STØRED IN BCDNUM. INTEGER PART ØF NØ.
                *MUST BE LESS THAN 99999. IF SØ, B IS SET TØ 1,
                *ØTHERWISE B IS SET TØ 0.
7F 0000   A BSCBCD CLR     SIGN      *CLEAR SIGN ØF BCD NØ.
86 05     A        LDAA    #5        *5 DIGITS ØNLY
CE 0000   A        LDX     #BCDNUM
6F 00     A CLRBCD CLR     X         *CLEAR BCDNUM
4A                 DECA
27 03 0010         BEQ     LØDPWR
08                 INX
20 F8 0008         BRA     CLRBCD
FE 0000   A LØDPWR LDX     BSCQL     *GET NØ. ØF DIGITS
E6 05     A        LDAB    5,X       **IN INTEGRAL PART
2E 06 001D         BGT     CMPPW5    *CANNØT BE  <1 UNLESS
27 03 001C         BEQ     ØBSCO     **IT IS 0
7E 00A4   P        JMP     BSCERR    *IT IS NEG., WRØNG
39          ØBSCO  RTS               *VARIABLE IS 0
C1 05     A CMPPW5 CMPB    #5        *IS PØWER <= 5?
2F 03 0024         BLE     SBCDNØ    *YES,CØRRECT
7E 00A4   P        JMP     BSCERR    *NØ,INCØRRECT
CE 0004   A SBCDNØ LDX     #BCDNUM+4
FF 0000   N        STX     BCDDGL
FE 0000   A        LDX     BSCQL
17                 TBA
85 01     A        BITA    #1
26 07 0039         BNE     CLRNIB
C6 01     A        LDAB    #1        *START WITH LEFT NIBBLE
F7 0005   N        STAB    NIBBLE
20 03 003C         BRA     BSCLAD
7F 0005   N CLRNIB CLR     NIBBLE    *START WITH RIGHT NIBBLE
16          BSCLAD TAB               *GET ADR. ØF LAST BASIC
54                 LSRB              **VARIABLE BYTE TØ BE
BD 0000   A        JSR     BLDADR    **LØADEDED
FF 0003   N        STX     BSCQDL
F6 0005   N MBCDDG LDAB    NIBBLE
```

BSCBCD

```
27 0F 0058          BEQ    LØDRGH
E6 00    A          LDAB   X          *GET LEFT NIBBLE
54                  LSRB
54                  LSRB
54                  LSRB
54                  LSRB
7F 0005  N          CLR    NIBBLE     *INDICATE RIGHT NIBBLE
09                  DEX               *GØ TØ NEXT MØST
FF 0003  N          STX    BSCQDL     *SIGNIFICANT BYTE
20 07 005F          BRA    STØBCD
E6 00    A LØDRGH   LDAB   X          *GET RIGHT NIBBLE
C4 0F    A          ANDB   #$F
7C 0005  N          INC    NIBBLE     *INDICATE LEFT NIBBLE
FE 0000  N STØBCD   LDX    BCDDGL     *STØRE NIBBLE INTØ BCDNUM
E7 00    A          STAB   X
4A                  DECA              *ALL NIBBLES TRANSFERRED?
27 0C 0073          BEQ    ØBSCPQ     *YES
FE 0000  N          LDX    BCDDGL     *TRANSFER NEXT NIBBLE
09                  DEX
FF 0000  N          STX    BCDDGL
FE 0003  N          LDX    BSCQDL
20 D1 0044          BRA    MBCDDG
FE 0000  A ØBSCPQ   LDX    BSCQL      *IS DATA NEGATIVE?
A6 00    A          LDAA   X
84 F0    A          ANDA   #$F0
27 25 00A1          BEQ    BSCBDF     *PØSITIVE
86 01    A          LDAA   #1         *NEGATIVE, SET SIGN
B7 0000  A          STAA   SIGN
E6 05    A          LDAB   5,X
CE 0004  A          LDX    #BCDNUM+4
7F 0002  N          CLR    BØRRØW
86 0A    A SUBBD    LDAA   #10        *CØNVERT TØ + BCD
A0 00    A          SUBA   X
B0 0002  N          SUBA   BØRRØW
81 0A    A          CMPA   #10        *PREVIØUS DIGITS 0?
27 07 009B          BEQ    SUBBDQ     *YES
A7 00    A          STAA   X          *STØRE IF NØT 10
86 01    A          LDAA   #1         *INDICATE PREVIØUS DIGIT
B7 0002  N          STAA   BØRRØW     **NØT 0
5A          SUBBDQ  DECB              *SUBTRACTIØN FINISHED?
27 03 00A1          BEQ    BSCBDF     *YES
09                  DEX               *NØ, SUBTRACT NEXT DIGIT
20 E8 0089          BRA    SUBBD
C6 01    A BSCBDF   LDAB   #1         *INDICATE CØNVERSIØN
39                  RTS               **CØRRECT
5F          BSCERR  CLRB              *INDICATE BASIC DATA
39                  RTS               **ERRØR
                    END
```

ST00UT

```
                    NAM     ST00UT
                    0PT     REL,PAGE=49
              *SCRATCH PAD FOR LEVEL 1 SUBROUTINES
              SCRAT2 COMM   DSCT
   0001   A PORTTP RMB      1          PORT DATA FORMAT
   0001   A MSBCDD RMB      1          MOST SIGNIICANT BCD DIGIT
   0004   A MAXBCD RMB      4          MAXIMUM BCD NUMBER
   0003   A MAXHEX RMB      3          MAXIMUM HEXADECIMAL NO.
                    XREF    0PORTL,0DATAL
                    XREF    HEXNUM,HBYTES,LBYTES
                    XREF    BCDNUM,SIGN,BLDADR,BCDHEX,CPBYTS
                    XREF    CLINES,NLINES
                    XDEF    ST00UT
              *SUBROUTINE TO STORE DATA TO BE OUTPUTED
              *INTO LOCATIONS POINTED TO BY 0DATAL, DATA
              *IS BCD NUMBER STORED IN SIGN, BCDNUM
              *LOCATION OF PORT IN THE PORTST TABLE HAS
              *TO BE GIVEN IN 0PORTL
                    PSCT
FE 0000   A ST00UT LDX      0PORTL
A6 06     A        LDAA     6,X
44                 LSRA
B7 0000   N        STAA     PORTTP     *STORE TYPE OF PORT
81 01     A        CMPA     #1
26 03 0010         BNE      0PBCD      *BCD OR SIGNED BCD PORT
7E 00D2   P        JMP      STOHEX     *HEX FORMAT PORT
BD 0000   A 0PBCD  JSR      CLINES     *COUNT NO. OF DATA
7F 0001   N        CLR      MSBCDD     **IN PORT, FIND MOST
74 0000   A        LSR      NLINES     **SIG. DIGIT OF MAXIMUM
49                 ROLA                **NO. THAT CAN BE OUTPUTED
74 0000   A        LSR      NLINES
79 0001   N        ROL      MSBCDD
47                 ASRA
79 0001   N        ROL      MSBCDD
CE 0002   N        LDX      #MAXBCD
86 04     A        LDAA     #4
6F 00     A CLRMBD CLR      X          *CLEAR MAXBCD
4A                 DECA
27 03 0031         BEQ      WBCDDQ
08                 INX
20 F8 0029         BRA      CLRMBD
CE 0005   N WBCDDQ LDX      #MAXBCD+3 *FIND MAX BCD NUMBER
7D 0000   A        TST      NLINES     *ANY DIGITS OF 9?
27 0C 0045         BEQ      TSTMSB
86 09     A        LDAA     #9
A7 00     A FMAXBD STAA     X          *STORE A DIGIT OF 9
09                 DEX                 **INTO MAXBCD
7A 0000   A        DEC      NLINES
27 02 0045         BEQ      TSTMSB     *NO MORE DIGIT OF 9
```

STØØUT

```
20 F6 003B          BRA    FMAXBD    *MØRE DIGITS ØF 9
B6 0001  N TSTMSB LDAA    MSBCDD
27 0B 0055          BEQ    CPBCDA    *MØST SIG DIGID WAS 9
5F                  CLRB
0D           FMSBCD SEC              *FIND MØST SIG BCD DIGIT
59                  RØLB
4A                  DECA
27 03 0053          BEQ    SMSBCD
08                  INX
20 F8 004B          BRA    FMSBCD
E7 00     A SMSBCD STAB    X         *STØRE MØST SIG. DIGIT
B6 0000   A CPBCDA LDAA    BCDNUM    *CØMPARE MAX BCD WITH
27 03 005D          BEQ    CMPBCD    **DATA TØ BE ØUTPUTED
7E 0117   P         JMP    STØERR    *DATA HAS 5 DIG.,WRØNG
CE 0002   N CMPBCD LDX     #MAXBCD   *CØMPARE DATA & MAXBCD
FF 0000   A         STX    HBYTES
CE 0001   A         LDX    #BCDNUM+1
FF 0000   A         STX    LBYTES
C6 04     A         LDAB   #4
BD 0000   A         JSR    CPBYTS    *CØMPARE
26 03 0073          BNE    STØBCD    *DATA<ØR=MAXBCD
7E 0117 P          JMP    STØERR   *DATA GREATER
B6 0001   A STØBCD LDAA    BCDNUM+1 *STØRE DATA IN BCDNUM INTØ
48                  ASLA             **LØCATIØNS PØINTED TØ
48                  ASLA             **BY ØDATAL
48                  ASLA
48                  ASLA
BA 0002   A         ØRAA   BCDNUM+2 *MERGE 2 DIGITS INTØ
FE 0000   A         LDX    ØDATAL    **1 BYTE
F6 0000   N         LDAB   PØRTTP
C1 03     A         CMPB   #3        *SIGNED BCD PØRT?
26 03 008A          BNE    STBCDH    *ØTHER DATA FØRMAT
6F 00     A         CLR    X         *CLEAR 1ST BYTE ØF 3 BYTES
08                  INX
A7 00     A STBCDH STAA    X         *STØRE FIRST 2 DIGITS
B6 0003   A         LDAA   BCDNUM+3 **(CAN BE 0)
48                  ASLA
48                  ASLA
48                  ASLA
48                  ASLA
BA 0004   A         ØRAA   BCDNUM+4
A7 01     A         STAA   1,X       *STØRE LAST 2 DIGITS
B6 0000   N         LDAA   PØRTTP
81 03     A         CMPA   #3        *SIGNED PØRT?
27 03 00A2          BEQ    ØBSCPQ    *DATA NEGATIVE?
7E 0114   P         JMP    STØUTF    *FINISHED
B6 0000   A ØBSCPQ LDAA    SIGN      *IS DATA NEGATIVE?
26 02 00A9          BNE    ØBSCNG    *YES
20 6B 0114          BRA    STØUTF    *PØSITIVE, SIGN BIT IS 0
```

ST00UT

```
FE 0000  A ØBSCNG LDX    ØPØRTL
BD 0000  A        JSR    CLINES
C6 02    A        LDAB   #2
B6 0000  A        LDAA   NLINES
4C                INCA              *INCREMENT 1 FØR SIGN LINE
81 08    A SGNBYØ CMPA   #8         *FIND LØCATIØN ØF BYTE
2F 05 00BE        BLE    GETSGØ     **WHICH STØRES THE SIGN
80 08    A        SUBA   #8
5A                DECB
20 F7 00B5        BRA    SGNBYØ
FE 0000  A GETSGØ LDX    ØDATAL
BD 0000  A        JSR    BLDADR
C6 01    A        LDAB   #1         *SHIFT SIGN BIT INTØ B
4A         SFSGØQ DECA              *NEED TØ SHIFT SIGN BIT?
27 03 00CC        BEQ    STØSGN     *NØ
58                ASLB              *YES
20 FA 00C6        BRA    SFSGØQ
EA 00    A STØSGN ØRAB   X          *MERGE SIGN WITH DATA
E7 00    A        STAB   X
20 42 0114        BRA    STØUTF     *FINISHED
BD 0000  A STØHEX JSR    CLINES     *PØRT IS HEX FØRMAT
BD 0000  A        JSR    BCDHEX     *CØNVERT BCD TØ HEX
B6 0000  A        LDAA   NLINES
7F 0006  N        CLR    MAXHEX
7F 0007  N        CLR    MAXHEX+1
0D         FMAXHX SEC               *FIND MAXIMUM HEX NØ.
79 0007  N        RØL    MAXHEX+1   *THAT CAN BE ØUTPUTTED
79 0006  N        RØL    MAXHEX
4A                DECA
27 02 00ED        BEQ    CPMAXH     *MAX HEX NØ. FØUND
20 F4 00E1        BRA    FMAXHX
B6 0006  N CPMAXH LDAA   MAXHEX     *CØMPARE HEX NØ.'S
B1 0000  A        CMPA   HEXNUM     *CØMPARE 1ST BYTE
22 12 0107        BHI    STHEX
27 03 00FA        BEQ    CPLHEX
7E 0117  P        JMP    STØERR     *DATA > MAX. HEX NØ.
B6 0007  N CPLHEX LDAA   MAXHEX+1   *CØMPARE 2ND BYTES
B1 0001  A        CMPA   HEXNUM+1
22 05 0107        BHI    STHEXA
27 03 0107        BEQ    STHEXA
7E 0117  P        JMP    STØERR
B6 0000  A STHEXA LDAA   HEXNUM     *STØRE HEX. DATA
FE 0000  A        LDX    ØDATAL
A7 00    A        STAA   X
B6 0001  A        LDAA   HEXNUM#1
A7 01    A        STAA   1,X
86 01    A STØUTF LDAA   #1
39                RTS
4F         STØERR CLRA              *INDICATE ERRØR
39                RTS
                  END
 00000
```

IDABSC

```
                    NAM     IDABSC
                    ØPT     REL,PAGE=49
              *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
              SCRAT2 CØMM   DSCT
   0001   A NIBBLE RMB      1          LEFT ØR RIGHT NIBBLE
   0001   A SGNMSK RMB      1          USED TØ STRIP SIGN
   0002   A BCDDGL RMB      2          BCD DIGIT LØCATIØN
   0002   A IBSCDL RMB      2          BASIC BYTE LØCATIØN
   0001   A DIGCTR RMB      1          DIGIT CØUNTER
   0001   A PØRTTP RMB      1          PØRT DATA FØRMAT
   0001   A BØRRØW RMB      1
                    XREF    IPØRTL,IBSCQL,INDATL
                    XREF    HEXNUM,SIGN,BCDNUM
                    XREF    BSCQL,HEXBSC,NLINES,CLINES,BLDADR
                    XDEF    IDABSC
              *SUBRØUTINE TØ STØRE DATA INTØ BASIC VARIABLE
              *INDATL PØINTS TØ LØCATIØN ØF DATA, IBSCQL
              *PØINTS TØ LØCATIØN ØF VARIABLE AND IPØRTL
              *PØINTS TØ LØCATIØN ØF PØRT IN PØRTST TABLE
                    PSCT
86 06        A IDABSC LDAA   #6
FE 0000      A        LDX    IBSCQL
6F 00        A CLIBSC CLR    X          *CLEAR BASIC VARIABLE
4A                    DECA
27 03 000D            BEQ    IPØTTP
08                    INX
20 F8 0005            BRA    CLIBSC
FE 0000      A IPØTTP LDX    IPØRTL     *STØRE DATA FØRMAT
A6 06        A        LDAA   6,X
44                    LSRA
B7 0007      N        STAA   PØRTTP
81 01        A        CMPA   #1
27 06 0020            BEQ    IDAHEX     *INPUT PØRT IS HEX FØRMAT
81 02        A        CMPA   #2
27 5C 007A            BEQ    IBCDPS     *INPUT PØRT UNSIGNED BCD
20 17 0037            BRA    IDASBD     *INPUT PØRT IS SIGNED BCD
FE 0000      A IDAHEX LDX    INDATL     *STØRE HEX DATA INTØ
A6 00        A        LDAA   X          **BASIC VARIABLE
B7 0000      A        STAA   HEXNUM
A6 01        A        LDAA   1,X
B7 0001      A        STAA   HEXNUM+1
FE 0000      A        LDX    IBSCQL
FF 0000      A        STX    BSCQL
BD 0000      A        JSR    HEXBSC
39                    RTS
7F 0000      A IDASBD CLR    SIGN       *STØRE SIGNED BCD NØ.
FE 0000      A        LDX    IPØRTL     **INTØ BASIC VARIABLE
BD 0000      A        JSR    CLINES
C6 02        A        LDAB   #2
```

IDABSC

```
B6 0000  A         LDAA   NLINES
4C                 INCA
81 08    A SGNBYT  CMPA   #8        *GET BIT PØSITIØN ØF SIGN
2F 05 004F         BLE    GETSGN
80 08    A         SUBA   #8
5A                 DECB
20 F7 0046         BRA    SGNBYT
FE 0000  A GETSGN  LDX    INDATL
BD 0000  A         JSR    BLDADR
C6 FE    A         LDAB   #$FE
F7 0001  N         STAB   SGNMSK
E6 00    A         LDAB   X         *GET DATA BYTE WITH
20 04 0062         BRA    SFSGNI    **SIGN BIT
0D         SHSGNI  SEC
79 0001  N         RØL    SGNMSK    *SHIFT A 0 TØ THE
54         SFSGNI  LSRB             **PØSITIØN ØF THE SIGN
4A                 DECA
26 F8 005E         BNE    SHSGNI
24 0C 0074         BCC    LDIDL2    *SIGN IS 0 ØR PØSITIVE
86 01    A         LDAA   #1
B7 0000  A         STAA   SIGN      *INDICATE NEGATIVE
B6 0001  N         LDAA   SGNMSK
A4 00    A         ANDA   X         *STRIP SIGN FRØM DATA
A7 00    A         STAA   X
FE 0000  A LDIDL2  LDX    INDATL
08                 INX              *3 BYTE WØRD
20 06 0080         BRA    GFINBY
7F 0000  A IBCDPS  CLR    SIGN
FE 0000  A         LDX    INDATL
A6 00    A GFINBY  LDAA   X
16                 TAB
84 F0    A         ANDA   #$F0
44                 LSRA
44                 LSRA
44                 LSRA
44                 LSRA
B7 0001  A         STAA   BCDNUM+1 *STØRE DATA INTØ BCDNUM
C4 0F    A         ANDB   #$F
F7 0002  A         STAB   BCDNUM+2
A6 01    A         LDAA   1,X
16                 TAB
84 F0    A         ANDA   #$F0      *GET LEFT NIBBLE
44                 LSRA
44                 LSRA
44                 LSRA
44                 LSRA
B7 0003  A         STAA   BCDNUM+3
C4 0F    A         ANDB   #$F
F7 0004  A         STAB   BCDNUM+4
```

IDABSC

```
CE 0001  A          LDX    #BCDNUM+1
C6 04    A          LDAB   #4
A6 00    A CBCDDG   LDAA   X          *IS DATA 0?
26 09 00B4          BNE    IBCDNQ
08                  INX
5A                  DECB
27 02 00B1          BEQ    IBCDEO     *YES
20 F6 00A7          BRA    CBCDDG
C6 01    A IBCDEO   LDAB   #1         *INDICATE DATA CØRRECT
39                  RTS
FE 0000  A IBCDNQ   LDX    IBSCQL     *BCD DATA IS NEGATIVE?
E7 05    A          STAB   5,X
B6 0000  A          LDAA   SIGN
27 22 00E0          BEQ    STØBSC     *PØSITIVE
86 90    A          LDAA   #$90       *NEGATIVE
A7 00    A          STAA   X          *STØRE '9' INTØ LEFT
CE 0004  A          LDX    #BCDNUM+4 **NIBBLE ØF 1ST BYTE
7F 0008  N          CLR    BØRRØW
86 0A    A SUBBCD   LDAA   #10        *CØNVERT TØ NEGATIVE
A0 00    A          SUBA   X          **FØRMAT ØF BASIC
B0 0008  N          SUBA   BØRRØW
A7 00    A          STAA   X
81 0A    A          CMPA   #10
27 05 00DA          BEQ    SUBCDQ     *PREVIØUS DIGITS ARE 0
86 01    A          LDAA   #1
B7 0008  N          STAA   BØRRØW
5A          SUBCDQ  DECB               *SUBTRACTIØNS FINISHED?
27 03 00E0          BEQ    STØBSC     *NØ, 4 DIGITS SUBTRACTED
09                  DEX                *SUBTRACT NEXT MØRE
20 E8 00C8          BRA    SUBBCD     **SIGNIFICANT DIGIT
7F 0000  N STØBSC   CLR    NIBBLE     *STØRE INTØ VARIABLE
FE 0000  A          LDX    IBSCQL
E6 05    A          LDAB   5,X        *NØ. ØF DIGITS IS
F7 0006  N          STAB   DIGCTR     **SAME AS EXPØNENT
FF 0004  N          STX    IBSCDL
C6 05    A          LDAB   #5
F0 0006  N          SUBB   DIGCTR
CE 0000  A          LDX    #BCDNUM
BD 0000  A          JSR    BLDADR     *GET MØST SIG. DIGIT
FF 0002  N          STX    BCDDGL     BYTE IN BCDNUM TØ BE LØADE
A6 00    A SNXBCD   LDAA   X          *LØAD DIGIT
FE 0004  N          LDX    IBSCDL
F6 0000  N          LDAB   NIBBLE
27 0B 0111          BEQ    STRGHT     *STØRE IN RIGHT NIBBLE
48                  ASLA
48                  ASLA
48                  ASLA
48                  ASLA
A7 00    A          STAA   X          *STØRE IN LEFT NIBBLE
```

IDABSC

```
7F 0000  N          CLR    NIBBLE   *INDICATE RIGHT NIBBLE
20 0B 011C          BRA    SNBCDQ   *ANY MØRE DIGITS?
AA 00    A  STRGHT  ØRAA   X        *MERGE WITH LEFT NIBBLE
A7 00    A          STAA   X
7C 0000  N          INC    NIBBLE
08                  INX
FF 0004  N          STX    IBSCDL
7A 0006  N  SNBCDQ  DEC    DIGCTR
27 09 012A          BEQ    IDBSCF   *NØ MØRE DIGITS
FE 0002  N          LDX    BCDDGL   *MØRE DIGITS
08                  INX
FF 0002  N          STX    BCDDGL
20 D2 00FC          BRA    SNXBCD
39          IDBSCF  RTS             *FINISHED
                    END
```

BSCVTQ

```
                    NAM    BSCVTQ
                    ØPT    REL
            *SUBRØUTINE TØ CHECK IF MEMØRY PØINTED TØ BY X
            *REG. IS A BASIC VECTØR WITH NAME Q0, Q1,...
            *ØR Q9. X AND A REG. ARE NØT CHANGED. IF IT
            *IS REG. B IS 1, ØTHERWISE IT IS 0.
                    XDEF   BSCVTQ
                    PSCT
C6 51    A  BSCVTQ LDAB   #'Q      *VECTØR START WITH Q?
E1 00    A         CMPB   X
26 13 0019         BNE    NBSCVT   *NØT Q VECTØR
C6 F0    A         LDAB   #$F0
E4 01    A         ANDB   1,X
C1 B0    A         CMPB   #$B0      *A VECTØR WITH NØ.?
26 0B 0019         BNE    NBSCVT   *NØ
C6 0F    A         LDAB   #$F       *DIGIT <9?
E4 01    A         ANDB   1,X
C1 09    A         CMPB   #9
2E 03 0019         BGT    NBSCVT   *NØ
C6 01    A         LDAB   #1        *YES, A Q VECTØR WITH
39                 RTS              **DIGITS
5F          NBSCVT CLRB
39                 RTS
                   END
```

ØUTCMD

```
                    NAM     ØUTCMD
                    ØPT     REL,PAGE=49
                *SUBRØUTINE TØ ØUTPUT TØ CØMMAND PØRT
                *ØUTBCD ØUTPUTS BCD NØ. IN ACC. A
                *ØUTBDB ØUTPUTS BCD AND A BLANK
                *ØUTBDC ØUTPUTS BCD, LINE FEED AND A C/R
                *ØUTSTR ØUTPUTS CØNTENTS ØF MEMØRY PØINTED TØ BY
                *X. A, B AND X REG. ARE CHANGED
                    XREF    CMPØRT,BSCHSA
                    XREF    MULPLY,BLDADR
                    XDEF    ØUTBCD,ØUTBDB,ØUTBDC,ØUTSTR
                *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
                SCRAT2 CØMM DSCT
   0001   A ØUTFLG RMB     1          ENTRY PØINT FLAG
   0002   A CMDØHA RMB     2          CØMMAND ØUTPUT HDR. ADR.
   0001   A CHARAC RMB     1          RIGHT NIBBLE ØF A
   0002   A STRADR RMB     2          MESSAGE STRING ADRESS
                    PSCT
   0A     A LFCRTN FCB     $A         LINE FEED
   0D     A        FCB     $D         CARRIAGE RETURN
   00     A        FCB     0,0,0,0,0,0,0,4 NULLS, END DELIMITE
7F 0000   N ØUTBCD CLR     ØUTFLG     *SET FLAG TØ 0
20 0C 001B         BRA     SAVCHR
C6 01     A ØUTBDB LDAB    #1         *ØUTPUT BCD AND BLANK
F7 0000   N        STAB    ØUTFLG
20 05 001B         BRA     SAVCHR
C6 02     A ØUTBDC LDAB    #2         *ØUTPUT BCD AND C/R
F7 0000   N        STAB    ØUTFLG
B7 0003   N SAVCHR STAA    CHARAC     *SAVE CHARACTER
20 08 0028         BRA     CMDHAD     *GET CØMMAND HANDLR ADR.
86 03     A ØUTSTR LDAA    #3         *ØUTPUT MEMØRY PØINTED
B7 0000   N        STAA    ØUTFLG     **TØ BY X
FF 0004   N        STX     STRADR     *SAVE MEMØRY ADDRESS
B6 0000   A CMDHAD LDAA    CMPØRT     *GET CØMMAND PØRT NØ.
C6 09    A        LDAB    #9         *GET ØFFSET
BD 0000   A        JSR     MULPLY
CE 0000   A        LDX     #BSCHSA    *GET BASIC HANDLERS ADR.
BD 0000   A        JSR     BLDADR     *ADD ØFFSET TØ BEGINNING
FF 0001   N        STX     CMDØHA     *SAVE ØUTPUT HANDLER ADR.
B6 0000   N        LDAA    ØUTFLG     *ØUTPUT WHAT?
81 02     A        CMPA    #2
2E 32 0072         BGT     ØUTSTG     *ØUTPUT MEMØRY
B6 0003   N        LDAA    CHARAC     *ØUTPUT BCD DIGITS
44                 LSRA               *ØUTPUT LEFT BCD DIGIT
44                 LSRA
44                 LSRA
44                 LSRA
8B 30     A        ADDA    #$30
FE 0001   N        LDX     CMDØHA
```

ØUTCMD

```
AD 00     A          JSR    X          *ØUTPUT DIGIT
B6 0003   N          LDAA   CHARAC     *GET RIGHT BCD DIGIT
84 0F     A          ANDA   #$F
8B 30     A          ADDA   #$30
FE 0001   N          LDX    CMDØHA
AD 00     A          JSR    X          *ØUTPUT RIGHT BCD DIGIT
B6 0000   N          LDAA   ØUTFLG
26 01 0060           BNE    ØBLANQ     *ØUTPUT BLANK?
39                   RTS               *JUST ØUTPUT BCD DIGITS
81 02     A ØBLANQ   CMPA   #2
27 08 006C           BEQ    ØLFCR      *ØUTPUT LINE FEED & C/R
86 20     A          LDAA   #$20       *ØUTPUT BCD AND BLANK
FE 0001   N          LDX    CMDØHA
AD 00     A          JSR    X
39                   RTS
CE 0000   P ØLFCR    LDX    #LFCRTN    *ØUTPUT LINE FEED & C/R
FF 0004   N          STX    STRADR
FE 0004   N ØUTSTG   LDX    STRADR     *ØUTPUT STRING
A6 00     A GETCHR   LDAA   X          *GET MEMØRY CØNTENTS
81 04     A          CMPA   #4         *END ØF STRING?
26 01 007C           BNE    ØUTCH1     *NØ
39                   RTS               *YES
FE 0001   N ØUTCH1   LDX    CMDØHA     *ØUTPUT A MEMØRY LØCATIØN
AD 00     A          JSR    X
FE 0004   N          LDX    STRADR     *GØ TØ NEXT LØCATIØN IN
08                   INX               **MEMØRY
FF 0004   N          STX    STRADR
20 EB 0075           BRA    GETCHR
                     END
```

BREAK

```
                     NAM    BREAK
                     OPT    REL
                     XREF   BCMMOD
                     XDEF   BREAK
            *SUBRØUTINE TØ SEE IF BREAK KEYS HAVE
            *BEEN DEPRESSED
37          BREAK    PSHB              *SAVE A AND B
36                   PSHA
B6 FCF4   A          LDAA   $FCF4      *READ ACIA STATUS
46                   RØRA
24 12 001A           BCC    RET        *NØ INPUT
DF 8A     A          STX    $8A        *SAVE X
CE 8004   A          LDX    #$8004
BD 0352   A          JSR    $352       *INPUT CHARACTER
DE 8A     A          LDX    $8A        *RESTØRE X
B1 0156   A          CMPA   $156       *BREAK CHARACTER?
26 03 001A           BNE    RET        *NØT BREAK CHARACTER
7E 0BAB   A          JMP    8CMMOD     *GØ TØ BASIC CØMMAND
32          RET      PULA              *RESTØRE A & B
33                   PULB
39                   RTS
                     END
```

SRPØRT

```
                    NAM     SRPØRT
                    ØPT     REL,PAGE=49
              *SCRATCH PAD FØR LEVEL 1 SUBRØUTINES
              SCRAT2 CØMM   DSCT
   0002   A NAMEL   RMB     2
   0002   A NAMECL  RMB     2
   0002   A PØRTNL  RMB     2
   0001   A PØRTCN  RMB     1
   0002   A PØRTTL  RMB     2
   0002   A CHARCN  RMB     2
                    XREF    NPØRTS,PØRTST,BLDADR
                    XDEF    SRPØRT
              *SUBRØUTINE TØ SEARCH FØR PØRT ENTRY IN PØRTST
              *LØCATIØN ØF NAME ØF PØRT SPECIFIED IN REG. X
              *IF PØRT IS IN PØRTST, X WILL CØNTAIN THE LØCATIØ
              *ØF PØRT IN PØRTST AND B WILL CØNTAIN THE ENTRY
              *NØ. ØF THE PØRT IN PØRTST. IF PØRT NAME DØES NØT
              *EXIST, X IS NØT CHANGED AND B IS EQUAL TØ 0.
                    PSCT
FF 0000   N SRPØRT STX     NAMEL     *SAVE X
B6 0000   A         LDAA   NPØRTS    *ANY ENTRIES IN PØRTST?
27 47 004F          BEQ    PØRTNM
B7 0006   N         STAA   PØRTCN    *YES
CE 0000   A         LDX    #PØRTST   *START WITH FIRST ENTRY
FF 0004   N         STX    PØRTNL
86 06     A CMPNAM  LDAA   #6        *CØMPARE 6 CHARACTERS
B7 0009   N         STAA   CHARCN    **ØF EACH NAME
FE 0004   N         LDX    PØRTNL
FF 0007   N         STX    PØRTTL
FE 0000   N         LDX    NAMEL
FF 0002   N         STX    NAMECL
A6 00     A CMPCHR  LDAA   X
FE 0007   N         LDX    PØRTTL
A1 00     A         CMPA   X         *CØMPARE 1 CHR.
26 12 003D          BNE    NPØRTQ    *NØT EQUAL
7A 0009   N         DEC    CHARCN    *ALL 6 CHR. MATCH?
27 21 0051          BEQ    NAMEMA    *YES
08                  INX              *CØMPARE NEXT CHR.
FF 0007   N         STX    PØRTTL
FE 0002   N         LDX    NAMECL
08                  INX
FF 0002   N         STX    NAMECL
20 E5 0022          BRA    CMPCHR
7A 0006   N NPØRTQ  DEC    PØRTCN    *ANY MØRE ENTRIES?
27 0D 004F          BEQ    PØRTNM    *NØ
C6 0C     A         LDAB   #12       *CØMPARE WITH NEXT ENTRY
FE 0004   N         LDX    PØRTNL
BD 0000   A         JSR    BLDADR
FF 0004   N         STX    PØRTNL
```

SRPØRT

```
20 C2 0011          BRA     CMPNAM
5F            PØRTNM CLRB             *INDICATE PØRT DØES NØT
39                  RTS               **EXIST IN PØRTST
A6 01       A NAMEMA LDAA   1,X
F6 0000     A       LDAB    NPØRTS    *GET ENTRY NØ. ØF PØRT
F0 0006     N       SUBB    PØRTCN
FE 0004     N       LDX     PØRTNL    *GET LØCATIØN ØF PØRT
5C                  INCB
39                  RTS
                    END
```

ADJHMS

```
                    NAM     ADJHMS
                    ØPT     REL
                    XDEF    ADJHMS
            *SUBRØUTINE TØ CØNVERT 1 SEC TØ 1ØØ MS
            *X PØINTS TØ THE LØCATIØN WHERE TIME
            *IS STØRED. IF TIME IS 0, B IS SET TØ 0.
            *IF TIME IS GREATER THAN ØR EQUAL TØ 1 SEC
            *B IS SET TØ 1.
            *SUBRØUTINE TØ ADJUST TIME
A6 02     A ADJHMS LDAA   2,X
27 04 0008         BEQ     ADJMIN
6A 02       A      DEC     2,X
20 16 001E         BRA     S100MS
6 01      A ADJMIN LDAA   1,X
27 04 0010         BEQ     ADJHR
6A 01       A      DEC     1,X
20 0A 001A         BRA     ST059S
A6 00       A ADJHR LDAA   X
27 13 0027         BEQ     ZERØTM
6A 00       A      DEC     X
86 3B       A      LDAA    #59
A7 01       A      STAA    1,X
86 3B       A ST059S LDAA  #59
A7 02       A      STAA    2,X
86 64       A S100MS LDAA  #100
AB 03       A      ADDA    3,X
A7 03       A      STAA    3,X
C6 01       A      LDAB    #1
39                 RTS
5F            ZERØTM CLRB
39                 RTS
                   END
```

RPGPIC

```
                    NAM     RPGPIC
                    ØPT     REL
                    XREF    PICADT
                    XDEF    RPGPIC,RPCPIC
              *SUBRØUTINE TØ PRØGRAM SØME LINES ØF
              *PIC NØ. 1 AS INPUTS. PIC
              *NØ. 1 IS ALSØ CØNNECTED TØ THE CLØCK.
                    PSCT
C6 03       A RPCPIC LDAB    #3
FE 0000     A        LDX     PICADT
E7 01       A        STAB    1,X
A4 00       A        ANDA    X
A7 00       A        STAA    X
C6 07       A        LDAB    #7
E7 01       A        STAB    1,X
39                   RTS
             *SUBRØUTINE TØ PRØGRAM SØME LINES ØF
              *A PIC(EXCEPT NØ. 1) AS INPUTS.
              *X CØNTAINS THE LØCATIØN WHERE THE
              *ADDRESS ØF THE PIC IS STØRED. THE BITS IN A
              *WHICH ARE SET TØ 1 REPRESENT THE LINES TØ
              *BE PRØGRAM AS INPUTS
EE 00       A RPGPIC LDX     X
6F 01       A        CLR     1,X
A4 00       A        ANDA    X
A7 00       A        STAA    X
86 04       A        LDAA    #4
A7 01       A        STAA    1,X
39                   RTS
                     END
```

LEVEL 2 SUBROUTINES

BCDHEX

```
                        NAM     BCDHEX
                        ØPT     REL,PAGE=49
                *SCRATCH PAD FØR LEVEL 2 SUBRØUTINES
                SCRAT3 CØMM     DSCT
   0002   A PWR10L RMB     2
   0002   A BCDDGL RMB     2          BCD NØ. DIGIT LØC.
   0001   A BCDDGC RMB     1          BCD DIGIT CØUNTER
                        XREF    HEXNUM,BCDNUM,PWRS10
                        XDEF    BCDHEX
                *SUBRØUTINE TØ CØNVERT THE BCD NØ. STØRED IN
                *SIGN, BCDNUM INTØ A 16 BIT HEX NØ. AND
                *STØRE IT IN HEXNUM. IF THERE IS NØ
                *ØVERFLØW, B IS SET TØ 1. IF ØVERFLØW, B IS 0
                *ACC. A AND X ARE CHANGED
                        PSCT
7F 0000   A BCDHEX CLR  HEXNUM    *CLEAR MØST SIG. BYTE ØF
B6 0004   A        LDAA BCDNUM+4 **HEXNUM AND STØRE LEAST
B7 0001   A        STAA HEXNUM+1 **SIG. DIGIT IN HEXNUM+1
CE 0003   A        LDX  #BCDNUM+3 *START WITH PØWER ØF 10
FF 0002   N        STX  BCDDGL
86 04     A        LDAA #4
B7 0004   N        STAA BCDDGC
CE 0000   A        LDX  #PWRS10
FF 0000   N        STX  PWR10L
FE 0002   N NBCDDG LDX  BCDDGL    *CØNVERT A DIGIT TØ HEX
E6 00     A        LDAB X
27 18 0039 BCDDFQ  BEQ  NBCDDQ    *0, NØ NEED TØ CØNVERT
FE 0000   N        LDX  PWR10L    *ADD PØWER ØF 10 UNTIL
A6 01     A        LDAA 1,X       **BCD DIGIT GØES TØ 0
BB 0001   A        ADDA HEXNUM+1
B7 0001   A        STAA HEXNUM+1
A6 00    A        LDAA   X
B9 0000   A        ADCA HEXNUM
25 1F 0052         BCS  ØVERFL    *HEX EQUIVALENT >$FFFF
B7 0000   A        STAA HEXNUM
5A                 DECB
20 E6 001F         BRA  BCDDFQ
7A 0004   N NBCDDQ DEC  BCDDGC    *CØNTINUE WITH NEXT
27 11 004F         BEQ  BCDHXF    **BCD DIGIT?
FE 0000   N        LDX  PWR10L    *YES
08                 INX
08                 INX
FF 0000   N        STX  PWR10L
FE 0002   N        LDX  BCDDGL
09                 DEX
FF 0002   N        STX  BCDDGL
20 CB 001A         BRA  NBCDDG
D6 01     A BCDHXF LDAB 1          *INDICATE NØ ØVERFLØW
39                 RTS
5F          ØVERFL CLRB            *INDICATE ØVERFLØW
39                 RTS
                   END
```

HEXBSC

```
                    NAM     HEXBSC
                    ØPT     REL
              *SCRATCH PAD FØR LEVEL 2 SUBRØUTINES
              SCRAT3 CØMM   DSCT
   0002   A PWR10L RMB      2
   0001   A PØWER  RMB      1
   0001   A PWRCTR RMB      1
   0002   A DIVSØR RMB      2
   0001   A QUØTNT RMB      1
   0001   A BØRRØW RMB      1
   0001   A NØNZRØ RMB      1
   0001   A NIBBLE RMB      1
   0002   A BSCQDL RMB      2
                    XREF    PWRS10,HEXNUM,BSCQL
                    XDEF    HEXBSC
              *SUBRØUTINE TØ CØNVERT FRØM HX TØ BCD
              *AND STØRE INTØ BASIC VARIABLE WHØSE LØCATIØN
              *IS IS BSCQL
                    PSCT
FE 0000   A HEXBSC LDX      BSCQL     *CLEAR BASIC VARIABLE
C6 06     A        LDAB     #6
6F 00     A CLRVAR CLR      X
5A                 DECB
27 03 000D         BEQ      HEXBC0
08                 INX
20 F8 0005         BRA      CLRVAR
B6 0000   A HEXBC0 LDAA     HEXNUM    *IS HEX NØ. 0?
26 06 0018         BNE      INIPWR
B6 0001   A        LDAA     HEXNUM+1
26 01 0018         BNE      INIPWR
39                 RTS                *HEX NØ. IS 0
86 05     A INIPWR LDAA     #5
B7 0002   N        STAA     PØWER
7F 0008   N        CLR      NØNZRØ
7F 0009   N        CLR      NIBBLE
CE 0006   A        LDX      #PWRC10+6 *START WITH 10000
FF 0000   N        STX      PWR10L
86 04     A        LDAA     #4
B7 0003   N        STAA     PWRCTR
FE 0000   A        LDX      BSCQL
FF 000A   N        STX      BSCQDL
FE 0000   N DIVPWR LDX      PWR10L    *DIVIDE BY A PØWER
EE 00     A        LDX      X         **ØF 10
FF 0004   N        STX      DIVSØR
7F 0006   N        CLR      QUØTNT
7F 0007   N        CLR      BØRRØW
B6 0000   A CMPPWR LDAA     HEXNUM    *CØMPARE 1ST 2 BYTES
B1 0004   N        CMPA     DIVSØR
22 10 005A         BHI      DIVIDE
27 02 004E         BEQ      CMP2BY    *IF =, CØMPARE 2ND BYTE
23 35 0083         BLS      DPWRF
```

HEXBSC

```
B6 0001  A CMP2BY LDAA   HEXNUM#1 *CØMPARE NEXT 2 BYTES
B1 0005  N        CMPA   DIVSØR+1
22 13 0069        BHI    SUBLSB
27 11 0069        BEQ    SUBLSB
20 29 0083        BRA    DPWRF
B6 0001  A DIVIDE LDAA   HEXNUM+1 *HEX NØ. IS GREATER
B1 0005  N        CMPA   DIVSØR+1 **THAN THE PØWER ØF 10
27 07 0069        BEQ    SUBLSB
22 05 0069        BHI    SUBLSB
C6 01    A        LDAB   #1
F7 0007  N        STAB   BØRRØW
B0 0005  N CUBLSB SUBA   DIVSØR+1 *SUBTRACT LESS SIGNFICANT
B7 0001  A        STAA   HEXNUM+1 **BYTE FIRST
B6 0000  A        LDAA   HEXNUM   *SUBTRACT MØST SIGNIFICANT
B0 0007  N        SUBA   BØRRØW   **BYTE
7F 0007  N        CLR    BØRRØW
B0 0004  N        SUBA   DIVSØR
B7 0000  A        STAA   HEXNUM
7C 0006  N        INC    QUØTNT
20 BF 0042        BRA    CMPPWR
B6 0006  N DPWRF  LDAA   QUØTNT   *DIVISIØN BY PØWER FIN.
27 0C 0094        BEQ    NØZRØQ   *DIGIT IS 0
F6 0008  N        LDAB   NØNZRØ   *PREVIØUS DIGITS 0?
26 11 009E        BNE    STØQUØ   *NØ
C6 01    A        LDAB   #1       *INDICATE NØN 0 QUØTIENT
B7 0008  N        STAA   NØNZRØ
20 0A 009E        BRA    STØQUØ
B6 0008  N NØZRØQ LDAA   NØNZRØ   *DIGIT IS 0
26 05 009E        BNE    STØQUØ   *PREVIØUS DIGITS 0?
7A 0002  N        DEC    PØWER    *YES
20 21 00BF        BRA    NXPWRQ   *DECREASE PØWER BY 1
FE 000A  N STØQUØ LDX    BSCQDL   *STØRE DIGIT INTØ BASIC
B6 0006  N        LDAA   QUØTNT   **VARIABLE
F6 0009  N        LDAB   NIBBLE
27 0B 00B4        BEQ    STØRGH   *STØRE INTØ RIGHT
48                ASLA            **NIBBLE ØF BYTE
48                ASLA
48                ASLA
48                ASLA
A7 00    A        STAA   X
7F 0009  N        CLR    NIBBLE
20 0B 00BF        BRA    NXPWRQ
AA 00    A STØRGH ØRAA   X
A7 00    A        STAA   X
7C 0009  N        INC    NIBBLE
08                INX
FF 000A  N        STX    BSCQDL
7A 0003  N NXPWRQ DEC    PWRCTR   *DIVISIØN BY ALL PØWERS
```

HEXBSC

```
27  B 00CF          BEQ     STØPWR    **ØF 10 FINISHED?
FE 0000  N          LDX     PWR10L    *CØNTINUE DIVIDING BY
09                  DEX               **NEXT PØWER ØF 10
09                  DEX
FF 0000  N          STX     PWR10L
7E 0034  P          JMP     DIVPWR
FE 000A  N STØPWR   LDX     BSCQDL    *STØRE LEAST SIGNIFICANT
B6 0001  A          LDAA    HEXNUM+1  **DIGIT INTØ VARIABLE
F6 0009  N          LDAB    NIBBLE
27 08 00E2          BEQ     STØRIG
48                  ASLA
48                  ASLA
48                  ASLA
48                  ASLA
A7 00    A          STAA    X
20 04 00E6          BRA     STØPW
AA 00    A STØRIG   ØRAA    X
A7 00    A          STAA    X
B6 0002  N STØPW    LDAA    PØWER     *STØRE EXPØNENT
FE 0000  A          LDX     BSCQL
A7 05    A          STAA    5,X
39                  RTS
                    END
```

CLINES

```
                    NAM     CLINES
                    ØPT     REL
                    XREF    NLINES
                    XDEF    CLINES
              *SUBRØUTINE TØ CØUNT THE NØ. ØF DATA
              *LINES ØF A PØRT. NØ. STØRED AN NLINES.
              *LØCATIØN ØF PØRT IN THE PØRTST
              *TABLE IS STØRED IN ACC. X. ACC. X IS NØT
              *CHANGED. ACC. A AND B ARE CHANGED.
                    PSCT
A6 08     A CLINES LDAA    8,X
7F 0000   A        CLR     NLINES
E6 09     A        LDAB    9,X        *GET 1ST. BIT MAP
54           SHFFLN LSRB               *SHIFT ØUT 0'S
24 FD 0007         BCC     SHFFLN
7C 0000   A INLNS1 INC     NLANES     *CØUNT NØ. ØF LINES
54                 LSRB               **USED IN THE FIRST PIC
25 FA 000A         BCS     INLNS1
81 01     A        CMPA    #1         *PØRT SPANS 1 PIC?
27 16 002A         BEQ     DNLNSQ     *YES
81 02     A        CMPA    #2         *PØRT SPANS 2 PIC?
27 08 0020         BEQ     INLNS2     *YES
C6 08     A        LDAB    #8         *PØRT SPANS 3 PIC
FB 0000   A        ADDB    NLINES     *ADD 8 FØR LINES USED
F7 0000   A        STAB    NLINES     **IN THE 2ND PIC
E6 0A     A INLNS2 LDAB    10,X       *CØUNT NØ. ØF LINES
54           INLNS3 LSRB               **ESED IN LAST PIC
24 05 002A         BCC     DNLNSQ     **SPANNED BY THE PØRT
7C 0000   A        INC     NLINES
20 F8 0022         BRA     INLNS3
A6 06     A DNLNSQ LDAA    6,X        *SIGNED BCD PØRT?
44                 LSRA
81 03     A        CMPA    #3
26 03 0034         BNE     CNLINF     *NØ
7A 0000   A        DEC     NLINES     *DECREASE 1 FØR SIGN
39           CNLINF RTS
                    END
```

LEVEL 3 SUBROUTINES

TSFBYT

```
                    NAM     TSFBYT
                    ØPT     REL
                    XREF    SØURCE
                    XDEF    TSFBYT
              *SUBRØUTINE TØ CØPY FRØM MEMØRY TØ MEMØRY
              *LØCATIØN ØF MEMØRY TØ BE CØPIED FRØM STØRED
              *IN SØURCE. LØCATIØN ØF MEMØRY TØ BE CØPIED TØ
              *STØRED IN DESTIN. NØ. ØF LØCATIØNS STØRED IN ACC
              *B. ACC. A, B AND X ARE ALL CHANGED.
              *SCRATCH PAD FØR LEVEL 3 SUBRØUTINES
              SCRAT4 CØMM    DSCT
   0002   A DESTIN RMB      2
                    PSCT
FF 0000   N TSFBYT STX      DESTIN
FE 0000   A MØVBYT LDX      SØURCE
A6 00     A         LDAA    X
08                 INX
FF 0000   A         STX     SØURCE
FE 0000   N         LDX     DESTIN
A7 00     A         STAA    X
08                  INX
FF 0000   N         STX     DESTIN
5A                  DECB
26 EB 0003          BNE     MØVBYT
39                  RTS
                    END
```

MULPLY

```
                    NAM     MULPLY
                    ØPT     REL
              SCRAT4 CØMM   DSCT
   0001   A RA      RMB     1
   0001   A RB      RMB     1
              *SUBRØUTINE TØ MULTIPLY ACC. A AND ACC. B
              *ASSUMING  NØ CARRY. PRØDUCT IS
              *STØRED AS UNSIGNED NØ. IN A.
              *ACC. B AND X ARE NØT CHANGED
                    XDEF    MULPLY
                    PSCT
B7 0000   N MULPLY STAA     RA
F7 0001   N         STAB    RB
5F                  CLRB
81 00     A         CMPA    #0
27 06 0011 ADDQ     BEQ     MULPLF
FB 0001   N         ADDB    RB
4A                  DECA
20 F8 0009          BRA     ADDQ
B6 0000   N MUPLF LDAA    RA
39                  RTS
                    END
```

SUBTRA

```
                        NAM     SUBTRA
                        ØPT     REL
                SCRAT4  CØMM    DSCT
   0002   A     SUBEND  RMB     2
   0001   A     BØRRØW  RMB     1
                        XREF    MINUEN
                        XDEF    SUBTRA
                *SUBRØUTINE TØ SUBTRACT 16 BIT HEX NØ.
                *IN MINUEN FRØM ACC. X, ASSUMING NØ. IN MINUEN
                *IS SMALLER THAN ACC. X. ACC. X
                *STØRES THE RESULT. ACC. A AND B ARE CHANGED
                        PSCT
FF 0000   N     SUBTRA  STX     SUBEND
7F 0002   N             CLR     BØRRØW
B6 0001   N             LDAA    SUBEND+1
B1 0001   A             CMPA    MINUEN+1
22 07 0015              BHI     SUBTRT
27 05 0015              BEQ     SUBTRT
C6 01     A             LDAB    #1
F7 0002   N             STAB    BØRRØW
B0 0001   A     SUBTRT  SUBA    MINUEN+1
B7 0001   N             STAA    SUBEND+1
B6 0000   N             LDAA    SUBEND
B0 0002   N             SUBA    BØRRØW
B0 0000   A             SUBA    MINUEN
B7 0000   N             STAA    SUBEND
FE 0000   N             LDX     SUBEND
39                      RTS
                        END
```

BLDADR

```
                        NAM     BLDADR
                        ØPT     REL
                SCRAT4  CØMM    DSCT
   0001   A     RA      RMB     1
   0001   A     RB      RMB     1
   0002   A     RX      RMB     2
                *SUBRØUTINE TØ ADD ACC. B TØ ACC. X
                        PSCT
                        XDEF    BLDADR
F7 0001   N     BLDADR  STAB    RB
FF 0002   N             STX     RX
FB 0003   N             ADDB    RX+1
F7 0003   N             STAB    RX+1
24 03 0011              BCC     LØADRX
7C 0002   N             INC     RX
FE 0002   N     LØADRX  LDX     RX
F6 0001   N             LDAB    RB
39                      RTS
                        END
```

CPBYTS

```
                    NAM     CPBYTS
                    ØPT     REL
                    XREF    HBYTES,LBYTES
                   XDEF    CPBYTC
             *SUBRØUTANE TØ CØMPARE 2 HEX NØ.'S. THE
             *NØ. IS STØRED IN THE LØCATIØN STØRED IN
             *HYBYTES, THE SECØND NØ. IS AT THE LØCATIØN
             *STØRED IN LBYTES. THE NØ. ØF BYTES IN IN
             *ACC. B. IF FIRST NØ.>2ND NØ., B IS SET
             *TØ 2, IF =, SET TØ 1, IF <, SET TØ 0. ACC.
             *X AND A ARE CHANGED
                    PSCT
FE 0000   A CPBYTS LDX     HBYTES
A6 00     A CMPUNT LDAA    X
FE 0000   A        LDX     LBYTES
A1 00     A        CMPA    X
22 19 0025         BHI     SETRB2
27 02 0010         BEQ     CPNXBQ
23 10 0020         BLS     SETBO
5A           CPNXBQ DECB
27 0F 0022         BEQ     SETB1
08                 INX
FF 0000   A        STX     LBYTES
FE 0000   A        LDX     HBYTES
08                 INX
FF 0000   A        STX     HBYTEC
20 E3 0003         BRA     CMPUNT
5F           SETBO CLRB
39                 RTS
C6 01     A SETB1  LDAB    #1
39                 RTS
C6 02     A SETRB2 LDAB    #2
39                 RTS
                   END
```

BLDAD2

```
                        NAM     BLDAD2
                        ØPT     REL
                SCRAT4  CØMM    DSCT
   0002       A ADDRST  RMB     2
                        XREF    ADDEND
                        XDEF    BLDAD2
                *SUBRØUTINE TØ ADD 16 BIT HEX NØ.'S IN
                *ACC. X AND ADDEND. X CØNTAINS RESULT AND
                *A IS CHANGED
                        PSCT
FF 0000         N BLDAD2 STX    ADDRST
B6 0001         A        LDAA   ADDEND+1
BB 0001         N        ADDA   ADDRST+1
B7 0001         N        STAA   ADDRST+1
B6 0000         A        LDAA   ADDEND
B9 0000         N        ADCA   ADDRST
B7 0000         N        STAA   ADDRST
FE 0000         N        LDX   ADDRST
39                       RTS
                         END
```