LOADING THE NUMERICAL CONTROL MACHINE CODE FROM AD-APT ONTO A MICROCOMPUTER CONTROLLED FLOPPY DISK

by

HSIAO-WEN LI

B. S. (I.E.), Tunghai University, Taiwan, 1976

A MASTER'S THESIS

submitted in partial fulfillment of the
requirements for the degree
MASTER OF SCIENCE

Department of Industrial Engineering
KANSAS STATE UNIVERSITY
Manhattan, Kansas

1981

Approved by:

John E. Biegel
Major Professor

TABLE OF CONTENTS

		Page
ACKNOWLEDGEMENT		iv
LIST OF FIGURES		iii
CHAPTER 1.	INTRODUCTION	2
1.1	HISTORICAL BACKGROUND	2
1.2	TYPES OF NC MEDIA	4
1.3	PROBLEMS	6
1.4	METHOD	6
CHAPTER 2.	USING CMS	9
2.1	GENERAL CONCEPT OF CMS	9
2.2	THE /* ROUTE PUNCH VMXXX CARD	9
2.3	SIGN-ON AND SIGN-OFF PROCEDURES	9
CHAPTER 3.	INTERFACING THE EXIDY SORCERER MICROCOMPUTER WITH THE ITEL SYSTEM	12
3.1	GENERAL DESCRIPTION OF ADDRESS, INPUT/OUTPUT PORTS, AND DATA NOTATION	12
3.2	THE UART FUNCTION AND I/O PORT	12
3.3	MODIFIED KEYBOARD SCAN ROUTINE	13
3.4	FULL-DUPLEX OPERATION	14
3.5	ADDING FOUR SPECIAL FUNCTIONS	15
3.6	PROGRAM FLOW CHART	19
3.7	CONNECTION BETWEEN EXIDY SORCERER AND MULTI-TECH SYSTEMS FM30 DATA COUPLER	46
CHAPTER 4.	INTERFACING THE EXIDY SORCERER WITH THE MICROPOLIS DISK OPERATING SYSTEM (MDOS)	50
4.1	HARDWARE INSTALLATION	50
4.2	PREPARING THE SYSTEM FOR USE WITH THE FLOPPY DISK SUBSYSTEM	50

4.3	STORING PROGRAMS ON DISK	51
4.4	LOADING A PROGRAM INTO EXIDY MEMORY	52
4.5	THE EXIDY SORCERER MEMORY MAP AND THE MDOS MEMORY MAP	53
CHAPTER 5.	USING THE COMPUTER ASSISTED CONTOURING SYSTEM (AD-APT)	57
5.1	GENERAL CONCEPT OF COMPUTER ASSISTED CONTOURING SYSTEM	57
5.2	COMPUTER PART PROGRAMMING	57
CHAPTER 6.	RESULTS	61
CHAPTER 7.	FURTHER RESEARCH	63
REFERENCES		64
APPENDICES		
A.	AD-APT PROGRAM	66
B.	Z80 SOFTWARE PROGRAM	68
C.	OUTPUT OF AD-APT	83
D.	ASCII FILES OF AD-APT OUTPUT	87

LIST OF FIGURES

Figure		Page
1.	Connections Between Exidy Sorcerer & FM30 Data Coupler	47
2.	Sorcerer Memory Map	54
3.	MDOS Memory Map	55
4.	The Concept of the NC Operation Planning Process	58
5.	Sample Part Cutter Path	59

ACKNOWLEDGEMENT

The author expresses his gratitude to his friends, H. L. Hsang, Danny Hsu and C. T. Chen, for their encouragement over the past year. In addition the author is very thankful to Professor John E. Biegel for his guidance and suggestions during the course of this research.

THIS BOOK CONTAINS NUMEROUS PAGES THAT WERE BOUND WITHOUT PAGE NUMBERS.

THIS IS AS RECEIVED FROM CUSTOMER.

CHAPTER 1

INTRODUCTION

INTRODUCTION

1.1 HISTORICAL BACKGROUND OF NUMERICAL CONTROLLED MACHINES

Before numerically controlled machine tools were commercially available to manufacturers, existing machine tools were more or less manually operated. The skill of the machinist could influence the quality of the manufactured part and also the rate at which it was produced. The desire for eliminating machine-operator intervention and increasing the production rate led to the development of numerical control machines.

The history of numerical control (NC) began in the late 1940s, when John T. Parsons proposed that a method of automatic machine control be developed which would guide a milling cutter to generate a smooth curve [1]. As he conceived it, the co-ordinate points would be coded onto punched cards and fed to a machine controller which would cause a modified milling machine to move in small incremental steps to achieve the desired path. In 1949 the U. S. Air Force became aware of the highly complex shapes that were needed for spaceage aircraft and missiles [2]. The introduction of new materials required new machining methods with greater accuracy and precision than could be achieved with conventional production machine tools [2]. It was recognized that the costs of tooling and manufacturing under the existing method would be prohibitive.

By 1957, the first successful NC installations were being used in production [1]; however, many users were experiencing difficulty in generating part programs for input to the machine controller. To remedy this situation, M.I.T. began the develop-

ment of a computer based part programming language called APT — Automatically Programmed Tools [1]. The objective was to devise a symbolic language which would enable the part programmer to specify mathematical relationships in a straightforward manner. In 1962 the first APT programming system was released for general industrial applications [1].

The development of numerical control technology has taken place on two major fronts. Hardware development concentrated on improved control systems and machine tools. By 1965 sophisticated NC machines and control systems were available for every major machine tool configuration. Software development concentrated on improvements to the APT language as well as the origination of other NC programming systems.

A change in overall philosophy began in the 1970s, and numerical control was then viewed as part of a larger concept — computer-aided-manufacturing (CAM) [1]. Although numerically controlled machine tools are essential for the development of operational CAM systems, conventional NC machines must be modified so that information may be passed between the machine control unit (MCU) and computer based control system. This modification has resulted in one major development derived from the NC concept: computer managed numerical control.

Computer managed numerical control is a generic term that encompasses DNC (Direct Numerical Control) and CNC (Computerized Numerical Control). DNC and CNC are methods that distribute programmable computing responsibility between a control computer and an NC machine tool. Neither system changes the functional characteristics of the NC machine; instead, each provides a means for communication of process data and commands outside the NC machine control loop.

1.2 TYPES OF NC MEDIA

1.2.1 PUNCHED CARDS

The punched card now has limited application as an input media for NC. However, since it was first originated by Herman Hollerith in 1887 [1], the punched card has provided an efficient and accurate means of data storage and transfer. The punched card is a low density storage medium and can only be input to the MCU at rates that are much slower than other media types. The intrinsic ability to remove and replace cards also leads to the possibility of mis-sequencing or loss of an NC block*.

1.2.2 PUNCHED TAPE

Paper tape perforators were first developed in the early 1940s, and have evolved to sophisticated units such as the Flexowriter [1]. This automatic machine reads and prints tapes at high speed. Input methods using punched tape require a tape reading device. One such device is the photoelectric tape reader that can interpret one hundred to one thousand punched characters per second from a perforated tape passed under a reading head. The reading head consists of a light source and an array of light sensitive elements. As a hole passes

* The NC block is used to pass the numerical control information to the MCU. It must be capable of specifying dimension and nondimension data. Dimension data comprise linear and angular motion commands, and nondimension data are the preparatory functions used to describe types of movement.

over the light source, an element of the array is turned "on". The pattern of "on" elements is decoded by the MCU and is either acted upon immediatelly or transferred to the information storage section of the MCU.

1.2.3 MAGNETIC MEDIA

The use of a magnetic material as a recording and storage medium was proposed by Poulsen in the late 1800s [1]. Modern magnetic tapes are made by coating a polyester or Mylar film with an iron oxide coating. Such magnetic tapes have extremely high storage densities: 1600 to 6250 bytes per inch are common in computer applications. Higher densities are possible.

The use of magnetic tape for NC applications has been limited by the fragile nature of the media. Dirt, oil, or dust can cause read errors, and in the industrial environment ferrous metal filing and/or magnetized tools cause damage to the data on the tape. In recent years sealed magnetic tape cassettes, like those used in sound recording equipment, have been used with some NC systems. However, because a slight movement of the tape away from the reading heads causes a substantial reduction in signal and data input errors result, it is not likely to see much use in the foreseeable furture.

A more reliable magnetic storage device, the flexible or floppy disk, has recently been introduced as an NC communication medium. A single 5.25 inch diameter disk contains the equivalent of about 1000 feet of punched tape. This is about 139 K bytes of information [3].

1.2.4 COMPUTER TRANSMITTED NC DATA

CNC and DNC are outgrowths of computer and NC technology. In a CNC system, the MCU is replaced by a programmable minicomputer. A number of NC programs may be stored in the computer's memory. After the NC instructions are stored, however, the minicomputer performs most or all of the basic numerical control functions.

In a DNC system, a large scale computer is used to control many NC machines simultaneously. All NC programs are stored using peripheral devices which have very large storage capacities. A computer program executed by the central processing unit (CPU) of a digital computer is often used to replace the MCU. Because data is internally stored and accessed, the use of physical media is greatly reduced.

1.3 PROBLEMS

The problem addressed in this research is to load the Numerical Control Machine Code from Adaptation of Automatically Programmed Tools (AD-APT) on a microcomputer controlled floppy disk.

1.4 METHOD

The process of loading AD-APT output, in NC machine code, onto a microcomputer controlled floppy disk was divided into four phases - 1) using the Conversational Monitor System (CMS), 2) interfacing the Exidy Sorcerer Microcomputer with the Itel System through a modem, 3) using the Micropolis Disk Operating System (MDOS) to save the data output from the Itel System on a floppy disk, 4) getting AD-APT output from the Itel System onto a floppy disk in a form that is usable in controlling an NC machine.

CHAPTER 2

USING CMS

2.1 GENERAL CONCEPT OF CMS

CMS, Conversational Monitor System, is a powerful tool for doing interactive computing. Using CMS, a remote typewriter terminal provides the complete facilities of a powerful computer. All the work can be kept in files, and the debugging work can be done without using cards or a keypunch.

2.2 THE /* ROUTE PUNCH VMXXX CARD

Operating System (OS) job traffic uses a /* ROUTE card to direct output files to desired locations [4]. The /* ROUTE card must appear in the job control stream right after the JOB card. If only the punched output is to be routed to the visual memory, the /* ROUTE PUNCH will be used in job control stream [4].

The use of the /* ROUTE PUNCH VMXXX card is to direct punched output to the CMS machine from an OS batch job submitted to the system in any manner; i.e. hands-on card reader, through the dispatcher, or from CMS [4].

2.3 SIGN-ON AND SIGN-OFF PROCEDURES

From dial-up terminals, the computer is called at 9-5374141. When the computer responds with a high-pitched tone, place the receiver in the coupler. The red `Carrier Detect´ light on the acoustic coupler indicates that the connection has been established. Press the RETURN key to trigger the `VM/370 online´ message.

Respond to the `VM/370 online´ message by typing:

"logon VMXXX" [4].

The system will then respond with an account number prompting message and a mask. Type in the account number over the mask and press the RETURN key. The system will respond with the SSN (Social Security Number) prompting message and a mask. Type in the SSN. The system will produce about four lines of logon information. After the appearance of the ready message (R;), the system is ready to begin the CMS session.

To sign off, simply enter `logoff´ [4].

CHAPTER 3

INTERFACING THE EXIDY SORCERER MICROCOMPUTER WITH THE ITEL SYSTEM

3.1 GENERAL DESCRIPTION OF ADDRESS, INPUT/OUTPUT PORTS, AND DATA NOTATION

The Sorcerer's Z80 central processing unit (CPU) has sixteen address lines; every memory storage location has a sixteen digit binary number as its address. This means that the Sorcerer will recognize up to 65536 different memory addresses [5]. Whenever reference is made to a memory address it is necessary to use a four digit hexadecimal number.

The Z80 has eight data lines — one for each bit of storage in a memory location. Every memory location contains a number from 0 to 255 (decimal). It uses hexadecimal notation for the memory contents, and writes each such number as two hexadecimal digits.

Similarly, the first eight address lines can designate an input/output port (I/O port). Since these eight lines can be set to any number from 0 to 255, there are 256 input and 256 output ports. They are identified by the hexadecimal numbers, ØØH* to FFH.

3.2 THE UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART) FUNCTION AND I/O PORT

The Exidy's UART function can be selected in software to communicate either with the cassette recorder or through the RS-232 interface [6]. (RS-232 is the Electronic Industries Association (EIA) standard interface between data terminal

* All hexadecimal number will have an "H" suffix.

equipment and data communication equipment employing serial binary data interchange.) The choice is controlled by bit 7*, the most significant bit present at port FEH. For operation of the RS-232 interface this bit must always be set "on" (set to a 1) [5].

This same port, FEH, is also used to control the baud rate with bit 6 ("on" for 1200 baud and "off" for 300 baud) [5]. Port FDH is the UART status port. Bit 0 indicates the transmit buffer status; 0 for empty, 1 for full. Data is available to be transmitted when bit 1 is a 1, when no data is available bit 1 is a 0 [5]. Port FCH is the serial data I/O port, and through it the system transmits an eight bit data word.

3.3 MODIFIED KEYBOARD SCAN ROUTINE — KEYBD

The RS-232 is enabled by setting bit 7 of I/O port FEH. However, this is the same port that the keyboard uses, so when Sorcerer scans the keyboard, bit 7 is reset and the RS-232 capability is disabled. Therefore to correct a "bug" in the monitor ROM (Read-Only Memory), the keyboard scan routine is copied in its entirety into RAM (Random-Access Memory). Then it is necessary to modify the few places where the offending bits exist. Line 8** of the KEYBD subroutine is changed from the original keyboard scan routine. It was "LD A, 1"; which is changed to "LD A, 81". Line 26 was changed in a similar way [7].

* The bits in an 8-bit "data word" are numbered from 0 through 7 going from right (least significant bit-LSB) to left (most significant bit-MSB).

** All the line numbers refer to the programs in Appendix B.

There is second keyboard routine in the monitor, QUIKCK. This subroutine does a quick check of CTRL/C, ESC, and RUN/STOP keys. It uses the output port FEH also. It leaves the most significant bits in the state specified by the bytes located at IY+45H and IY+3DH in memory. Thus, QUIKCK does not need to be replaced, it is only necessary to change these two bytes. (See lines 185 and 186.)

3.4 FULL-DUPLEX OPERATION

Full-duplex means that the keyboard's output is sent to the modem and, simultaneously, the modem's received signal is displayed on the video monitor. This simultaneous I/O means that the microcomputer must always be listening to both the modem and the keyboard and talking to both the video monitor and the modem.

The problem can be divided as two seperate operations:

1. If a key is pressed, send its value out on the RS-232 bus.
2. If a character is received from the modem, display it on the video monitor.

Of the two; the second operation is faster and is the basis of the routine INIFR (input if ready). This subroutine checks the UART data available status, and if a character has been received, it is displayed on the video monitor.

Placing INIFR at the end of KEYBD and calling the INIFR elsewhere in the program catches all of the operation; namely the transmission of data. The top level of the program is TERMINAL; it calls the KEYBD until a character is typed. SNOUT (the RS-232 output driver) is called when a character is eventually typed. If all lines beyond 192 were removed, and the object code of line 192 changed to 18E9H, the Exidy would be a complete full-duplex, dumb terminal [7].

3.5 ADDING FOUR SPECIAL FUNCTIONS

There are four principal functions that have been implemented:

1. To send a large ASCII (American Standard Code for Information Interchange) file from the Sorcerer's memory to the Itel System*.
2. To receive a large ASCII file from the Itel and place it in the RAM section of the Exidy.
3. To view the contents of the Exidy RAM on the monitor.
4. To exit to the monitor to do local chores.

It is assumed that the keyboard will never send graphic characters to the control computer. For graphics output, the Exidy sets bit 7 (the parity bit) to 1, but the Itel System ignores bit 7. The program filters out these characters (using line 189 and 190) before sending them to SNOUT.

The graphic-shift-period combination (ØEFH) was chosen as a special-graphics character to signify that one of the previously mentioned four special functions was desired. When this special character is typed a prompt, <<<, is displayed on the video monitor, indicating that the microcomputer is ready for one of the four types of functions. The monitor's input routine, MINPUT, makes use of the keyboard scan routine that contains the

* The Itel System is the university mainframe computer.

software bug previously mentioned. The program is patched to store the starting address of KEYBD at the memory locations addressed by IY+41H and IY+42H (lines 196 to 199).

Starting with line 206, MINPUT is used, setting the HL register pair to the beginning of the line buffer and calling FTOKEN, which returns with the Z-flag* set if the line is empty; otherwise, register A contains the first non-blank character in the line, which should be S, R, V or X for the four possible commands respectively. If none of these four characters is found, the program will display two question marks and ignore the request (starting with line 217).

3.5.1 SENDING FILES

The IBM 3705-E1 Communications Controller on campus reacts as a half-duplex terminal and has a 1200 baud (max) data transmission rate over the phone line. It does not have a control-S (or XOFF), which means that the data transmission should be halted while the Itel System catches up by processing its input buffer. Also it does not have a control-Q (or XON), which indicates that it is ready for the data to be received [8]. In the next year, these characters which control the data transmission rate will be installed.

The use of the control-S and control-Q will permit the Itel System to catch up and avoid the loss of subsequent data. The current approach is much more successful and should

* The Z-flag indicates result of an instruction, if the result has the value 0, this flag is set; otherwise it is reset.

work for almost any computer installation. It never lets the echo* fall more than one line behind.

During block transmissions, SOUT is used instead of SNOUT. SOUT acts like SNOUT except that when it must transmit a (CR), it does not return to the calling program (lines 302 to 304) but proceeds to WAIT, which waits for a (LF) before allowing a return to the program that called SOUT to avoid missing the display of the last few characters in each line, the WAIT takes every character it receives and forwards it to FSEND, which acts much like the SEND in the monitor's ROM except that it filters out control characters for special attention. The (CR) and (LF) cause no problems for SEND; they are not filtered out.

Usually at the beginning of a new line the (null)s and (bell)s are received. The FSEND is used to ignore the former and display "<BELL>" for the latter (using lines 335 through 346). The system will also type out the hexadecimal value of any such control characters and follow with four equal signs for an eye-catching effect (lines 327 through 334).

If the block transmission is terminated because QUIKCK has detected a keypress, the next RAM address that was to be used will be displayed. The rest of the line on the monitor will be filled with >s. (These tasks are done by ADONE, starting at line 226.) Otherwise, the block transmission will terminate when the MEOF is found in RAM, then control goes to ADONE.

* The echo means the response signal that comes from the mainframe computer.

3.5.2 RECEIVING FILES

This feature is used to load a file from the mainframe computer into the microcomputer RAM starting at location 3ØØØH. For example, a command can be input to the mainframe computer to cause it to type the file without pause on the Exidy but the (CR), which completes the command, will not follow the command immediately. The mainframe computer will not act upon a command until it receives this (CR). Then following the command by a special-graphics character, R, a space and the address 3ØØØH should be typed. The video will display: T D F <<< R 3ØØØH. The first part of the line (T D F) depends upon the command that the mainframe computer recognizes and the file-name and filetype desired.

If every thing appears to be in order, type a (CR). The microcomputer program then determines what is wanted and sends a (CR) to the mainframe computer. The Itel proceeds to output file D, which appears on the video monitor as well as being stored in the RAM. When the output is complete or the file is no longer wanted, hit the key RUN/STOP. The program will place a (CR), a (LF) and the MEOF(ØØH) into RAM to complete the record. Control then passes to ADONE, which shows the memory address of the next available byte.

The RECEIVING input routine is necessary to filter the incoming stream of characters to eliminate the (null)s before they enter RAM. Otherwise, the files in RAM will be messy.

3.5.3 VIEWING FILES

The program permits an examination of ASCII files in RAM without interaction with the mainframe computer. After the special-graphics character, V, space, a hexadecimal address and a (CR) are typed on the Exidy keyboard, the ASCII files will be displayed on the monitor at a rate of about 100 characters per second. As with sending files, the display will terminate when MEOF is found in RAM or when QUIKCK sees a keypress. Upon termination, a hexadecimal address of the next available byte will be displayed. The program is written in Z80 assembly language; a detailed Z80 instruction set is contained in [9] and [10].

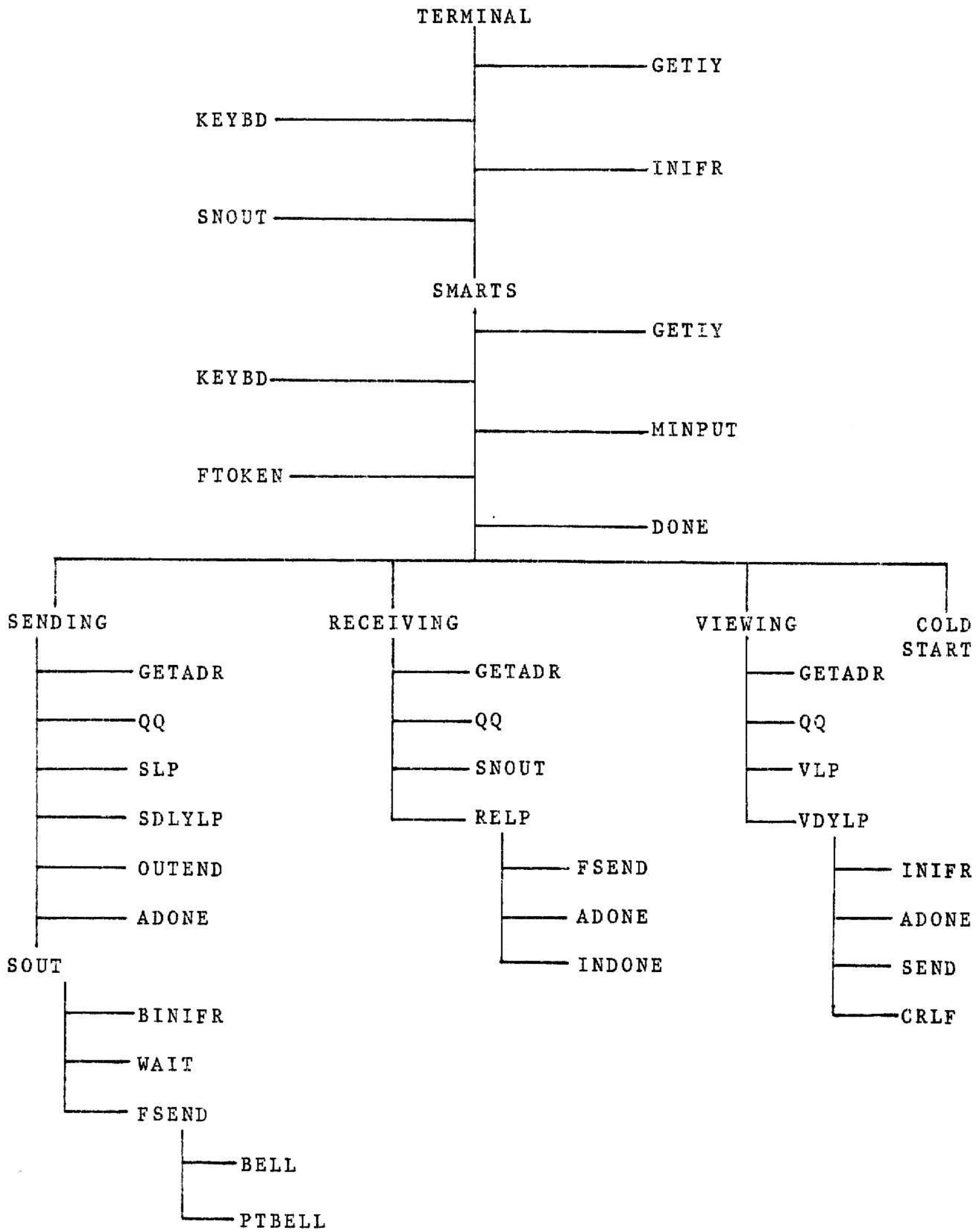
3.6 PROGRAM FLOW CHART

The program tree that follows shows how the various subroutine are used by main program (TERMINAL). Subroutines that called from the Exidy MDOS are not listed.

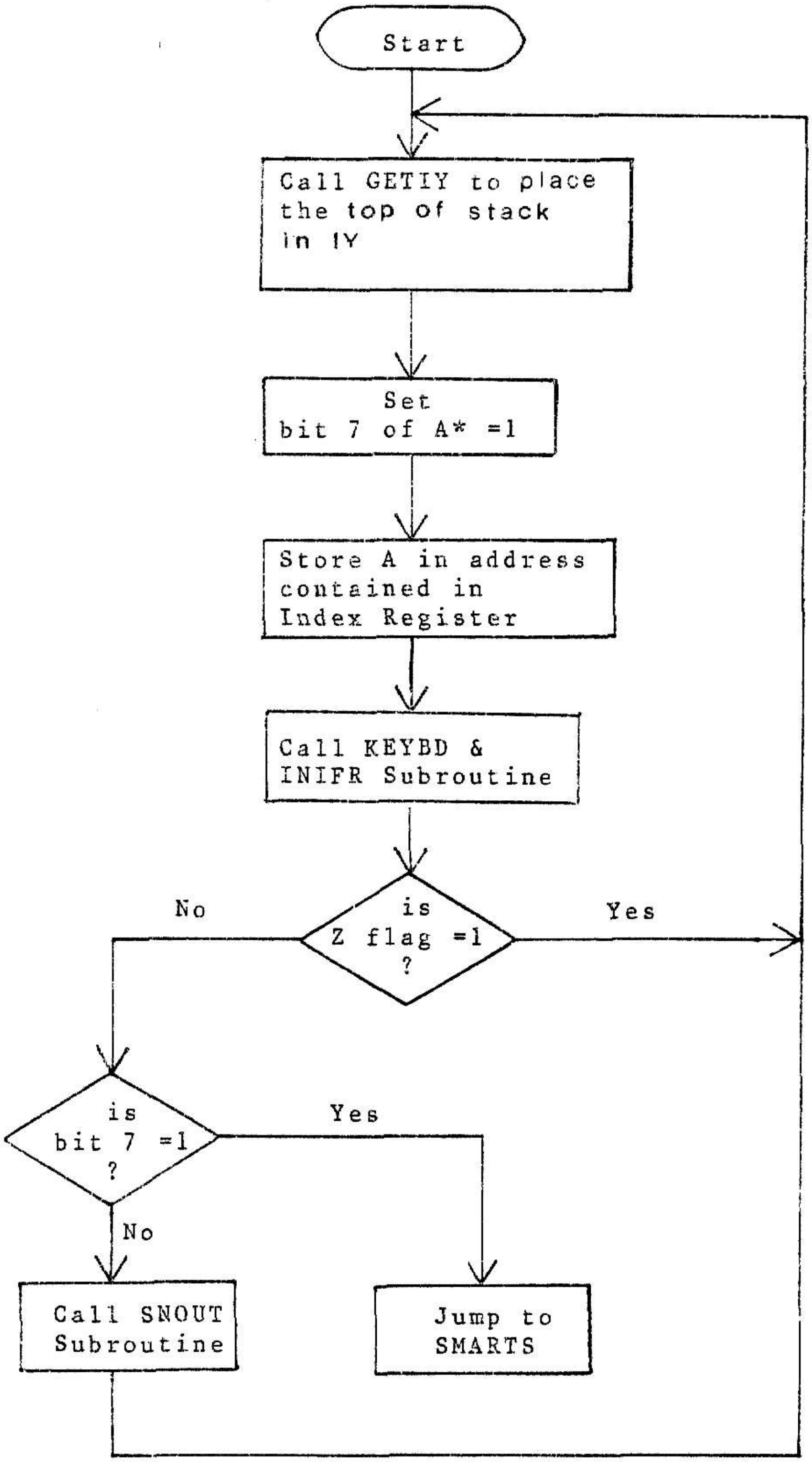
THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE.

THIS IS AS RECEIVED FROM CUSTOMER.

PROGRAM TREE

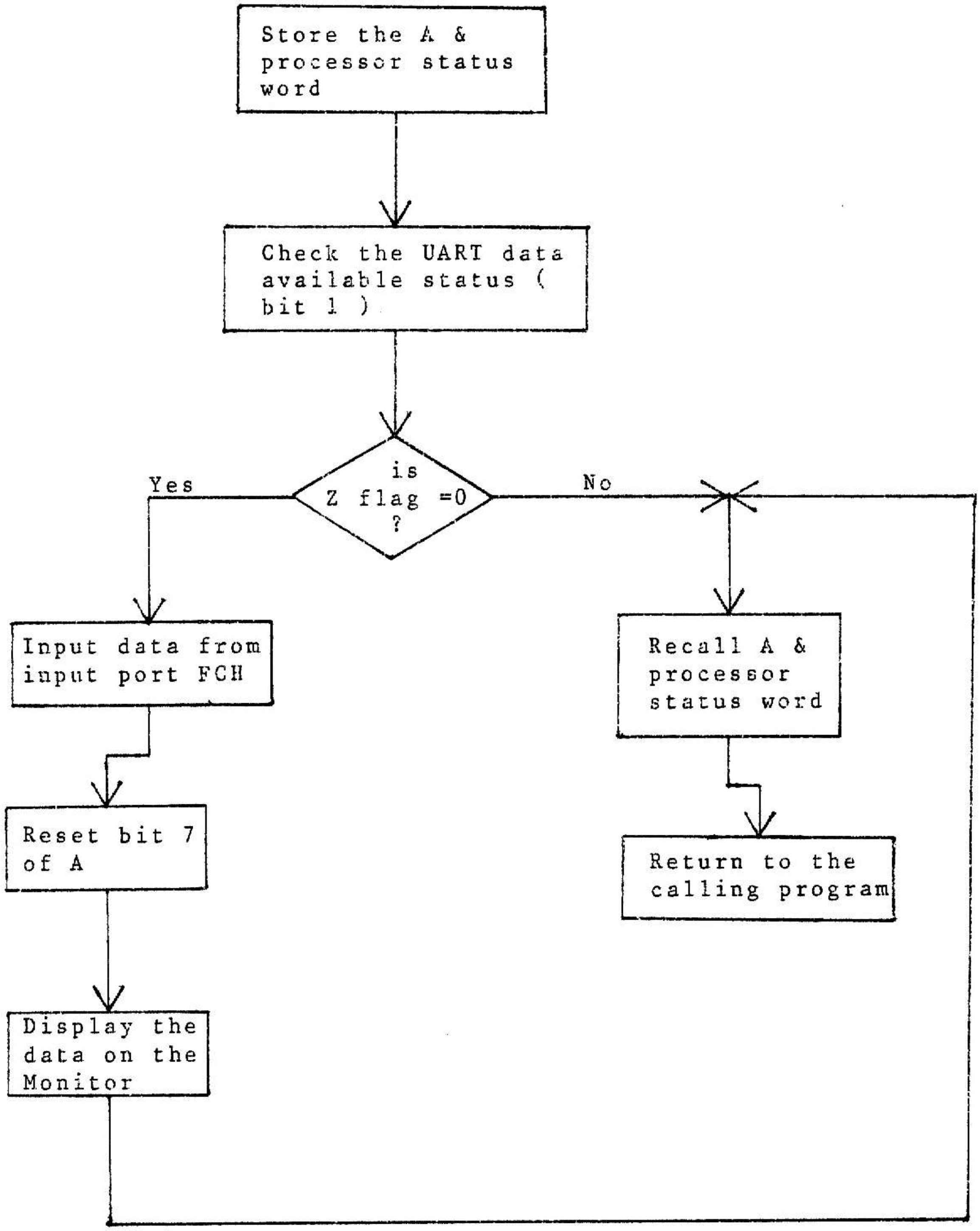


TERMINAL

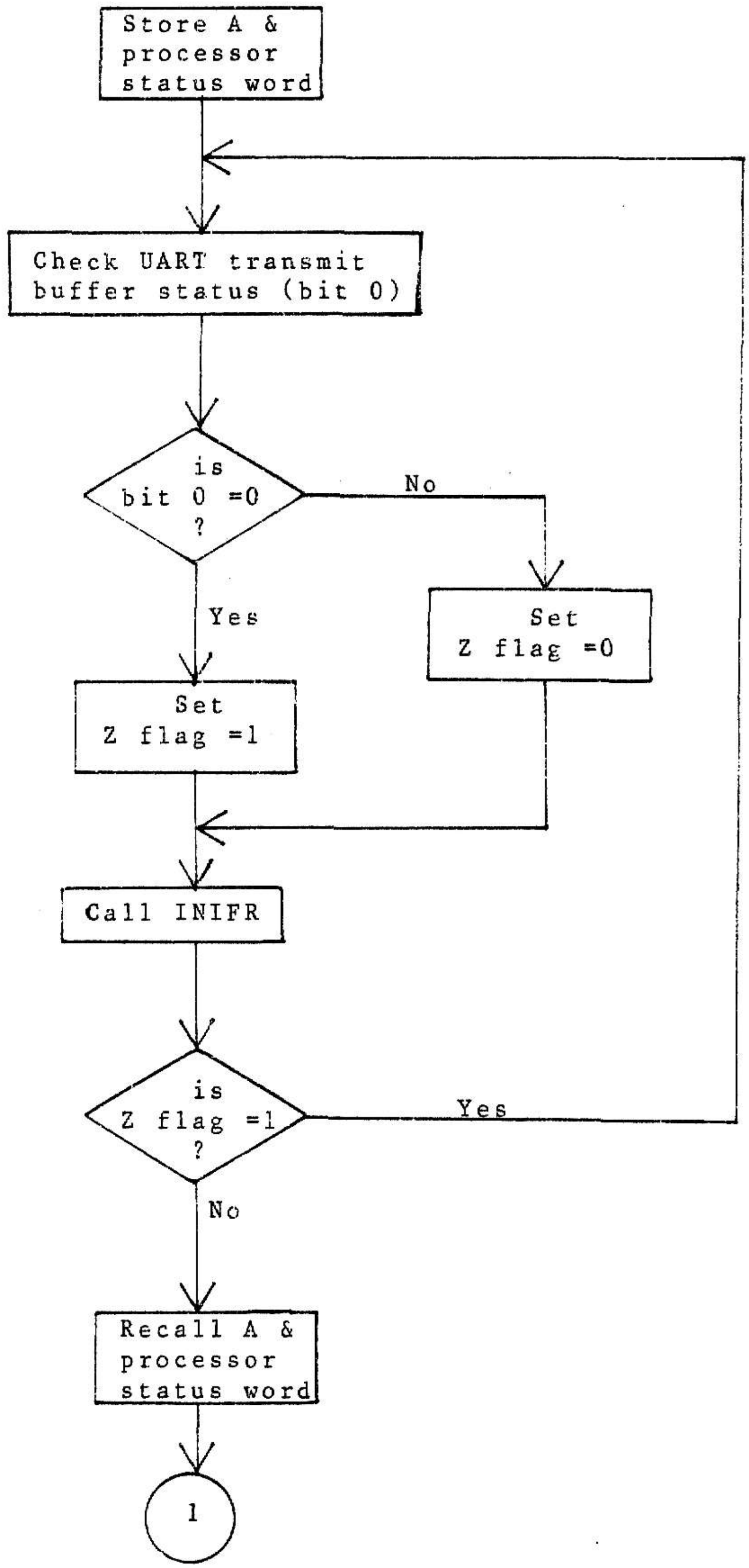


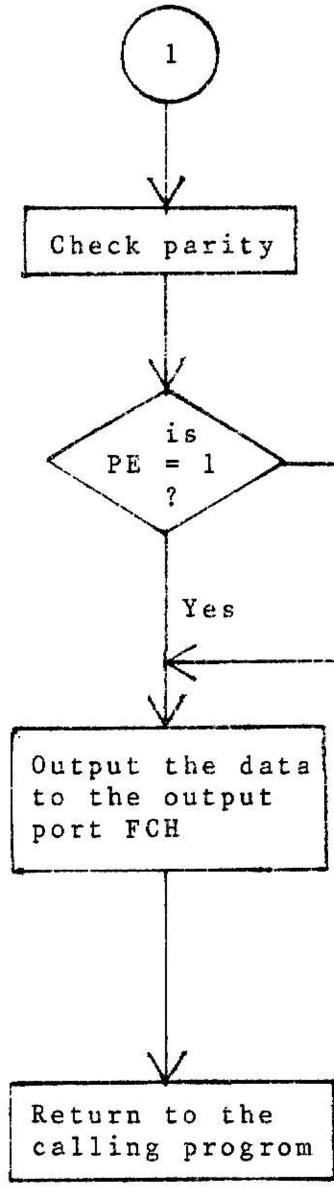
* A is the Accumulator.

INIFR

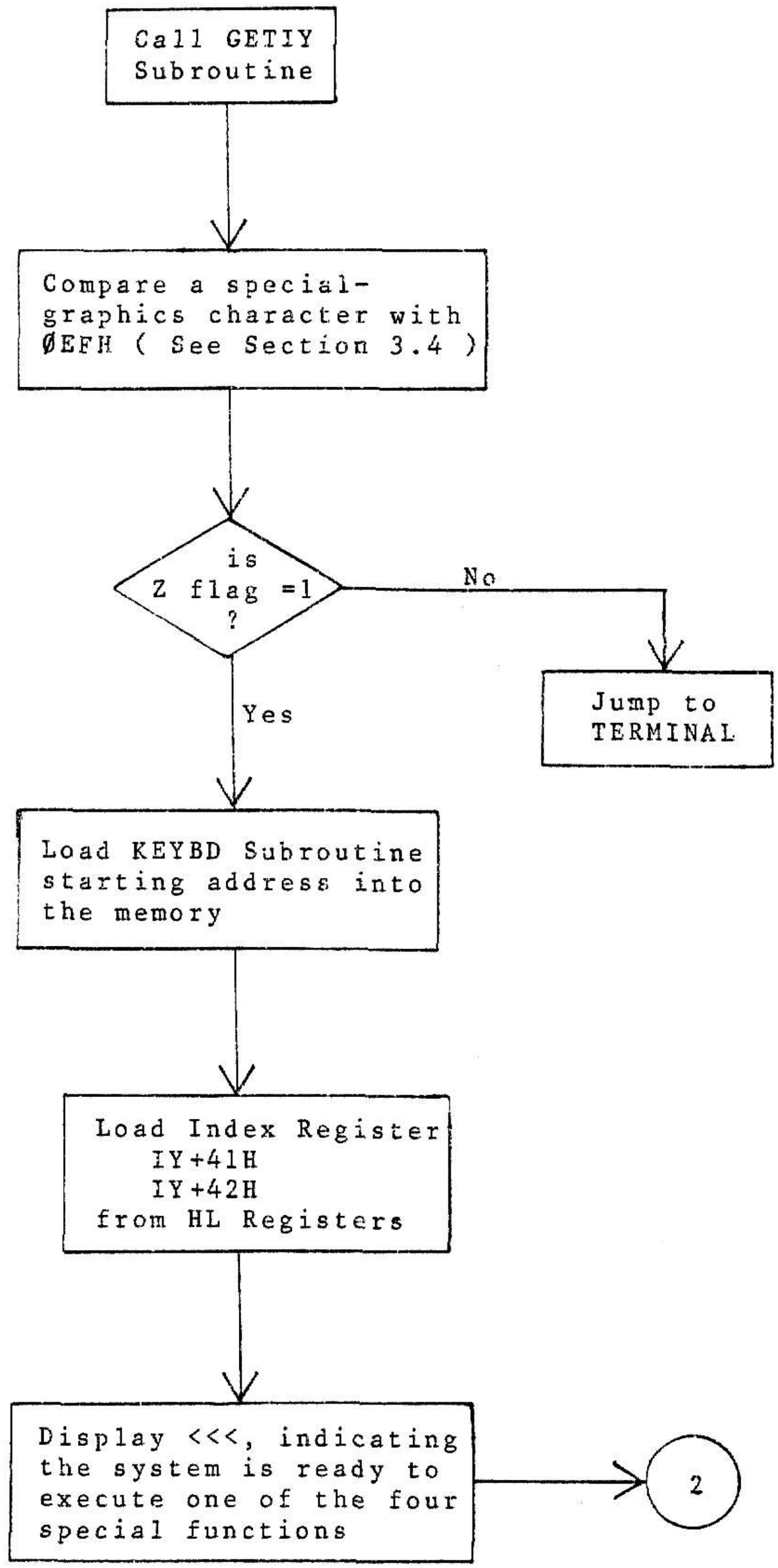

Store the A & processor status word
Check the UART data available status (bit 1)
is Z flag =0 ?
Yes
No
Input data from input port FCH
Reset bit 7 of A
Display the data on the Monitor
Recall A & processor status word
Return to the calling program

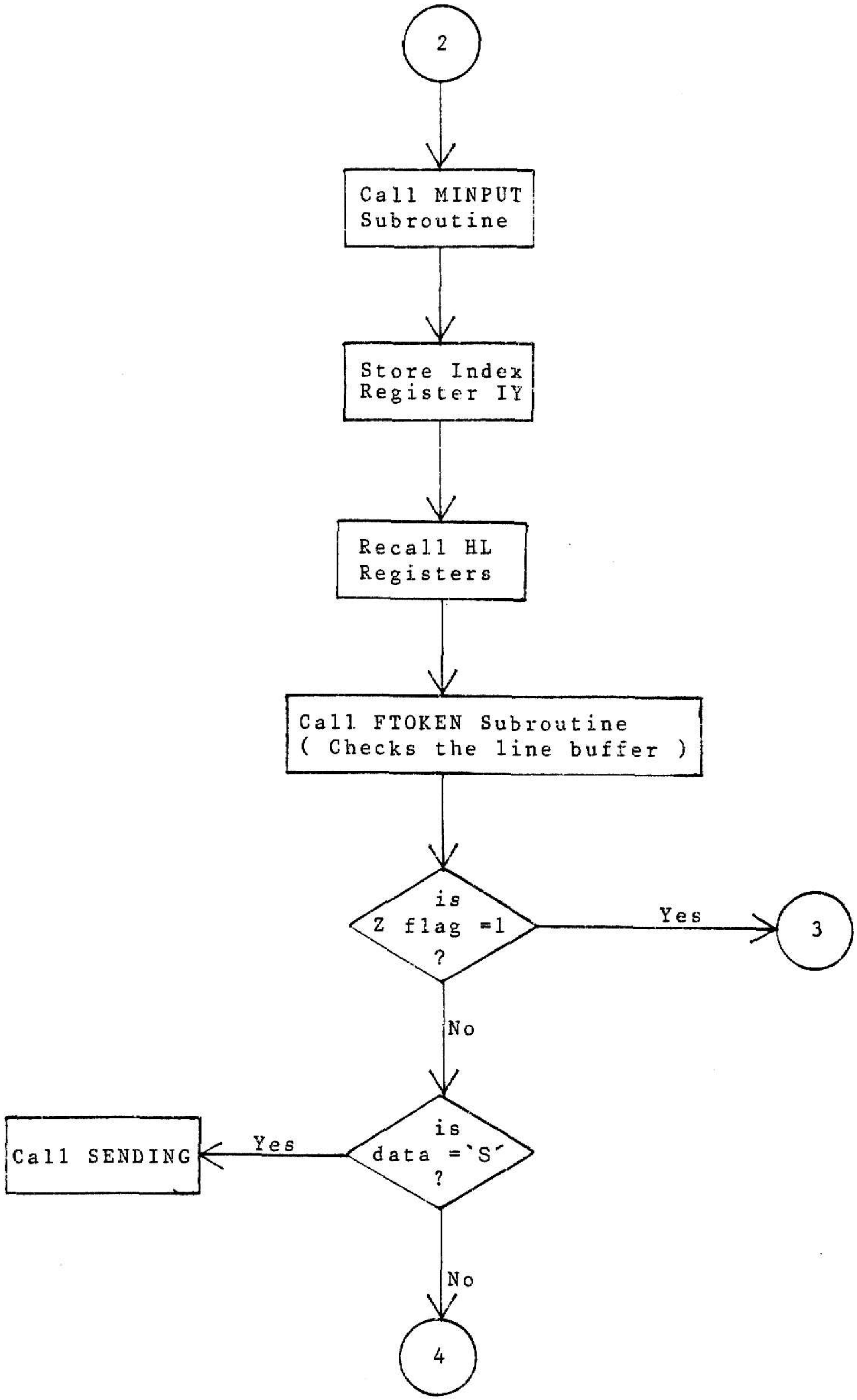
SNOUT

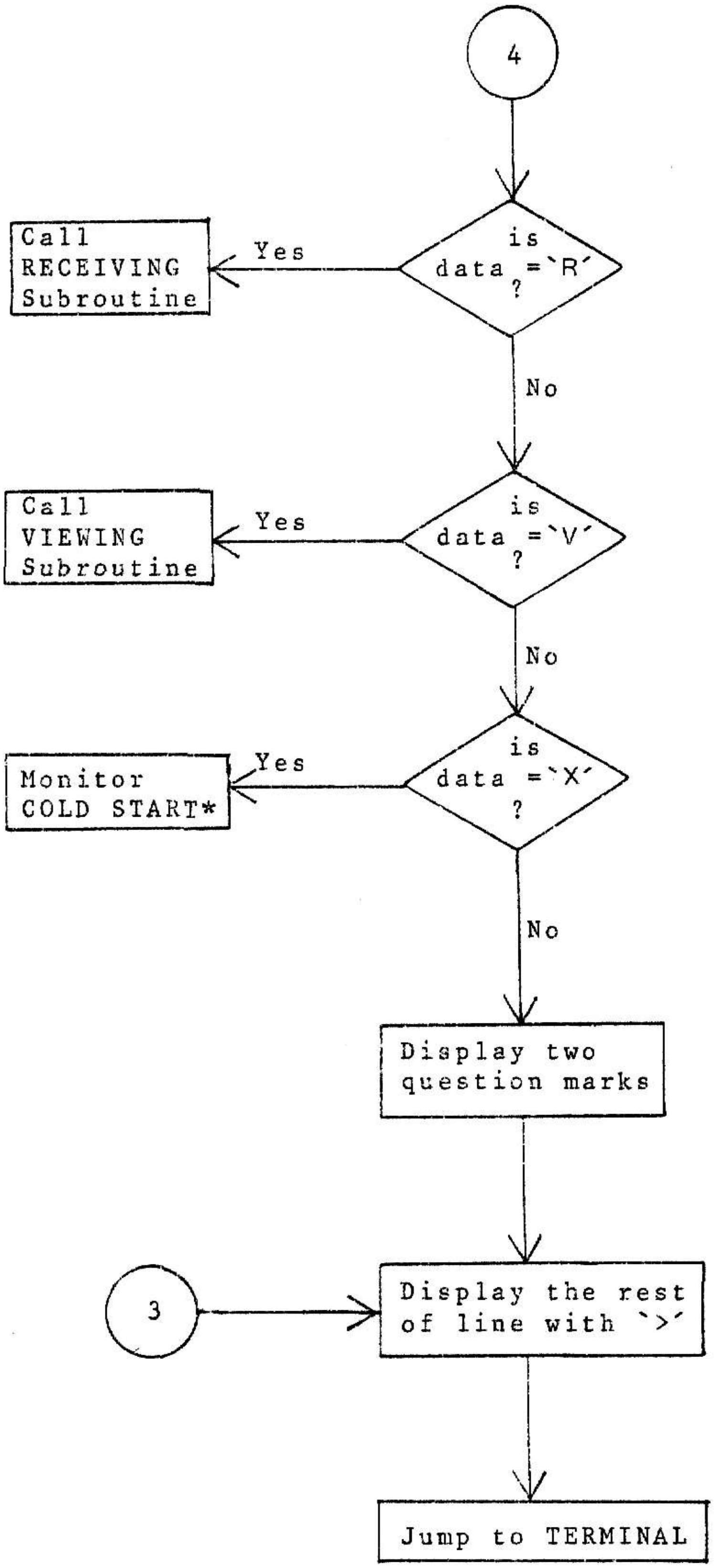

Store A & processor status word
Check UART transmit buffer status (bit 0)
is bit 0 =0 ?
No
Yes
Set Z flag =0
Set Z flag =1
Call INIFR
is Z flag =1 ?
Yes
No
Recall A & processor status word
1


1
Check parity
is
PE = 1
?
No
Set bit 7 =1
Yes
Output the data
to the output
port FCH
Return to the
calling progrom

SMARTS

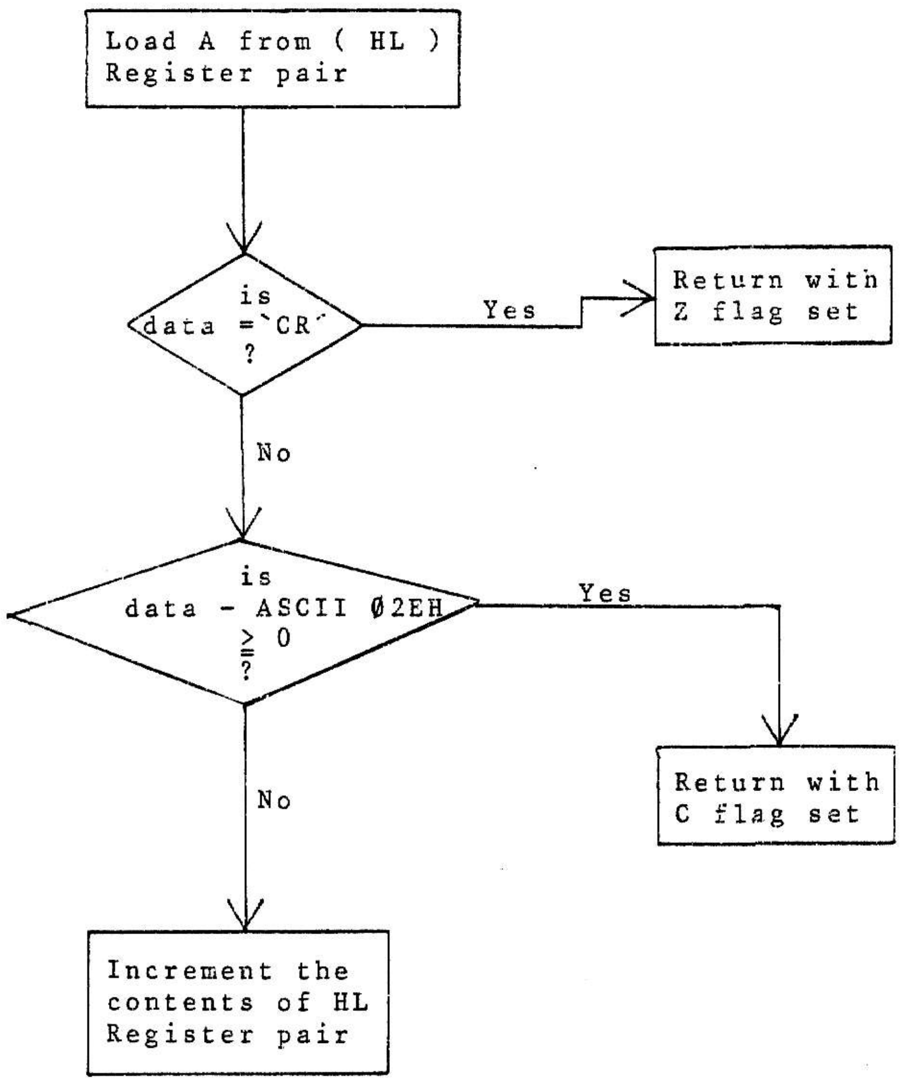

Call GETIY Subroutine
Compare a special-graphics character with ØEFH (See Section 3.4)
is Z flag =1 ?
No
Jump to TERMINAL
Yes
Load KEYBD Subroutine starting address into the memory
Load Index Register IY+41H IY+42H from HL Registers
Display <<<, indicating the system is ready to execute one of the four special functions
2


2
Call MINPUT Subroutine
Store Index Register IY
Recall HL Registers
Call FTOKEN Subroutine (Checks the line buffer)
is Z flag =1 ?
Yes
3
No
is data =`S´ ?
Yes
Call SENDING
No
4

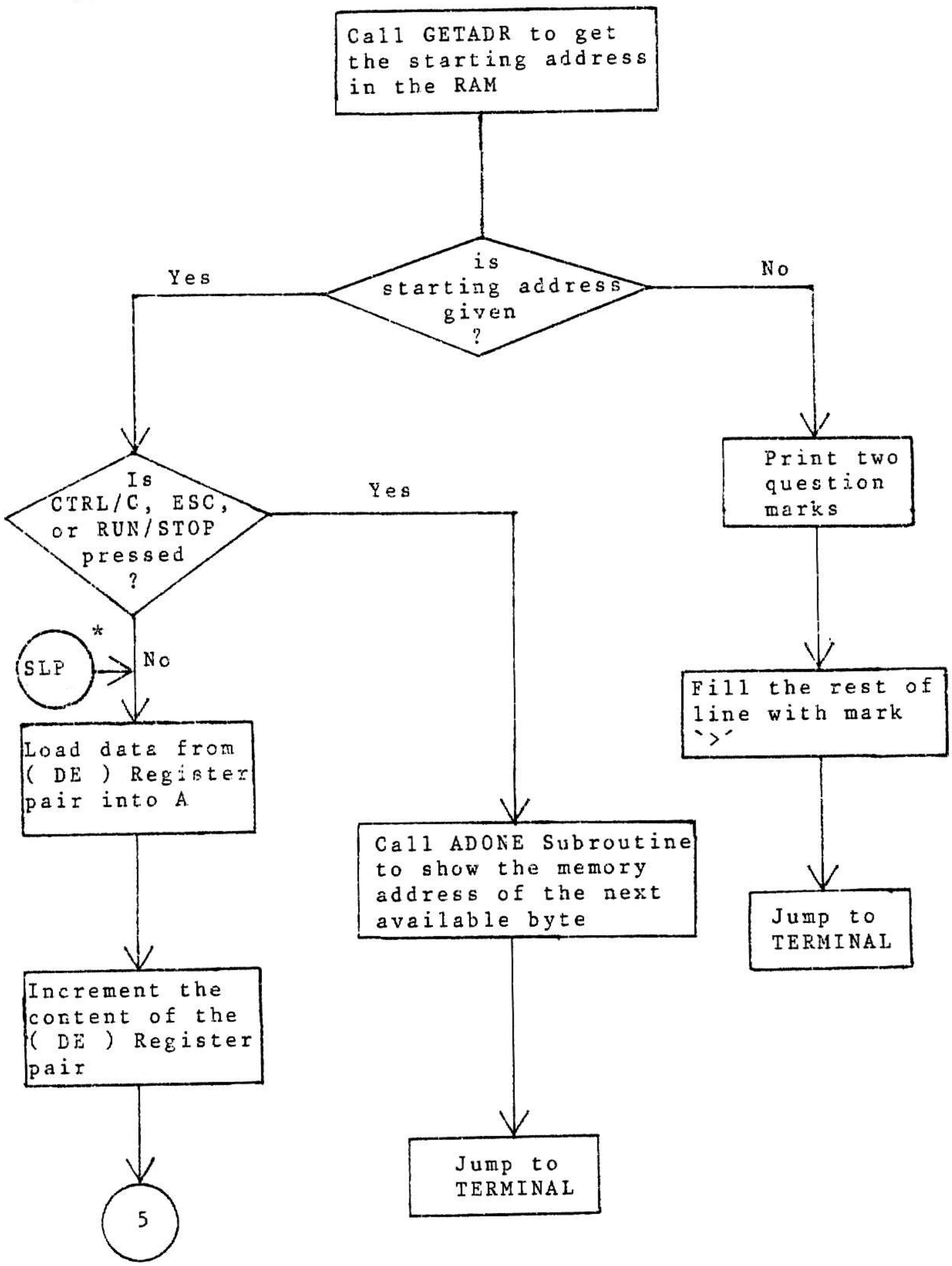


* Sorcerer enters at this address, E000H , when first turned on, or when the RESET key is depressed

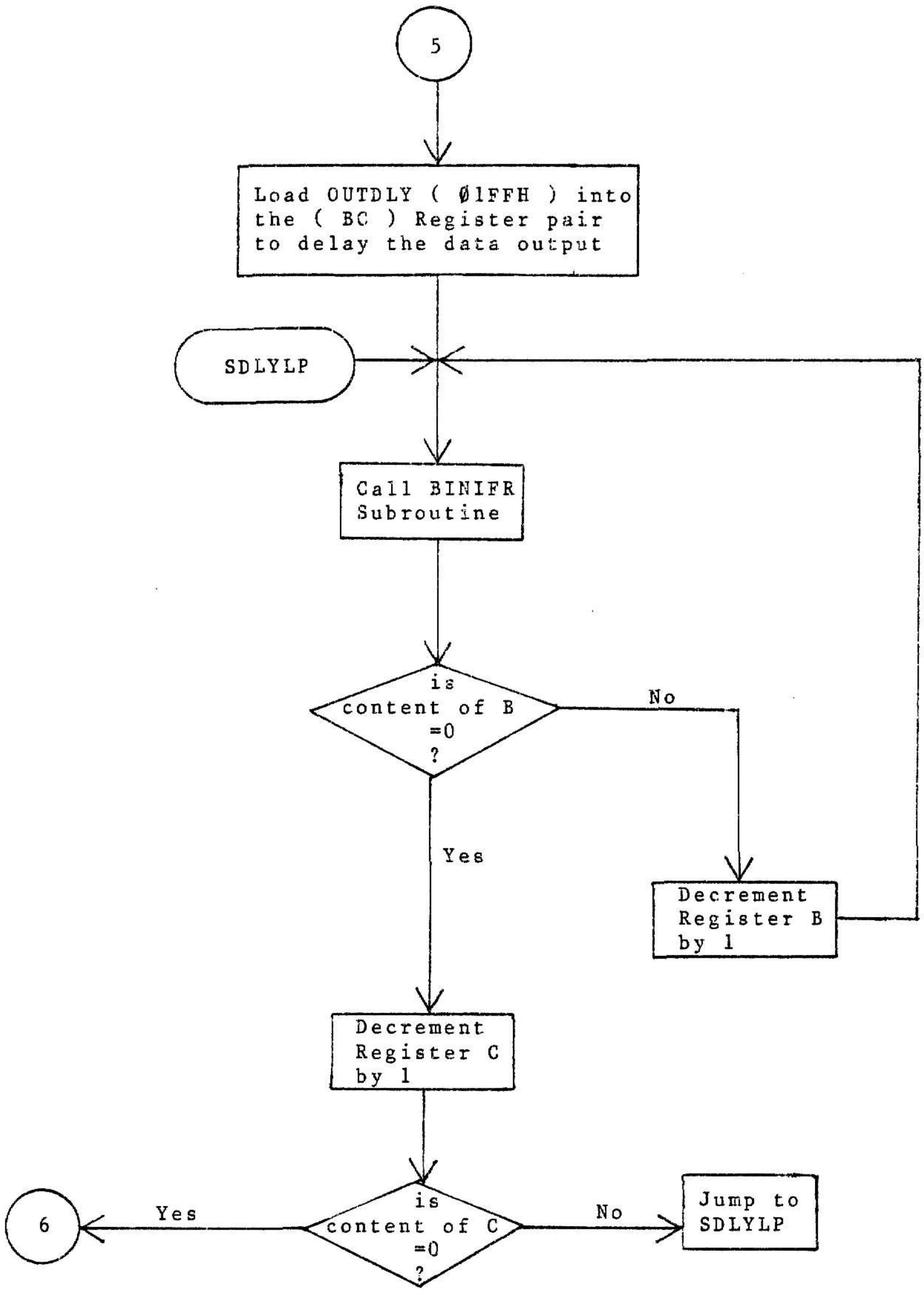
FTOKEN

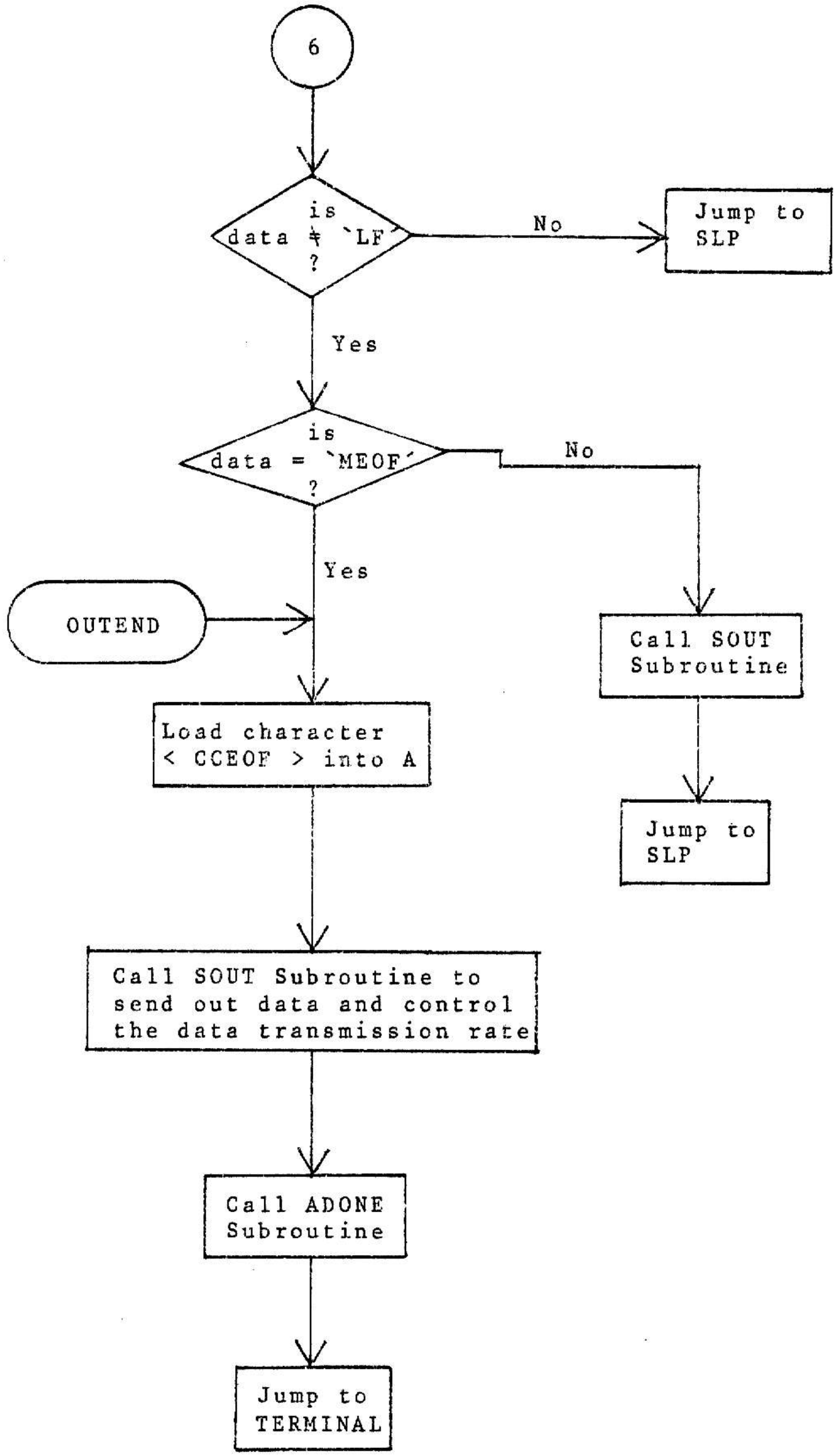

Load A from (HL)
Register pair
is
data =`CR`
?
Yes
Return with
Z flag set
No
is
data - ASCII Ø2EH
≥ 0
?
Yes
Return with
C flag set
No
Increment the
contents of HL
Register pair

SENDING

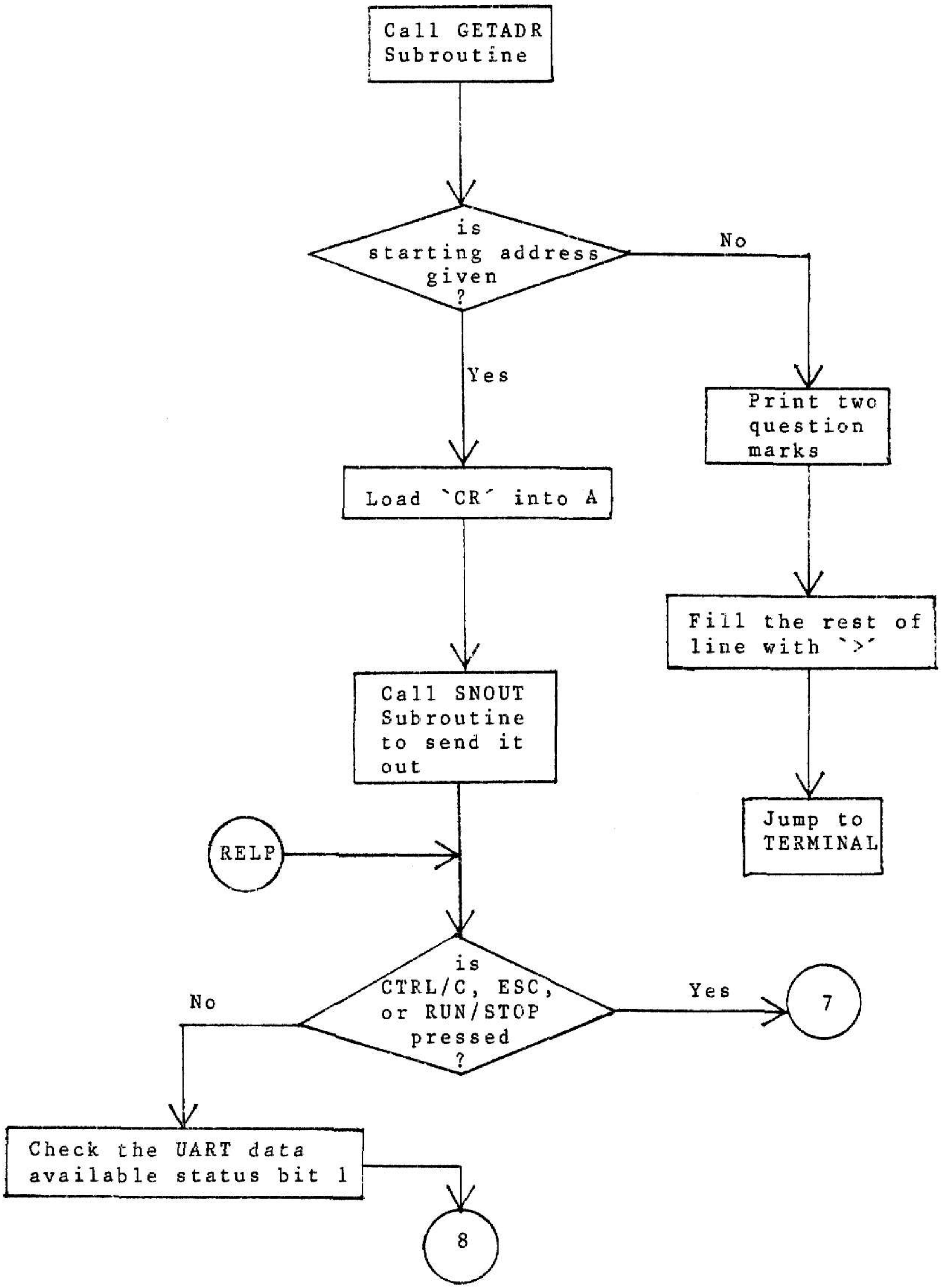


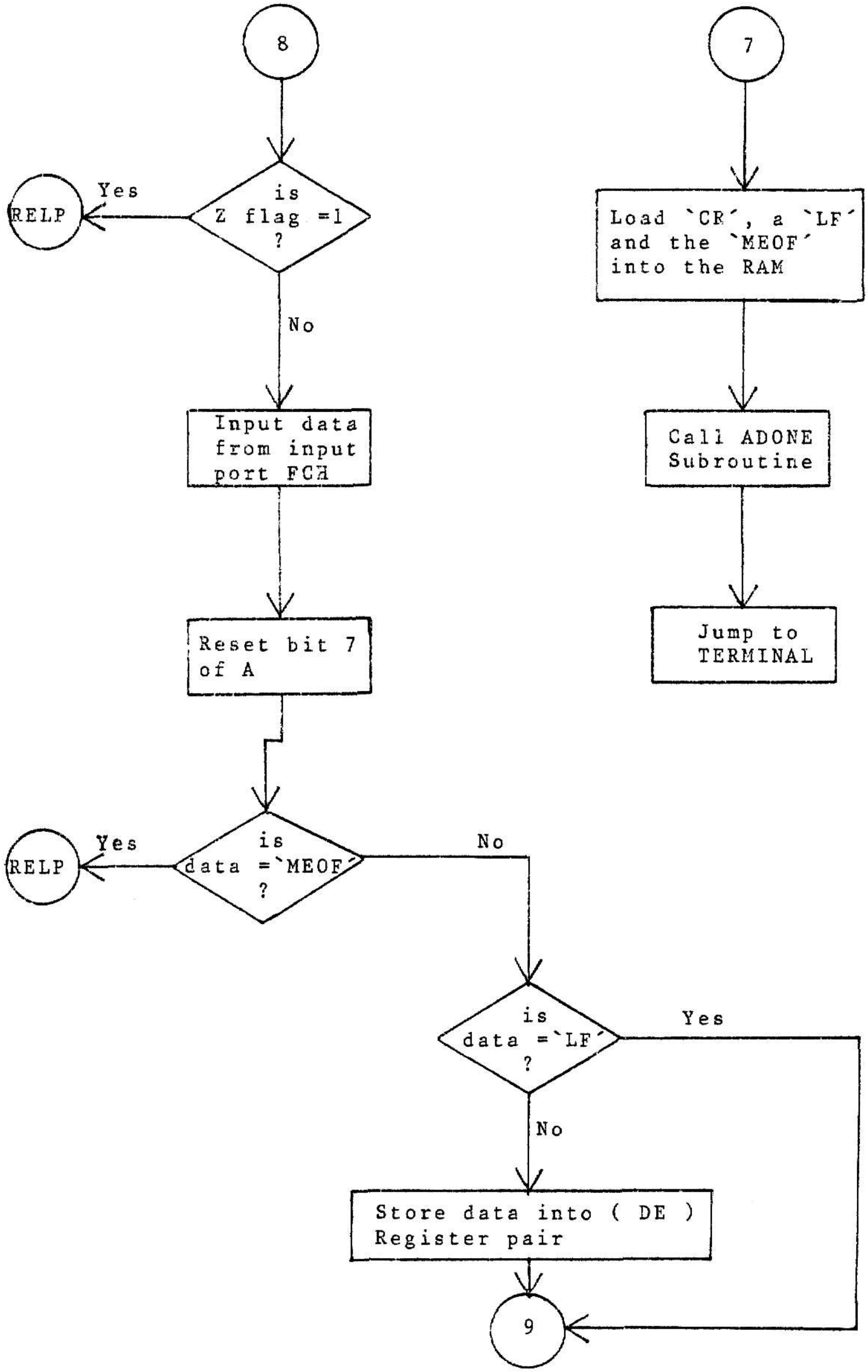
* This means we are returning from the subroutine, SLP

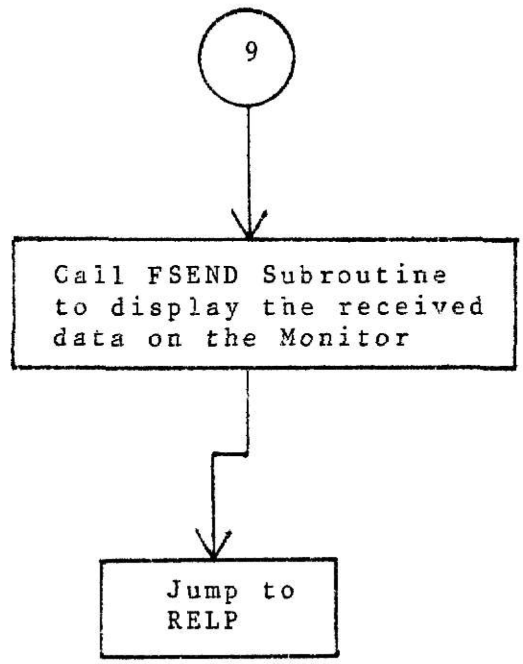

5
Load OUTDLY (Ø1FFH) into the (BC) Register pair to delay the data output
SDLYLP
Call BINIFR Subroutine
is content of B =0 ?
No
Yes
Decrement Register B by 1
Decrement Register C by 1
is content of C =0 ?
Yes
No
6
Jump to SDLYLP


6
is
data ≠ `LF´
?
No
Jump to
SLP
Yes
is
data = `MEOF´
?
No
Yes
OUTEND
Call SOUT
Subroutine
Load character
< CCEOF > into A
Jump to
SLP
Call SOUT Subroutine to
send out data and control
the data transmission rate
Call ADONE
Subroutine
Jump to
TERMINAL

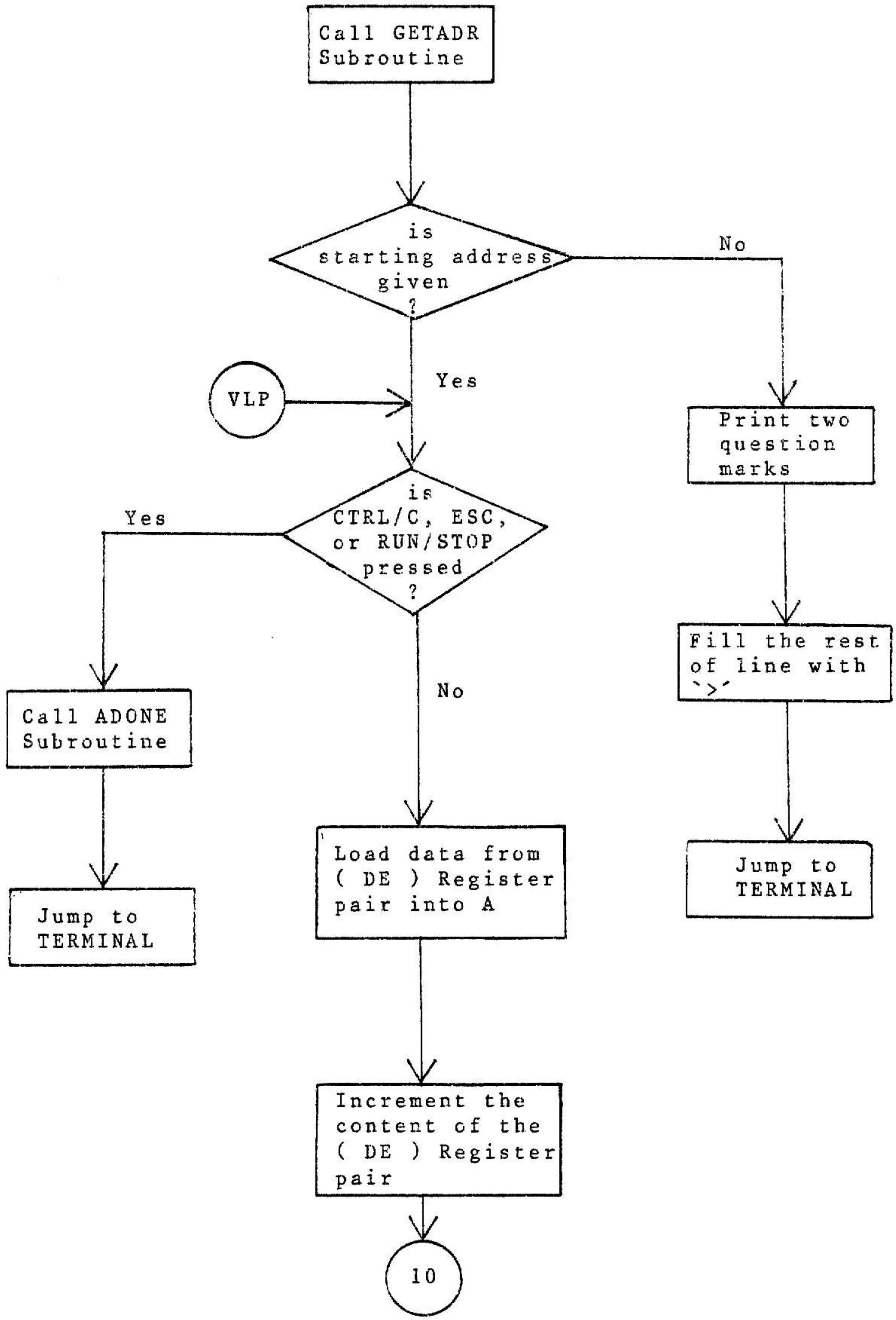
RECEIVING


Call GETADR Subroutine
is starting address given ?
No
Yes
Print two question marks
Load `CR´ into A
Fill the rest of line with `>´
Call SNOUT Subroutine to send it out
Jump to TERMINAL
RELP
is CTRL/C, ESC, or RUN/STOP pressed ?
No
Yes
7
Check the UART data available status bit 1
8

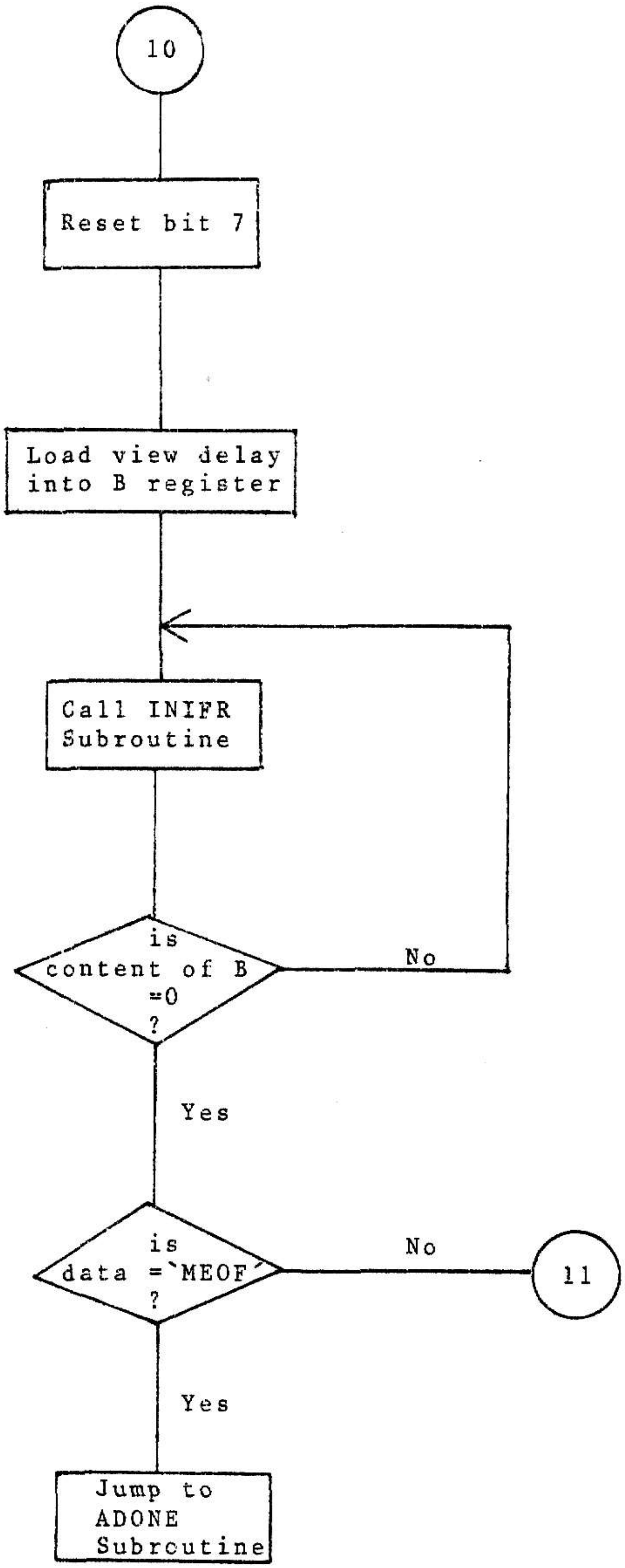

8
is
Z flag =1
?
Yes
RELP
No
Input data
from input
port FCH
Reset bit 7
of A
is
data =`MEOF´
?
Yes
RELP
No
is
data =`LF´
?
Yes
No
Store data into (DE)
Register pair
9
7
Load `CR´, a `LF´
and the `MEOF´
into the RAM
Call ADONE
Subroutine
Jump to
TERMINAL

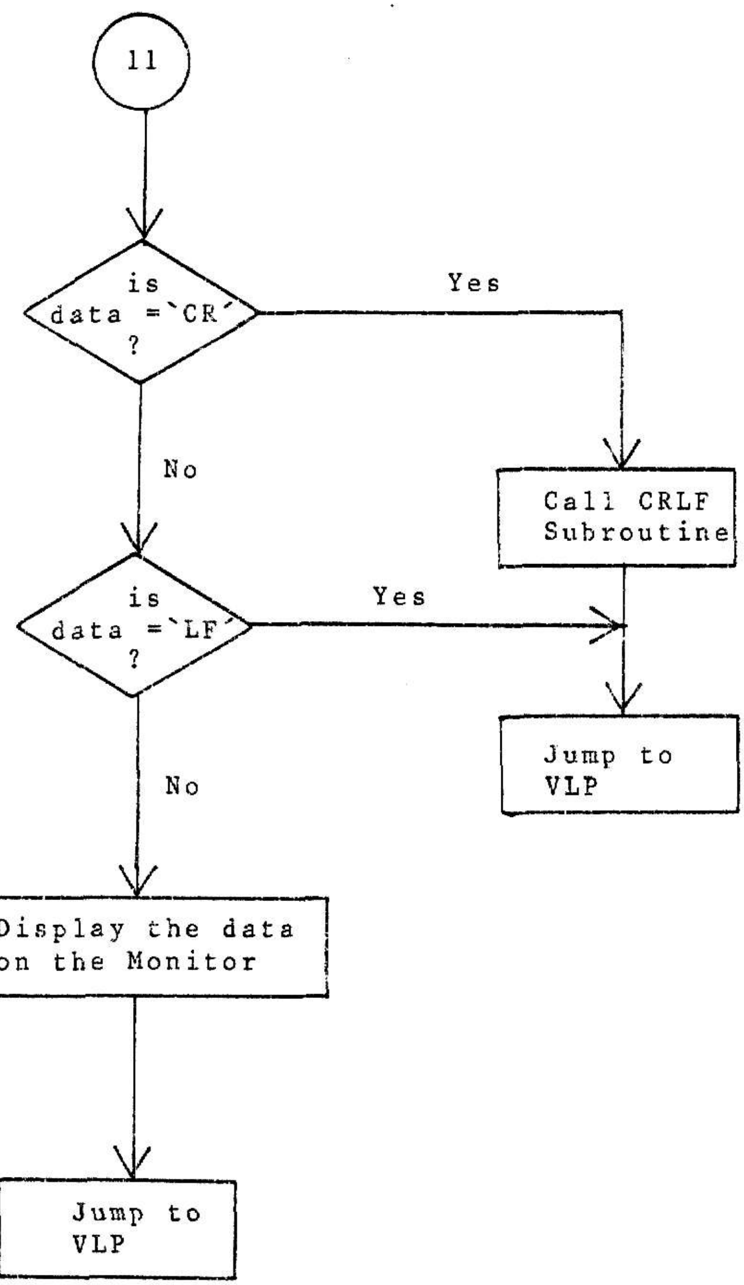

9
Call FSEND Subroutine to display the received data on the Monitor
Jump to RELP

VIEWING

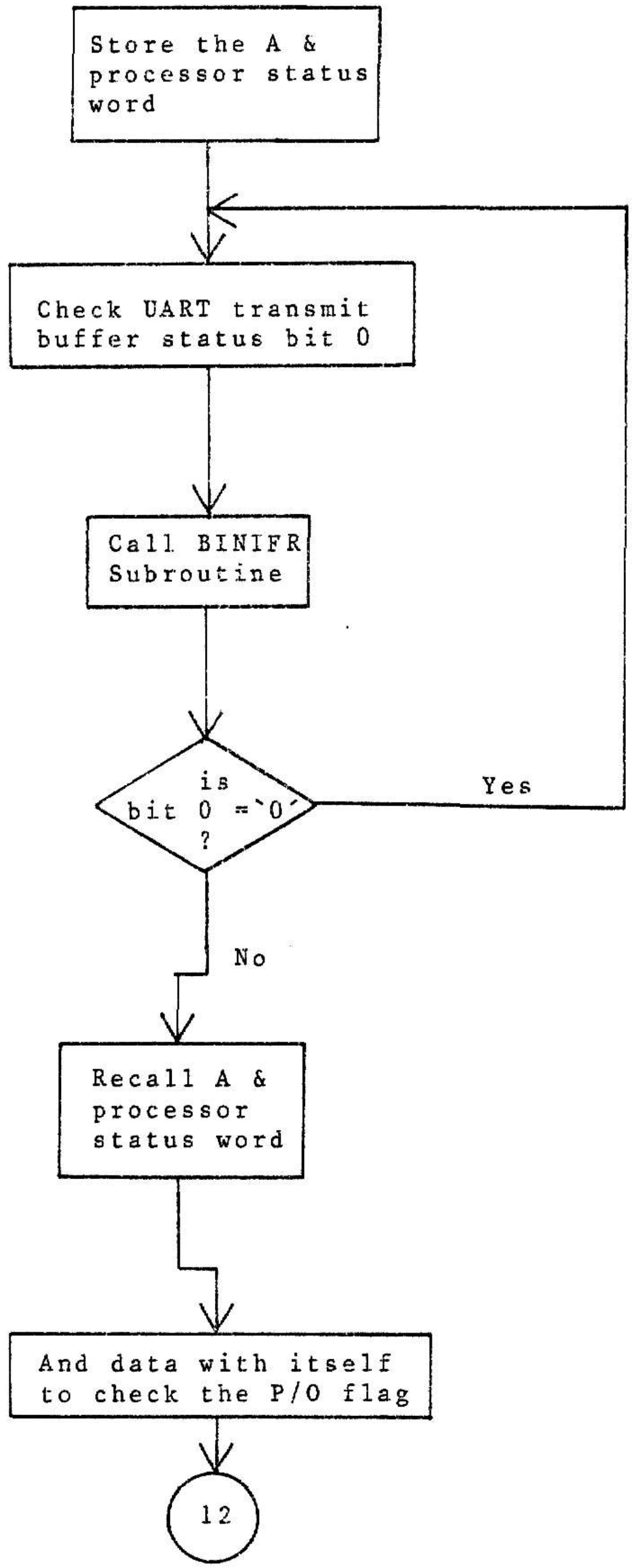


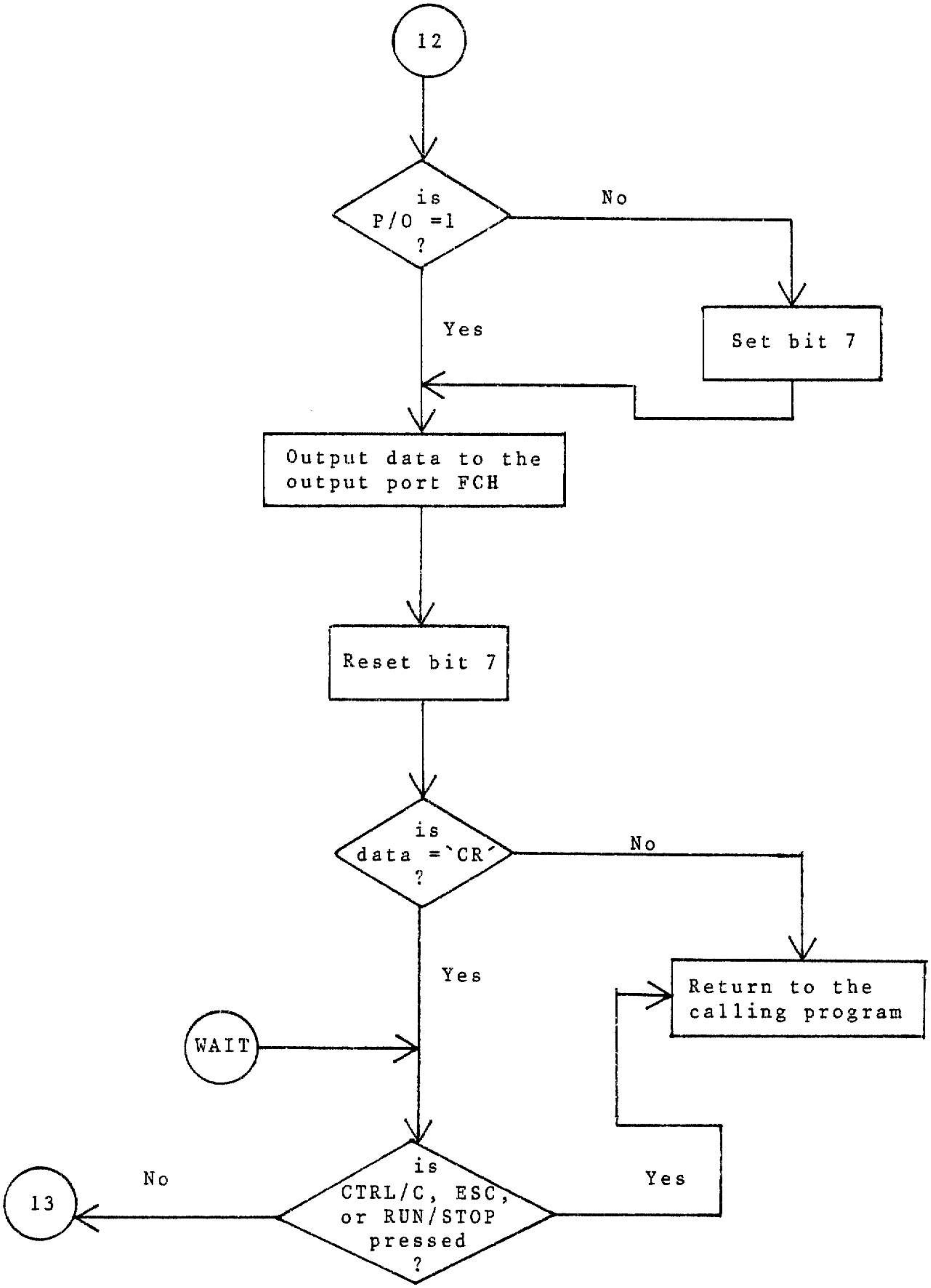
Call GETADR Subroutine
is starting address given ?
No
Yes
VLP
Print two question marks
is CTRL/C, ESC, or RUN/STOP pressed ?
Yes
No
Fill the rest of line with `>'
Call ADONE Subroutine
Load data from (DE) Register pair into A
Jump to TERMINAL
Jump to TERMINAL
Increment the content of the (DE) Register pair
10

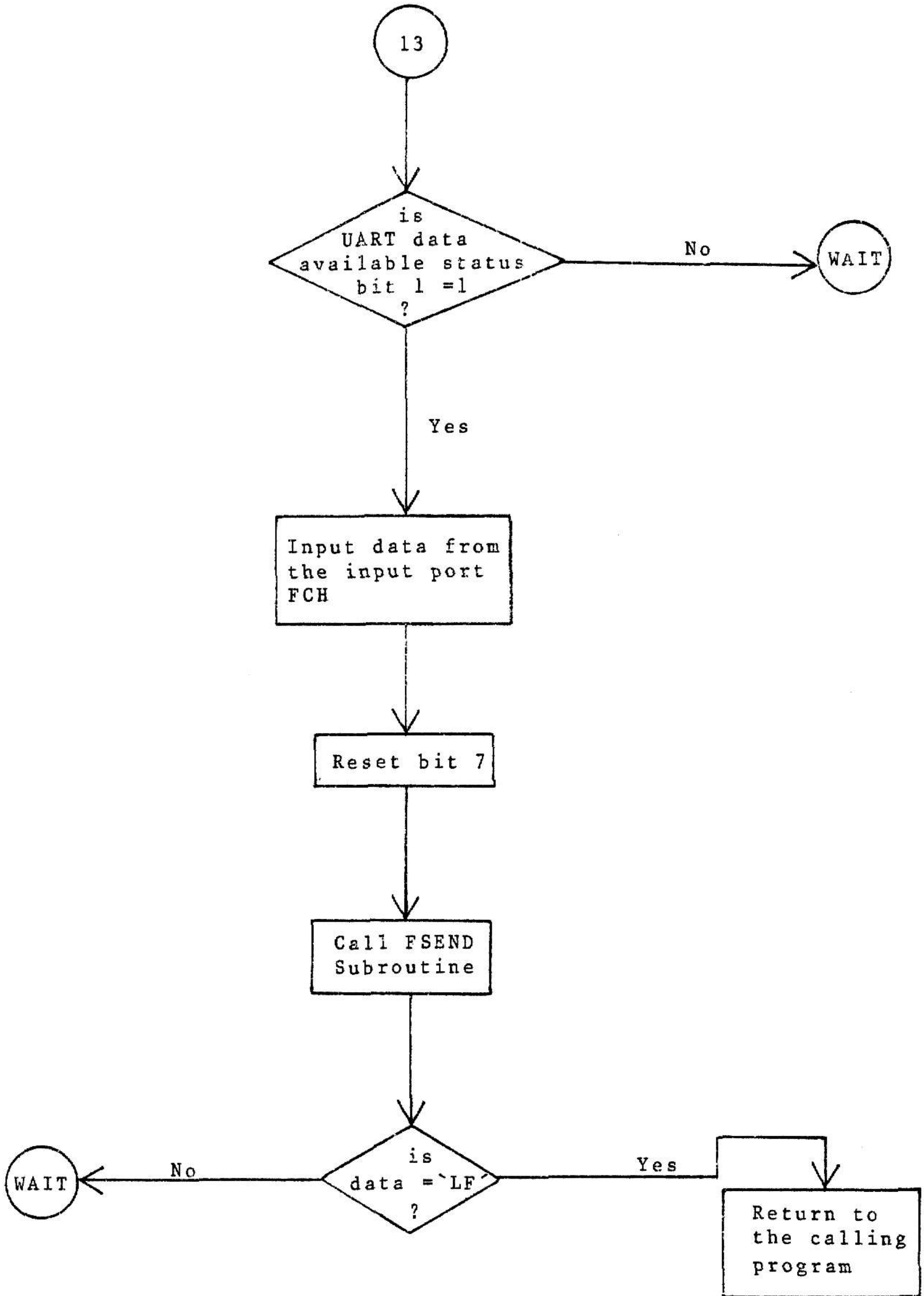

10
Reset bit 7
Load view delay into B register
Call INIFR Subroutine
is content of B =0 ?
No
Yes
is data =`MEOF´ ?
No
11
Yes
Jump to ADONE Subroutine


11
is
data = `CR`
?
Yes
No
Call CRLF
Subroutine
is
data = `LF`
?
Yes
No
Jump to
VLP
Display the data
on the Monitor
Jump to
VLP

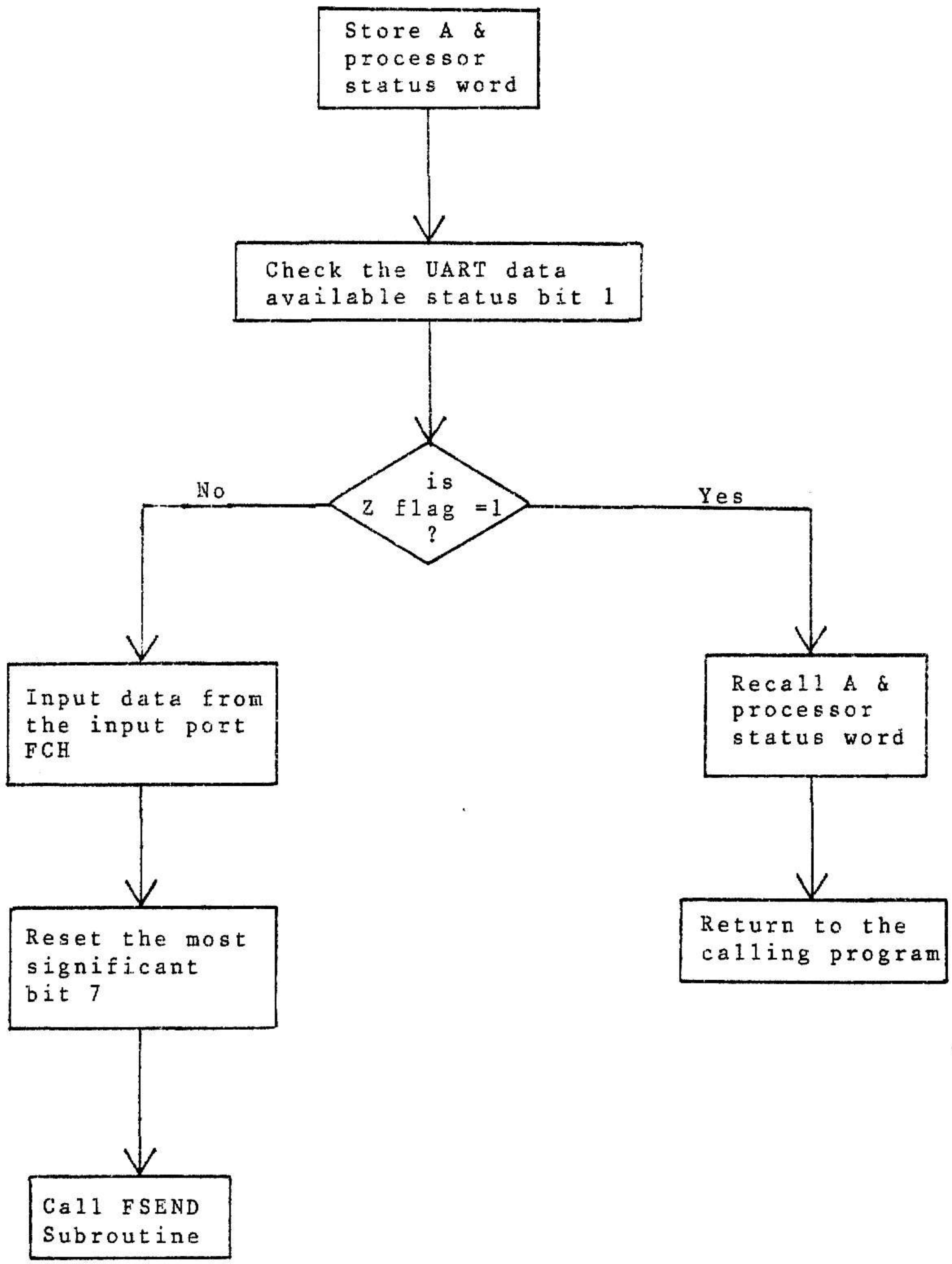
SOUT


Store the A & processor status word
Check UART transmit buffer status bit 0
Call BINIFR Subroutine
is bit 0 = `0´ ?
Yes
No
Recall A & processor status word
And data with itself to check the P/O flag
12

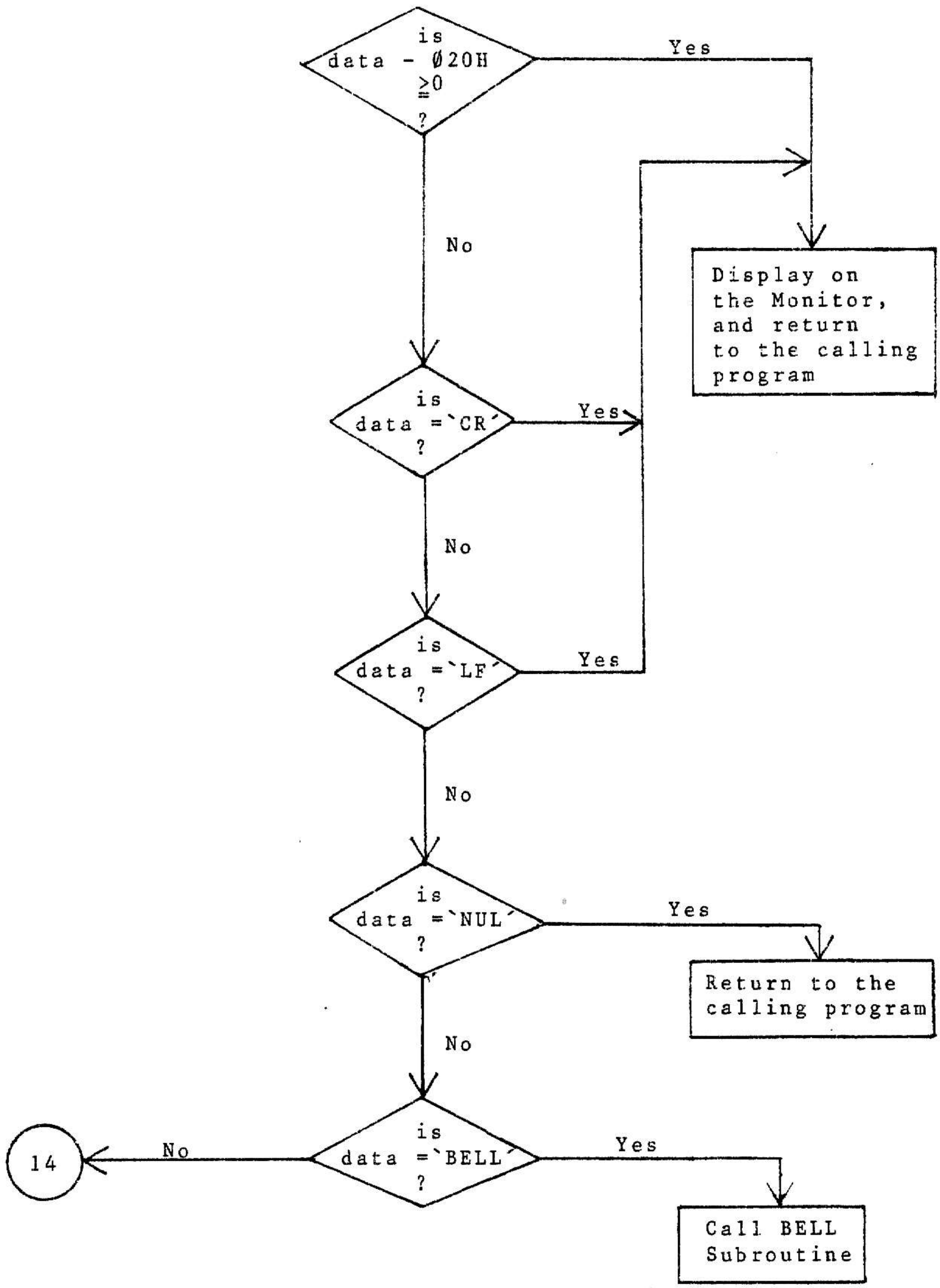

12
is
P/O =1
?
No
Set bit 7
Yes
Output data to the
output port FCH
Reset bit 7
is
data = `CR´
?
No
Return to the
calling program
Yes
WAIT
is
CTRL/C, ESC,
or RUN/STOP
pressed
?
No
Yes
13

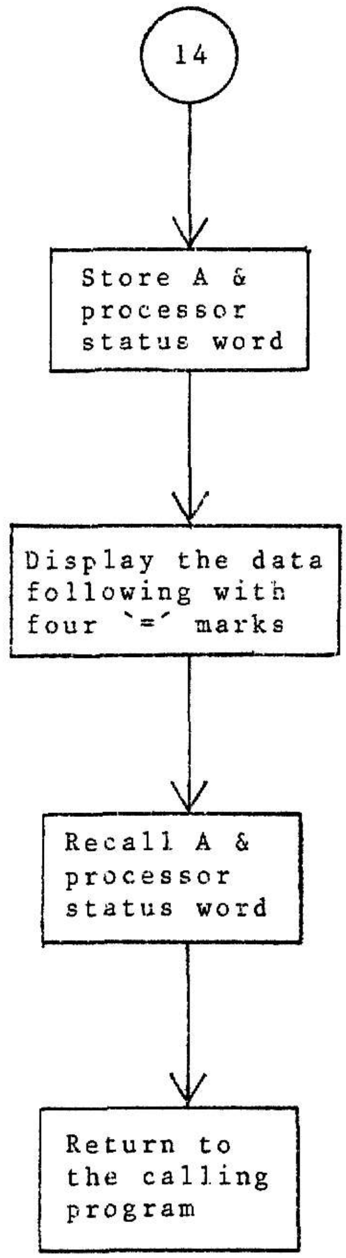

13
is UART data available status bit 1 =1 ?
No
WAIT
Yes
Input data from the input port FCH
Reset bit 7
Call FSEND Subroutine
is data =`LF´ ?
No
WAIT
Yes
Return to the calling program

BINIFR

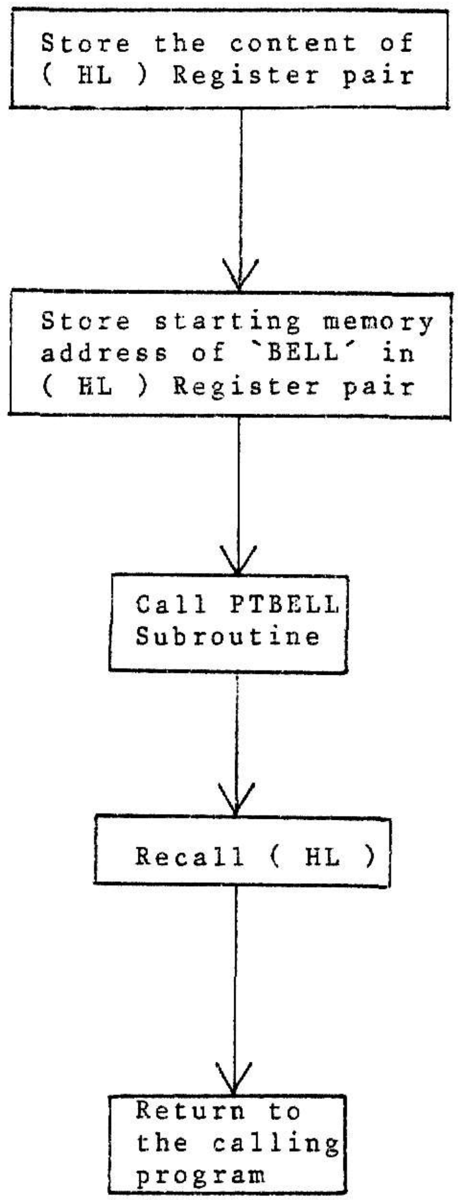

Store A &
processor
status word
Check the UART data
available status bit 1
is
Z flag =1
?
No
Yes
Input data from
the input port
FCH
Reset the most
significant
bit 7
Call FSEND
Subroutine
Recall A &
processor
status word
Return to the
calling program

FSEND

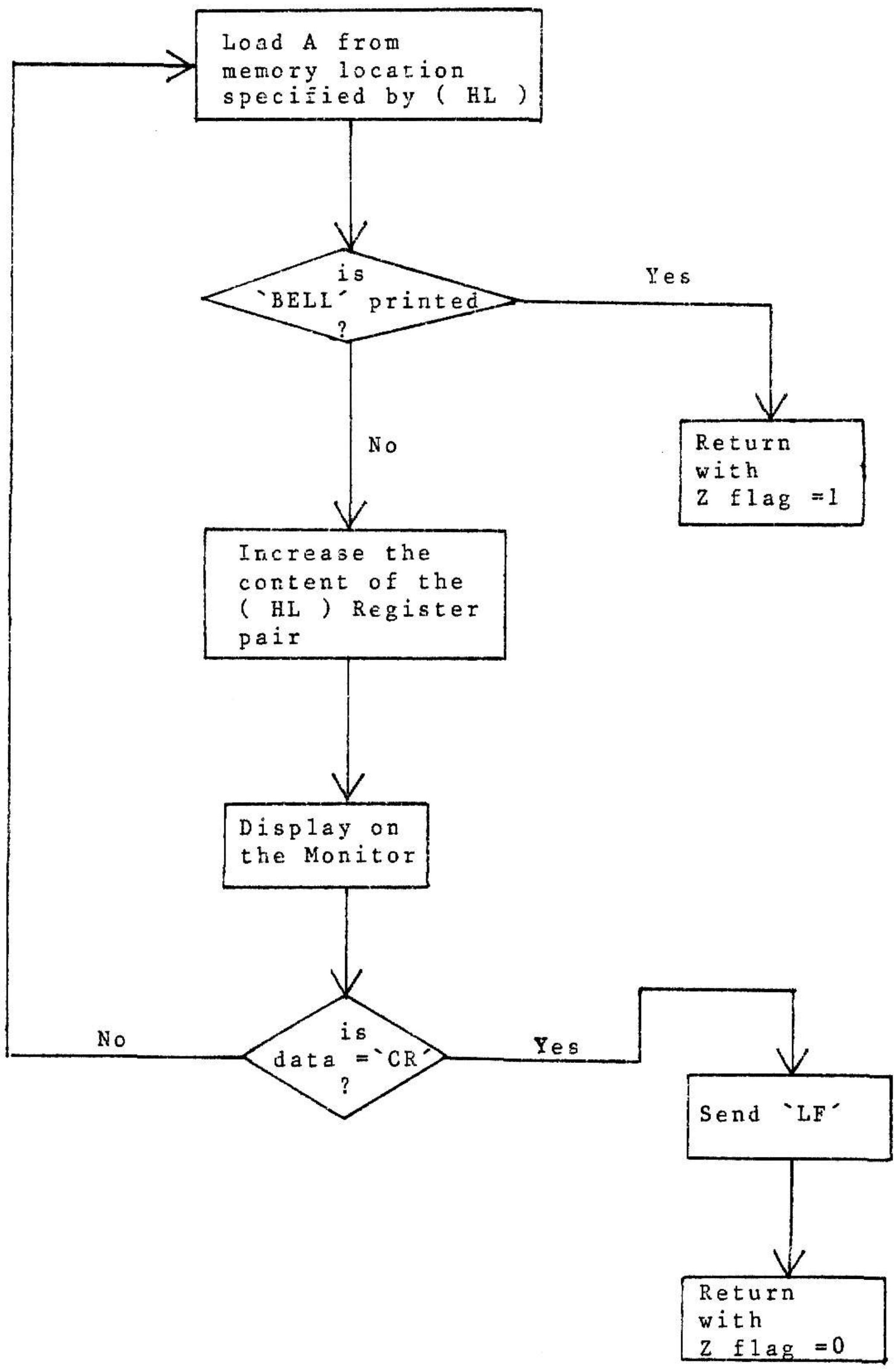

is
data - Ø20H
≥0
?
Yes
No
Display on
the Monitor,
and return
to the calling
program
is
data =`CR´
?
Yes
No
is
data =`LF´
?
Yes
No
is
data =`NUL´
?
Yes
Return to the
calling program
No
is
data =`BELL´
?
Yes
Call BELL
Subroutine
No
14


14
Store A & processor status word
Display the data following with four `=' marks
Recall A & processor status word
Return to the calling program

BELL


Store the content of (HL) Register pair
Store starting memory address of `BELL´ in (HL) Register pair
Call PTBELL Subroutine
Recall (HL)
Return to the calling program

PTBELL


Load A from
memory location
specified by (HL)
is
`BELL´ printed
?
Yes
No
Return
with
Z flag =1
Increase the
content of the
(HL) Register
pair
Display on
the Monitor
No
is
data =`CR´
?
Yes
Send `LF´
Return
with
Z flag =0

3.7 CONNECTION BETWEEN EXIDY SORCERER AND MULTI-TECH SYSTEMS FM 30 DATA COUPLER

3.7.1 DESCRIPTION, TECHNICAL SPECIFICATIONS AND OPERATION

The coupler is housed in a plastic enclosure with a metal base. All of the functional circuitry is contained on a single printed circuit board. The speaker side is marked CORD. Indicators (light emitting diodes) for power and carrier are provided in the front of the case. The power switch is located on the back of the coupler, and the Half/Full duplex switch is located on the bottom of the coupler.

Input Data Format : Serial, Binary, Asynchronous
Transmission Rate : 0 to 300 bps
Interface : EIA RS-232

Turn the power switch 'on' to start operation. The power indicator should turn 'on'. Dial 9-5374141 and listen for the high pitched answer tone. When tone is heard, place the handset firmly into the rubber cups and wait until the carrier indicator turns 'on'. This indicates that the communications link has been established and that the logon procedure can proceede. After logoff, replace the telephone handset on the telephone cradle and turn the power switch to the OFF position.

3.7.2 CONNECTING EXIDY SERIAL INTERFACE BOARD WITH FM 30 EIA RS-232 CONNECTOR

There are six pins on the FM 30 EIA RS-232 connector that must be connected with the Exidy Serial Interface connector or to itself (see Fig. 1).

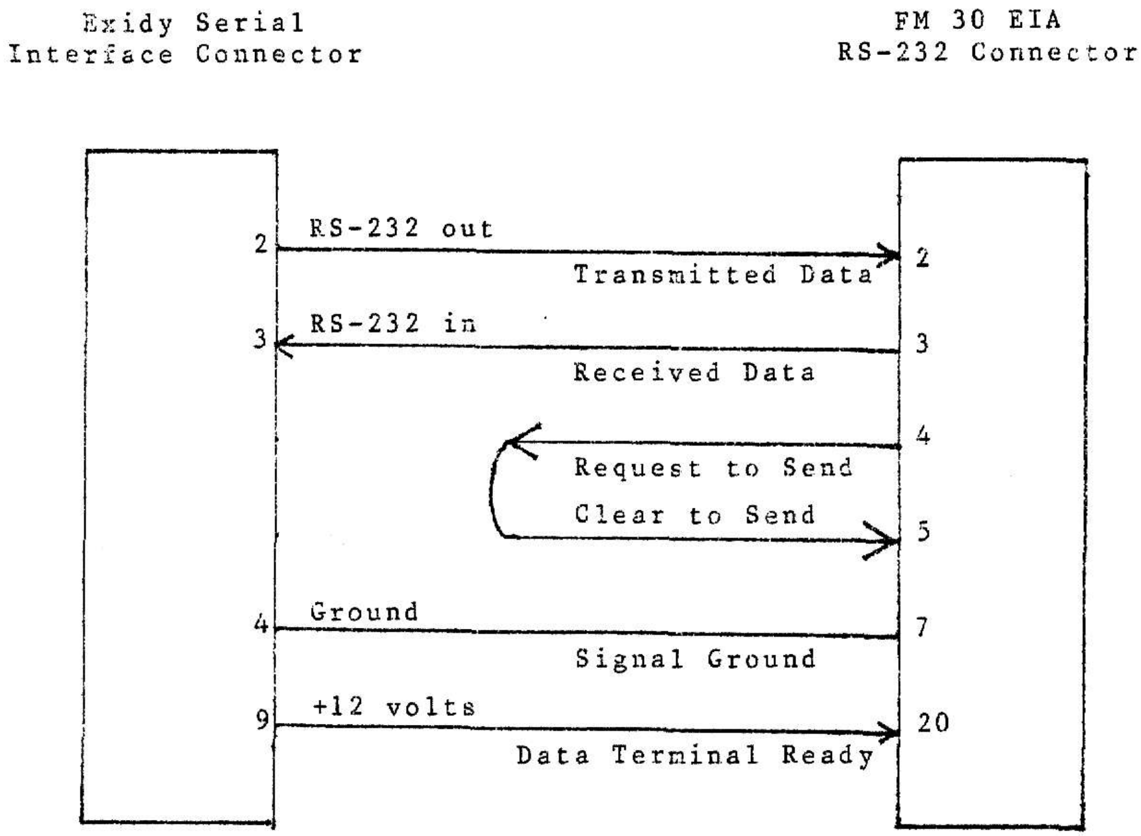


Fig. 1. Connections Between Exidy Sorcerer & FM 30 Data Coupler

1) Connect pin 2 (Transmitted Data) with Exidy pin 2 (RS-232 out).
2) Connect pin 3 (Received Data) with Exidy pin 3 (RS-232 in).
3) Connect pin 7 (Signal Ground) with Exidy pin 4 (Ground).
4) Connect pin 20 (Data Terminal Ready) with Exidy pin 9 (+12 volts).
5) Connect pin 4 (Request to Send) with pin 5 (Clear to Send).

CHAPTER 4

INTERFACING THE EXIDY SORCERER WITH THE MICROPOLIS DISK OPERATING SYSTEM (MDOS)

4.1 HARDWARE INSTALLATION

Installation of the Micropolis disk subsystem hardware requires a compatible computer system. The computer must have an S-100 bus system using an 8080, 8085, or Z80 central processor [3]. The Exidy's S-100 interface connection with the Exidy S-100 Expansion Unit is through a 50-pin male connector printed onto the edge of the logic board. It mates with a female 50-pin edge connector. The S-100 bus and storge modules in the Expansion Unit are connected to the disk controller. Hardware installation must be complete before system software configuration can begin.

4.2 PREPARING THE SYSTEM FOR USE WITH THE FLOPPY DISK SUBSYSTEM

There are four steps to be followed :

1. Make all necessary hardware connections. The necessary components are the Exidy Sorcerer primary unit, the S-100 Expansion Unit, the video display device, and the floppy disk device. Turn power ON on all units.
2. Place the Basic 4.0 System Disk into the floppy disk main frame making sure that the disk label is up and the arrow on the label is pointing into the opening in disk unit. Push the disk into the opening until it is seated; indicated by an audible click.
3. a. Load the disk by pressing down on the actuator bar, located just under the opening on the front of the main frame, slowly and firmly, as far as it will go. The actuator should remain down when the disk is

loaded.

b. To unload the disk, depress the load actuator and allow it to rise to the up position. Eject the disk by flipping the actuator bar (in the up position) up.

4. Load the PDS (Program Development Software) MDOS System into memory by typing "GO 9000" and pressing the RETURN key on keyboard [11]. When the system driver is loaded the display should read as follows :

MICROPOLIS MDOS VS. 4.0 - COPYRIGHT 1978 [10].

The system disk may now be removed and reinserted into the floppy disk main frame only as needed.

4.3 STORING PROGRAMS ON DISK*

1. Before a program can be stored on a disk, the disk must be initialized as follows :

a After loading the MICROPOLIS MDOS driver, remove the System Disk and load a blank disk into the floppy disk driver.

b Type INITØ and press the RETURN key. The monitor will respond with :

ARE YOU SURE ?

c Type Y and press the RETURN key. The prompt, >, will be displayed when the disk is initialized.

2. A program can be saved on an initialized disk by using the following command,

```
SAVE "[unit :] <filename>" <starting address>
     <ending address>
```

* The disks are 16 sector hard-sectored single-sided disks.

a The unit number designates the floppy disk unit containing the initialized disk on which the program will be stored. This number may be omitted in which case unitØ is assumed.

b The name of file may be from 1 to 10 characters long and may contain digits, letters, and the special characters < > = . / ? , - @.

c The file will be stored on at least one of the 34 available (assuming no other files are already stored) tracks on the disk. A track consists of 16 sectors of 256 bytes per sector. A program may use as many available tracks as needed but can not use any part of a track that has previously been assigned even though some space (on the track) has not been used.

d The file is written sequentially from the memory starting address through the memory ending address into full sequential records.

4.4 LOADING A PROGRAM INTO EXIDY MEMORY

The format of the LOAD command is :

LOAD "[unit :] <filename>" <starting address>

The starting address parameter is mandatory. If a starting address is not specified a SYNTAX ERROR message will be displayed. If the starting address is less than 2BØØH a LOAD ADDRESS ERROR will result because the file must be loaded beneath the MDOS application areas; that is from 2BØØH to F4ØØH [3].

4.5 THE EXIDY SORCERER MEMORY MAP AND THE MDOS MEMORY MAP

Comparing the Exidy memory map (Fig. 2) with the MDOS memory map (Fig. 3), the unused addresses in both memory maps are from 2B∅∅H to BFFFH [6]. In the range of these memory addresses, if the Z80 is instructed to read the data in a memory address which is not connected to any RAM or ROM, it will usually assume the data is FFH.

After checking the memory addresses from 2B∅∅H to BFFFH, it was found that the useful addresses (for storing files) are actually from 2CE6H to 8∅∅∅H. Therefore, the data output from the Itel stored in the Exidy RAM must start at or above 2CE6H and must end below or at 8∅∅∅H. Use the MDOS command SAVE to save the file on the initialized blank disk.

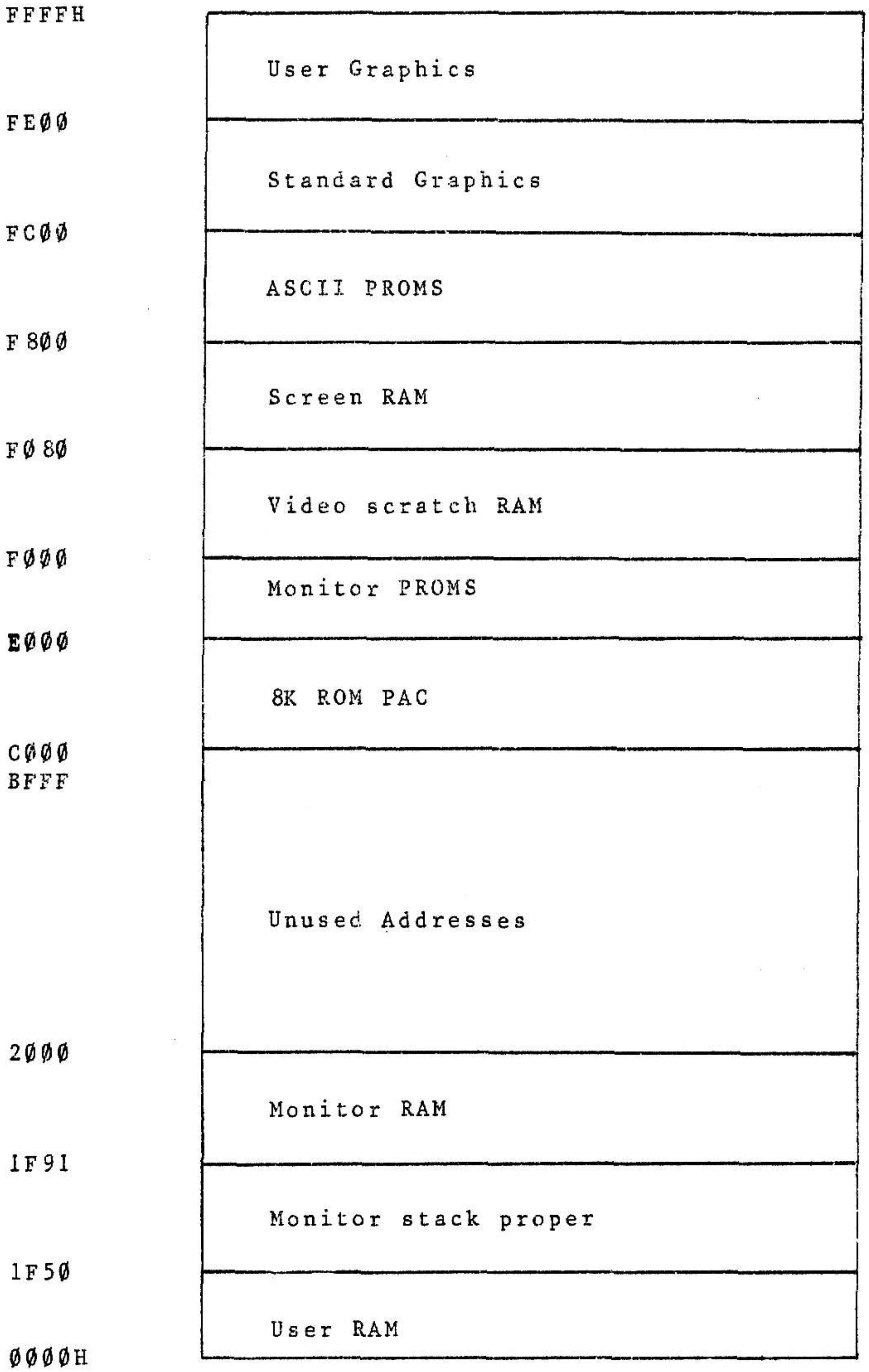


Fig. 2. Sorcerer Memory Map Using ROM PAC [5]

F8ØØH	
	Micropolis Disk Controller
F6ØØ	
	BOOTSTRAP ROM as supplied by Micropolis
F4ØØ	
	The applications program extends from here to the end of contiguous memory.
2BØØ	
	The MDOS module contains the command executive and all user callable routines not in RES.
1598	
	The RES module contains all permanently resident I/O and Disk Control Routines and associated buffers.
Ø1AØ	
	Initially used by bootstrap loader. After the system is loaded, this space is used for the system stack.
ØØ6A	
	Not used by Micropolis hardware/software
ØØØØH	

Fig. 3. MDOS Memory Map [3]

CHAPTER 5

USING THE COMPUTER ASSISTED CONTOURING SYSTEM (AD-APT)

5.1 GENERAL CONCEPT OF COMPUTER ASSISTED CONTOURING SYSTEM

The contouring system is used when the route taken by the tool is of consequence to the configuration through-out the entire tool movement. Contouring is used to control machine tools such as the lathe, milling machine, and machining center. When they remove metal continuously over the surface of a workpiece, as in the milling of a cam. Contouring may also apply to flame cutting, sawing a continuous path, welding, and to such processes as the application of adhesives. A continuous path system requires a very detailed instructional information to control the tool while it is moving along two or more machine axes simultaneously. An automatic data processing technique to program a contouring system is, therefore, very essential and useful.

5.2 COMPUTER PART PROGRAMMING

Using the IBM System/360 AD-APT to get the NC machine code of the cutter path and save it on a floppy disk. Figure 4 depicts, in general terms, the concept of the NC Operation Planning Process. An example of a program to obtain the contouring path of a cutter to mill and drill a part is shown in Appendix A. Figure 5 is the sample part cutter path. A detailed AD-APT instruction set and explanation of programming in AD-APT is contained in [12] and [13].

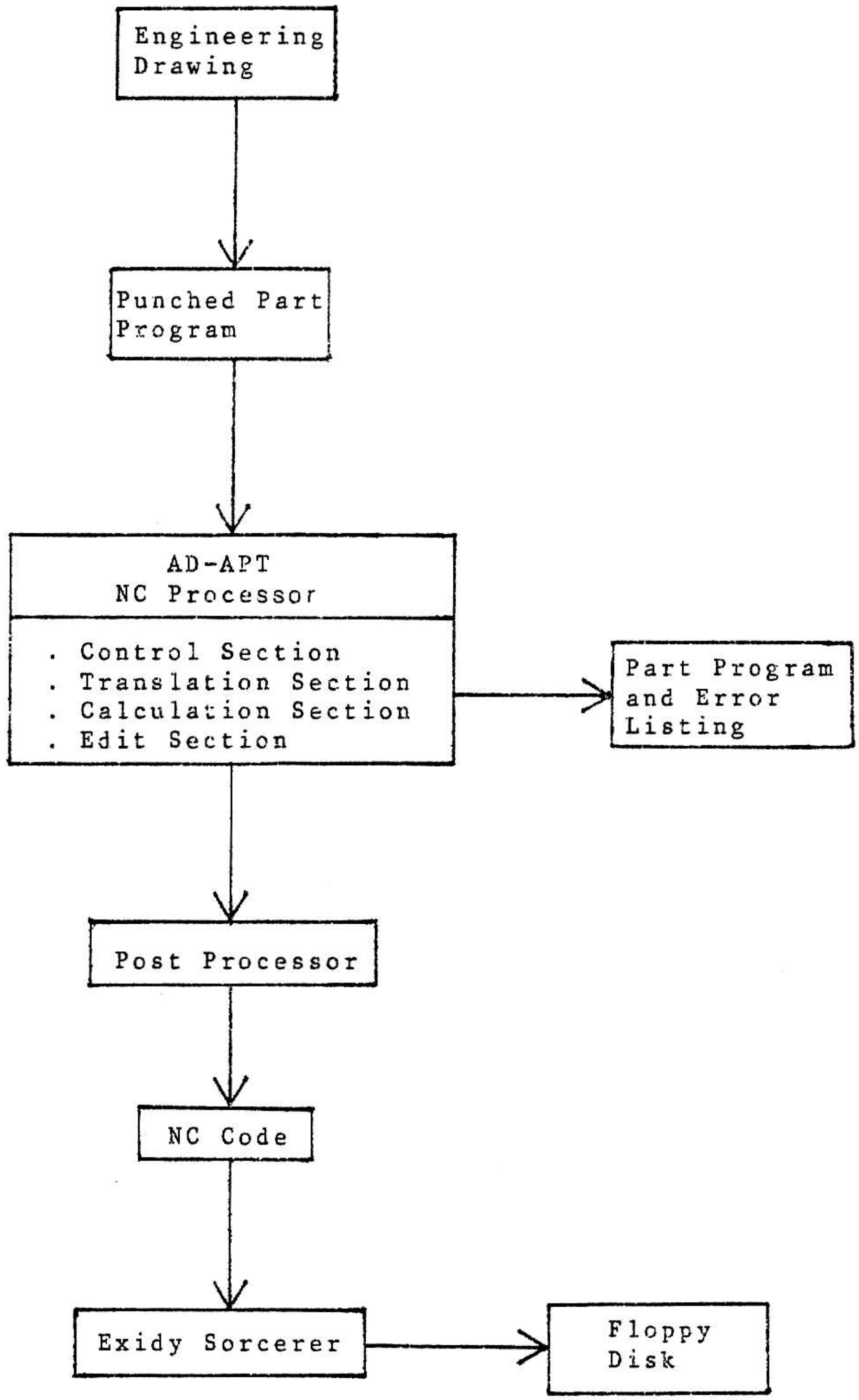


Fig. 4. The Concept of the NC Operation Planning Process

Cutter R = 0.25 "

Step 1 : cut a circular plate, R1 = 2 "

Step 2 : drill 9 holes in the plate

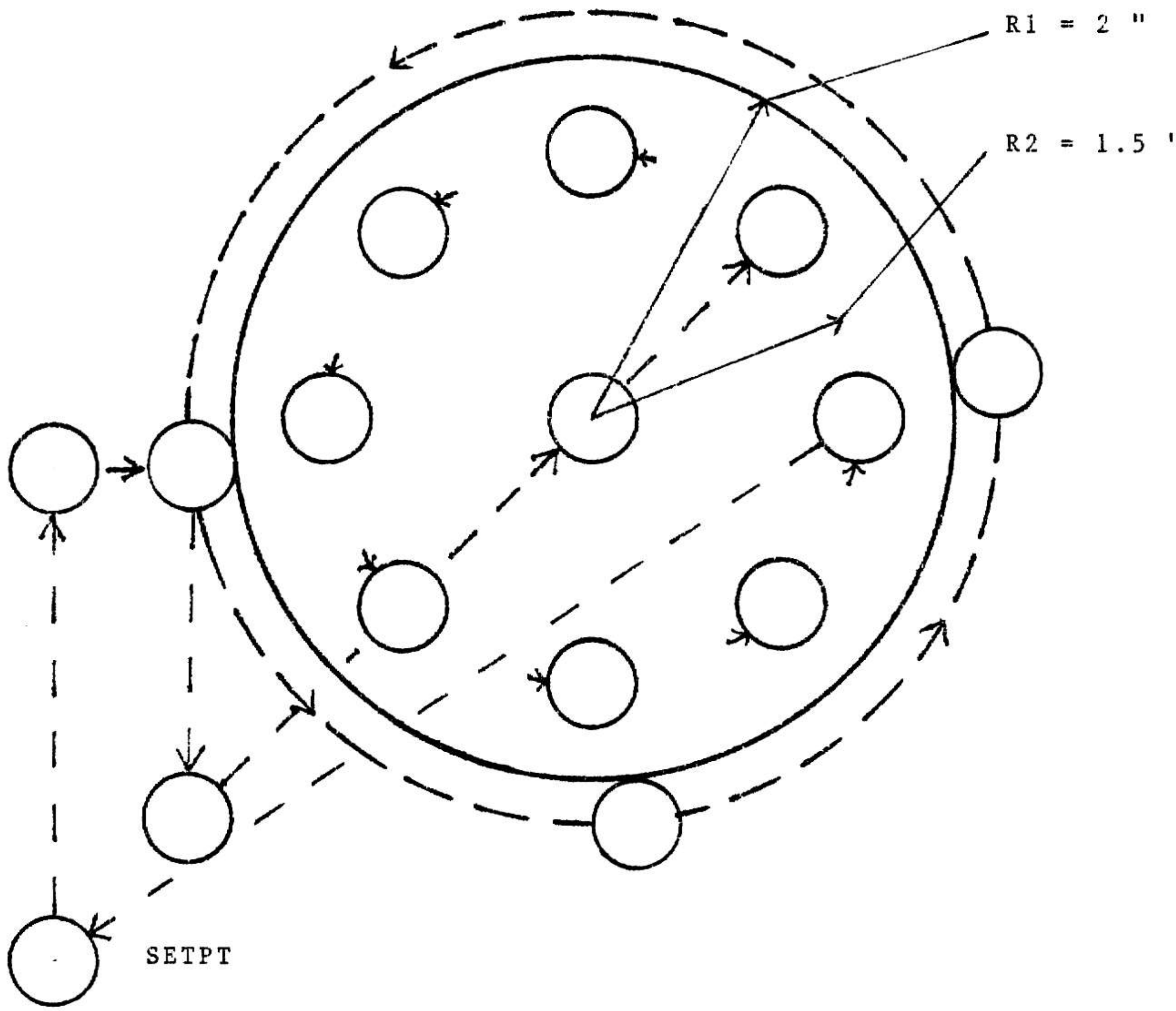


Fig. 5. Sample Part Cutter Path

CHAPTER 6

RESULTS

RESULTS

The interface of the hardware connections and the software program between Itel System and Exidy Sorcerer has been successfully completed through the modem. The output* of NC machine code from AD-APT has been also successfully loaded onto a microcomputer (Exidy Sorcerer) controlled floppy disk.

It is significant to note that the techniques applied here also can be used to load any ASCII file from the Itel System onto a floppy disk, or to save a file from floppy disk in the Itel System.

* See Appendix C & D.

CHAPTER 7

FURTHER RESEARCH

FURTHER RESEARCH

For the further research, the ASCII file of NC machine code must be converted to the EIA 244A code of NC machine by a code conversion subroutine. The interface hardware design between the Exidy Sorcerer and NC machine (made by Pratt & Whitney) must be designed in a similar way by using RS-232. Also, the digitizing pad (Houston Instruments) must be connected with the Exidy Sorcerer. The coordinates (x,y) of a drawing can be generated by the digitizing pad and stored in the RAM of Exidy Sorcerer for use in controlling microcomputer controlled machine tools.

REFERENCES

1. Pressman, R. S., and J. E. Williams, Numerical Control and Computer Aided Manufacturing, John Wiley & Sons, New York, 1977, pp. 2-4 & 152-158.

2. Olling, G., " An Introduction to Numerically Controlled Machining Systems, " The Manufacturing Productivity Education Committee, August 1977, pp. 3-5.

3. ________, Micropolis 1040/1050 S-100 Floppy Disk Subsystems User's Manual, Micropolis Corporation, Canoga Park, Ca., January 1979, pp. 2-1 thru 2-16 and p. 4-13.

4. Conrow, Kenneth, The CMS Cookbook, Computing Center of Kansas State University, August 1978, pp. 1-5 and 37-38.

5. ________, Sorcerer Technical Manual, Exidy, Inc., Sunnyvale, Ca., March 1979, pp. 9-34.

6. ________, EIA Standard RS-232-C, Engineering Department of Electronic Industries Association, August 1969, pp. 3-15.

7. Bergmann, E. E., " Use Your Exidy As a Smart Terminal. " Microcomputing, July 1980, pp. 142-146.

8. ________, CP/CMS Guide, Computing Center of Kansas State University, January 1980, p. 17.

9. Leventhal, Lance A., Z80 Assembly Language Programming, Adam Osborne & Associates, Inc., Berkely, Ca., 1979, pp. 3-40 thru 3-163.

10. Osborne, Adam, Jerry Kane, Russell Rector, and Susanna Jacobson, Z80 Programming for Logic Design, Osborne/ Mcgraw-Hill, Inc., Berkely, Ca., 1978, pp. 6-20 thru 6-145.

11. ________, Guide to Use of the Exidy Sorcerer Microcomputer Utilizing the Micropolis 1040/1050 S-100 Floppy Disk Subsystem, (unpublished-3 pages) 1979.

12. Hori, S., APT Part Programming, IIT Research Institute, Chicago, Illinois, 1967, pp. 1-79.

13. ________, System/360 AD-APT/AUTOSPOT (OS) Numerical Control Processor (360A-CN-12X) Part Programming Manual, 2nd Edition, IBM Corporation, November 1970, pp. 1.0 thru 9.6.3.

APPENDIX A

AD-APT PROGRAM

```
PARTNO APT PROGRAMMING

            CLPRNT
            CUTTER/0.25
            OUTTOL/0.005
            INTOL/0.005
            N=8
            R1=2
            R2=1.5

SETPT     = POINT/-3,-3,0

PT1       = POINT/0,0,0.5
L1        = LINE/1,0,0,-1,0,0
L2        = LINE/PT1,PERPTO,L1
C1        = CIRCLE/0,0,0,R1
            FROM/SETPT
            SPINDL/2500,CLW
            FEDRAT/20
            COOLNT/ON
            RAPID,GO/TO,L1
            RAPID,GO/TO,C1
            TLRGT,GORGT/C1,PAST,L2
            GOFWD/C1,PAST,L1
            GOFWD/C1,PAST,L1
            RAPID,GOTO/-2.5,-2.5,0
            RAPID,GOTO/-2.5,-2.5,0.5
            RAPID,GOTO/PT1
            GOTO/0,0,0
            RAPID,GOTO/PT1
            RESERV/X, 8,Y, 8,M, 8,PT, 8
            I=1
            M(I) =360/N
            LOOPST
1)          X(I) =R2*COSF(M(I))
            Y(I) =R2*SINF(M(I))
            PT(I) =POINT/X(I),Y(I),0.5
            RAPID,GOTO/PT(I)
            GOTO/X(I),Y(I),0
            RAPID,GOTO/PT(I)
            IF(I-N) 2,3,3
2)          I=I+1
            M(I) =M(I-1) + 360/N
            JUMPTO/1
3)          LOOPND
            RAPID,GOTO/-3,-3,0.5
            RAPID,GOTO/SETPT
            SPINDL/OFF
            COOLNT/OFF
            MACHIN/PRATTW,1
            FINI
```

APPENDIX B

Z80 SOFTWARE PROGRAM

APPENDIX B :

ADDRESS	OBJECT		INSTRUCTION	
0000	C35601	DUPLEX	JP	TERMINAL
0003	FDE5	KEYBD	PUSH	IY
0005	CDA2E1		CALL	GETIY
0008	C5		PUSH	BC
0009	D5		PUSH	DE
000A	E5		PUSH	HL
000B	DDE5		PUSH	IX
000D	3E81		LD	A,81H
000F	D3FE		OUT	(0FEH),A
0011	DBFE		IN	A,(0FEH)
0013	CB4F		BIT	1,A
0015	200E		JR	NZ,NORPT-$
0017	018813		LD	BC,5000
001A	0B	REPET	DEC	BC
001B	78		LD	A,B
001C	B1		OR	C
001D	20FB		JR	NZ,REPET-$
001F	FD7E6C		LD	A,(IY+6CH)
0022	C3F700		JP	FINEND
0025	AF	NORPT	XOR	A
0026	0EFE		LD	C,(0FEH)
0028	5F		LD	E,A
0029	CBFB		SET	7,E
002B	1600	MLOOP	LD	D,0
002D	42		LD	B,D

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
002E	26 80	0026		LD	H,80H
0030	DD211EEC	0027		LD	IX,INSTBL
0034	ED61	0028	SLOOP	OUT	(C),H
0036	2E01	0029		LD	L,1
0038	ED78	0030	BLOOP	IN	A,(C)
003A	A5	0031		AND	L
003B	C2C900	0032		JP	NZ,ABIT1
003E	E5	0033		PUSH	HL
003F	210001	0034		LD	HL,100H
0042	2D	0035	DEBOUN	DEC	L
0043	20FD	0036		JR	NZ,DEBOUN-$
0045	25	0037		DEC	H
0046	20FA	0038		JR	NZ,DEBOUN-$
0048	E1	0039		POP	HL
0049	ED78	0040		IN	A,(C)
004B	A5	0041		AND	L
004C	C2C900	0042		JP	NZ,ABIT1
004F	DD5600	0043		LD	D,(IX+0)
0052	CB7A	0044		BIT	7,D
0054	281A	0045		JR	Z,CODED-$
0056	CB62	0046		BIT	4,D
0058	2802	0047		JR	Z,NONGRA-$
005A	CBF0	0048		SET	6,B
005C	CB5A	0049	NONGRA	BIT	3,D
005E	2802	0050		JR	Z,NONCON-$
0060	CBE8	0051		SET	5,B
0062	CB52	0052	NONCON	BIT	2,D

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
0064	2802	0053		JR	Z,NONSHI-$
0066	CBE0	0054		SET	4,B
0068	CB4A	0055	NONSHI	BIT	1,D
006A	285D	0056		JR	Z,ABIT1-$
006C	CBD8	0057		SET	3,B
006E	1859	0058		JR	ABIT1-$
0070	E5	0059	CODED	PUSH	HL
0071	D5	0060		PUSH	DE
0072	DDE5	0061		PUSH	IX
0074	E1	0062		POP	HL
0075	111EEC	0063		LD	DE,INSTBL
0078	B7	0064		OR	A
0079	ED52	0065		SBC	HL,DE
007B	D1	0066		POP	DE
007C	CB70	0067		BIT	6,B
007E	2815	0068		JR	Z,NOGRAP-$
0080	CB72	0069		BIT	6,D
0082	2811	0070		JR	Z,NOGRAP-$
0084	D5	0071		PUSH	DE
0085	116EEC	0072		LD	DE,GRATBL
0088	19	0073		ADD	HL,DE
0089	7E	0074		LD	A,(HL)
008A	CBFF	0075		SET	7,A
008C	D1	0076		POP	DE
008D	CB60	0077		BIT	4,B
008F	2826	0078		JR	Z,FINOP-$

ADDRESS	OBLECT	LINE NO.		INSTRUCTION	
0091	CBF7	0079		SET	6,A
0093	1822	0080		JR	FINOP-$
0095	D5	0081	NOGRAP	PUSH	DE
0096	CB68	0082		BIT	5,B
0098	2805	0083		JR	Z,SKIP1-$
009A	11BEEC	0084		LD	DE,CONTBL
009D	1815	0085		JR	SKIP4-$
009F	CB60	0086	SKIP1	BIT	4,B
00A1	2805	0087		JR	Z,SKIP2-$
00A3	110EED	0088		LD	DE,SHITBL
00A6	180C	0089		JR	SKIP4-$
00A8	CB58	0090	SKIP2	BIT	3,B
00AA	2805	0091		JR	Z,SKIP3-$
00AC	115EED	0092		LD	DE,SLOTBL
00AF	1803	0093		JR	SKIP4-$
00B1	11AEED	0094	SKIP3	LD	DE,UNSTBL
00B4	19	0095	SKIP4	ADD	HL,DE
00B5	D1	0096		POP	DE
00B6	7E	0097		LD	A,(HL)
00B7	CBE3	0098	FINOP	SET	4,E
00B9	CBDB	0099		SET	3,E
00BB	CBD3	0100		SET	2,E
00BD	CBBB	0101		RES	7,E
00BF	E1	0102		POP	HL
00C0	F5	0103		PUSH	AF
00C1	ED78	0104	WAITK	IN	A,(C)

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
00C3	A5	0105		AND	L
00C4	28FB	0106		JR	Z,WAITK-$
00C6	F1	0107		POP	AF
00C7	180B	0108		JR	BITEND-$
00C9	CB05	0109	ABIT1	RLC	L
00CB	3E20	0110		LD	A,20H
00CD	BD	0111		CP	L
00CE	2002	0112		JR	NZ,SKIP5-$
00D0	CBD3	0113		SET	2,E
00D2	DD23	0114	SKIP5	INC	IX
00D4	CB53	0115	BITEND	BIT	2,E
00D6	CA3800	0116		JP	Z,BLOOP
00D9	CB93	0117		RES	2,E
00DB	CB5B	0118		BIT	3,E
00DD	2007	0119		JR	NZ,SECEND-$
00DF	24	0120		INC	H
00E0	3E90	0121		LD	A,90H
00E2	BC	0122		CP	H
00E3	C23400	0123		JP	NZ,SLOOP
00E6	CB9B	0124	SECEND	RES	3,E
00E8	37	0125		SCF	
00E9	CB7B	0126		BIT	7,E
00EB	2803	0127		JR	Z,SKIP6-$
00ED	AF	0128		XOR	A
00EE	CBE3	0129		SET	4,E

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
00F0	CB63	0130	SKIP6	BIT	4,E
00F2	2003	0131		JR	NZ,FINEND-$
00F4	C32B00	0132		JR	MLOOP
00F7	DDE1	0133	FINEND	POP	IX
00F9	E1	0134		POP	HL
00FA	D1	0135		POP	DE
00FB	C1	0136		POP	BC
00FC	B7	0137		OR	A
00FD	2803	0138		JR	Z,KEYRET-$
00FF	FD776C	0139		LD	(IY+6CH),A
0102	FDE1	0140	KEYRET	POP	IY
0104	F5	0141	INIFR	PUSH	AF
0105	DBFD	0142		IN	A,(0FDH)
0107	CB4F	0143		BIT	1,A
0109	2002	0144		JR	NZ,SHOW-$
010B	F1	0145		POP	AF
010C	C9	0146		RET	
010D	DBFC	0147	SHOW	IN	A,(0FCH)
010F	CBBF	0148		RES	7,A
0111	CD0CE0	0149		CALL	SEND
0114	F1	0150		POP	AF
0115	C9	0151		RET	
0116	F5	0152	SNOUT	PUSH	AF
0117	DBFD	0153	WBE	IN	A,(0FDH)
0119	CB47	0154		BIT	0,A

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
011B	CD0401	0155		CALL	INIFR
011E	28F7	0156		JR	Z,WBE-$
0120	F1	0157		POP	AF
0121	A7	0158		AND	A
0122	EA2701	0159		JP	PE,OUTOK
0125	CBFF	0160		SET	7,A
0127	D3F7	0161	OUTOK	OUT	(OFCH),A
0129	C9	0162		RET	
012A	CDC601	0163	VIEWING	CALL	GETADR
012D	2879	0164		JR	Z,QQ-$
012F	CD15E0	0165	VLP	CALL	QUIKCK
0132	C2BE01	0166		JP	NZ,ADONE
0135	1A	0167		LD	A,(DE)
0136	13	0168		INC	DE
0137	CBBF	0169		RES	7,A
0139	06FF	0170		LD	B,OFFH
013B	CD0401	0171	VDLYLP	CALL	INIFR
013E	10FB	0172		DJNZ	VDLYLP-$
0140	FE00	0173		CP	OOH
0142	287A	0174		JR	Z,ADONE-$
0144	FEOD	0175		CP	CR
0146	2809	0176		JR	Z,VCRLF-$
0148	FEOA	0177		CP	LF
014A	28E3	0178		JR	Z,VLP-$
014C	CD0CE0	0179		CALL	SEND

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
014F	18DE	0180		JR	VLP
0151	CD05E2	0181	VCRLF	CALL	CRLF
0154	18D9	0182		JR	VLP
0156	CDA2E1	0183	TERMINAL	CALL	GETIY
0159	3E80	0184		LD	A,80H
015B	FD7745	0185		LD	(IY+45H),A
015E	FD773D	0186		LD	(IY+3DH),A
0161	CD0300	0187		CALL	KEYBD
0164	28F0	0188		JR	Z,TERMINAL-$
0166	CB7F	0189		BIT	7,A
0168	2005	0190		JR	NZ,SMARTS-$
016A	CD1601	0191		CALL	SNOUT
016D	18E7	0192		JR	TERMINAL-$
016F	CDA2E1	0193	SMARTS	CALL	GETIY
0172	FEEF	0194		CP	0FEH
0174	20E0	0195		JR	NZ,TERMINAL-$
0176	210300	0196		LD	HL,KEYBD
0179	FD7541	0197		LD	(IY+41H),L
017C	FD7442	0198		LD	(IY+42H),H
017F	FD364300	0199		LD	(IY+43H),0
0183	3E3C	0200		LD	A,`<'
0185	0603	0201		LD	B,3
0187	CD0CE0	0202	PLP	CALL	SEND
018A	10FB	0203		DJNZ	PLP-$
018C	CD3AE1	0204		CALL	MINPUT

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
018F	FDE5	0205		PUSH	IY
0191	E1	0206		POP	HL
0192	CD25E2	0207		CALL	FTOKEN
0195	2819	0208		JR	Z,DONE-$
0197	FE53	0209		CP	\`S´
0199	2832	0210		JR	Z,SENDING-$
019B	FE52	0211		CP	\`R´
019D	2859	0212		JR	Z,RECEIVING-$
019F	FE56	0213		CP	\`V´
01A1	2887	0214		JR	Z,VIEWING-$
01A3	FE58	0215		CP	\`X´
01A5	CA00E0	0216		JP	Z,0E000H
01A8	3E3F	0217	QQ	LD	A,\`?´
01AA	CD0CE0	0218		CALL	SEND
01AD	CD0CE0	0219		CALL	SEND
01B0	3E3E	0220	DONE	LD	A,\`>´
01B2	0640	0221		LD	B,64
01B4	CD0CE0	0222	DONELP	CALL	SEND
01B7	10FB	0223		DJNZ	DONELP-$
01B9	CD05E2	0224		CALL	CRLF
01BC	1898	0225		JR	TERMINAL-$
01BE	CD05E2	0226	ADONE	CALL	CRLF
01C1	CDE8E1	0227		CALL	MUNCNV
01C4	18EA	0228		JR	DONE-$
01C6	CD2FE2	0229	GETADR	CALL	GTOKN2

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
01C9	C8	0230		RET	Z
01CA	C33DE2	0231		JP	MCONVA
01CD	CDC601	0232	SENDING	CALL	GETADR
01D0	28D6	0233		JR	Z,QQ-$
01D2	CD15E0	0234	SLP	CALL	QUIKCK
01D5	20E7	0235		JR	NZ,ADONE-$
01D7	1A	0236		LD	A,(DE)
01D8	13	0237		INC	DE
01D9	0101FF	0238		LD	BC,01FFH
01DC	CD2F02	0239	SDLYLP	CALL	BINIFR
01DF	10FB	0240		DJNZ	SDLYLP-$
01E1	0D	0241		DEC	C
01E2	20F8	0242		JR	NZ,SDLYLP-$
01E4	FE0A	0243		CP	LF
01E6	28EA	0244		JR	Z,SLP-$
01E8	FE00	0245		CP	00H
01EA	2805	0246		JR	Z,OUTEND-$
01EC	CD4102	0247		CALL	SOUT
01EF	18E1	0248		JR	SLP-$
01F1	3E1A	0249	OUTEND	LD	A,CCEOF
01F3	CD4102	0250		CALL	SOUT
01F6	18C6	0251		JR	ADONE-$
01F8	CDC601	0252	RECEIVING	CALL	GETADR
01FB	28AB	0253		JR	Z,QQ-$
01FD	3E0D	0254		LD	A,`CR´

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
01FF	CD1601	0255		CALL	SNOUT
0202	CD15E0	0256	RELP	CALL	QUIKCK
0205	2019	0257		JR	NZ,INDONE-$
0207	DBFD	0258		IN	A,(0FDH)
0209	CB4F	0259		BIT	1,A
020B	28F5	0260		JR	Z,RELP-$
020D	DBFC	0261		IN	A,(0FCH)
020F	CBBF	0262		RES	7,A
0211	FE00	0263		CP	00H
0213	28ED	0264		JR	Z,RELP-$
0215	FE0A	0265		CP	LF
0217	2800	0266		JR	Z,INMEM-$
0219	12	0267		LD	(DE),A
021A	13	0268	INMEM	INC	DE
021B	CD7002	0269		CALL	FSEND
021E	18E2	0270		JR	RELP-$
0220	3E0D	0271	INDONE	LD	A,CR
0222	12	0272		LD	(DE),A
0223	13	0273		INC	DE
0224	3E0A	0274		LD	A,LF
0226	12	0275		LD	(DE),A
0227	13	0276		INC	DE
0228	3E00	0277		LD	A,00H
022A	12	0278		LD	(DE),A

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
022B	13	0279		INC	DE
022C	C3BE01	0280		JP	ADONE
022F	F5	0281	BINIFR	PUSH	AF
0230	DBFD	0282		IN	A,(OFDH)
0232	CB4F	0283		BIT	1,A
0234	2002	0284		JR	NZ,BSHOW-$
0236	F1	0285	SNDRT	POP	AF
0237	C9	0286		RET	
0238	DBFC	0287	BSHOW	IN	A,(OFCH)
023A	CBBF	0288		RES	7,A
023C	CD7002	0289		CALL	FSEND
023F	F1	0290		POP	AF
0240	C9	0291		RET	
0241	F5	0292	SOUT	PUSH	AF
0242	DBFD	0293	BWBE	IN	A,(OFDH)
0244	CB47	0294		BIT	0,A
0246	CD2F02	0295		CALL	BINIFR
0249	28F7	0296		JR	Z,BWBE-$
024B	F1	0297		POP	AF
024C	A7	0298		AND	A
024D	EA5202	0299		JP	PE,BOUTOK
0250	CBFF	0300		SET	7,A
0252	D3FC	0301	BOUTOK	(OFCH),A	
0254	CBBF	0302		RES	7,A
0256	FE0D	0303		CP	CR

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
0258	C0	0304		RET	NZ
0259	CD15E0	0305	WAIT	CALL	QUIKCK
025C	C0	0306		RET	NZ
025D	DBFD	0307		IN	A,(0FDH)
025F	CB4F	0308		BIT	1,A
0261	28F6	0309		JR	Z,WAIT-$
0263	DBFC	0310		IN	A,(0FCH)
0265	CBBF	0311		RES	7,A
0267	CD7002	0312		CALL	FSEND
026A	FE0A	0313		CP	LF
026C	C8	0314		RET	Z
026D	18EA	0315		JR	WAIT-$
026F	C9	0316		RET	
0270	FE20	0317	FSEND	CP	SP
0272	D20CE0	0318		JP	NC,SEND
0275	FE0D	0319		CP	CR
0277	CA0CE0	0320		JP	Z,SEND
027A	FE0A	0321		CP	LF
027C	CA0CE0	0322		JP	Z,SEND
027F	FE00	0323		CP	NUL
0281	C8	0324		RET	Z
0282	FE07	0325		CP	07H
0284	2816	0326		JR	Z,BELL-$
0286	F5	0327		PUSH	AF
0287	CDEDE1	0328		CALL	0E1EDH

ADDRESS	OBJECT	LINE NO.		INSTRUCTION	
028A	3E3D	0329		LD	A,`=´
028C	0604	0330		LD	B,4
028E	CD0CE0	0331	SMR	CALL	SEND
0291	10FB	0332		DJNZ	SMR-$
0293	F1	0333		POP	Af
0294	C9	0334		RET	
0295	3C	0335	DF `<´		
0296	42	0336	DF `B´		
0297	45	0337	DF `E´		
0298	4C	0338	DF `L´		
0299	4C	0339	DF `L´		
029A	3E	0340	DF `>´		
029B	00	0341	DF 00		
029C	E5	0342	BELL	PUSH	HL
029D	219502	0343		LD	HL,0295
02A0	CDBAE1	0344		CALL	PTBELL
02A3	E1	0345		POP	HL
02A4	C9	0346		RET	

********* END *********

APPENDIX C

OUTPUT OF AD-APT

APPENDIX C :

```
TOF:
N1G1G91F300
N2M33
N3F2000
N4M7
N5G0
N6Y28750
N7G0
N8X8768Y365
N9G1
N10X136Y-2057
N11X962Y-4010
N12X1717Y-3748
N13X2407Y-3348
N14X3006Y-2822
N15X3494Y-2189
N16X3850Y-1475
N17X4062Y-706
N18X2849Y62
N19X2057Y136
N20X3992Y1032
N21X3719Y1781
N22X3305Y2464
N23X2769Y3055
N24X2129Y3531
N25X1408Y3875
N26X635Y4074
N27X-97Y2481
N28X-125Y2058
N29X-1032Y3992
N30X-1781Y3718
N31X-2465Y3306
N32X-3055Y2769
N33X-3531Y2128
N34X-3875Y1408
N35X-4074Y636
N36X-4121Y-162
N37X-4011Y-952
N38X-3753Y-1708
N39X-3354Y-2398
N40X-2829Y-3000
N41X-2198Y-3488
N42X-1484Y-3847
N43X-716Y-4060
N44X56Y-2901
N45G0
N46X-3820Y-23750
```

```
N47G0
N48Z5000
N49G0
N50X25001Y25001
N51G1
N52Z-5000
N53G0
N54Z5000
N55G0
N56X10607Y10607
N57G1
N58Z-5000
N59G0
N60Z5000
N61G0
N62X-10607Y4393
N63G1
N64Z-5000
N65G0
N66Z5000
N67G0
N68X-10608Y-4393
N69G1
N70Z-5000
N71G0
N72Z5000
N73G0
N74X-4393Y-10607
N75G1
N76Z-5000
N77G0
N78Z5000
N79G0
N80X4393Y-10608
N81G1
N82Z-5000
N83G0
N84Z5000
N85G0
N86X10607Y-4393
N87G1
N88Z-5000
N89G0
N90Z5000
N91G0
N92X10608Y4393
N93G1
```

```
N94Z-5000
N95G0
N96Z5000
N97G0
N98X4393Y10607
N99G1
N100Z-5000
N101G0
N102Z5000
N103G0
N104X-45001Y-30000
N105G0
N106Z-5000
N107G1
N108M30
N109M9
N110M2


R;
```

APPENDIX D

ASCII FILES OF AD-APT OUTPUT

APPENDIX D :

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3000:	0D	0D	0A	20	0D	0A	7F	7F	4E	31	47	31	47	39	31	46
3010:	33	30	30	0D	0A	7F	7F	4E	32	4D	33	33	0D	0A	7F	7F
3020:	4E	33	46	32	30	30	30	0D	0A	7F	7F	4E	34	4D	37	0D
3030:	0A	7F	7F	4E	35	47	30	0D	0A	7F	7F	4E	36	59	32	38
3040:	37	35	30	0D	0A	7F	7F	4E	37	47	30	0D	0A	7F	7F	4E
3050:	38	58	38	37	36	38	59	33	36	35	0D	0A	7F	7F	4E	39
3060:	47	31	0D	0A	7F	7F	4E	31	30	58	31	33	36	59	2D	32
3070:	30	35	37	0D	0A	7F	7F	4E	31	31	58	39	36	32	59	2D
3080:	34	30	31	30	0D	0A	7F	7F	4E	31	32	58	31	37	31	37
3090:	59	2D	33	37	34	38	0D	0A	7F	7F	4E	31	33	58	32	34
30A0:	30	37	59	2D	33	33	34	38	0D	0A	7F	7F	4E	31	34	58
30B0:	33	30	30	36	59	2D	32	38	32	32	0D	0A	7F	7F	4E	31
30C0:	35	58	33	34	39	34	59	2D	32	31	38	39	0D	0A	7F	7F
30D0:	4E	31	36	58	33	38	35	30	59	2D	31	34	37	35	0D	0A
30E0:	7F	7F	4E	31	37	58	34	30	36	32	59	2D	37	30	36	0D
30F0:	0A	7F	7F	4E	31	38	58	32	38	34	39	59	36	32	0D	0A

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3100:	7F	7F	4E	31	39	58	32	30	35	37	59	31	33	36	0D	0A
3110:	7F	7F	4E	32	30	58	33	39	39	32	59	31	30	33	32	0D
3120:	0A	7F	7F	4E	32	31	58	33	37	31	39	59	31	37	38	31
3130:	0D	0A	7F	7F	4E	32	32	58	33	33	30	35	59	32	34	36
3140:	34	0D	0A	7F	7F	4E	32	33	58	32	37	36	39	59	33	30
3150:	35	35	0D	0A	7F	7F	4E	32	34	58	32	31	32	39	59	33
3160:	35	33	31	0D	0A	7F	7F	4E	32	35	58	31	34	30	38	59
3170:	33	38	37	35	0D	0A	7F	7F	4E	32	36	58	36	33	35	59
3180:	34	30	37	34	0D	0A	7F	7F	4E	32	37	58	2D	39	37	59
3190:	32	34	38	31	0D	0A	7F	7F	4E	32	38	58	2D	31	32	35
31A0:	59	32	30	35	38	0D	0A	7F	7F	4E	32	39	58	2D	31	30
31B0:	33	32	59	33	39	39	32	0D	0A	7F	7F	4E	33	30	58	2D
31C0:	31	37	38	31	59	33	37	31	38	0D	0A	7F	7F	4E	33	31
31D0:	58	2D	32	34	36	35	59	33	33	30	36	0D	0A	7F	7F	4E
31E0:	33	32	58	2D	33	30	35	35	59	32	37	36	39	0D	0A	7F
31F0:	7F	4E	33	33	58	2D	33	35	33	31	59	32	31	32	38	0D

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3200:	0A	7F	7F	4E	33	34	58	2D	33	38	37	35	59	31	34	30
3210:	38	0D	0A	7F	7F	4E	33	35	58	2D	34	30	37	34	59	36
3220:	33	36	0D	0A	7F	7F	4E	33	36	58	2D	34	31	32	31	59
3230:	2D	31	36	32	0D	0A	7F	7F	4E	33	37	58	2D	34	30	31
3240:	31	59	2D	39	35	32	0D	0A	7F	7F	4E	33	38	58	2D	33
3250:	37	35	33	59	2D	31	37	30	38	0D	0A	7F	7F	4E	33	39
3260:	58	2D	33	33	35	34	59	2D	32	33	39	38	0D	0A	7F	7F
3270:	4E	34	30	58	2D	32	38	32	39	59	2D	33	30	30	30	0D
3280:	0A	7F	7F	4E	34	31	58	2D	32	31	39	38	59	2D	33	34
3290:	38	38	0D	0A	7F	7F	4E	34	32	58	2D	31	34	38	34	59
32A0:	2D	33	38	34	37	0D	0A	7F	7F	4E	34	33	58	2D	37	31
32B0:	36	59	2D	34	30	36	30	0D	0A	7F	7F	4E	34	34	58	35
32C0:	36	59	2D	32	39	30	31	0D	0A	7F	7F	4E	34	35	47	30
32D0:	0D	0A	7F	7F	4E	34	36	58	2D	33	38	32	30	59	2D	32
32E0:	33	37	35	30	0D	0A	7F	7F	4E	34	37	47	30	0D	0A	7F
32F0:	7F	4E	34	38	5A	35	30	30	30	0D	0A	7F	7F	4E	34	39

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3300:	47	30	0D	0A	7F	7F	4E	35	30	58	32	35	30	30	31	59
3310:	32	35	30	30	31	0D	0A	7F	7F	4E	35	31	47	31	0D	0A
3320:	7F	7F	4E	35	32	5A	2D	35	30	30	30	0D	0A	7F	7F	4E
3330:	35	33	47	30	0D	0A	7F	7F	4E	35	34	5A	35	30	30	30
3340:	0D	0A	7F	7F	4E	35	35	47	30	0D	0A	7F	7F	4E	35	36
3350:	58	31	30	36	30	37	59	31	30	36	30	37	0D	0A	7F	7F
3360:	4E	35	37	47	31	0D	0A	7F	7F	4E	35	38	5A	2D	35	30
3370:	30	30	0D	0A	7F	7F	4E	35	39	47	30	0D	0A	7F	7F	4E
3380:	36	30	5A	35	30	30	30	0D	0A	7F	7F	4E	36	31	47	30
3390:	0D	0A	7F	7F	4E	36	32	58	2D	31	30	36	30	37	59	34
33A0:	33	39	33	0D	0A	7F	7F	4E	36	33	47	31	0D	0A	7F	7F
33B0:	4E	36	34	5A	2D	35	30	30	30	0D	0A	7F	7F	4E	36	35
33C0:	47	30	0D	0A	7F	7F	4E	36	36	5A	35	30	30	30	0D	0A
33D0:	7F	7F	4E	36	37	47	30	0D	0A	7F	7F	4E	36	38	58	2D
33E0:	31	30	36	30	38	59	2D	34	33	39	33	0D	0A	7F	7F	4E
33F0:	36	39	47	31	0D	0A	7F	7F	4E	37	30	5A	2D	35	30	30

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3400:	30	0D	0A	7F	7F	4E	37	31	47	30	0D	0A	7F	7F	4E	37
3410:	32	5A	35	30	30	30	0D	0A	7F	7F	4E	37	33	47	30	0D
3420:	0A	7F	7F	4E	37	34	58	2D	34	33	39	33	59	2D	31	30
3430:	36	30	37	0D	0A	7F	7F	4E	37	35	47	31	0D	0A	7F	7F
3440:	4E	37	36	5A	2D	35	30	30	30	0D	0A	7F	7F	4E	37	37
3450:	47	30	0D	0A	7F	7F	4E	37	38	5A	35	30	30	30	0D	0A
3460:	7F	7F	4E	37	39	47	30	0D	0A	7F	7F	4E	38	30	58	34
3470:	33	39	33	59	2D	31	30	36	30	38	0D	0A	7F	7F	4E	38
3480:	31	47	31	0D	0A	7F	7F	4E	38	32	5A	2D	35	30	30	30
3490:	0D	0A	7F	7F	4E	38	33	47	30	0D	0A	7F	7F	4E	38	34
34A0:	5A	35	30	30	30	0D	0A	7F	7F	4E	38	35	47	30	0D	0A
34B0:	7F	7F	4E	38	36	58	31	30	36	30	37	59	2D	34	33	39
34C0:	33	0D	0A	7F	7F	4E	38	37	47	31	0D	0A	7F	7F	4E	38
34D0:	38	5A	2D	35	30	30	30	0D	0A	7F	7F	4E	38	39	47	30
34E0:	0D	0A	7F	7F	4E	39	30	5A	35	30	30	30	0D	0A	7F	7F
34F0:	4E	39	31	47	30	0D	0A	7F	7F	4E	39	32	58	31	30	36

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
3500:	30	38	59	34	33	39	33	0D	0A	7F	7F	4E	39	33	47	31
3510:	0D	0A	7F	7F	4E	39	34	5A	2D	35	30	30	30	0D	0A	7F
3520:	7F	4E	39	35	47	30	0D	0A	7F	7F	4E	39	36	5A	35	30
3530:	30	30	0D	0A	7F	7F	4E	39	37	47	30	0D	0A	7F	7F	4E
3540:	39	38	58	34	33	39	33	59	31	30	36	30	37	0D	0A	7F
3550:	7F	4E	39	39	47	31	0D	0A	7F	7F	4E	31	30	30	5A	2D
3560:	35	30	30	30	0D	0A	7F	7F	4E	31	30	31	47	30	0D	0A
3570:	7F	7F	4E	31	30	32	5A	35	30	30	30	0D	0A	7F	7F	4E
3580:	31	30	33	47	30	0D	0A	7F	7F	4E	31	30	34	58	2D	34
3590:	35	30	30	31	59	2D	33	30	30	30	30	0D	0A	7F	7F	4E
35A0:	31	30	35	47	30	0D	0A	7F	7F	4E	31	30	36	5A	2D	35
35B0:	30	30	30	0D	0A	7F	7F	4E	31	30	37	47	31	0D	0A	7F
35C0:	7F	4E	31	30	38	4D	33	30	0D	0A	7F	7F	4E	31	30	39
35D0:	4D	39	0D	0A	7F	7F	4E	31	31	30	4D	32	0D	0A	7F	7F
35E0:	20	0D	0A	7F	7F	20	0D	0A	7F	7F	20	0D	0A	7F	7F	52

LOADING THE NUMERICAL CONTROL MACHINE CODE FROM AD-APT ONTO A MICROCOMPUTER CONTROLLED FLOPPY DISK

by

HSIAO-WEN LI

B. S. (I.E.), Tunghai University, Taiwan, 1976

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1981

ABSTRACT

This study was to load the Numerical Control Machine Code from Adaptation of Automatically Programmed Tools (AD-APT) on a microcomputer controlled floppy disk. This included four phases: 1) using the CMS, 2) interfacing the Exidy Sorcerer with the Itel System, 3) using MDOS to save the data output from the Itel System onto a floppy disk, and 4) using AD-APT to get the output from the Itel System in a form that is usuable in controlling an NC machine. The interface of the hardware connections and the software program between Itel System and Exidy Sorcerer has been successfully completed through the modem. The output of NC machine code from AD-APT has been also successfully loaded onto a microcomputer (Exidy Sorcerer) controlled floppy disk.