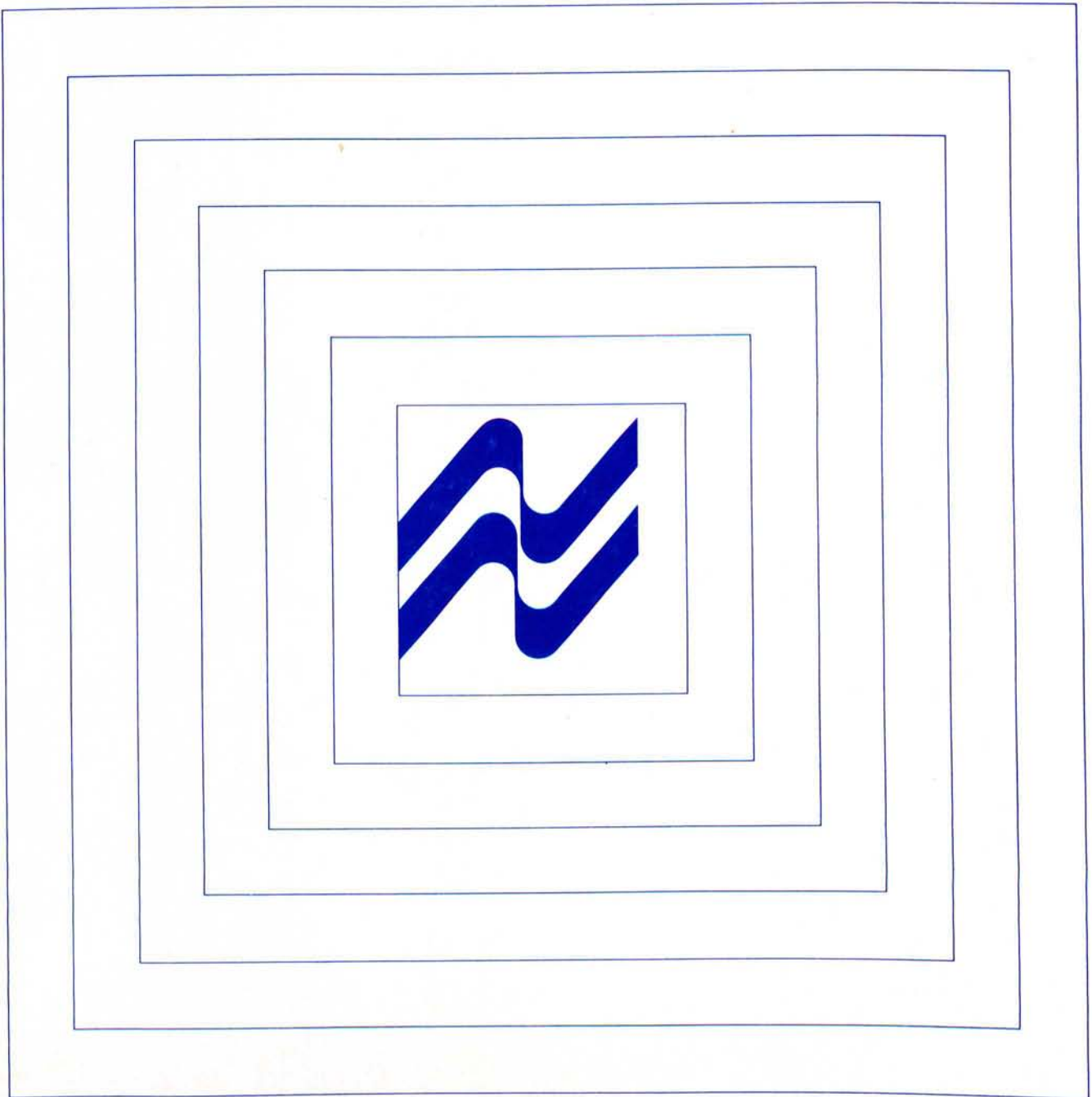


# Users Manual

---

National  
Semiconductor

SC/MP  
Low Cost  
Development  
System



This digitised copy has been provided by Museum Victoria free of charge. You may use this work for any Non-Commercial Use with attribution to Museum Victoria and the creator.



SC/MP  
LOW COST DEVELOPMENT SYSTEM  
(LCDS)

USERS MANUAL

November 1976

© National Semiconductor Corporation  
2900 Semiconductor Drive  
Santa Clara, California 95051

## PREFACE

This manual comprises user information pertaining to the SC/MP Low Cost Development System (LCDS). The following coverage is provided:

- A general description of the LCDS
- Installation and operational instructions
- Programming examples keyed to the routines associated with the operational procedures
- Functional descriptions that interrelate the resident firmware program, the keyboard and display panel, and the logic and buffering circuits used for transferring control back and forth between the resident firmware program and a user-entered application program
- Servicing guidelines

For purposes of user applications system development, the user should have a working knowledge of computer programming, digital circuit logic, and integrated circuits.

The material in this manual is subject to change without notice. Circuit details and other data presented in the engineering documentation supplied with the SC/MP LCDS take precedence over the information presented in this manual.

Copies of the following publications that are relevant to the SC/MP LCDS may be obtained from the local National Semiconductor sales office.

- SC/MP Programming and Assembler Manual (NSC Order No. ISP-8S/994Y)
- SC/MP Technical Description (Publication No. 4200079)
- Data Sheet — ISP-8A/500D Single-Chip 8-bit Microprocessor (SC/MP)
- Data Sheet — ISP-8C/100 CPU Application Card
- Data Sheet — ISP-8C/004P 4K-by-8 PROM Application Card & ISP-8C/004B 4K-by-8 ROM/PROM Socket Application Card
- Data Sheet — ISP-8C/002 2K-by-8 Read/Write Memory Application Card



TABLE OF CONTENTS

Chapter		Page
1	GENERAL DESCRIPTION	
1.1	MAJOR ITEMS AND OPERATIONAL CHARACTERISTICS . . . . .	1-1
1.2	LEADING PARTICULARS . . . . .	1-2
1.3	BASIC EQUIPMENT SUPPLIED . . . . .	1-3
1.4	OPTIONS . . . . .	1-3
2	INSTALLATION	
2.1	DC POWER REQUIREMENTS . . . . .	2-1
2.2	PREPARATION FOR USE . . . . .	2-1
2.3	PANEL CHECKOUT . . . . .	2-3
2.4	TELETYPE INTERCONNECTION AND CHECKOUT . . . . .	2-8
2.4.1	Teletype Interconnection . . . . .	2-8
2.4.2	Paper Tape Reader Relay Option . . . . .	2-11
2.4.3	Teletype Checkout . . . . .	2-12
2.5	INSTALLATION OF OPTIONAL CONNECTORS . . . . .	2-16
3	OPERATION	
3.1	CONTROLS, CONNECTORS, AND INDICATORS . . . . .	3-1
3.2	OPERATING MODE DESCRIPTION . . . . .	3-7
3.2.1	Power-up and Initialization . . . . .	3-9
3.2.2	Panel/TTY Selection . . . . .	3-9
3.3	PANEL OPERATING PROCEDURES . . . . .	3-9
3.3.1	RUN/DEBUG Mode Selection . . . . .	3-10
3.3.2	DEBUG Commands . . . . .	3-10
3.3.3	DEBUG Operation . . . . .	3-11
3.3.3.1	Examining SC/MP Program Counter, Registers, and Accumulator . . . . .	3-11
3.3.3.2	Altering SC/MP Program Counter, Registers, and Accumulator . . . . .	3-11
3.3.3.3	Examining Contents of a Memory Location . . . . .	3-12
3.3.3.4	Altering Contents of a Memory Location . . . . .	3-13
3.4	TTY OPERATING PROCEDURES . . . . .	3-13
3.4.1	TTY DEBUG Commands . . . . .	3-15
3.4.2	RUN/DEBUG Mode Selection . . . . .	3-20
3.4.3	Examining SC/MP Program Counter, Registers, and Accumulator . . . . .	3-21
3.4.4	Altering SC/MP Program Counter, Registers, and Accumulator . . . . .	3-21
3.4.5	Examining Contents of a Memory Location . . . . .	3-21
3.4.6	Altering Contents of a Memory Location . . . . .	3-21
3.4.7	Punching a Paper Tape . . . . .	3-21
3.4.8	Loading a Paper Tape . . . . .	3-22
3.5	APPLICATION SYSTEM INTERFACING . . . . .	3-22
3.5.1	Interface Bus Signal Description . . . . .	3-25
3.5.2	SC/MP CPU Application Card Memory Control . . . . .	3-32
3.5.3	Application System Address Assignment . . . . .	3-32
3.5.4	DEBUG* Signal Implementation . . . . .	3-32
3.5.5	RUN Mode Considerations . . . . .	3-36
3.6	PROGRAM DEVELOPMENT ON LCDS . . . . .	3-36
3.6.1	TTY Utilities . . . . .	3-36
3.6.1.1	GETC Subroutine . . . . .	3-40
3.6.1.2	GETP Subroutine . . . . .	3-41
3.6.1.3	GECHO Subroutine . . . . .	3-41
3.6.1.4	PECHO Subroutine . . . . .	3-41
3.6.1.5	PUTC Subroutine . . . . .	3-44
3.6.1.6	MESG Subroutine . . . . .	3-44

TABLE OF CONTENTS (Continued)

Chapter		Page
3 (cont'd)	3.6.1.7 GHEX Subroutine . . . . .	
	3.6.1.8 GHEXE Subroutine . . . . .	3-47
	3.6.1.9 PHEX Subroutine . . . . .	3-51
	3.6.1.10 PHEXB Subroutine . . . . .	3-52
	3.6.2 Digital Readout . . . . .	3-52
	3.6.3 Keyboard Pushbuttons . . . . .	3-52
		3-53
4	PROGRAMMING EXAMPLES	
	4.1 INTRODUCTION . . . . .	4-1
	4.2 PROGRAMMING CONVENTIONS AND INPUT/OUTPUT DATA ASSIGNMENTS . . . . .	4-1
	4.3 SUBROUTINE DESCRIPTION	
	4.3.1 GETC, GETP, GECHO, and PECHO Subroutines . . . . .	4-1
	4.3.2 PUTC Subroutine . . . . .	4-2
	4.3.3 MESH Subroutine . . . . .	4-3
	4.3.4 GHEX and GHEXE Subroutines . . . . .	4-4
	4.3.5 PHEXB and PHEX Subroutines . . . . .	4-5
5	FUNCTIONAL DESCRIPTION	
	5.1 GENERAL DESCRIPTION . . . . .	5-1
	5.1.1 CPU and Data I/O Control . . . . .	5-3
	5.1.2 Hardware-Forced DEBUG Save Routine . . . . .	5-3
	5.1.3 DEBUG Mode Termination . . . . .	5-7
	5.1.4 Hardware-Forced DEBUG Return Routine . . . . .	5-7
	5.1.5 TTY Interface Circuit . . . . .	5-8
	5.2 DETAILED DESCRIPTION . . . . .	5-9
	5.2.1 Mode Control Logic . . . . .	5-10
	5.2.1.1 DEBUG Mode Selection . . . . .	5-10
	5.2.1.2 RUN Mode Selection . . . . .	5-10
	5.2.1.3 Initialization . . . . .	5-17
	5.2.2 Address Decode Logic . . . . .	5-17
	5.2.3 Read-Only Memory . . . . .	5-17
	5.2.4 Read/Write Memory . . . . .	5-17
	5.2.5 Data Buffers . . . . .	5-17
	5.2.6 Digital Readout . . . . .	5-17
	5.2.7 Keyboard . . . . .	5-18
	5.2.8 HALT MODE Switch . . . . .	5-18
	5.3 CIRCUIT DESCRIPTION . . . . .	5-18
	5.3.1 RUN/DEBUG Mode Enable Logic . . . . .	5-18
	5.3.2 DEBUG Address Generator . . . . .	5-21
6	SERVICE INFORMATION	
APPENDIX A — PROGRAM FOR PUNCHING COMPLEMENTED-BINARY PAPER TAPE		

LIST OF ILLUSTRATIONS

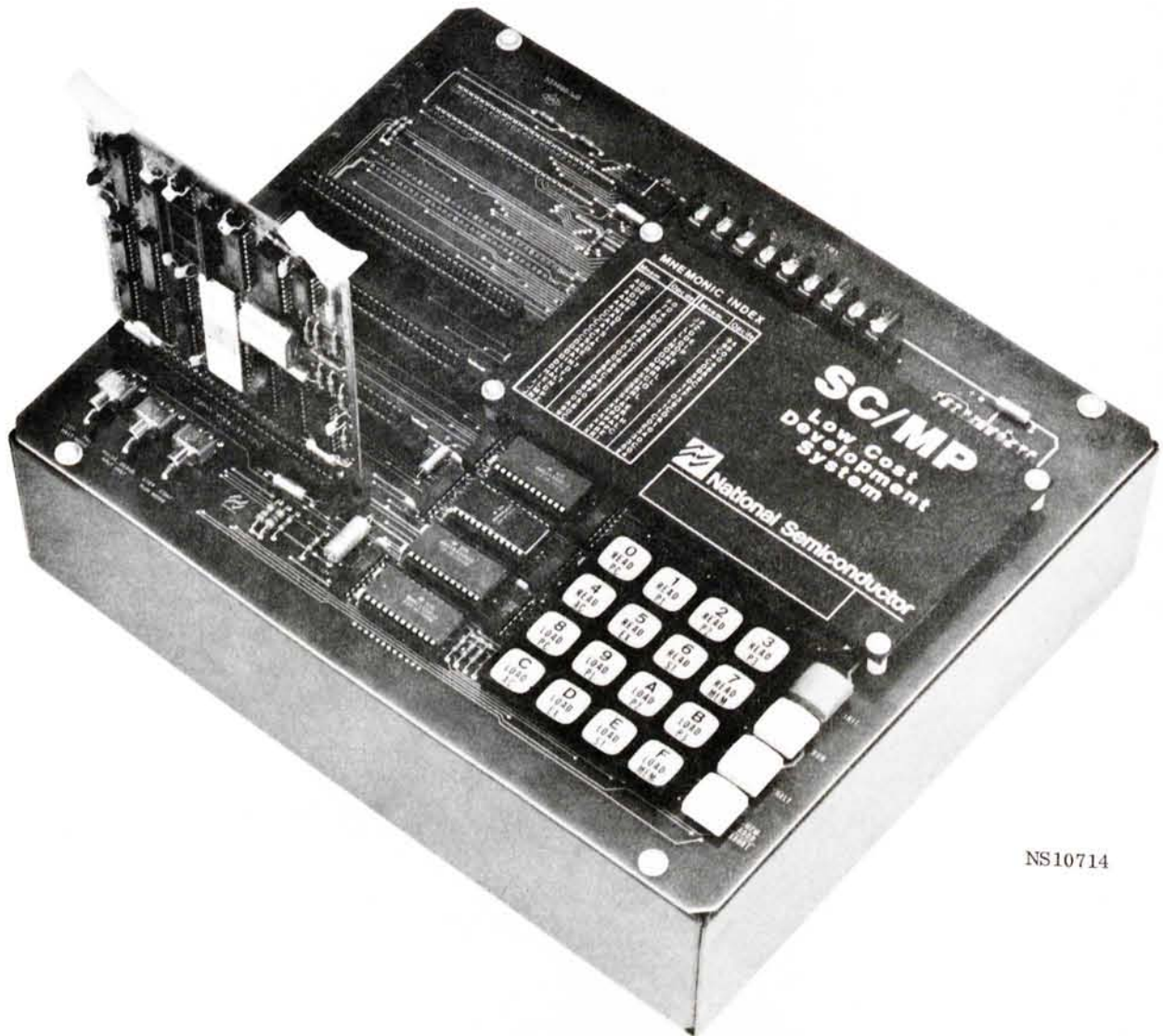
Figure		Page
2-1	LCDS Chassis Layout . . . . .	2-2
2-2	TTY Layout . . . . .	2-9
2-3	TTY Current Source Resistor . . . . .	2-9
2-4	TTY Terminal Strip . . . . .	2-10
2-5	Disabling TTY Auto-Answerback . . . . .	2-10
2-6	Reader Relay Schematic . . . . .	2-11
2-7	TTY Connector Panel . . . . .	2-12
3-1	LCDS Controls, Connectors and Indicators . . . . .	3-2
3-2	Application System Interface Bus . . . . .	3-24
3-3	Memory Control Circuit for SC/MP CPU Application Card . . . . .	3-33
3-4	Flags, BSENSE, and Interrupt Control . . . . .	3-37
3-5	GETP, GETC, GECHO, and PECHO Subroutines . . . . .	3-42
3-6	PUTC Subroutine . . . . .	3-45
3-7	MESG Subroutine . . . . .	3-46
3-8	GHEX and GHEXE Subroutines . . . . .	3-49
3-9	PHEX and PHEXE Subroutines . . . . .	3-55
3-10	LCDS Digital Readout and Control Circuit . . . . .	3-56
4-1	GETP, GETC, GECHO, and PECHO Subroutine Detailed Flowchart . . . . .	4-6
4-2	GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing . . . . .	4-10
4-3	PUTC Subroutine — Detailed Flowchart . . . . .	4-19
4-4	PUTC Subroutine — Annotated Instruction Listing . . . . .	4-21
4-5	MESG Subroutine — Detailed Flowchart . . . . .	4-27
4-6	MESG Subroutine — Annotated Instruction Listing . . . . .	4-29
4-7	GHEX and GHEXE Subroutine — Detailed Flowchart . . . . .	4-35
4-8	GHEX and GHEXE Subroutine — Annotated Instruction Listing . . . . .	4-42
4-9	GHEX and GHEXE Subroutine — Detection of Space, ALTMODE, and ESC . . . . .	4-55
4-10	ASCII-to-Hexadecimal Conversion Routine for GHEX and GHEXE Subroutines . . . . .	4-56
4-11	Four-Digit Save Routine for GHEX and GHEXE Subroutines . . . . .	4-58
4-12	PHEX and PHEXB Subroutine — Detailed Flowchart . . . . .	4-60
4-13	PHEX and PHEXB Subroutine — Annotated Instruction Listing . . . . .	4-64
4-14	Hexadecimal-to-ASCII Conversion Routine for PHEX and PHEXB Subroutines . . . . .	4-73
5-1	LCDS Overall Block Diagram . . . . .	5-2
5-2	DEBUG Save Routine Timing . . . . .	5-5
5-3	DEBUG Return Routine Timing . . . . .	5-8
5-4	LCDS Detailed Block Diagram . . . . .	5-11/5-12
5-5	Mode Control Logic Timing for DEBUG Save and Restore Routines . . . . .	5-16
5-6	RUN/DEBUG Mode Enable Logic Circuit Diagram . . . . .	5-20
5-7	DEBUG Address Generator and Address Buffer Circuit Diagram . . . . .	5-22



LIST OF TABLES

Table		Page
1-1	Leading Particulars . . . . .	1-2
1-2	Basic Equipment Supplied . . . . .	1-3
1-3	Options Available . . . . .	1-3
2-1	LCDS Power Requirements . . . . .	2-1
2-2	Panel Checkout Description . . . . .	2-6
2-3	TTY Checkout Description . . . . .	2-15
3-1	Description of Controls, Connectors, and Indicators . . . . .	3-1
3-2	Jumper Options Available at Connector J8 . . . . .	3-7
3-3	DEBUG Save Addresses for SC/MP Program Counter, Registers, and Accumulator . . . . .	3-8
3-4	TTY DEBUG Command Notations and Symbols . . . . .	3-14
3-5	TTY DEBUG Commands . . . . .	3-16
3-6	SC/MP Microprocessor Page Designations . . . . .	3-19
3-7	Sources of Accessory Equipment . . . . .	3-23
3-8	SC/MP Application Card Interface Bus Description . . . . .	3-25
3-9	Ribbon Connector J6 (Optional) Pin Assignments . . . . .	3-30
3-10	Ribbon Connector J7 (Optional) Pin Assignments . . . . .	3-31
3-11	SC/MP CPU Application Card Memory Control Options . . . . .	3-34
3-12	DEBUG TTY Utility Subroutines . . . . .	3-39
3-13	USA Standard Code for Information Interchange (ASCII) . . . . .	3-40
5-1	Hardware-Forced DEBUG Save Routine Address Functions . . . . .	5-4
5-2	Hardware-Forced DEBUG Return Routine Address Functions . . . . .	5-9
5-3	DEBUG Save and Return Routine Address Generation . . . . .	5-23





NS10714

## Chapter 1

### GENERAL DESCRIPTION

The SC/MP Low Cost Development System (hereinafter referred to as the LCDS) is a low-cost SC/MP controller that provides all the features necessary for developing and debugging SC/MP hardware and software applications designs. It includes an SC/MP CPU Application Card and an interface structure that permits the CPU Application Card to function as part of the LCDS or as part of a user-fabricated applications system. Overall addressing capability of the applications system is 65,536 locations. Common memory/peripheral address and data buses permit the whole class of SC/MP memory-reference instructions to be used for peripheral control. Command capability for entry and test of user-generated applications software is afforded via a resident firmware program, an Operators Control Panel, and a 20-milliampere interface that permits connection of an optional Teletype <sup>®</sup>.

#### 1.1 MAJOR ITEMS AND OPERATIONAL CHARACTERISTICS

Following are salient operating characteristics of the major items comprising the LCDS.

- SC/MP CPU APPLICATION CARD — provides CPU interface for execution of user-generated applications programs and LCDS resident firmware program.
- PREWIRED APPLICATIONS SYSTEMS INTERFACE — four prewired sockets provide a plug-in interface for SC/MP family application cards, and permit interconnection of additional SC/MP applications hardware via user-fabricated cabling.
- INTERFACE LOGIC — provides control and monitor functions that permit transfer-of-control between development-system resident firmware program and user-generated applications programs.
- DEVELOPMENT SYSTEM RESIDENT FIRMWARE PROGRAM — contains LCDS control-panel and Teletype subroutines that permit entry of software debug commands via the Operators Control Panel or via an optional Teletype.
- OPERATORS CONTROL PANEL — provides the following software debug capabilities.
  - Display contents of SC/MP program counter, accumulator, and other registers in hexadecimal format
  - Display contents of any memory location in hexadecimal format
  - Alter contents of SC/MP program counter, accumulator, and other registers
  - Alter contents of any memory location
  - Initiate execution of user-generated applications program at any memory address
  - Select single-instruction or normal execution of user-generated applications program
  - Interrupt execution of user-generated applications program at any point
- 20-MILLIAMPERE TELETYPE INTERFACE — provides standard interface for interconnection of optional Teletype. Expanded software debug capabilities associated with Teletype option include the following.
  - Print contents of SC/MP program counter, accumulator, and other registers
  - Print contents of any single memory location or selected range of memory locations
  - Alter contents of SC/MP program counter, accumulator, and other registers
  - Alter contents of any memory location or selected range of memory locations
  - Set breakpoint halts for user-generated applications program

<sup>®</sup> Registered trademark of Teletype Corporation



- 20-MILLIAMPERE TELETYPE INTERFACE — (continued)
  - Initiate execution of user-generated applications program at any memory address
  - Obtain applications program load module by punching selected memory range to paper tape
  - Load LCDS-generated paper-tape load module into memory
  - Load IMP-16 or FORTRAN Cross Assembler-generated paper-tape load module into memory

1.2 LEADING PARTICULARS

LCDS electrical and physical characteristics are listed in table 1-1.

Table 1-1. Leading Particulars

Feature	Description
Data Word Length	8 bits
Addressing	16 bits
Instruction Set	46 Instructions
Arithmetic	Parallel, binary, fixed point, twos complement and parallel 2-digit BCD
Memory	8-bit bytes of semiconductor memory expandable to a maximum of 65,536 bytes
Addressing Modes	PC-Relative Immediate Indexed Auto-Indexed
Typical Operating Speed	2.0 microseconds/microcycle
Input/Output and Control	8-bit buffered data bus 16-bit buffered address bus 1 serial data-input port 1 serial data-output port 3 general-purpose control flag outputs 1 general-purpose sense input 1 general-purpose sense/interrupt input Bus-request output and bus-enable input (permit use of allocation logic for Direct Memory Access and multi-processor applications)
Input Power (at 25°C)	+5 VDC (current specifications are provided -12 VDC in chapter 2)
Humidity	Maximum of 90-percent relative humidity without condensation
Dimensions of Chassis	4 inches (10.2 centimeters) high 12 inches (30.0 centimeters) wide 10 inches (25.4 centimeters) deep

### 1.3 BASIC EQUIPMENT SUPPLIED

Equipment supplied as part of the basic LCDS configuration is listed in table 1-2.

Table 1-2. Basic Equipment Supplied

Item	Description
LCDS Chassis	Contains LCDS circuit and DEBUG firmware, and provides four prewired interface sockets for connection of SC/MP family application cards and/or custom applications circuits.
SC/MP CPU Application Card	Provides a CPU interface used for execution of LCDS DEBUG firmware program and user-generated applications programs.

### 1.4 OPTIONS

Optional equipment that may be used to expand the capabilities of the basic LCDS configuration is listed in table 1-3.

Table 1-3. Options Available

Item	Description
Teletype	The LCDS is designed to operate with a standard Teletype model ASR 3320/JC or TU without XON, XOFF, and with the automatic answerback disabled. If desired, this model Teletype may be ordered from the National Semiconductor Corporation under order number IMP-00/810.
Teletype Tape Reader Relay	If an application requires that the LCDS control the TTY tape reader, a relay can be installed to permit the LCDS to turn the tape reader on and off, and thereby control the reading of one frame of data at a time. A board containing the required circuit may be ordered from the National Semiconductor Corporation under order number IPC-16P/810R. Detailed instructions for installing the board or an equivalent user-fabricated circuit are provided in chapter 2.
Connector J5	A user-supplied 72-pin connector may be installed at location J5 to permit plug-in interconnection of an additional SC/MP family application card. Manufacturer part numbers for the connector are provided in chapter 2 along with detailed installation instructions.
Connectors J6 and J7	User-supplied 50-pin ribbon connectors may be installed at locations J6 and J7 to facilitate interconnection of custom applications systems. A manufacturer part number for the connectors is provided in chapter 2 along with detailed installation instructions.

## Chapter 2

### INSTALLATION

#### 2.1 DC POWER REQUIREMENTS

The LCDS is designed to operate on user-supplied +5VDC and -12VDC input power. Since the LCDS provides a plug-in interface for user-fabricated SC/MP application systems, overall current requirements are determined by adding the current requirements of the LCDS chassis to the current requirements of the applications hardware with which it is interconnected. Current requirements for the LCDS chassis and for SC/MP applications cards are listed in table 2-1; current requirements for other applications hardware should be specified in the manufacturer's literature.

#### 2.2 PREPARATION FOR USE

To prepare the LCDS for initial operation, refer to figure 2-1 and proceed as follows.

#### CAUTION

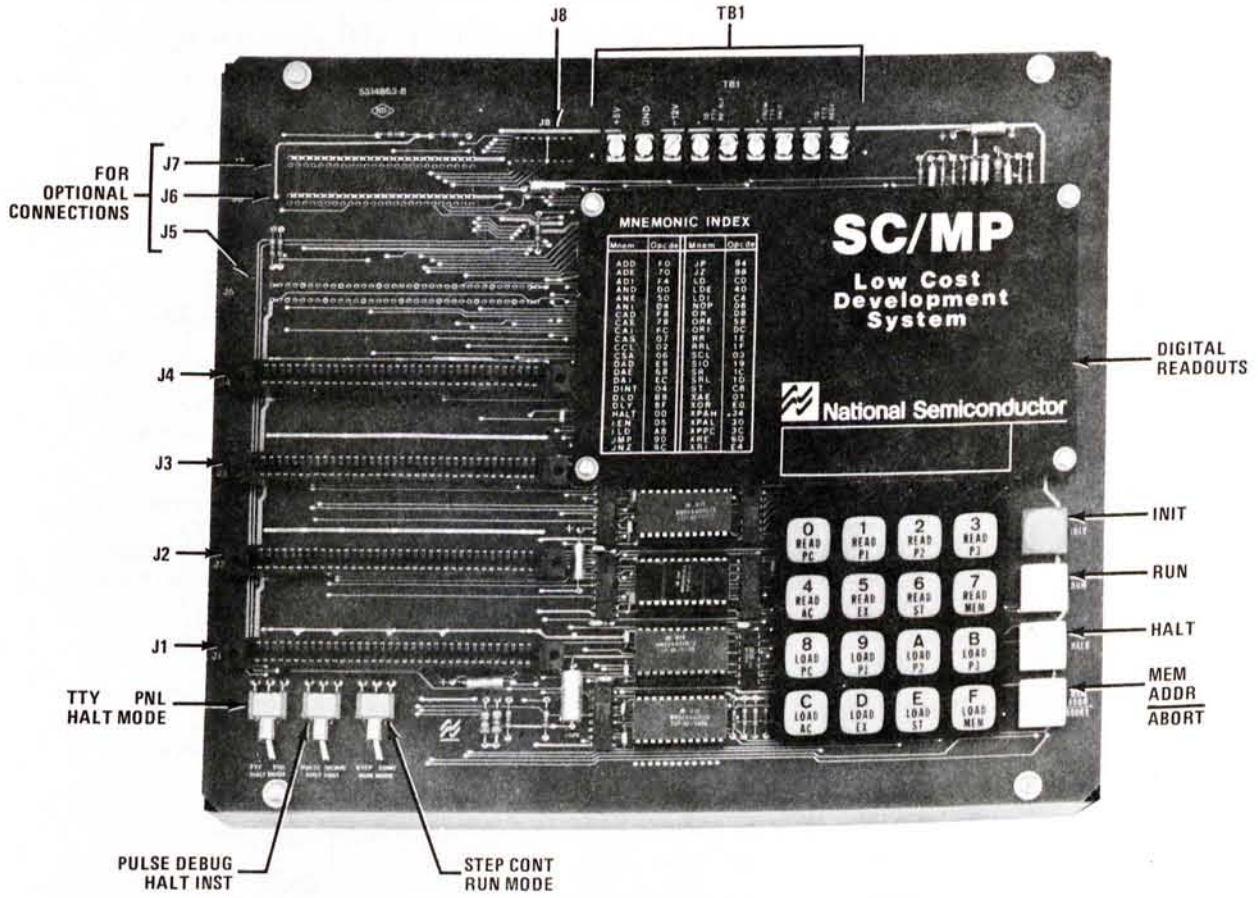
To prevent damage to equipment, always turn off power to LCDS before connecting or disconnecting SC/MP Application Cards or other SC/MP applications hardware.

1. Insert the SC/MP CPU Application Card into one of the four prewired sockets (J1 through J4) with the component side of the card facing the front of the LCDS chassis.
2. Verify that a jumper is installed between pins 5 and 12 on socket J8 and that the remaining pins on the socket are unterminated. (The functions of the jumper options available at socket J8 are described in table 3-2 of chapter 3, Operation, and instructions for using the jumper options to effect various special-purpose configurations are provided in 3.5, Application System Interfacing.)
3. Using 16-gauge or heavier wire, connect +5VDC and -12VDC power to LCDS. If separate external power supplies are employed, both power supplies must be referenced to common GND terminal.

Table 2-1. LCDS Power Requirements

Item	Current Requirements	
	+5VDC	-12VDC
Low-Cost Development System Chassis (without SC/MP CPU Application Card installed)	1.5A	300mA
SC/MP CPU Application Card (with MM5204 in PROM socket)	600mA	150mA
SC/MP Read/Write Memory Application Card	1.45A	NA
SC/MP ROM or ROM/PROM Application Card (with eight MM5204 PROMs installed)	550mA	320mA





NS10715

Figure 2-1. LCDS Chassis Layout

To verify the proper operation of the LCDS controls and indicators, perform the steps listed below. The purpose of each step is listed in table 2-2, along with a reference to the appropriate description provided in chapter 3, Operation.

1. Set HALT MODE switch to PNL. Then turn on +5VDC and -12VDC power to LCDS, and verify that digital readout displays 0001 PC.
2. Press READ P1 pushbutton, and verify that digital-readout display changes to 0000 P1.
3. Press READ P2 pushbutton, and verify that digital-readout display changes to 0000 P2.
4. Press READ P3 pushbutton, and verify that digital-readout display changes to 0000 P3.
5. Press READ AC pushbutton, and verify that digital-readout display changes to 00 A.
6. Press READ EX pushbutton, and verify that digital-readout display changes to 00 E.

NOTE

The term "XX" or "XXXX" is used to denote uncertainty of hexadecimal value displayed.

7. Press READ ST pushbutton, and verify that digital-readout display changes to 10 S.
8. Press READ MEM pushbutton, and verify that digital-readout display changes to 0000 "XX".
9. Press LOAD PC pushbutton, and verify that digital-readout display changes to ---- PC. Then, key in any 2-digit value (XX--PC), press MEM ADDR/ABORT pushbutton, and verify that digital-readout display changes to 0000 00.
10. Press READ PC pushbutton, and verify that digital-readout display changes to 0001 PC.
11. Press MEM ADDR/ABORT pushbutton, and verify that digital-readout display changes to ----. Then, key in value 7800, and verify that last two digits of digital-readout display C8 (contents of memory location 7800).

NOTE

If a wrong digit is accidentally keyed in when a numeric value is being entered for a LOAD Command, press the MEM ADDR/ABORT pushbutton to terminate the command (digital readout will display 0000 00). Then, press the LOAD pushbutton again to reselect the command, and key in the correct data value.

Address 7800 is a Read-Only memory location.

12. Press LOAD MEM pushbutton, and verify that digital-readout display changes to 7800 --. Then, key in any value except C8, and verify that digital readout display changes to 0000 00.
13. Press MEM ADDR/ABORT pushbutton, and verify that digital-readout display changes to ----. Then, key in value 5555, and verify that last two digits of digital readout display "XX" (contents of memory location 5555).
14. Press LOAD MEM pushbutton, and verify that digital-readout display changes to 5555 --. Then key in value 55 and verify that digital-readout display changes to 5555 55.
15. Press MEM ADDR/ABORT pushbutton, and verify that digital-readout display changes to ----. Then, key in value AAAA, and verify that last two digits of digital readout display "XX" (contents of memory location AAAA).
16. Press LOAD MEM pushbutton, and verify that digital-readout display changes to AAAA --. Then, key in value AA, and verify that digital-readout display changes to AAAA AA.



17. Press LOAD PC pushbutton, and verify that digital-readout display changes to ---- PC. Then, key in value 0123.
18. Press LOAD P1 pushbutton, and verify that digital-readout display changes to ---- P1. Then, key in value 4567.

NOTE

The LCDS digital readout displays the hexadecimal values B and D in lower case characters (b and d) that are readily distinguished from the values 8 and 0.

19. Press LOAD P2 pushbutton, and verify that digital-readout display changes to ---- P2. Then, key in value 89AB.
20. Press LOAD P3 pushbutton, and verify that digital-readout display changes to ---- P3. Then, key in value CDEF.
21. Press LOAD AC pushbutton, and verify that digital-readout display changes to -- A. Then, key in value 01.
22. Press LOAD EX pushbutton, and verify that digital-readout display changes to -- E. Then, key in value 02.
23. Press LOAD ST pushbutton, and verify that digital-readout display changes to -- S. Then, key in value 04.
24. Press READ PC pushbutton, and verify that digital-readout display changes to 0123 PC.
25. Press READ P1 pushbutton, and verify that digital-readout display changes to 4567 P1.
26. Press READ P2 pushbutton, and verify that digital-readout display changes to 89Ab P2.
27. Press READ P3 pushbutton, and verify that digital-readout display changes to CdEF P3.
28. Press READ AC pushbutton, and verify that digital-readout display changes to 01 A.
29. Press READ EX pushbutton, and verify that digital-readout display changes to 02 E.
30. Press READ ST pushbutton, and verify that digital-readout display changes to 04 S.
31. Press INIT pushbutton, and verify that digital-readout display changes to 0001 PC. Then, repeat steps 2 through 8 before proceeding to step 32.

NOTE

After the LOAD MEM pushbutton is pressed, the first two hexadecimal digits keyed in are automatically entered into the memory location whose address is displayed on the first four digits of the digital readout.

32. Press MEM ADDR/ABORT pushbutton, and key in value 7701 to initialize memory loading at address 7701. Then, enter program listed on next page into specified memory locations using LOAD MEM pushbutton — that is, press LOAD MEM pushbutton, key in data value specified, and, then, press LOAD MEM pushbutton to access next memory location.

Memory Location	Data Value	Instruction	Comments
7701	C4	LDI 020	Load AC with low-order digital-readout address.
7702	20		
7703	31	XPAL P1	Exchange P1-low with AC.
7704	C4	LDI 070	Load AC with high-order digital-readout address.
7705	70		
7706	35	XPAH P1	Exchange P1-high with AC.
7707	C4	LDI 077	Load AC with digital-readout display code for hexadecimal value A.
7708	77		
7709	C9	ST 0(P1)	Display hexadecimal value A on first digit of digital readout.
770A	00		
770B	C4	LDI 07	Load AC with loop-count value 07.
770C	07		
770D	C8	ST .+100	Store contents of AC at memory location "LOOP" (current PC address +100 <sub>10</sub> ).
770E	64		
770F	8F	DLY 0FF	Delay Instruction.
7710	FF		
7711	B8	DLD .+96	Decrement memory location "LOOP" (current PC address +96 <sub>10</sub> ).
7712	60		
7713	9C	JNZ .-6	Repeat delay/decrement loop until contents of memory location "LOOP" are decremented to zero.
7714	FA		
7715	00	HLT	Halt Instruction.
7716	C4	LDI 07C	Load AC with digital-readout display code for hexadecimal value b.
7717	7C		
7718	C9	ST 0(P1)	Display hexadecimal value b on first digit of digital readout.
7719	00		
771A	C4	LDI 07	Load AC with loop-count value of 07.
771B	07		
771C	C8	ST .+85	Store contents of AC at memory location "LOOP" (current PC address +85 <sub>10</sub> ).
771D	55		
771E	8F	DLY 0FF	Delay Instruction.
771F	FF		
7720	B8	DLD .+87	Decrement memory location "LOOP" (current PC address +81 <sub>10</sub> ).
7721	51		
7722	9C	JNZ .-6	Repeat delay/decrement loop until contents of memory location "LOOP" are decremented to zero.
7723	FA		
7724	90	JMP .-31	Jump back to instruction that loads AC with digital-readout display code for hexadecimal value A.
7725	E1		

$P_1 = 7020$

= Loop-count display

$7020 = 77 = A$

770E  
64

$770E = 7$

33. Press MEM ADDR/ABORT pushbutton, and key in value 7701.
34. Use READ MEM pushbutton to examine memory locations 7701 through 7725 sequentially, and verify that the program was loaded properly. If an error is detected, press MEM ADDR/ABORT pushbutton, and key in address of memory location containing erroneous data value. Then, press LOAD MEM pushbutton, and key in correct data value.



35. Set the following switches to the positions listed:
  - HALT MODE — PNL
  - HALT INST — PULSE
  - RUN MODE — STEP
36. Press INIT pushbutton, and verify that 0001 PC is displayed on digital readout. Then, press LOAD PC pushbutton, key in 7701, and verify that digital-readout indication changes to 7701 PC.
37. Press RUN pushbutton, and verify that digital-readout indication changes to 7703 PC.
38. Set RUN MODE switch to CONT. Then, press RUN pushbutton, and verify that first digit of digital readout alternately displays hexadecimal values A and b.
39. Set HALT INST switch to DEBUG, and verify that program halts with 7716 PC displayed on digital readout.
40. Set HALT INST switch to PULSE, press RUN pushbutton, and observe that hexadecimal values A and b are again displayed alternately on first digit of digital readout.
41. To complete the panel checkout procedure, press HALT pushbutton, and verify that program halts with "XXXX" PC displayed on digital readout ("XXXX" will be a value between 7707 and 7724).

Table 2-2. Panel Checkout Description

Step	Purpose	Reference
1	This step verifies that the Power-Up Initialization Circuit on the SC/MP CPU Application Card will initialize the SC/MP Micro-processor and will place the LCDS in the DEBUG Mode when power is turned on.	3.2.1
2-7	These steps verify that the LCDS saves the contents of the SC/MP program counter, registers, and accumulator before entering the DEBUG Mode, and that the saved contents can be examined via the READ keyboard pushbuttons.	3.2 and 3.3.3.1
8	This step verifies that the contents of a memory location can be examined via the READ MEM pushbutton.	3.3.3.3
9, 10	These steps verify that a Load Command can be aborted by pressing the MEM ADDR/ABORT pushbutton.	3.3.2
11	This step verifies that the contents of a memory location can be examined via the MEM ADDR/ABORT pushbutton when a command is not in progress.	3.3.3.3
12	This step verifies that the LCDS does a "read-after-write" check when a LOAD MEM command is executed.	3.3.3.4
13-16	These steps verify that all segments of the digital readout are operable.	3.1
17-30	These steps verify that the saved contents of the SC/MP program counter, registers, and accumulator can be altered via the LOAD keyboard pushbuttons.	3.3.3.2
31	This step verifies that pressing the INIT pushbutton will initialize the SC/MP Microprocessor and cause the LCDS to enter the DEBUG Mode after the initialized contents of the SC/MP program counter, registers, and accumulator are saved.	3.2.1

Table 2-2. Panel Checkout Description (Continued)

Step	Purpose	Reference
32-34	These steps verify that a memory-reference address can be stored via the MEM ADDR/ABORT pushbutton and that a program can be loaded via the LOAD MEM pushbutton.	3.3.3.4
35, 36, 37	These steps verify the LCDS program-execution STEP function; that is, one instruction of the program is executed each time that the RUN pushbutton is pressed; then the LCDS returns to the DEBUG Mode.	3.3.1
38	This step verifies that an applications program can be executed normally when the RUN pushbutton is pressed and the RUN MODE switch is set to CONTIN.	3.3.1
39	This step verifies that the LCDS will terminate the RUN Mode and enter the DEBUG Mode when the HALT INST switch is set to DEBUG and a Halt Instruction is executed.	3.3.1
40	This step verifies that RUN Mode operation will not be affected by execution of a Halt Instruction when the HALT INST switch is set to PULSE.	3.3.1
41	This step verifies that the LCDS will terminate the RUN Mode and enter the DEBUG Mode when the HALT pushbutton is pressed.	3.3.1



## 2.4 TELETYPE INTERCONNECTION AND CHECKOUT

The LCDS is designed to operate with a standard Teletype<sup>®</sup> (TTY) model ASR 3320/JC or TU without XON, XOFF, and with the automatic answerback option disabled. This model TTY is equipped with a paper tape reader that is configured to be controlled manually by the operator. With this configuration, the tape reader immediately begins reading data from the tape when it is turned on and transmits the data continually to the LCDS. The LCDS, in turn, accepts data from the tape reader until the input requirements are met (as determined by the resident LCDS firmware program in accordance with the format of the data on the tape). The tape reader, however, continues to advance tape and read data from the tape until the reader is turned off by the operator — regardless of whether the LCDS is still accepting the data.

If an application requires that the LCDS control the tape reader, an optional reader relay can be installed. The LCDS then will energize the reader relay to turn the tape reader on and, thereby, to control the reading of one frame of data at a time.

### 2.4.1 Teletype Interconnection

The TTY must be set to operate in the full-duplex mode with a 20-milliampere current loop interface. Instructions for TTY setup and connection to the LCDS are provided below. Figure 2-2 is a view of the layout of the top of the TTY showing the location of the assemblies that are referenced in these instructions. Figures 2-3 and 2-4 show the details of the TTY terminal strip and the current source resistor. In these figures, the dotted lines indicate the connections for half-duplex and 60-milliampere current-loop operation. This is the configuration in which the TTY is shipped normally from the Teletype Corporation. The solid lines indicate the desired connections for full-duplex and 20-milliampere current-loop operation. Ensure that power is removed from the TTY before performing the following steps. (If the TTY is obtained from National Semiconductor Corporation (order number IMP-00/810), the following steps 1-4 already will have been accomplished.)

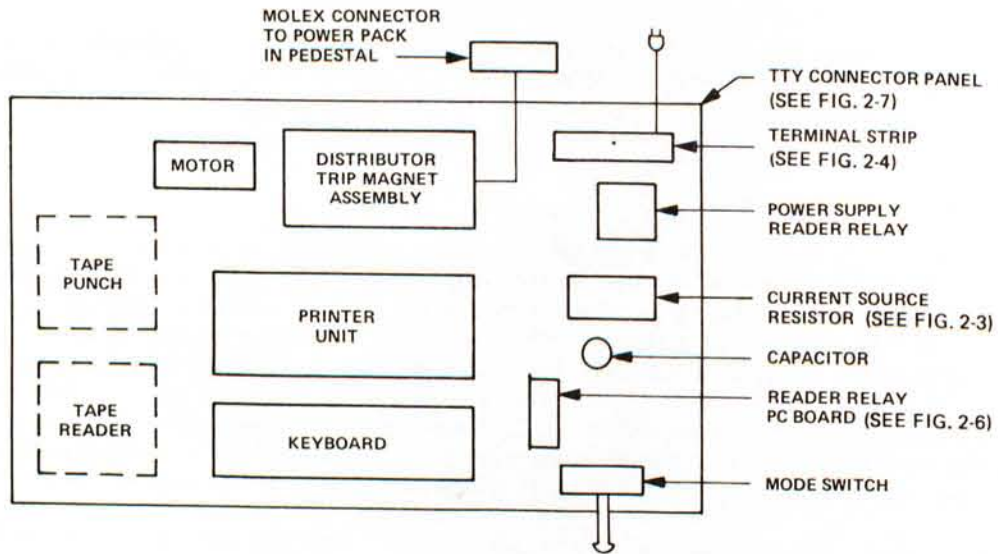
1. To set TTY current source to 20 milliamperes, move blue wire from terminal 3 to terminal 4 of current-source resistor (figure 2-3).
2. To set receive current to 20 milliamperes, move purple wire from pin 8 to pin 9 on terminal strip located at rear of TTY (figure 2-4).
3. To configure TTY for full-duplex, (a) move white-blue wire from pin 4 to pin 5 on the terminal strip, and (b) move brown-yellow wire from pin 3 to pin 5 (figure 2-4).
4. To disable the auto-answerback option, lift the print-station paper cover and locate the cavity behind the keyboard. Directly beneath the carriage is a set of nine codebars. At the front of this assembly is a tie-bar (figure 2-5). The auto-answerback is disabled by placing a clip over the tie-bar so that the third slot from the right is covered. On some models, one of these copper-colored clips already may be placed over the second slot; if so, move it to the third slot. If no clip is provided, it can be obtained from your local Teletype dealer.
5. Connect FROM TTY XMIT - and + inputs of LCDS to pins 3 and 4, respectively, of TTY terminal strip (figure 2-4).
6. Connect TO TTY RECV - and + outputs of LCDS to pins 6 and 7, respectively, of TTY terminal strip (figure 2-4).

#### NOTE

Cable length from TTY to LCDS should not exceed 12 feet. Recommended cable type is standard twisted-pair, 22 AWG.

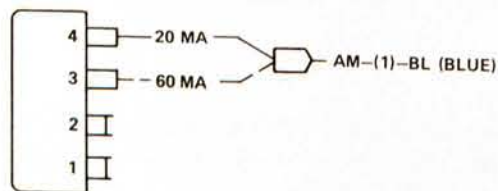
This completes the standard setup and interconnection of the TTY to the LCDS.

<sup>®</sup> Registered trademark of Teletype Corporation



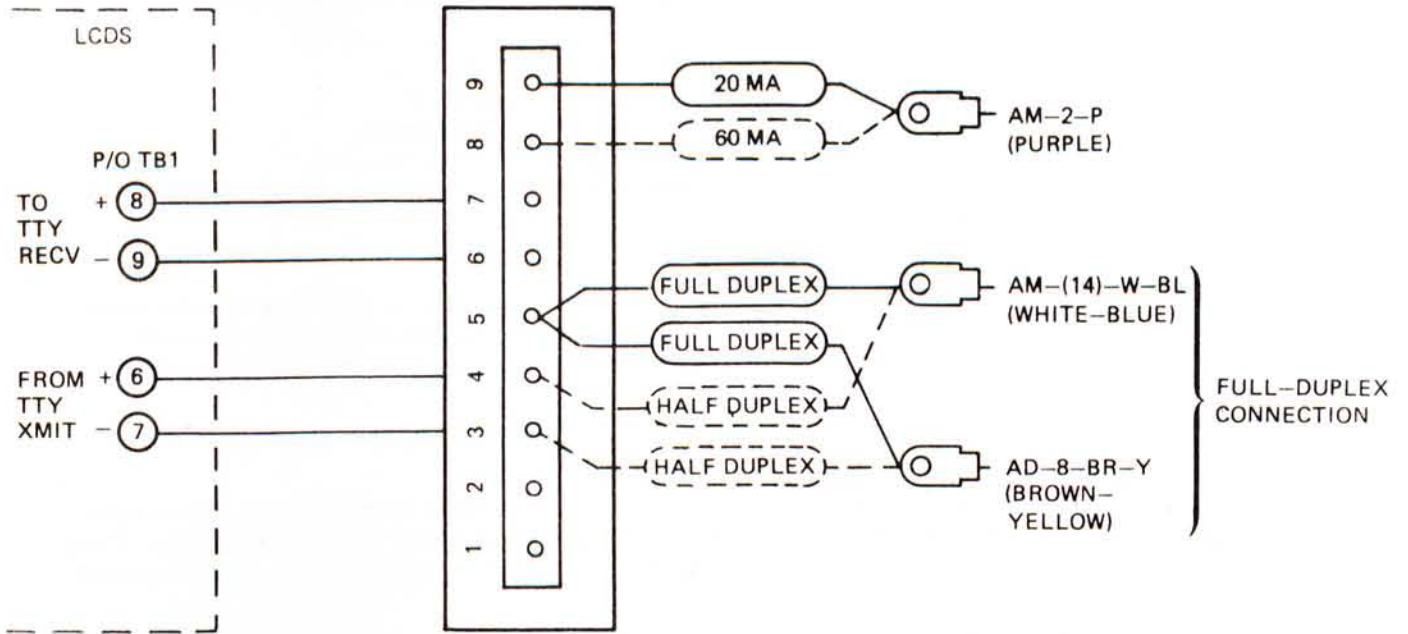
NS10505

Figure 2-2. TTY Layout



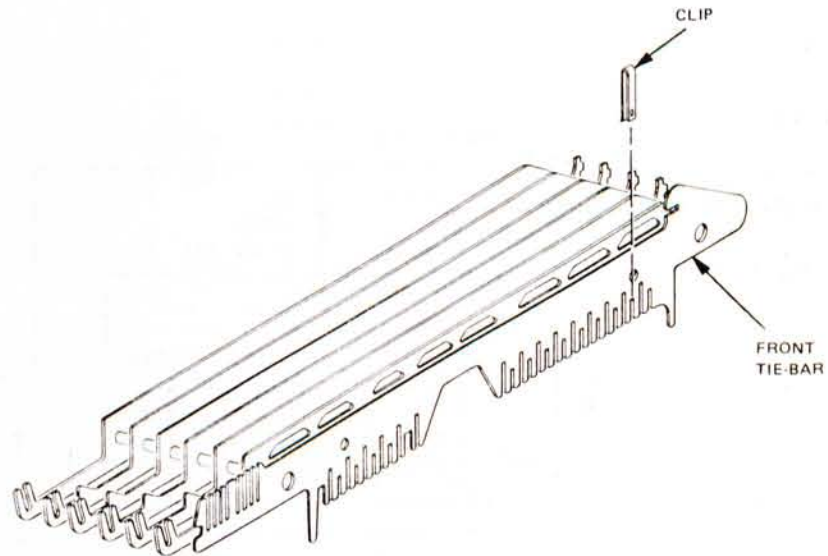
NS10506

Figure 2-3. TTY Current Source Resistor



NS10507

Figure 2-4. TTY Terminal Strip



CODEBAR BASKET ASSEMBLY

NS10508

Figure 2-5. Disabling TTY Auto-Answerback



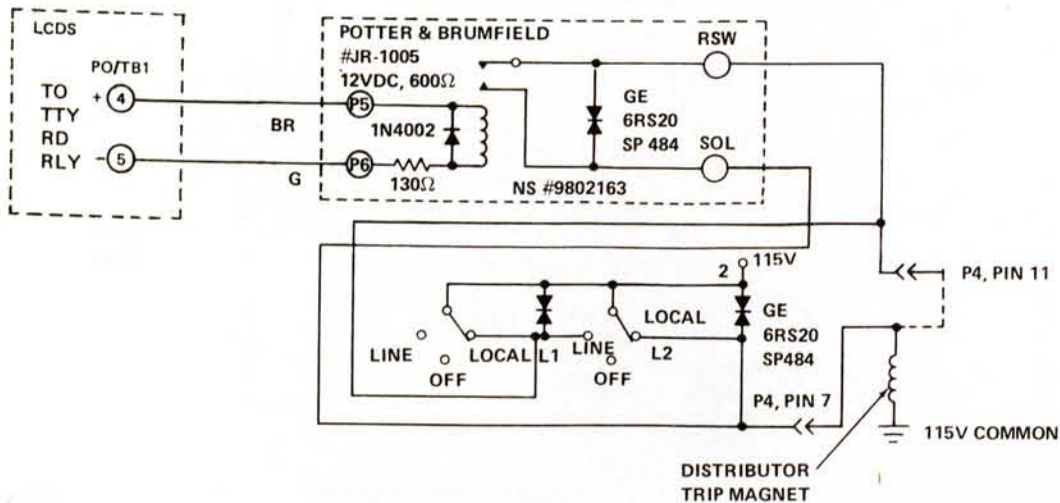
### 2.4.2 Paper Tape Reader Relay Option

If the paper tape reader relay option is required, the following additional steps must be performed.

#### NOTE

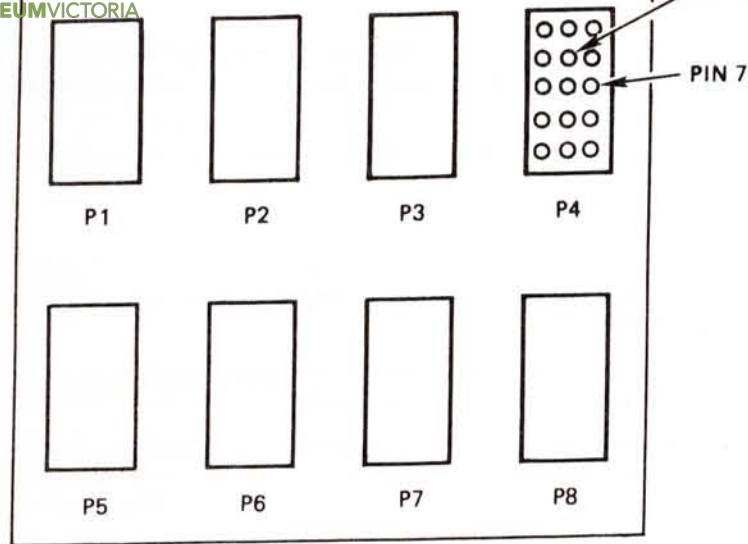
A board containing the required reader relay and associated circuits can be obtained from the National Semiconductor Corporation. The order number is IPC-16P/810R. Alternately, the user can provide the relay and circuit. Details of the relay board are shown in figure 2-6.

1. Install relay board, NS Number IPC-16P/810R or equivalent, in Paper Tape Reader drive circuit. Mounting tab with holes for mounting board is located in lower corner next to keyboard. Mount board with components facing away from keyboard (figure 2-2) utilizing two number 6 screws and lockwashers.
2. On jack 4 (male) at rear of Teletype, remove brown wire and pin from pin 11 and install at pin 7 (figure 2-7).
3. Connect wire from plug 4 (female), pin 7, at rear of Teletype to SOL terminal lug on relay board, using Molex termination at pin 7 and clip-on termination at SOL terminal lug. Then, connect second wire from SOL terminal lug to terminal L2 on Line/Local switch at front of Teletype (figure 2-6).
4. Run wire from L1 on Line/Local Switch to RSW terminal lug on relay board using clip-on termination.
5. Connect TO TTY RD RLY + output of LCDS to P5 terminal lug on relay board (figure 2-6).
6. Connect TO TTY RD RLY - output of LCDS to P6 terminal lug on relay board (figure 2-6).
7. Install two thyrectors (transient suppressors) on Line/Local Switch. Connect one between terminals L2 and 2 and the other between terminals L1 and 2 (figure 2-6).



NS10509

Figure 2-6. Reader Relay Schematic



NS10510

Figure 2-7. TTY Connector Panel

### 2.4.3 Teletype Checkout

To verify the proper operation of the TTY interface and firmware, perform the steps listed below. The purpose of each step is listed in table 2-3, along with a reference to the appropriate description provided in chapter 3, Operation.

1. Set TTY keyboard to LINE, TTY tape punch to OFF, and TTY tape reader to STOP.
2. Set the following LCDS switches to the positions indicated:
  - HALT MODE — PNL
  - HALT INST — PULSE
  - RUN MODE — CONT
3. Turn on power to LCDS and key in program listed for step 32 of 2.3 (Panel Checkout).
4. Load PC with 7701; then, press RUN pushbutton and verify that hexadecimal values A and b are displayed alternately on first digit of digital readout. If these values are not displayed properly, refer to steps 33 and 34 of 2.3, Panel Checkout, and correct any erroneous program data values.
5. On LCDS, set HALT MODE switch to TTY and press INIT pushbutton. Verify that TTY prints out the following:

```
CL 0001
-
```

#### NOTE

If a wrong digit is accidentally typed when a numeric value is being entered in one of the steps that follow, press the ESCAPE key and observe that the TTY prints out a hyphen (-) to prompt for new input. Then, repeat the step to enter the correct value.

6. On TTY, type T 7701:7725 and press RETURN key. Verify that TTY prints out as follows:

```
7701 C4 20 31 C4 70 35 C4 77 C9 00 C4 07 C8 64 8F
7710 FF B8 60 9C FA 00 C4 7C C9 00 C4 07 C8 55 8F FF
7720 B8 51 9C FA 90 E1
```



7. On TTY, type G7701 and press RETURN key. Verify that first digit of digital readout alternately displays hexadecimal values A and b, and that A is first value displayed after RETURN key is pressed.
8. On LCDS, set HALT INST switch to DEBUG. Verify that after a few seconds digital readout blanks and TTY prints out:  
CL 7716  
-
9. On TTY, type G and press RETURN key. Verify that the following indications are obtained in the sequence listed:
  - a. Hexadecimal value b is displayed on first digit of digital readout for a few seconds, then display changes to hexadecimal value A.
  - b. After hexadecimal value A is displayed for a few seconds, digital readout blanks and TTY prints out:  
CL 7716  
-
10. On TTY, type A 7715, 08 and press RETURN key. Then, type H7724, and press RETURN key a second time.
11. On TTY, type G7701 and press RETURN key. Verify that the following indications are obtained in the sequence listed:
  - a. Hexadecimal value A is displayed on first digit of digital readout for a few seconds, then display changes to hexadecimal value b.
  - b. After hexadecimal value b is displayed for a few seconds, digital readout blanks and TTY prints out:  
CL 7724  
-
12. On TTY, type H and press RETURN key. Then type G and press RETURN key a second time. Verify that first digit of digital readout alternately displays hexadecimal values A and b, and that A is the first value displayed after the RETURN key is pressed.
13. On LCDS, set RUN MODE switch to STEP. Verify that digital readout blanks and that TTY prints out:  
CL "XXXX" (any 4-digit hexadecimal value between 7707 and 7724)  
-
14. On LCDS, press INIT switch and verify that TTY prints out:  
CL 0001  
-
15. On TTY, type G7701 and press RETURN key. Verify that TTY prints out:  
CL 7703  
-
16. On TTY, type G and press RETURN key. Verify that TTY prints out:  
CL 7704  
-
17. Turn off TTY and LCDS to erase stored program. Then turn TTY and LCDS back on, press INIT pushbutton, and verify that TTY prints out:  
CL 0001  
-
18. On TTY, type in program data value list as indicated below, then press RETURN key.  
-A7701, C4, 20, 31, C4, 70, 35, C4, 77, C9, 00, C4, 07, C8, 64, 8F, FF, B8, 60, 9C, FA, 00

19. On LCDS, set RUN MODE switch to CONT.
20. On TTY, type G7701 and press RETURN key. Verify that first digit of digital readout displays hexadecimal value A for a few seconds, then blanks. When digital readout blanks, verify that TTY prints out:  
 CL 7716  
 -
21. On TTY, type P7701:7716 and press RETURN key. Then set TTY tape punch to ON, type any character and verify that paper tape is punched. When tape halts, return TTY tape punch to OFF.
22. Momentarily turn off TTY and LCDS to erase stored program. When TTY and LCDS are turned back on, press INIT pushbutton; then tear off punched tape and insert tape into tape reader.
23. On TTY, type L and press RETURN key. Then set TTY tape reader to START and verify that tape advances through tape reader. When tape halts, return tape reader to STOP and remove tape from tape reader.
24. On TTY, type G7701 and press RETURN key. Verify that first digit of digital readout displays hexadecimal value A for a few seconds, then blanks. When digital readout blanks, verify that TTY prints out:  
 CL 7716  
 -
25. On TTY, type in alter register value list as indicated below; then press RETURN key.  
 -A77F5, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF, FF
26. On LCDS, set HALT MODE switch to PNL, then press TTY RETURN key and verify that digital readout displays FFFF PC.
27. Press the following pushbuttons and verify that the indications obtained on the digital readout are as specified.

<u>Switch</u>	<u>Indication</u>
READ PC	FFFF PC
READ P1	FFFF P1
READ P2	FFFF P2
READ P3	FFFF P3
READ AC	FF AC
READ EX	FF E
READ ST	FF S

28. Set HALT MODE switch to TTY. Then type TR on TTY, press RETURN key, and verify that the following printout is obtained:  
 PC      P1      P2      P3      AC   EX   SR  
 77F5 FF FF FF FF FF FF FF FF FF FF FF
29. Reinsert punched tape into TTY tape reader.
30. Press INIT pushbutton, type L7731 on TTY, and press TTY RETURN key. Then set TTY tape reader to START and verify that tape advances through tape reader. When tape halts, return tape reader to STOP, and remove tape from tape reader.
31. To complete the TTY checkout procedure, type G7731 and press RETURN key. Verify that first digit of digital readout displays hexadecimal value A for a few seconds, then blanks. When LCDS digital readout blanks, verify that TTY prints out:

CL 7746  
 -

Table 2-3. TTY Checkout Description

Step	Purpose	Reference
1-4	These are preliminary steps that set up the LCDS for checkout of TTY.	Not Applicable
5	This step verifies that TTY prints out the saved PC address following initialization of SC/MP Microprocessor.	3.2.1
6	This step verifies that Type Address Range Command causes the contents of a selected range of memory addresses to be printed out by TTY.	3.4.5
7	This step verifies that GO command can be used to select RUN Mode operation.	3.4.2
8, 9	These steps verify that GO (Continue) command initiates RUN Mode operation at saved PC address.	3.4.2
10, 11	These steps verify that the Breakpoint Halt command can be used to enter a Halt Instruction at a desired memory address, and that GO (Start Select) command can be used to initiate RUN Mode operation at a specific starting address.	3.4.2
12	This step verifies that Halt Reset command can be used to reset a breakpoint Halt Instruction.	3.4.2
13-16	These steps verify that TTY printout is proper when single-instruction execution of a program is selected.	3.4.2
17-20	These steps verify that contents of a selected range of memory locations can be altered via Alter Address Range command.	3.4.6
21, 22	These steps verify that the Punch command can be used to cause contents of a series of memory locations to be punched into paper tape.	3.4.7
23, 24	These steps verify that the Load command can be used to read a punched tape into memory starting at address specified on tape.	3.4.8
25-28	These steps verify that Alter Memory Address Range command can be used to alter the values saved for SC/MP program counter, registers, and accumulator and that Type Registers command can be used to cause saved values to be printed out on TTY keyboard.	3.4.4
29-31	These steps verify that Load at Specified Address command can be used to read a punched tape into memory starting at memory address specified in command.	3.4.8



## 2.5 INSTALLATION OF OPTIONAL CONNECTORS

If desired, an optional 72-pin connector may be installed at location J5 to permit plug-in interconnection of one additional SC/MP application card, and 50-pin ribbon connectors may be installed at locations J6 and J7 to facilitate interconnection of custom application systems. Manufacturer's part numbers for these connectors are listed below.

1. <u>72-Pin Connector</u>	
Source	Part Number
C. D. C.	VPB04B36A00A1E
Stanford Applied Eng.	CDP7000-72
2. <u>50-Pin Connectors</u>	
Source	Part Number
3M	3433-2003

To minimize the possibility of damaging the LCDS printed-circuit connections, a 40-watt (maximum) soldering iron and 60/40 resin-core solder should be used in installing the optional connectors. To install the connectors, proceed as follows.

1. If applicable, turn off power to the LCDS; then tag and remove all external connectors from TB1.
2. Remove SC/MP CPU Application Card from motherboard connector.
3. Using a Number-2 Phillips-head screwdriver, loosen and remove the four outermost screws that secure the motherboard to the LCDS chassis.
4. Remove the motherboard from the LCDS chassis to gain access to the bottom of the board for soldering purposes.
5. Orient the connector to be installed such that pin 1 is aligned with the square mounting hole at the desired motherboard location; then insert the connector pins into the predrilled holes provided on the motherboard. (For connectors J6 and J7, the orientation specified will result in the keyed slots facing the rear of the motherboard.)

### CAUTION

Use extreme care when soldering connector pins to lands on motherboard. Excess heat or force could cause lands to peel from motherboard.

6. Solder the connector pins to the lands provided on the bottom of the motherboard. After all pins are soldered, clean excess flux using cleaning agent such as Freon, acetone, or isopropyl alcohol (100% dry).

### CAUTION

If a continuity checker is used to verify that the connector pins were soldered properly, the output of the continuity checker must not exceed 1.5 volts. An output of greater than 1.5 volts could possibly damage the LCDS circuitry even though power is not applied to the LCDS.

7. Visually examine each solder connection for bridges, cold joints, and so forth; if a connection is in doubt, use a continuity checker having an output of 1.5 volts (or less) to test for opens and shorts.
8. Replace motherboard on LCDS chassis and reinstall Phillips-head screws to secure motherboard in place.
9. If applicable, reinstall all tagged external connections to TB1.
10. Replace SC/MP CPU Application Card in motherboard connector.

Chapter 3  
OPERATION

This chapter covers basic SC/MP LCDS operation in detail, and further describes how the LCDS may be used with an SC/MP applications system for hardware and software development purposes. The information is presented in the following sequence.

- CONTROLS, CONNECTORS, AND INDICATORS (3.1). Illustrates the physical location and describes the purpose and function of each LCDS control, connector, and indicator — including the jumper options available for applications system control.
- OPERATING MODE DESCRIPTION (3.2). Describes basic LCDS operation in the RUN and DEBUG Modes, LCDS Power-Up and Initialization, and transfer of operating control between the LCDS Panel and the optional Teletype<sup>®</sup> (TTY).
- PANEL OPERATING PROCEDURES (3.3). Provides detailed instructions for operating the LCDS via the Panel controls.
- TTY OPERATING PROCEDURES (3.4). Provides detailed instructions for operating the LCDS via the optional Teletype (TTY).
- APPLICATIONS SYSTEM INTERFACING (3.5). Covers applications system interconnection and address assignment.
- PROGRAM DEVELOPMENT ON LCDS (3.6). Describes the use of special features designed into the LCDS to simplify development of application programs.

3.1 CONTROLS, CONNECTORS, AND INDICATORS

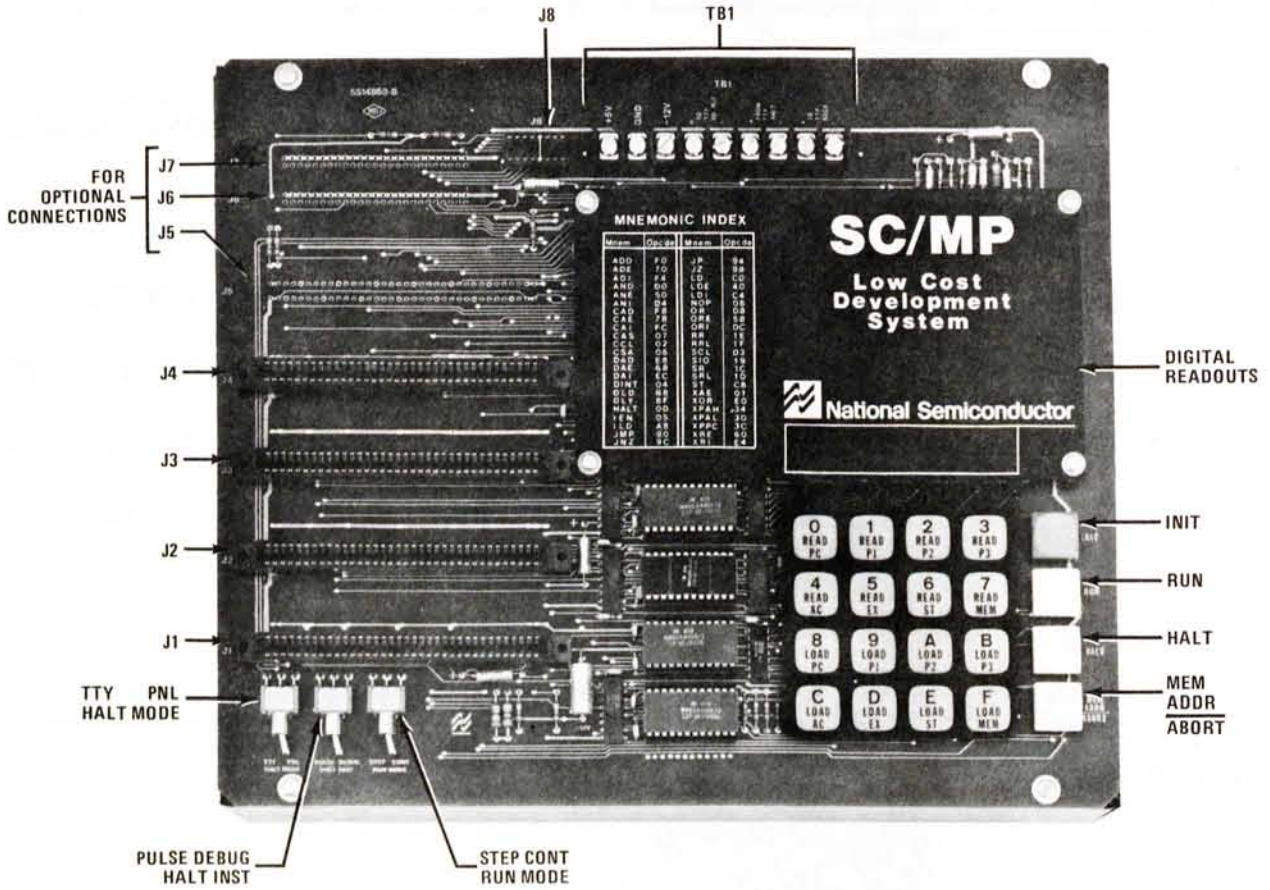
LCDS controls, connectors, and indicators are shown and described in figure 3-1 and table 3-1, respectively. Under the "reference" column of table 3-1, the heading number(s) under which operational use of the subject control, connector, or indicator is described. The functions of the jumper options available at connector J8 are covered in table 3-2.

Table 3-1. Description of Controls, Connectors, and Indicators

Control, Connector, or Indicator	Description	Reference
INIT pushbutton	Provides initialization output over SC/MP Application Card bus to initialize SC/MP Microprocessor and any application system circuitry utilizing INIT* signal. Thus, when pressed during RUN Mode Operation, terminates program execution and causes LCDS to enter DEBUG Mode; when pressed during DEBUG Mode Operation, terminates command in progress and causes LCDS to return to DEBUG Mode entry point.	3.2.1

<sup>®</sup> Registered trademark of the Teletype Corporation.





NS10716

Figure 3-1. LCDS Controls, Connectors, and Indicators



Table 3-1. Description of Controls, Connectors, and Indicators (Continued)

Control, Connector, or Indicator	Description	Reference
HALT pushbutton	Functional only during RUN Mode Operation of LCDS. When pressed in RUN Mode, terminates program execution (after instruction in progress is completed), and causes LCDS to enter DEBUG Mode.	3.3.1 and 3.4.2
MEM ADDR/ABORT pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is enabled. When pressed while numeric input is in progress, terminates command and causes digital readout to display small zeroes (0000 00). When pressed while numeric input is not in progress, causes digital readout to display hyphens (----) to prompt for memory-address input.	3.3.2, 3.3.3.3, and 3.3.3.4
RUN pushbutton	<p>Functional only when LCDS is in DEBUG Mode and Panel Operation is enabled. When pressed, terminates DEBUG Mode, and causes execution of user-entered program to be initiated at stored SC/MP program-counter address.</p> <p style="text-align: center;">NOTE</p> <p>In addition to providing the functions listed below, the digital readout and the keyboard switches may also be employed by the user's applications program to provide special-purpose RUN Mode functions. Instructions for programming the digital readout and the keyboard switches are provided under 3.6, Program Development on LCDS.</p>	3.3.1
Digital Readout	Six-character, 7-segment display associated with examining and altering the contents of memory locations and the values saved for the SC/MP program counter, registers, and accumulator.	Not Applicable
READ PC/0 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored PC value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 0.	3.3.2 and 3.3.3.1
READ P1/1 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored P1 value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 1.	3.3.2 and 3.3.3.1

Table 3-1. Description of Controls, Connectors, and Indicators (Continued)

Control, Connector, or Indicator	Description	Reference
READ P2/2 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored P2 value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 2.	3.3.2 and 3.3.3.1
READ P3/3 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored P3 value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 3.	3.3.2 and 3.3.3.1
READ AC/4 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored AC value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 4.	3.3.2 and 3.3.3.1
READ EX/5 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored Extension (E) Register value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 5.	3.3.2 and 3.3.3.1
READ ST/6 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes stored Status (S) Register value to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 6.	3.3.2 and 3.3.3.1
READ MEM/7 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, causes address and contents of stored memory reference address to be displayed on digital readout; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 7.	3.3.2 and 3.3.3.3



Table 3-1. Description of Controls, Connectors, and Indicators (Continued)

Control, Connector, or Indicator	Description	Reference
LOAD PC/8 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored PC value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 8.	3.3.2 and 3.3.3.2
LOAD P1/9 pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored P1 value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value 9.	3.3.2 and 3.3.3.2
LOAD P2/A pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored P2 value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value A.	3.3.2 and 3.3.3.2
LOAD P3/B pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored P3 value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value B.	3.3.2 and 3.3.3.2
LOAD AC/C pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored AC value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value C.	3.3.2 and 3.3.3.2
LOAD EX/D pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored Extension (E) Register value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value D.	3.3.2 and 3.3.3.2
LOAD ST/E pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored Status (S) Register value; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value E.	3.3.2 and 3.3.3.2



Table 3-1. Description of Controls, Connectors, and Indicators (Continued)

Control, Connector, or Indicator	Description	Reference
LOAD MEM/F pushbutton	Functional only when LCDS is in DEBUG Mode and Panel Operation is selected, or when addressed by the user's application program while RUN Mode Operation is enabled. When pressed while keyboard is in Command Mode, enables alteration of stored memory-reference address contents; when pressed while keyboard is in Numeric Mode, outputs hexadecimal value F.	3.3.2 and 3.3.3.4
RUN MODE switch	Associated with RUN Mode operation of LCDS. When set to STEP, enables single-instruction execution of user-entered program; when set to CONTIN enables normal execution of user-entered program.	3.3.1
HALT INST switch	Associated with RUN Mode operation of LCDS. When set to DEBUG, LCDS terminates RUN Mode and enters DEBUG Mode upon execution of Halt Instruction; when set to PULSE, RUN Mode operation is not affected by execution of Halt Instruction.	3.3.1 and 3.4.2
HALT MODE switch	Associated with DEBUG operation of LCDS. When set to PNL, enables DEBUG operational commands to be entered via Panel keyboard; when set to TTY, enables DEBUG operational commands to be entered via TTY keyboard.	3.3.1 and 3.4.2
Connectors J1-J4	Prewired interface connectors that enable plug-in interconnection of SC/MP family Application Cards for hardware- and software-development purposes.	3.5
Connector J5 (optional, user-supplied)	Expands plug-in interface capability for SC/MP family Application Cards.	3.5
Connectors J6 and J7 (optional, user-supplied)	Allows use of flat cable to expand Applications Card Bus for user's system.	3.5
Connector J8	Enables jumper interconnection for various control applications.	3.5.2 and 3.5.5
Terminal Strip TB1	Enables interconnection of primary dc input power and optional TTY.	Chapter 2

Table 3-2. Jumper Options Available at Connector J8

Pin	Signal	Description
1	BA09	Buffered address bit 09 — available for external address decoding.
2	BA11	Buffered address bit 11 — available for external address decoding.
3	BA13	Buffered address bit 13 — available for external address decoding.
4	BA15	Buffered address bit 15 — available for external address decoding.
5	TTY	Factory-installed jumper permits buffered TTY output data to be transmitted to SC/MP Microprocessor via BSENSE input. See pin 12 description.
6	BA12	Buffered address bit 12 — available for external address decoding.
7	BA10	Buffered address bit 10 — available for external address decoding.
8	DEBUG*	LCDS control signal input; when low, causes LCDS to terminate RUN Mode operation and enter DEBUG Mode.
9	INDBG*	LCDS status signal output; held continuously low while LCDS is in DEBUG Mode.
10	BA14	Buffered address bit 14 — available for external address decoding.
11	USER 4	Provided for user implementation in special-purpose applications; not presently used by LCDS or SC/MP family application cards.
12	BSENSE	Sense B input to SC/MP Microprocessor; normally jumpered to pin 5 to permit reception of TTY data.
13	EN1	SC/MP CPU Application Card general-purpose driver enable input/output (EN1 input is inverted on SC/MP CPU Application Card to provide EN1* output); used typically for control of SC/MP CPU Application Card memory.
14	EN1*	
15	ROMSEL	SC/MP CPU Application Card Read-Only Memory Select Input. When high, enables Read-Only Memory on CPU card if MEMSEL* signal is also high; when low, disables Read-Only Memory on CPU Card.
16	RAMSEL	SC/MP CPU Application Card Read/Write Memory Select Input. When high, enables CPU card Read/Write Memory if MEMSEL* signal is also high; when low, disables CPU card Read/Write Memory.

### 3.2 OPERATING MODE DESCRIPTION

The LCDS has two basic modes of operation, the RUN Mode and the DEBUG Mode. In the RUN Mode, the LCDS circuits provide a monitor function and the SC/MP CPU Application Card operates as part of the users application system, so a user-entered program may be tested in an actual operating environment. In the DEBUG Mode, the LCDS circuits provide control functions that enable the SC/MP CPU Application Card to execute the resident DEBUG firmware without affecting the status of the user's applications hardware or software. Execution of the DEBUG firmware, in turn, permits the user to communicate with the LCDS either via the Operators Control Panel (Panel) or via an optional TTY. The DEBUG Mode capabilities include the following.

#### NOTE

The LCDS reserves addresses 7000 through 7FFF for DEBUG operations. All of the remaining addresses (0000 through 6FFF and 8000 through FFFF) are available for assignment by the user as desired.



Panel Capabilities

- Display contents of SC/MP program counter, registers, and accumulator in hexadecimal format
- Display contents of any memory location in hexadecimal format
- Alter contents of SC/MP program counter, registers, and accumulator
- Alter contents of any memory location
- Initiate execution of user-entered program at any memory address
- Interrupt execution of user-entered program at any point without loss of hardware or software status

Optional TTY

- Print contents of SC/MP program counter, registers, and accumulator
- Print contents of any single memory location, or any selected range of memory locations
- Alter contents of SC/MP program counter, registers, and accumulator
- Alter contents of any single memory location, or any selected range of memory locations
- Set breakpoint halts in Read/Write Memory that enable interruption of user-entered program at any point without loss of hardware or software status
- Initiate execution of user-entered program at any memory address
- Obtain paper tape Load Module by punching selected range of memory locations to paper tape
- Load LCDS generated paper-tape Load Module into memory
- Load IMP-16 or FORTRAN Cross Assembler generated paper-tape Load Module into memory

When the DEBUG Mode is selected, it is entered via a Save Routine that is initiated when the current RUN Mode instruction is completed. The purpose of the Save Routine is to store the contents of the SC/MP program counter, registers, and accumulator in memory locations 77F5 through 77FF (refer to table 3-3) to permit a subsequent return to the RUN Mode without loss of status. During the Save Routine, the contents of the program counter (PC) are incremented by one prior to storage so that they point to the address of the first instruction that will be fetched when the RUN Mode is subsequently selected; the contents of the remaining SC/MP registers and accumulator are not altered by the Save Routine. Upon completion of the Save Routine, the LCDS displays or prints out the stored PC contents to indicate entry into the DEBUG Mode.

Table 3-3. DEBUG Save Addresses for SC/MP Program Counter, Registers, and Accumulator

Memory Address	Stored Value
77F5	Program Counter high-order byte
77F6	Program Counter low-order byte
77F7	Register P1 high-order byte
77F8	Register P1 low-order byte
77F9	Register P2 high-order byte
77FA	Register P2 low-order byte
77FB	Register P3 high-order byte
77FC	Register P3 low-order byte
77FD	Contents of Accumulator
77FE	Contents of Extension Register
77FF	Contents of Status Register



All of the DEBUG capabilities associated with displaying and altering the contents of the SC/MP program counter, registers, and accumulator are with respect to the values stored in memory locations 77F5 through 77FF. Thus, the user can check the status that was present when the last RUN Mode instruction was completed and can make any necessary changes. When the RUN Mode is subsequently selected, the DEBUG Mode is terminated via a Return Routine that enters the contents of memory locations 77F5 through 77FF into the SC/MP program counter, registers, and accumulator. RUN Mode operation then is initiated at the restored PC address. (Users familiar with operation of the SC/MP CPU Applications Card should note that the DEBUG Return Routine decrements the value contained in memory locations 77F5 and 77F6 before restoring it to the PC. This compensates for the automatic PC increment that occurs prior to the fetch of the first RUN Mode instruction.)

### 3.2.1 Power-up and Initialization

When power is turned on to the LCDS, a Power-up Initialization Circuit on the SC/MP CPU Application Card provides a low-going INIT\* output that initializes the SC/MP microprocessor and places the LCDS in the DEBUG Mode. The LCDS then displays 0001 PC on the Panel or prints out CL 0001 on the TTY to indicate entry into the DEBUG Mode. After the power-up initialization cycle is completed, the SC/MP microprocessor may be reinitialized at any time by pressing the INIT switch or by applying a low-going INIT\* pulse of at least four micro-cycles duration to the SC/MP CPU Application Card (refer to SC/MP CPU Application Card Data Sheet). If reinitialization occurs while the LCDS is in the RUN Mode, the RUN Mode will be terminated after the instruction in progress is completed; then, the DEBUG Save Routine will be executed to store the initialized contents of the SC/MP program counter, registers, and accumulator in memory locations 77F5 through 77FF. After the DEBUG Save Routine is completed, the LCDS will provide one of the indications described previously (0001 PC or CL 0001) to indicate entry into the DEBUG Mode. If reinitialization occurs while the LCDS is in the DEBUG Mode, the command in progress will be terminated, and the LCDS will repeat the DEBUG Save Routine to store the initialized contents of the SC/MP program counter, registers and accumulator in memory locations 77F5 through 77FF. Then, the LCDS will display 0001 PC or print out CL 0001 to indicate a return to DEBUG operation.

### 3.2.2 Panel/TTY Selection

Operation of the LCDS via the Panel or the optional TTY is selected by setting the HALT MODE switch to PNL or TTY, respectively. Control will then be transferred to the appropriate location as follows:

1. If the HALT MODE switch setting is changed while the LCDS is in the RUN Mode, control will not be transferred to the selected location until the LCDS subsequently enters the DEBUG Mode. Upon entering the DEBUG Mode, the LCDS then will display or print out the saved PC contents at the appropriate location.
2. If the HALT MODE switch is changed from PNL to TTY while the LCDS is in the DEBUG Mode, control will be transferred automatically to the TTY unless the Panel is in the Numeric Mode (refer to 3.3.2, DEBUG Commands). When the Panel is in the Numeric Mode, control will not be transferred to the TTY until the command in progress is completed, or the MEM ADDR/ABORT switch is pressed to terminate the command.
3. If the HALT MODE switch setting is changed from TTY to PNL when the LCDS is in the DEBUG Mode, control will be transferred to the Panel when the TTY RETURN key is pressed.

#### NOTE

The INIT and HALT pushbuttons and the TTY HALT INST switch remain functional regardless of whether LCDS operation is being controlled via the TTY or the Panel.

## 3.3 PANEL OPERATING PROCEDURES

The paragraphs that follow provide detailed instructions for operating the LCDS via the Panel.



### 3.3.1 RUN/DEBUG Mode Selection

The RUN Mode is selected by pressing the RUN pushbutton and the DEBUG Mode is selected by default when the RUN Mode is terminated. Panel pushbuttons and switches associated with RUN Mode Operation are RUN, HALT, RUN MODE, and HALT INST. All of the remaining Panel pushbuttons and switches, except the INIT pushbutton, are associated with the DEBUG Mode Operation. Use of the INIT pushbutton is covered in 3.2.1.

To select the RUN Mode of operation, proceed as follows.

#### NOTE

The following procedure assumes that the user's application program has previously been loaded into memory.

1. Set the RUN MODE switch to CONTIN to enable normal execution of the user's applications program or to STEP to enable single-instruction execution of the user's applications program. When the RUN MODE switch is set to STEP, one instruction will be executed each time that the RUN MODE switch is pressed, and the LCDS will return automatically to the DEBUG Mode. When the RUN MODE switch is set to CONTIN, the user's applications program will be executed continuously while the RUN Mode is enabled.
2. Set the HALT INST switch to DEBUG or PULSE, respectively, to enable or disable DEBUG halting of the user's applications program. When the HALT INST switch is set to DEBUG, execution of an applications program Halt Instruction causes the RUN Mode to be terminated and the DEBUG Mode to be entered. When the HALT INST switch is set to PULSE, RUN Mode Operation is not affected by execution of an applications program Halt Instruction.
3. Press the READ PC pushbutton and observe the PC address displayed on the digital readout. If this is the desired starting address for the user's application program, press the RUN pushbutton to initiate execution of the applications program at that address. If a different starting address is desired, press the LOAD PC pushbutton and key in the desired address. Then, press the RUN pushbutton to initiate program execution.

After the RUN pushbutton is pressed, the LCDS digital readout will remain blanked until the RUN Mode is terminated and the DEBUG Mode is entered. (Upon entry to the DEBUG Mode, the LCDS will display the saved PC address on the digital readout.) Termination of the RUN Mode and entry into the DEBUG Mode will occur under any of the following conditions.

1. The INIT pushbutton is pressed (refer to 3.2.1).
2. The HALT pushbutton is pressed. (When the HALT pushbutton is pressed, termination of the RUN Mode will occur when the instruction in progress is completed.)
3. The RUN/STEP switch is set to STEP, and one instruction is executed.
4. The HALT Mode switch is set to DEBUG, and a Halt Instruction is executed.
5. The DEBUG input to the LCDS is externally driven low (refer to 3.5.4).

### 3.3.2 DEBUG Commands

When the LCDS is in the DEBUG Mode, the keyboard pushbuttons on the Panel function under control of the DEBUG firmware to provide either command or numeric outputs. The command outputs of the pushbuttons are indicated by the screened legends (READ PC, READ P1, etc.) and the numeric outputs are indicated by the screened hexadecimal values (0, 1, 2, and so forth). Operation of the DEBUG firmware is such that the command outputs of the pushbuttons are always enabled except when a numeric value must be entered to complete a command (LOAD PC, LOAD P1, and so forth). When this occurs, the DEBUG firmware enables the numeric outputs of the pushbuttons, and the LCDS prompts for input by displaying hyphens (---- or --) in the appropriate digits of the digital readout. Pressing the keyboard switches then causes hexadecimal values to be entered (left to right) and displayed on the appropriate digits of the digital readout. When the last value is entered, the DEBUG firmware automatically executes the command and reenables the command outputs

of the switches. To abort a command while the numeric outputs of the switches are enabled (one or more hyphens are displayed on the digital readout), press the MEM ADDR/ABORT pushbutton. The LCDS then will display small zeroes (0000 00) on the digital readout to indicate that the command has been terminated without execution, and the DEBUG firmware will reselect the command outputs of the pushbuttons.

NOTE

Whenever data are loaded into a memory location via the LOAD MEM switch, the LCDS reads the stored data back and compares them with the original input to verify that the input was loaded properly. If the two values differ, 0000 00 is displayed on the digital readout to indicate the discrepancy. Thus, if an attempt is made inadvertently to write data into a Read-Only Memory location, the error will be indicated via the digital readout.

3.3.3 DEBUG Operation

Procedures for operating the LCDS in the DEBUG Mode, are provided in the paragraphs that follow.

3.3.3.1 Examining SC/MP Program Counter, Registers, and Accumulator

To examine the values stored for the SC/MP program counter, registers or accumulator, simply press the appropriate READ pushbutton. The contents of the selected register will then be displayed on the digital readout (as indicated below) until another command is subsequently entered.

NOTE

The term "XX" or "XXXX" is used to denote uncertainty of hexadecimal value displayed.

<u>Pushbutton Pressed</u>	<u>Indication</u>
READ PC	"XXXX" PC
READ P1	"XXXX" P1
READ P2	"XXXX" P2
READ P3	"XXXX" P3
READ AC	"XX" A
READ EX	"XX" E
READ ST	"XX" S

3.3.3.2 Altering SC/MP Program Counter, Registers, and Accumulator

To alter the values stored for the SC/MP program counter, registers, or accumulator, press the appropriate LOAD pushbutton and observe that the LCDS prompts for input as indicated below. Then, key in the desired hexadecimal digits via the keyboard pushbuttons and observe that corresponding values are displayed on the digital readout. If a wrong digit is accidentally keyed in, press the MEM ADDR/ABORT pushbutton to terminate the command in progress. Then, reselect the command via the appropriate LOAD pushbutton and key in the required number of digits. When the last digit is keyed in, the new value will be stored automatically in memory to replace the previous value.



<u>Pushbutton Pressed</u>	<u>Prompt Indication</u>
LOAD PC	---- PC
LOAD P1	---- P1
LOAD P2	---- P2
LOAD P3	---- P3
LOAD AC	-- A
LOAD EX	-- E
LOAD ST	-- S

### 3.3.3.3 Examining Contents of a Memory Location

The contents of a memory location may be examined either via the MEM ADDR/ABORT pushbutton or via the READ MEM pushbutton. Operation of the pushbuttons is as follows.

1. When the MEM ADDR/ABORT pushbutton is pressed while no command is in progress, the LCDS prompts for a 4-digit address input by displaying hyphens (----) on the digital readout. After the desired 4-digit address is keyed in, the contents of that memory location are displayed automatically on the last two digits of the digital readout.
2. When the READ MEM switch is pressed, the DEBUG firmware refers to a stored memory-reference address to determine which memory location was selected for display. The address of the selected memory location then is displayed on the first four digits of the digital readout, and the contents of the memory location are displayed on the last two digits. During DEBUG operation, the DEBUG firmware automatically sets the stored memory-reference address to the last 4-digit hexadecimal value displayed on the digital readout. In addition, each time that the contents of a memory location are examined via the MEM ADDR/ABORT or READ MEM pushbuttons, the DEBUG firmware also enables a READ MEM auto-increment loop. The auto-increment loop then remains enabled until some other command is entered (READ PC, READ AC, LOAD MEM, or another). Thus, if the last command processed was MEM ADDR or READ MEM, pressing the READ MEM pushbutton will cause the stored memory-reference address to be incremented by one before the current READ MEM command is executed; if the READ MEM pushbutton is pressed following any other command, the memory location accessed will correspond to the last 4-digit hexadecimal value previously displayed on the digital readout. Examples of using the READ MEM pushbutton are provided below.

<u>Command Sequence</u>	<u>Digital Readout Indication</u>
INIT	0001 PC
READ P1	0000 P1
READ MEM	0000 "XX"
READ MEM	0001 "XX"
READ MEM	0002 "XX"
READ AC	00 A
READ MEM	0002 "XX"
READ P1	0000 P1
MEM ADDR (enter 7800)	---- / 7800 C8
READ MEM	7801 C4
READ MEM	7802 08
READ P2	0000 P2
READ MEM	0000 "XX"
LOAD P2 (enter 7810)	---- P2 / 7810 P2
READ MEM	7810 31

### 3.3.3.4 Altering Contents of a Memory Location

The contents of a memory location are altered, first, by pressing the LOAD MEM pushbutton and, then, by keying in the desired two-digit hexadecimal value. When the LOAD MEM pushbutton is pressed, the DEBUG firmware refers to a stored memory-reference address to determine the memory location to be accessed; then, the DEBUG firmware causes the address of the selected memory location to be displayed on the first four digits of the digital readout followed by two hyphens ("XXXX" --). When the desired 2-digit hexadecimal value is then keyed in, it is entered automatically into the selected memory location. If a wrong digit is keyed in accidentally, press the MEM ADDR/ABORT pushbutton to terminate the command in progress. Then, press the MEM ADDR/ABORT pushbutton a second time and key in the appropriate 4-digit memory address. After the memory address is keyed in, press the LOAD MEM pushbutton and key in the correct data value.

Derivation of the stored memory-reference address is covered under 3.3.3.3, Examining Contents of a Memory Location. Use of the LOAD MEM pushbutton does not affect the basic derivation of this address; however, it does reset the READ MEM auto-increment loop (if it was in effect) and does enable a LOAD MEM auto-increment loop. The LOAD MEM auto-increment loop then remains enabled until some other command is entered (READ PC, LOAD AC, READ MEM, or another). Thus, if the last command processed was LOAD MEM, pressing the LOAD MEM pushbutton again will cause the stored memory-reference address to be incremented by one before the current LOAD MEM command is processed; if the LOAD MEM pushbutton is pressed following any other command, the memory location accessed will correspond to the last 4-digit hexadecimal value previously displayed on the digital readout. Examples of using the LOAD MEM switch are provided below.

<u>Command Sequence</u>	<u>Indication</u>
INIT	0001 PC
READ P1	0000 P1
LOAD P1 (enter 7700)	---- P1 / 7700 P1
LOAD MEM (enter AA)	7700 -- / 7700 AA
READ MEM	7700 AA
LOAD MEM (enter BB)	7700 -- / 7700 bb
LOAD MEM (enter CC)	7701 -- / 7701 CC
LOAD MEM (enter DD)	7702 -- / 7702 dd
MEM/ADDR (enter 7700)	---- / 7700 bb
READ MEM	7701 CC
LOAD MEM (enter 11)	7701 -- / 7701 11
READ AC	00 A
LOAD MEM (enter 22)	7701 -- / 7701 22
READ MEM	7701 22

#### NOTE

The following TTY operating procedures require that the Auto-Answerback Option on the TTY be disabled. Instruction for disabling the Auto-Answerback Option is provided in 2.4.1.

### 3.4 TTY OPERATING PROCEDURES

LCDS operation may be controlled via the TTY keyboard when the LCDS is in the DEBUG Mode, the TTY LINE/OFF/LOCAL switch is set to LINE and the LCDS HALT MODE switch is set to TTY. Descriptions of the TTY commands and examples showing their use are provided in the paragraphs that follow; notations and symbols used in the descriptions and examples are covered in table 3-4.



Table 3-4. TTY DEBUG Command Notations and Symbols

Notation/Symbol	Meaning
<p>Underscored characters, numbers, and symbols (XXXX)</p>	<p>Underscored characters, numbers, and symbols indicate inputs that are entered by the user at the TTY keyboard. For example:</p> <ul style="list-style-type: none"> <li>- <u>T 100</u> (DEBUG firmware prompt/user input)</li> <li>- <u>T 200:204</u> (DEBUG firmware prompt/user input)</li> </ul>
<p>Non-underscored characters, numbers, and symbols (XXXX)</p>	<p>Non-underscored characters, numbers, and symbols indicate outputs from the DEBUG firmware that are printed by the TTY. For example:</p> <ul style="list-style-type: none"> <li>- <u>T 100</u> <u>CR</u> (DEBUG firmware prompt/user input)</li> <li>0100 <u>XX</u> (DEBUG firmware output, output undefined)</li> <li>- <u>T 200:203</u> (DEBUG firmware prompt/user input)</li> <li>0200 <u>hv hv hv hv</u> (DEBUG firmware output and prompt for next command)</li> </ul>
<p>hv</p>	<p>Denotes a 2-digit hexadecimal value that is typed in or printed out via the TTY keyboard. If more than the desired number of digits are typed in, the DEBUG firmware only accepts the last two digits. Leading zeroes may be omitted or included, as desired.</p>
<p>ma</p>	<p>Denotes a 4-digit hexadecimal value that specifies a memory address. If more than four digits are typed in, the DEBUG firmware only accepts the last four digits. Leading zeroes may be omitted or included, as desired.</p>
<p>hvr</p>	<p>Denotes a value list comprised of 2-digit hexadecimal values separated by commas. Any digits omitted between commas are interpreted as zeroes. Thus, A would be interpreted as 0A, 00.</p>
<p>mar</p>	<p style="text-align: center;">NOTE</p> <p>A memory-address range must be located on a single "page" or "wraparound" will occur; refer to 3.4.1, TTY DEBUG Commands.</p>
<p>comm</p>	<p>Denotes a memory-address range consisting of a memory address (ma), a colon (:), and a second memory address (ma); that is, mar = ma:ma. All locations from the first entry through the last are included in the range. For example, 001A:02FF signifies all memory locations from 001A through 02FF including 001A and 002F. The memory address to the left of the colon represents the low limit of the range, and the address to the right represents the high limit. If the upper limit of the range is smaller than the lower limit, the DEBUG firmware accepts only the left number and executes the specified command at that address.</p>
<p>, (comma)</p>	<p>Denotes comment consisting of English-language text, including letters and numbers. Associated with TTY utilities covered under 3.6, Program Development on LCDS.</p> <p>Used during entry of commands to separate hexadecimal values (hv) from one another, or from another element of the command statement. When commas are used to separate hexadecimal values, any digits omitted are treated as zeroes; thus, AA, ,BB would be treated as AA, 00, BB.</p>



Table 3-4. TTY DEBUG Command Notations and Symbols (Continued)

Notation/Symbol	Meaning
: (colon)	Used during entry of commands to separate the first entry of a memory address range from the second entry.
[ ]	Indicates that the element(s) specified within the brackets are optional and can be included in or left out of the command statement, as desired.
- (hyphen)	Prompt symbol printed out by DEBUG firmware on TTY keyboard to indicate that it is ready to accept a command input.
ⒸR	Indicates that RETURN key on TTY keyboard should be pressed to complete command input.
ⒻF	Indicates that the LINE FEED key should be pressed to cause the TTY keyboard to index to the next line.

### 3.4.1 TTY DEBUG Commands

When TTY operation is selected and the LCDS is in the DEBUG Mode, the TTY keyboard switches function under control of the DEBUG firmware to provide either command or numeric functions. The command functions of the switches are described in table 3-5, and the numeric functions are indicated by the screened legends on the appropriate switches (0 through F). Operation of the DEBUG firmware is such that the command functions of the switches are always enabled except when a numeric value must be entered to complete a command. When this occurs, the DEBUG firmware will accept a numeric input from the switches until the RETURN key is pressed to enable execution of the command. After the command is executed, the DEBUG firmware then causes the TTY to print out a hyphen (-) to indicate that the command was executed properly and that a new command can be entered. If the DEBUG firmware detects an error when a command or numeric value is being entered, it automatically terminates the command and causes the TTY to print out a question mark (?) to indicate detection of the error. The question mark then will be followed by a hyphen to prompt for a new command input.

To abort a command, simply type one of the following illegal characters: ALTMODE, ESCAPE, or CTRL/X. When any of these characters are typed, the DEBUG firmware automatically terminates the command in progress and causes the TTY to print out a hyphen, thereby prompting for a new command input.

When a numeric value is being entered, typing more than the required number of digits causes the earlier digits to be discarded. If, for example, 123400124 were typed in as a 4-digit address, 0124 would be the value accepted by the DEBUG firmware. When less than the required number of digits is typed in before pressing the RETURN key, the DEBUG firmware automatically treats the missing digits as leading zeroes. Thus, for example, if 12 were typed in as a 4-digit address, the DEBUG firmware would accept the value as 0012.

Whenever the DEBUG firmware is printing out information via the TTY keyboard or outputting data to the tape punch, the TTY keyboard is also being scanned for input. If input is detected, the output in progress will be terminated, then the DEBUG firmware will print out a hyphen on the TTY keyboard to prompt for a new command input. This feature is especially useful for terminating excessive or undesired output.

NOTE

When an SC/MP Pointer Register (PC, P1, P2, or P3) is incremented, only the 12 low-order bits are affected; the 4 high-order bits remain at the value to which they were previously set via initialization of the SC/MP microprocessor, or via an XPPC, XPAH, or Transfer (JMP, JP, JZ, and JNZ) Instruction. The term 'page', therefore, is used in the following paragraphs to indicate the group of 4,096 memory locations specified by the value of the Pointer Register four high-order bits (that is, X000 through XFFF). For reference purpose, SC/MP page designations are listed in table 3-6.

The TTY Type Register Range and Punch DEBUG commands permit a range of memory addresses to be specified in the command. The only restriction on specifying a range is that the entire range must be contained within a single page to prevent "wraparound" from occurring. During execution of commands that specify a range, the DEBUG firmware employs an SC/MP Pointer Register as a reference counter (1) by setting the Pointer Register to the appropriate starting address and, then, (2) by automatically incrementing the Pointer Register to permit printout of the specified memory data. Wraparound will occur, therefore, if the Pointer Register is incremented through a page boundary; that is, if the Pointer Register reaches the end of the page to which it has been previously set, it will automatically loop back to the start of the same page and thereby cause an erroneous printout. For example, if memory addresses, 3FFE through 4001 were specified, the printout would actually indicate the contents of memory locations 3FFE, 3FFF, 3000, and 3001.

The Alter Address Range Command permits modification of data in a range of memory locations, so, again, the entire range must be on a single page to prevent wraparound from occurring during automatic incrementing of the SC/MP Pointer Register used as a reference counter. In addition, since new instructions or data values are entered via this command, the user must ensure that the existing Pointer Register page boundaries are observed within the applications program to prevent wraparound from occurring during PC-Relative, Indexed, or Auto-Indexed addressing. Detailed information on PC-Relative, Indexed, and Auto-Indexed addressing is provided in the SC/MP Programming and Assembler Manual.

The Load at Specified Address command causes a paper-tape Load Module to be read into contiguous memory locations starting at the address specified in the command instead of the address specified on the tape. The user must take care, therefore, to ensure that the starting address does not cause wraparound to occur during loading of the tape or during future execution of the applications program.

Table 3-5. TTY DEBUG Commands

Command	Format	Description
TYPE COMMANDS		
Type Address	T ma	<p>This command causes the address of the specified memory location to be printed out in 4-digit hexadecimal notation, and the contents of the memory location to be printed out in a 2-digit hexadecimal notation. For example:</p> <p style="margin-left: 40px;">- T 200 <b>CR</b> (DEBUG firmware prompt/user input)</p> <p style="margin-left: 40px;">0200 hv (DEBUG firmware output and prompt for next command)</p> <p style="margin-left: 40px;">- /</p> <p>(contents of 0200)</p>



Table 3-5. TTY DEBUG Commands (Continued)

Command	Format	Description
TYPE COMMANDS (Continued)		
Type Address Range	T mar	<p style="text-align: center;">NOTE</p> <p>The range specified by the Type Address Range command must be located on a single "page" or "wraparound" will occur; refer to 3.4.1, TTY DEBUG Commands.</p> <p>This command causes the contents of the specified memory locations to be printed out in sequence. The printout is formatted to 16 memory locations (maximum) per line. The first entry on each line is a 4-digit hexadecimal value that denotes the starting address of the line, and the remaining entries are 2-digit hexadecimal values that indicate the contents of the memory locations covered by the line. For example:</p> <pre> - T 200:215 CR (DEBUG firmware prompt/user input) 0200 hv, hv, hv, ..... hv       /   /   /   /   /   /       (contents of 0200) (contents of 0201) (contents of 020F)       }       }       } 0210 hv, hv, hv, hv, hv, hv - / / / / /   (contents of 0210) (contents of 0215)           </pre> <p style="text-align: right;">} DEBUG firmware output and prompt for next command</p>
Type Registers	TR	<p>This command automatically accesses memory locations 77F5 through 77FF to cause the values stored for the SC/MP program counter, registers, and accumulator to be printed out. Headings are included and the values of the registers are shown in 2-digit hexadecimal notation with the high-order byte of a 16-bit register preceding the low-order byte. For example:</p> <pre> - TR CR (DEBUG firmware prompt/user input) 77F5 PC P1 P2 P3 AC EX SR       hv hv hv hv hv hv hv hv hv hv hv hv - / / / / /   (high-order byte) (low-order byte)           </pre> <p style="text-align: right;">} DEBUG firmware output and prompt for next command</p>
ALTER COMMANDS		
Alter Address	A ma, hv	<p>This command causes a 2-digit hexadecimal value to be entered into the memory location specified; the previous contents of the memory location are lost. For example, - A 200, 1F CR would cause the hexadecimal value 1F to be stored at memory location 0200.</p> <p style="text-align: center;">NOTE</p> <p>The memory locations referenced by the Alter Address Range command must all be on the same page or wrap-around will occur; refer to 3.4.1, TTY DEBUG Commands.</p>
Alter Address Range	A ma, hvr	<p>This command causes 2-digit hexadecimal values to be entered into a series of memory locations starting at the memory address specified. For example, typing - A 100, 1F, 2F, 3F, 4F, 5F, 6F CR would cause the value 1F to be entered into memory location 0100, the value 2F to be entered into memory location 0101, and so forth.</p>



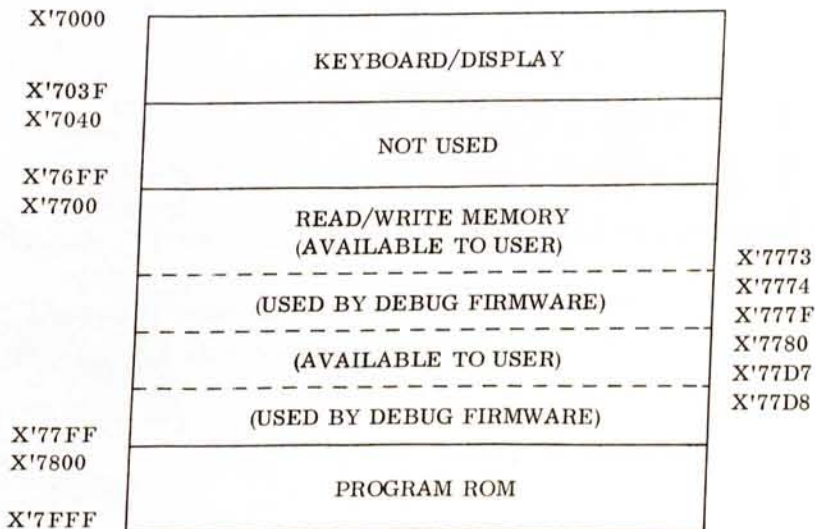
Command	Format	Description
<b>BREAKPOINT COMMANDS</b>		
Breakpoint Halt	H ma	This command enters a Halt Instruction into the specified memory location after saving the original contents of the memory location. For example typing - H 100 <u>CR</u> would cause a Halt Instruction to be stored at memory location 0100. Only one breakpoint halt may be in effect at one time; if a second breakpoint halt command is entered, the DEBUG firmware resets the first breakpoint halt by restoring the original contents to the memory location used for the first breakpoint halt. The Breakpoint Halt command will work properly only when the HALT INST switch is set to DEBUG.
Breakpoint Halt Reset	H	This command cancels a previous breakpoint halt command without setting a new breakpoint halt. Cancellation of the previous breakpoint halt command is effected by restoring the original contents to the memory location used for the previous breakpoint halt.
<b>GO COMMANDS</b>		
GO (Continue)	G	This command causes RUN Mode operation to be initiated at the PC address stored in memory locations 77F5 and 77F6.
GO (Start Select)	G ma	This command specifies the address at which RUN Mode operation will be initiated. For example, typing - G 100 will cause RUN Mode operation to be initiated at address 0100.
<b>PUNCH COMMAND</b>		
Punch	P mar	<p style="text-align: center;">NOTE</p> <p>The memory-address range specified by the Punch command must be located on a single memory "page" or "wraparound" will occur; refer to 3.4.1, TTY DEBUG Commands.</p> <p>This command causes the contents of the specified memory locations to be punched into paper tape in SC/MP Load Module Format (refer to SC/MP Programming and Assembler Manual). The contents of the specified memory locations are not altered.</p>
<b>LOAD COMMANDS</b>		
Load at Tape Address	L	This command enables an LM tape to be loaded into memory at the addresses as specified on the tape.  <p style="text-align: center;">NOTE</p> <p>The starting address selected via the Load at Specified Address command must permit the entire tape to be loaded on a single "page" or "wraparound" will occur; refer to 3.4.1, TTY DEBUG Commands.</p>
Load at Specified Address	L ma	This command causes the LCDS to ignore the starting address specified on the LM tape and, instead, loads the LM tape into memory in contiguous locations starting at the address specified in the command.

Table 3-6. SC/MP Microprocessor Page Designations

Page	Memory Range	Hexadecimal Value of PC Address Bits (Note 1)															
		15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0000 through 0FFF	0				X				X				X			
1	1000 through 1FFF	1				X				X				X			
2	2000 through 2FFF	2				X				X				X			
3	3000 through 3FFF	3				X				X				X			
4	4000 through 4FFF	4				X				X				X			
5	5000 through 5FFF	5				X				X				X			
6	6000 through 6FFF	6				X				X				X			
7	7000 through 7FFF (Note 2)	7				X				X				X			
8	8000 through 8FFF	8				X				X				X			
9	9000 through 9FFF	9				X				X				X			
A	A000 through AFFF	A				X				X				X			
B	B000 through BFFF	B				X				X				X			
C	C000 through CFFF	C				X				X				X			
D	D000 through DFFF	D				X				X				X			
E	E000 through EFFF	E				X				X				X			
F	F000 through FFFF	F				X				X				X			

- Notes:
1. X indicates don't care state for address bits.
  2. Addresses 7000 through 7FFF are reserved by the LCDS for DEBUG operations as shown below. Thus, the Read/Write Memory locations not used by the DEBUG firmware (7700 through 7773 and 7780 through 77D7) are available to the user without restriction; the remaining addresses in the range 7000 through 7FFF, however, may be specified only in a user's application program as indicated under 3.6, Program Development on LCDS.

LCDS Memory Map





### 3.4.2 RUN/DEBUG Mode Selection

The RUN Mode is selected by typing - G CR or G ma CR on the TTY, and the DEBUG Mode is selected by default when the RUN Mode is terminated. LCDS switches and TTY DEBUG commands associated with RUN Mode operation are the INIT pushbutton, the HALT switch, the HALT INST switch, the RUN Continue Command, the RUN Start Select Command, the Breakpoint Halt Command, and the Breakpoint Halt Reset Command. Use of the INIT pushbutton is covered under 3.2.1, Power-Up and Initialization. Use of the remaining switches and TTY commands are covered in the instructions that follow.

To select the RUN Mode of operation, proceed as follows.

1. Set RUN Mode switch to CONTIN to enable normal execution of applications program or to STEP to enable single-instruction execution of the applications program. When RUN MODE switch is set to CONTIN, applications program will be executed continuously until RUN Mode is terminated. When RUN Mode switch is set to STEP, one instruction will be executed each time that RUN Mode is selected; then, LCDS will return automatically to DEBUG Mode.
2. Set HALT INST switch to DEBUG or PULSE, respectively, to enable or disable DEBUG halting of applications program. When HALT INST switch is set to DEBUG, RUN Mode will be terminated and DEBUG Mode will be entered whenever a Halt Instruction is executed by applications program. When HALT INST switch is set to PULSE, RUN Mode Operation is not affected by execution of an applications program Halt Instruction. A Halt Instruction may be temporarily entered at desired applications program address via TTY Breakpoint Halt Command. The Halt Instruction then will remain in effect until either a second Breakpoint Halt Command or a Breakpoint Halt Reset Command is executed. Since only one Breakpoint Halt Command may be in effect at one time, each succeeding Breakpoint Halt Command resets previous Breakpoint Halt Command by restoring original contents of memory location at which Halt Instruction was stored temporarily. When a Breakpoint Halt Reset Command is executed, last Breakpoint Halt Command is reset by restoring original contents to appropriate memory location without loading a Halt Instruction into a new memory location. To enter a Breakpoint Halt Command, type

- H ma CR (for example, typing - H 100 CR would cause a Halt Instruction to be stored at memory location 0100)

To enter a Breakpoint Halt Reset Command, type

- H CR

3. Type - TR CR and observe resultant printout on the TTY. If PC value indicated is desired starting address for application program, type - G CR to initiate RUN Mode Operation at that address. If a different starting address is desired, type - G ma CR to initiate RUN Mode Operation at address specified by ma value.

After the RUN Command is entered, the TTY keyboard will be disabled until the DEBUG Mode is subsequently entered. Upon entry into the DEBUG Mode, the TTY will print out: - CL hv to indicate the saved PC value. Termination of the RUN Mode and entry into the DEBUG Mode will occur under any of the following conditions.

1. The INIT switch is pressed (refer to 3.2.1, Power-up and Initialization).
2. The HALT switch is pressed (RUN Mode termination occurs after the instruction in progress is completed).
3. The RUN/STEP switch is set to STEP. (RUN Mode termination occurs after one instruction is executed.)
4. The HALT INST switch is set to DEBUG and a Halt Instruction is executed.
5. The DEBUG input to the LCDS is externally driven low (refer to 3.5.4, DEBUG\* Signal Implementation).



### 3.4.3 Examining SC/MP Program Counter, Registers, and Accumulator

To examine the values stored for the SC/MP program counter, registers, and accumulator, type: - TR CR . The TTY will then print out the values stored in memory locations 77F5 through 77FF (refer to table 3-3) in the following format:

	PC	P1	P2	P3	AC	EX	SR
77F5	hv hv	hv hv	hv hv	hv hv	hv hv	hv hv	hv hv

### 3.4.4 Altering SC/MP Program Counter, Registers, and Accumulator

The values stored for the SC/MP program counter, registers, and accumulator may be altered via the Alter Memory Address or Alter Memory Address Range Commands. Thus, for example, typing - A 77F5, hv, hv CR will cause a new 4-digit PC value to be stored at memory locations 77F5 and 77F6, and typing - A 77FD, hv CR will cause a new AC value to be stored at memory location 77FD.

### 3.4.5 Examining Contents of a Memory Location

The contents of a single memory location may be examined via the Type Address Command and the contents of a series of memory locations may be examined via the Type Address Range Command. Examples of using these commands are provided in table 3-5.

### 3.4.6 Altering Contents of a Memory Location

The contents of a single memory location may be altered via the Alter Address Command and the contents of a series of memory locations may be altered via the Alter Address Range Command. Examples of using these commands are provided in table 3-5.

### 3.4.7 Punching a Paper Tape

To punch the contents of a series of memory locations onto paper tape, proceed as follows:

1. Type - P mar CR . (for example, typing - P 100:110 CR would cause the contents of memory locations 0100 through 0110 to be punched onto the paper tape after steps 2 and 3 are executed.)
2. Set Paper Tape Punch to ON.
3. Type any character and observe that tape is punched out.

When tape stops, tear off tape and set Paper Tape Punch to OFF.

#### NOTE

The LCDS will accept paper tapes generated via the IMP-16 and FORTRAN Cross Assemblers as well as tapes previously generated by the LCDS.

### 3.4.8 Loading a Paper Tape

To read a paper tape into memory, proceed as follows:

1. Raise cover on Tape Reader, install tape on sprocket, and then close cover to hold tape in place.
2. If it is desired to load the tape into memory using addresses specified on the tape, type - L CR. If it is desired to load into memory starting at some other address, type - L ma CR. (For example, typing - L 100 CR would cause the tape to be loaded into memory starting at address 0100.)

#### NOTE

No adjustment of the object code is attempted. The data on the tape are merely read into contiguous memory locations starting at the specified address.

3. Set Tape Reader to START and observe that tape feeds through. When tape stops, remove tape and set Tape Reader to STOP.

### 3.5 APPLICATION SYSTEM INTERFACING

The LCDS interface bus (figure 3-2) is prewired to permit an SC/MP CPU Application Card to function as part of the user's application system when the LCDS is in the RUN Mode, and as part of the LCDS when the LCDS is in the DEBUG Mode. The way that the LCDS accomplishes this is by reserving addresses 7000 through 7FFF for DEBUG operations and by permitting the remainder of the 64K addresses to be assigned as desired by the user. From a software standpoint, therefore, the RUN Mode covers the address range: 0000-6FFF and 8000-FFFF.

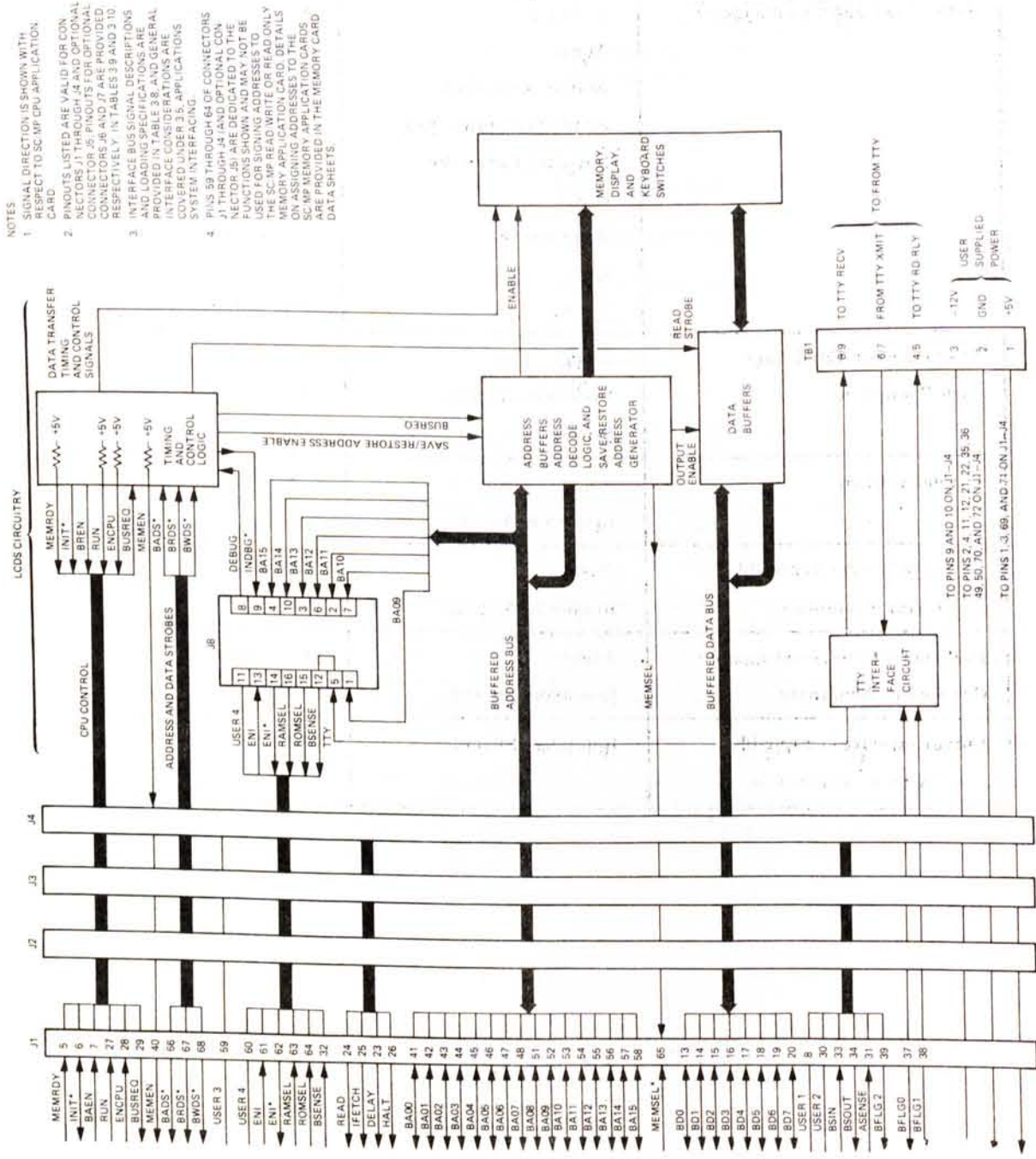
For definition purposes, the term "user's application system" is considered to include all circuits that provide an operational interface with the SC/MP CPU Applications Card when the LCDS is in the RUN Mode. Thus, for small-scale applications, the user's application system may be nothing more than the 256-by-8-bit Read/Write Memory on the SC/MP CPU Application Card. For large-scale applications, the user's application system may consist of a variety of SC/MP family application cards, peripherals, and/or special-purpose custom memory and control circuits.

Connectors J1 through J4 provide a 72-pin plug-in interface for SC/MP family application cards (all cards must be installed with component side facing front of LCDS). If desired, one more 72-pin connector may be installed at socket location J5 to accommodate an additional SC/MP family application card, and ribbon connectors may be installed at motherboard locations J6 and J7 to accommodate custom applications system designs (refer to chapter 2, Installation). Sources of accessory equipment useful for fabricating and interconnecting custom application cards are listed in table 3-7.

Table 3-7. Sources of Accessory Equipment

Equipment	Source	Part Number
72-contact Edge Connector	C. D. C.	VPB04B36A00A1E
	Augat	14005-17P3
	Robinson-Nugent	EC-721
	Stanford Applied Eng.	CDP7000-72
	National Connector	900100-36
	Cinch	50-72C-30
	Winchester	HW36C0111
	Elco	00-6307-072-309-001
	Viking	3VH36/LJND5
13-connector Card Cage with Backplane	Augat	8170-MG1
	Robinson-Nugent	MECA-1
	Scanbe	
Extender Card	Augat	8136-MG13
	Robinson-Nugent	EB-72
Universal wire-wrappable Card with Terminals	Augat	8136-UMG1
	Robinson-Nugent	UNI-24
High-density wire-wrappable Card with Terminals	Augat	8136-MG15
	Robinson-Nugent	
Universal wire-wrappable Card without Terminals	Robinson-Nugent	(Special)





- NOTES:
1. SIGNAL DIRECTION IS SHOWN WITH RESPECT TO SC MP CPU APPLICATION CARD.
  2. PINS LISTED ARE VALID FOR CONNECTORS J1 THROUGH J4 AND OPTIONAL CONNECTOR J5. PINS OUTS FOR OPTIONAL CONNECTORS J6 AND J7 ARE PROVIDED, RESPECTIVELY, IN TABLES 3.9 AND 3.10.
  3. INTERFACE BUS SIGNAL DESCRIPTIONS AND LOADING SPECIFICATIONS ARE PROVIDED IN TABLE 3.8, AND GENERAL INTERFACE CONSIDERATIONS ARE COVERED UNDER 3.5, APPLICATIONS SYSTEM INTERFACING.
  4. PINS 59 THROUGH 64 OF CONNECTIONS J1 THROUGH J4 (AND OPTIONAL CONNECTION J5) ARE DESIGNATED TO THE CPU CONTROL AND MAY NOT BE USED FOR ADDRESSING ADDRESSES TO THE SC MP READ WRITE OR READ ONLY MEMORY APPLICATION CARD. DETAILS ON ASSIGNING ADDRESSES TO THE SC MP MEMORY APPLICATION CARDS ARE PROVIDED IN THE MEMORY CARD DATA SHEETS.

NS10512

Figure 3-2. Application System Interface Bus

### 3.5.1 Interface Bus Signal Description

LCDS interface bus signal descriptions are provided in table 3-8; all signals are listed in numeric order according to pin assignment on connectors J1-J4. Tables 3-9 and 3-10 cross-reference the pin assignments of the optional ribbon connectors (J6 and J7) to the descriptions in table 3-8.

Table 3-8. SC/MP Application Card Interface Bus Description

Pin	Signal	Description
1	+5V	Five-volt primary dc power for SC/MP Application Cards.
2	GND	Power and signal ground for SC/MP Application Cards.
3	+5V	Five-volt primary dc power for SC/MP Application Cards.
4	GND	Power and signal ground for SC/MP Application Cards.
5	MEMRDY	Input to SC/MP CPU Application Card. When set low prior to trailing edge of BRDS* or BWDS* strobe, stretches strobe to extend data input/output cycle; that is, strobe is held low until MEMRDY is returned high. Pulled up to +5 volts via resistor on LCDS; not used on SC/MP Memory Application Cards.
6	INIT*	SC/MP CPU Application Card input/output signal used for SC/MP and external system initialization. When power is first turned on, Power-up Initialization Circuit on SC/MP CPU Application Card provides low-going INIT* output to initialize SC/MP Microprocessor and to place the LCDS in the DEBUG Mode. After power-up initialization, SC/MP may be reinitialized by driving INIT* line low for at least four microcycles or by pressing INIT pushbutton on LCDS. When reinitialization occurs while RUN Mode operation is enabled, RUN Mode is automatically terminated and LCDS is placed in DEBUG Mode. When reinitialization occurs while DEBUG Mode is enabled, command in progress is aborted and LCDS returns to DEBUG entry point. The INIT* signal is not used by SC/MP Memory Application Cards.
7	BAEN	Control input to SC/MP CPU Application Card. When high, enables card to output BA00-BA15 address during data input/output cycle; when low, holds address bits in high-impedance (TRI-STATE) state to allow forcing of externally generated addresses onto Address Bus. Set low by LCDS during DEBUG Save and Restore Routines to control operation of SC/MP CPU Application Card. Not used by SC/MP Memory Application Cards.
8	USER 1 <i>not used</i>	Provided for user implementation in special-purpose applications. Not now used by LCDS or SC/MP application cards; may be used by future SC/MP application cards.
9, 10	-12V	Twelve-volt primary dc power for SC/MP CPU and Read-Only Memory Application Cards.
11, 12 13-20	GND	Power and signal ground.  Buffered 8-bit input/output data bus, which is set to high-impedance state, except when actually in use by SC/MP CPU Application Card (BADS*, BRDS*, or BWDS* low). During address interval (BADS* low) of data input/output cycle, SC/MP CPU Application Card outputs address and status data over bus; during ensuing data-transfer interval (BRDS* or BWDS* low), SC/MP CPU Application Card transmits or receives 8-bit data byte to/from application system memory (RUN Mode) or LCDS memory (DEBUG Mode).



Table 3-8. SC/MP Application Card Interface Bus Description (Continued)





Pin	Signal	CPU Output at BADS* Time		CPU Input at BRDS* Time	CPU Output at BWDS* Time
		Functional Name	Description		
13	BD0	BA12	Fourth most significant bit of 16-bit address.	 Input data are expected on the eight (BD0-BD7) lines. 	 Output data are valid on the eight (BD0-BD7) lines. 
14	BD1	BA13	Third most significant bit of 16-bit address.		
15	BD2	BA14	Second most significant bit of 16-bit address.		
16	BD3	BA15	Most significant bit of 16-bit address		
17	BD4	READ	When high, data input cycle is starting; when low data output cycle is starting		
18	BD5	IFETCH	When high, first byte of instruction is being fetched; processed by LCDS to control entry into DEBUG Mode.		
19	BD6	DELAY	When high indicates that delay cycle is starting; that is, second byte of Delay Instruction is being fetched.		
20	BD7	HALT	When high, indicates that Halt Instruction has been executed; causes LCDS to terminate RUN Mode and enter DEBUG Mode when HALT INST switch is set to DEBUG.		
Pin	Signal	Description			
21, 22	GND	Power and signal ground for SC/MP Application Cards.			
23	DELAY	Latched Delay Status Flag output of SC/MP CPU Application Card. Latched high on leading edge of BADS* strobe to indicate that delay cycle is starting (that is, second byte of Delay Instruction is to be fetched during data input/output cycle in progress); otherwise, latched low on leading edge of BADS* strobe. Not used by LCDS or SC/MP Memory Application Cards.			
24	READ	Latched Read Status Flag output of SC/MP CPU Application Card. Latched high or low, respectively, on leading edge of BADS* strobe to indicate whether read or write data input/output cycle is in progress. Not used by LCDS or SC/MP Memory Application Cards.			



Table 3-8. SC/MP Application Card Interface Bus Description (Continued)

Pin	Signal	Description
25	IFETCH	Latched Instruction Fetch Status Flag output of SC/MP CPU Application Card. Latched high on leading edge of BADS* strobe when first byte of instruction is to be fetched during data input/output cycle in progress; otherwise latched low on leading edge of BADS* strobe. Not used by LCDS or SC/MP Memory Application Cards.
26	HALT	Latched Halt Status Flag output of SC/MP CPU Application Card. Latched high on leading edge of BADS* strobe to indicate that a Halt Instruction has been executed; otherwise latched low on leading edge of BADS* strobe. Not used by LCDS or SC/MP Memory Application Cards.
27	RUN	Run Enable Input to SC/MP CPU Application Card. When high, enables normal program execution by SC/MP Microprocessor; when low, SC/MP operation is suspended (after completion of current instruction) without loss of internal status. Pulled up to +5 volts via resistor on LCDS; not used by SC/MP Memory Application Cards.
28	ENCPU	Bus Access Enable input to SC/MP CPU Application Card. When high, enables SC/MP Microprocessor data input/output cycle execution; when low, inhibits SC/MP Microprocessor data input/output cycle execution. Pulled up to +5 volts via resistor on LCDS; not used by SC/MP Memory Application Cards.
29	BUSREQ	Bus Request output of SC/MP CPU Application Card. Goes high when SC/MP microprocessor requests bus to initiate data input/output cycle and remains high until data input/output cycle is completed. Processed by LCDS to control outputting of addresses during DEBUG Save and Restore Routines. Not used by SC/MP Memory Application Cards.
30	USER 2 <i>not used</i>	Provided for user implementation in special-purpose applications. Not now used by LCDS or SC/MP application cards; may be used by future SC/MP application cards.
31	ASENSE	Sense/Interrupt Request input to SC/MP CPU Application Card. Serves as interrupt request when SC/MP Microprocessor internal IE (interrupt enable) Flag is set, and as general-purpose sense input when IE Flag is reset. Not used by LCDS or SC/MP Memory Application Cards.
32	BSENSE	General-purpose sense condition input to SC/MP CPU Application Card. Factory-installed jumper on LCDS connector J8 (refer to table 3-2) dedicates this input to reception of TTY output data. Not used by SC/MP Memory Application Cards.
33	BSIN	Buffered Serial Data Input to SC/MP CPU Application Card. When SIO (Serial Input/Output) Instruction is executed, data on this line are right shifted into SC/MP microprocessor 8-bit Extension Register. Not used by LCDS or SC/MP Memory Application Cards.
34	BSOUT	Buffered Serial Data Output of SC/MP Microprocessor. When SIO (Serial Input/Output) Instruction is executed, data are right shifted onto this line from SC/MP Extension Register. Not used by LCDS or SC/MP Memory Application Cards.
35, 36	GND	Power and signal ground for SC/MP application cards.
37	BFLGO	Buffered General-Purpose Flag Output of SC/MP CPU Application Card. During DEBUG Mode, used by LCDS to transmit data to TTY. Not used by SC/MP Memory Application Cards.

Table 3-8. SC/MP Application Card Interface Bus Description (Continued)

Pin	Signal	Description
38	BFLG1	Buffered General-Purpose Flag Output of SC/MP CPU Application Card. During DEBUG Mode, used by LCDS to control TTY tape-reader relay. Not used by SC/MP Memory Application Cards.
39	BFLG2	Buffered General-Purpose Flag Output of SC/MP CPU Application Card. Not used by LCDS or SC/MP Memory Application Cards.
40	MEMEN	Address Compare Enable input to SC/MP Memory Application Cards. When high, enables addressing of SC/MP Memory Application Cards; when low, inhibits addressing of SC/MP Memory Application Cards. Pulled up to +5 volts via resistor on LCDS. Not used by SC/MP CPU Application Card.
41	BA00 } BA01 } BA02 } BA03 } BA04 } BA05 } BA06 } BA07 }	Buffered 16-bit Address Bus, which is set to high-impedance state except when data input/output cycle is in progress. Used by SC/MP CPU Application Card to communicate with application system or LCDS memory except during DEBUG Save and Restore Routines. During DEBUG Save and Restore Routines, LCDS sets BAEN Signal low to disable address output of SC/MP CPU Application Card, and forces addresses onto bus to control DEBUG Mode entry and exit. SC/MP Memory Application Cards have internal comparator circuit that continually compares address on bus with address assigned to card to enable or disable card, as appropriate.
42		
43		
44		
45		
46		
47		
48		
49	See below	
50	See below	
51	BA08 } BA09 } BA10 } BA11 } BA12 } BA13 } BA14 } BA15 }	
52		
53		
54		
55		
56		
57		
58		
49, 50	GND	Power and signal ground for SC/MP application cards.
NOTE		
<p>LCDS interface connectors J1 through J4 are prewired to enable installation of SC/MP CPU Application Card at any location. Thus, pins 61 through 64 are dedicated to the functions listed below and may not be used for externally assigning addresses to SC/MP Memory Application Cards. Details on assigning addresses to SC/MP Memory Application Cards via on-card jumper options are provided in memory card data sheets.</p>		
59	<i>not used</i> USER 3	Provided for user implementation in special-purpose applications; not used by LCDS or SC/MP CPU Application Cards.
60	<i>not used</i> USER 4	Provided for user implementation in special-purpose applications; not used by LCDS or SC/MP Memory Application Cards. Also available at LCDS connector J8 (refer to table 3-2).
61	EN1	SC/MP CPU Application Card input/output signals typically used for on-card memory control. (EN1 input is inverted to provide EN1* output.) Both signals are also accessible at LCDS connector J8 (refer to table 3-2).
62	EN1*	



Table 3-8. SC/MP Application Card Interface Bus Description (Continued)

Pin	Signal	Description
63	RAMSEL	SC/MP CPU Application Card Read/Write Memory Select Input. When high, enables on-card Read/Write Memory if MEMSEL* input to SC/MP CPU Application Card is also high; when low, disables on-card Read/Write Memory. RAMSEL is accessible at LCDS connector J8 for jumper implementation of desired control function (refer to table 3-2).
64	ROMSEL	SC/MP CPU Application Card Read-Only Memory Select Input. When high, enables on-card Read-Only Memory if MEMSEL* input is also high; when low, disables on-card Read-Only Memory. ROMSEL is accessible at LCDS connector J8 for jumper implementation of desired control function (refer to table 3-2).
65	MEMSEL*	Wired-AND Memory Select Input to SC/MP CPU Application Card from LCDS and SC/MP Memory Application Cards. Held low during data input/output cycle when LCDS or Memory Application Card is addressed; at all other times, pulled up to +5 volts via resistor on SC/MP CPU Application Card. Thus, enables SC/MP CPU Card memory whenever application system or LCDS memory is not addressed.
66	BADS*	Buffered active-low Address Strobe output of SC/MP CPU Application Card. When low, indicates that valid address and status are present on system buses. Used by LCDS and SC/MP Memory Application Cards to enable comparison of address present on bus with preassigned address.
67	BRDS*	Buffered active-low Read Data Strobe output of SC/MP CPU Application Card. When low, processed by LCDS, SC/MP Memory Application Cards, and CPU card memory control circuits to output data onto Buffered Data Bus from addressed memory location. On trailing edge, data present on Buffered Data Bus are input to SC/MP Microprocessor.
68	BWDS*	Buffered active-low Write Data Strobe output of SC/MP CPU Application Card. When low, indicates that valid output data are present on Buffered Data Bus. Processed by LCDS, SC/MP Memory Application Cards, and CPU card Read/Write Memory control circuits to write data into addressed memory location.
69	+5V	Five-volt primary dc power for SC/MP application cards.
70	GND	Power and signal ground for SC/MP application cards.
71	+5V	Five-volt primary dc power for SC/MP application cards.
72	GND	Power and signal ground for SC/MP application cards.



Table 3-9. Ribbon Connector J6 (Optional) Pin Assignments

Pin	Signal	Refer to Table 3-8, Pin:
1	GND	2
2	BRDS*	67
3	GND	2
4	MEMSEL*	65
5	GND	2
6	USER 3	59
7	GND	2
8	BA14	57
9	GND	2
10	BA12	55
11	GND	2
12	BA10	53
13	GND	2
14	BA08	51
15	GND	2
16	BA06	47
17	GND	2
18	BA04	45
19	GND	2
20	BA02	43
21	GND	2
22	BA00	41
23	GND	2
24	BFLG2	39
25	GND	2

Pin	Signal	Refer to Table 3-8, Pin:
26	BFLG0	37
27	GND	2
28	BSOUT	34
29	GND	2
30	BSENSE	32
31	GND	2
32	USER 2	30
33	GND	2
34	ENC PU	28
35	GND	2
36	HALT	26
37	GND	2
38	READ	24
39	GND	2
40	BD7	20
41	GND	2
42	BD5	18
43	GND	2
44	BD3	16
45	GND	2
46	BD1	14
47	GND	2
48	USER 1	8
49	GND	2
50	INIT*	6

Table 3-10. Ribbon Connector J7 (Optional) Pin Assignments

Pin	Signal	Refer to Table 3-8, Pin:
1	GND	2
2	BWDS*	68
3	GND	2
4	BADS*	66
5	GND	2
6	USER 4	60
7	GND	2
8	BA15	58
9	GND	2
10	BA13	56
11	GND	2
12	BA11	54
13	GND	2
14	BA09	52
15	GND	2
16	BA07	48
17	GND	2
18	BA05	46
19	GND	2
20	BA03	44
21	GND	2
22	BA01	42
23	GND	2
24	MEMEN	40
25	GND	2

Pin	Signal	Refer to Table 3-8, Pin:
26	BFLG1	38
27	GND	2
28	BSIN	33
29	GND	2
30	ASENSE	31
31	GND	2
32	BUSREQ	29
33	GND	2
34	RUN	27
35	GND	2
36	IFETCH	25
37	GND	2
38	DELAY	23
39	GND	2
40	BD6	19
41	GND	2
42	BD4	17
43	GND	2
44	BD2	15
45	GND	2
46	BD0	13
47	GND	2
48	BAEN	7
49	GND	2
50	MEMRDY	5

### 3.5.2 SC/MP CPU Application Card Memory Control

The SC/MP CPU Application Card contains a 256-by-8-bit read/write memory and has provision for installation of an optional 512-by-8-bit read-only memory. Selection of the read/write or the read-only memory is effected, respectively, by the RAMSEL and ROMSEL inputs to the card as shown in figure 3-3. Thus, when the read/write memory or the read-only memory is to serve as part of the user's applications system during RUN Mode operation, address decoding may be necessary for proper control of the RAMSEL and ROMSEL signals. Address decoding options for RAMSEL and ROMSEL control are listed in table 3-11.

### 3.5.3 Application System Address Assignment

The address ranges available for assignment to the user's application system are 0000-6FFF and 8000-FFFF. If the user's application system includes SC/MP memory application cards that are installed in connector(s) J1 through J4, the internal jumper options on the memory cards must be used for address assignment and pins 59 through 64 of the memory cards must be left unterminated to permit normal operation of the SC/MP CPU Application Card. Instructions for assigning addresses to the memory cards via the internal jumper options are provided in the data sheets for the SC/MP Read/Write Memory and the SC/MP ROM/PROM Memory Application Cards.

For SC/MP application systems based on discrete components or custom interconnection of the SC/MP memory applications cards, the method employed for address assignment is left to the discretion of the user since a variety of schemes are possible (refer to SC/MP Technical Description, publication number 4200079).

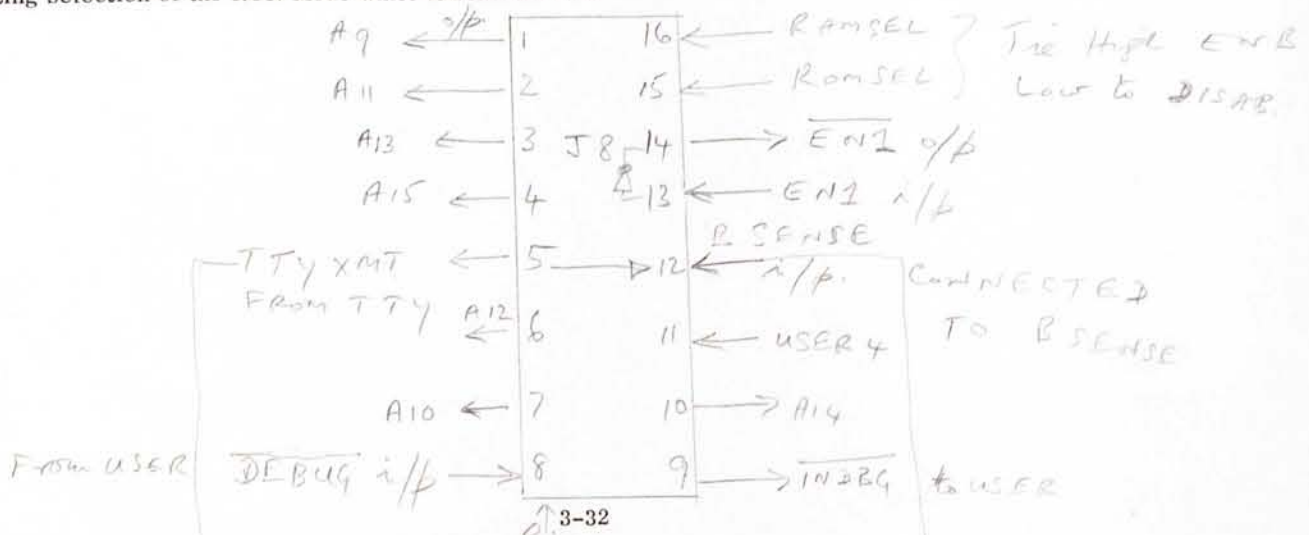
#### NOTE

As shown in figure 3-3, address bits BA09 through BA15 are brought out to LCDS connector J8 to provide an easy access for the user.

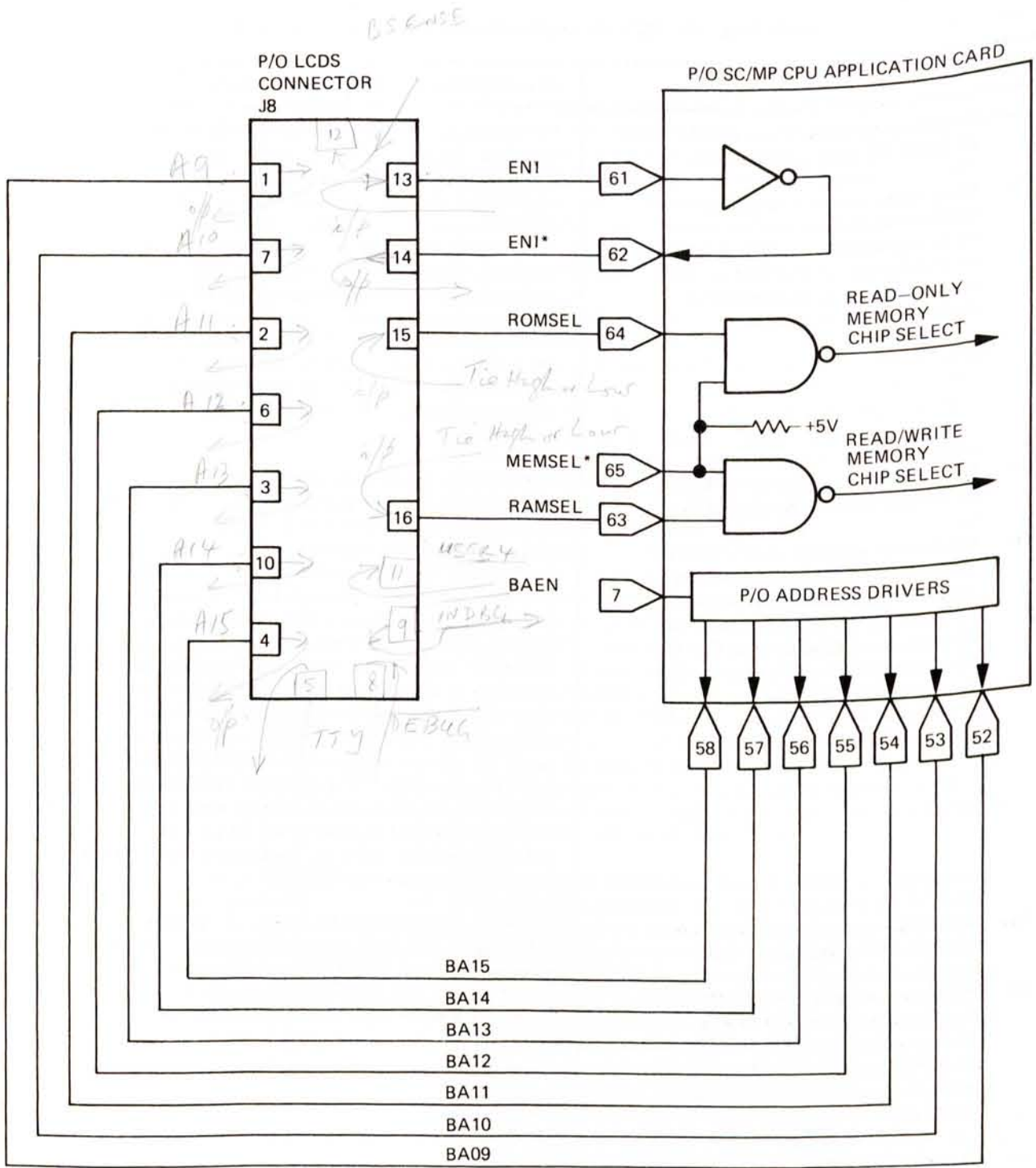
### 3.5.4 DEBUG\* Signal Implementation

For various applications system requirements it may be desirable to detect the occurrence of an asynchronous event and to monitor the operating conditions that were in effect at the time of the event. This can be readily accomplished by using the DEBUG\* input to the LCDS (available at pin 8 of connector J8). When the DEBUG\* signal goes low while the LCDS is in the RUN Mode, the RUN Mode is terminated and the DEBUG Mode is entered after the instruction in progress is completed. Thus, external circuits can be employed to drive the DEBUG\* signal low in response to an external event; then, the information desired can be obtained by examining the contents of memory locations and/or the values stored for the SC/MP program counter, registers, and accumulator. While the DEBUG\* signal remains low, the RUN Mode Operation is inhibited; the DEBUG\* signal must be reset high, therefore, before RUN Mode operation can be reenabled via the RUN switch or the TTY RUN Commands.

If the DEBUG\* signal goes low while the LCDS is in the DEBUG Mode, it will not affect LCDS operation except for inhibiting selection of the RUN Mode while it remains low.







NOTE: REFER TO TABLE 3-11 FOR A DESCRIPTION OF SC/MP CPU APPLICATION CARD MEMORY CONTROL OPTIONS

NS10513

Figure 3-3. Memory Control Circuit for SC/MP CPU Application Card

Table 3-11. SC/MP CPU Application Card Memory Control Options

Conditions	Decoding Required for RAMSEL and ROMSEL Signals
<p>1a. Read-only memory is not installed on SC/MP CPU Application Card.</p> <p>1b. Read/write memory on SC/MP CPU Application Card is used as stand-alone applications system or as part of applications system that consists exclusively of SC/MP application cards.</p>	<p>No special decoding is required for the RAMSEL and ROMSEL signals, but the RAMSEL signal should be tied to +5V via an 1-kilohm resistor to ensure a logic '1' input to the SC/MP CPU Application Card. For the conditions specified, the Read/Write Memory on the SC/MP CPU Application Card is controlled by the MEMSEL* input. When either the LCDS or the memory application cards are addressed, the MEMSEL* signal will be low and the read/write memory on the SC/MP CPU Application Card will be disabled; when neither the LCDS nor the memory application cards are addressed, the MEMSEL* signal will be high and the Read/Write Memory on the SC/MP CPU Application Card will be enabled.</p>
<p>2a. Read-only memory is not installed on SC/MP CPU Applications Card.</p> <p>2b. Read/write memory on SC/MP CPU Applications Card is used as part of applications system that contains custom memory and/or peripheral circuits in addition to or in lieu of SC/MP memory application card(s).</p>	<p>If the custom application system circuit is configured to provide a low-going MEMSEL* output to the SC/MP CPU Applications Card while addressed, no special decoding of the RAMSEL and ROMSEL signals is necessary, but the RAMSEL signal should be pulled up to +5V via a 1-kilohm resistor to ensure a logic '1' input to the SC/MP CPU Application Card. If the custom applications system circuit is not configured to provide a low MEMSEL* output while addressed, Address Bits BA09-BA15 can be decoded such that RAMSEL is driven high for a unique 256-address range and low at all other times, or RAMSEL can be tied to an address bit that is out of the address range of the applications system when RAMSEL goes high. For example, if the highest address assigned to the applications system were 4FFF, RAMSEL could be jumpered to address bit BA15. The low-order address of the SC/MP CPU Application Card Read/Write Memory then would be 8000.</p>
<p>3a. Read-only memory is installed on SC/MP CPU Applications Card.</p> <p>3b. Read-only memory on SC/MP CPU Application Card is used as stand-alone applications system or as part of applications system that consists exclusively of SC/MP application cards.</p>	<p>RAMSEL should be tied to ground and ROMSEL should be pulled up to +5V via a 1-kilohm resistor. The Read/Write Memory on the SC/MP CPU Application Card then will be held disabled, and the read-only memory will be controlled by the MEMSEL* input as described for condition 1.</p>
<p>3c. Read/write memory on SC/MP CPU Application Card is not used as part of applications system.</p>	



Table 3-11. SC/MP CPU Application Card Memory Control Options (Continued)

Conditions	Decoding Required for RAMSEL and ROMSEL Signals
<p>4a. Read-only memory is installed on SC/MP CPU Application Card.</p> <p>4b. Read-only memory on SC/MP CPU Application Card is used as part of applications system that contains custom memory and/or peripheral circuits in addition to or in lieu of SC/MP memory application cards.</p> <p>4c. Read/write memory on SC/MP CPU Application Card is not used as part of applications system.</p>	<p>Tie RAMSEL to ground to disable Read/Write Memory on SC/MP CPU Application Card. If the custom applications system circuit is configured to provide a low-going MEMSEL* output to the SC/MP CPU Application Card while addressed, no special decoding is necessary for the ROMSEL signal, but the ROMSEL signal should be pulled up to +5V via an 1-kilohm resistor to ensure a logic '1' input to the SC/MP CPU Application Card. If the custom applications system circuit is not configured to provide a low MEMSEL* output while addressed, address bits BA10-BA15 can be decoded such that ROMSEL is driven high for a unique 512-address range and low at all other times, or ROMSEL can be tied to an address bit that is out of the address range of the applications system when ROMSEL goes high. For example, if the highest address assigned to the applications system were 4FFF, ROMSEL could be jumpered to address bit BA15. The low-order address of the SC/MP CPU Application Card read-only memory would then be 8000.</p>
<p>5a. Read-only memory is installed on SC/MP CPU Application Card.</p> <p>5b. Read-only and read/write memories on SC/MP CPU Application Card are used as stand-alone applications system or as part of applications system that consists exclusively of SC/MP application cards.</p>	<p>One of the address bits available at J8 (BA09-BA15) should be jumpered to the EN1 and RAMSEL signals, and the EN1* signal should be jumpered to the ROMSEL signal. The read/write memory then will be enabled when the MEMSEL* signal and the selected address bit are high and the read-only memory will be enabled when the MEMSEL* signal is high and the selected address bit is low. The logical state of the MEMSEL* signal for the conditions specified is covered under condition 1; use of the EN1 and EN1* signals is covered in the data sheet for the SC/MP CPU Application Card.</p>
<p>6a. Read-only memory is installed on SC/MP CPU Application Card.</p> <p>6b. Read-Only and Read/Write Memories on SC/MP CPU Applications Card are used as stand-alone applications system or as part of applications system that consists exclusively of SC/MP application cards.</p>	<p>If the custom applications system circuit is configured to provide a low-going MEMSEL* output to the SC/MP CPU Applications Card while addressed, the RAMSEL and ROMSEL signals can be controlled as described for condition 5. If the custom applications system circuit is not configured to provide a low MEMSEL* output while addressed, address bits BA09-BA15 can be decoded such that RAMSEL and ROMSEL are driven high for unique 256- and 512-address ranges, respectively. Optionally, an address bit that is out of the range of the application system when the address bit is high can be ANDed externally with the BA09 address bit; the output of the AND gate is then connected to the RAMSEL and EN1 signals, and the ROMSEL signal is connected to the EN1* signal. If, for example, the highest address assigned to the applications system were 4FFF, the BA15 and BA09 address bits could be jumpered together. The address assigned to the SC/MP read/write memory then would be 8000-80FF and the low-order address of the read-only memory would be 8100.</p>



### 3.5.5 RUN Mode Considerations

Since the LCDS is primarily designed to permit the design of a user's application system to be tested in an operational environment, it enables all of the operating capabilities of the SC/MP CPU Application Card to be utilized during the RUN Mode Operation. For certain types of applications, however, special processing of the signals listed below may be necessary to ensure that the applications system does not interfere with LCDS operation during the DEBUG Mode, and the LCDS does not interfere with applications system operation during the RUN Mode.

1. BFLG0 and BFLG1 Signals. During DEBUG operation, the BFLG0 and BFLG1 outputs of the SC/MP CPU Application Card are used, respectively, to transmit data to the optional TTY and to control the TTY Reader Relay. Thus, if these signals are to be used also by the applications system during RUN Mode Operation, they should be applied to the applications system via a gate that is controlled by the INDBG\* status output of the LCDS (see figure 3-4).

#### NOTE

When the DEBUG Mode is entered, the contents of the SC/MP Status Register are saved in memory location 77FF. Thus, the logical states of BFLG0 and BFLG1 will be saved when the RUN Mode is terminated and reinstated when the RUN Mode subsequently is selected.

2. BSENSE Signal. During DEBUG Mode Operation, the BSENSE input to the SC/MP CPU Application Card is used for reception of data from the optional TTY. Thus, if the BSENSE signal is also to be used by the applications system during RUN Mode Operation, the jumper installed at the factory between pins 5 and 12 of connector J8 (refer to table 3-2) should be disconnected, and a single-throw switch should be installed instead. The switch then can be set to OFF to disconnect the BSENSE and TTY signals during RUN Mode Operation, and to ON to reconnect the signals during DEBUG Mode Operation.
3. Interrupts. If the SC/MP Internal IE Flag is set during RUN Mode Operation to enable interrupts, the ASENSE/Interrupt signal should be applied to the SC/MP CPU Applications Card via a gate that is controlled by the INDBG\* status output of the LCDS (see figure 3-4).

### 3.6 PROGRAM DEVELOPMENT ON LCDS

For detailed information on SC/MP program development and machine-language coding of SC/MP instructions, refer to the SC/MP Programming and Assembler Manual (publication number 4200094). The paragraphs that follow describe the use of special features designed into the LCDS to simplify applications program development and debug operations.

#### 3.6.1 TTY Utilities

The LCDS DEBUG firmware contains a number of TTY utility subroutines that can be called by the user's application program to send and receive data to and from the TTY keyboard and the TTY tape reader. A brief description of each of the subroutines available to the user is provided in table 3-12 along with the subroutine calling address.

Detailed instructions for using the subroutines in an applications program are provided in the paragraphs that follow and functional operation of the subroutines is shown in figures 3-5 through 3-9. The following general conventions are associated with all of the subroutines.

1. Input Data. All of the subroutines associated with receiving data from the TTY keyboard and/or the TTY tape reader employ the BSENSE input to the SC/MP Microprocessor as the input data line. When data are to be received from the TTY keyboard only, the Flag 1 output of the SC/MP Microprocessor is reset low to deenergize the TTY Reader Relay; when data are to be received from the TTY tape reader, the Flag 1 output of the SC/MP Microprocessor is set high to enable the TTY Reader Relay.

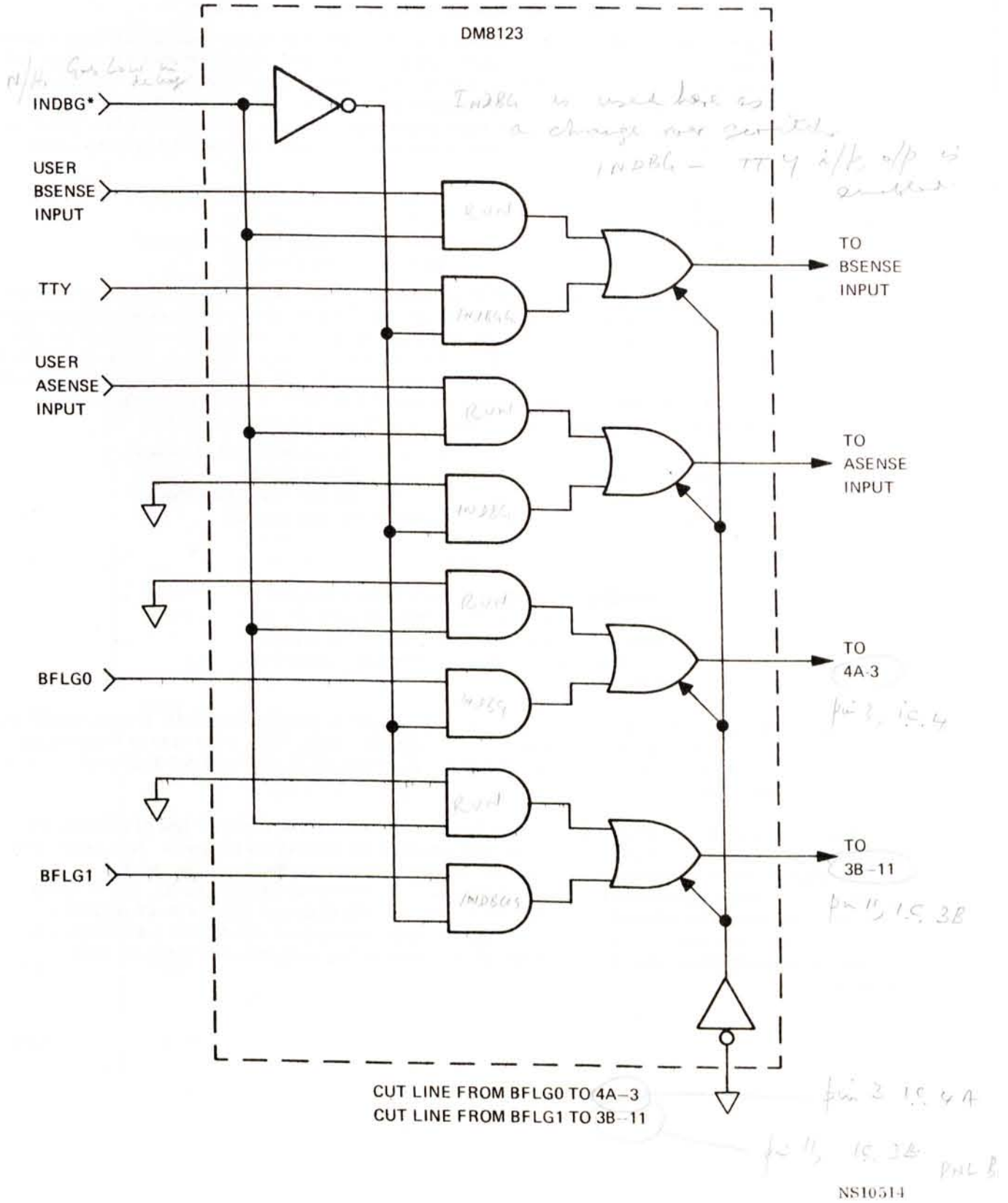


Figure 3-4. Flags, BSENSE, and Interrupt Control



2. Output Data. All of the subroutines associated with transmitting output data to the TTY employ the Flag 0 output of the SC/MP Microprocessor as the output data line.
3. Software Stack. All of the subroutines reference a SC/MP Pointer Register 2 (P2) as a stack pointer for development of a software stack that grows from high to low memory. Thus, before a subroutine is called P2 must be set to the high-order address of the stack. During execution of the subroutine, the stack then is used for saving and restoring SC/MP registers and for temporary storage of working and input/output data. The maximum stack length required is seven memory locations. The following instruction sequence may be used to load a 4-digit memory address into P2:

LDI	hv	Load AC with low-order P2 address
XPAL	2	Exchange P2-low with AC
LDI	hv	Load AC with high-order P2 address
XPAH	2	Exchange P2-high with AC

4. Subroutine Calls and Returns. All of the subroutines save the contents of the SC/MP Pointer Register 3 (P3) upon entry; subsequently, the subroutines effect a return to the calling program, first, by restoring the saved contents to P3 and, then, by exchanging the contents of P3 with the contents of the Program Counter. Thus, to ensure a proper return, a subroutine should be called, first, by loading the appropriate address into P3 and, second, by exchanging the contents of P3 with the contents of the Program Counter via an XPPC 3 instruction. The following instruction sequence may be used to load a 4-digit memory address into P3:

LDI	hv	Load AC with low-order P3 address - 1
XPAL	3	Exchange P3-low with AC
LDI	hv	Load AC with high-order P3 address
XPAH	3	Exchange P3-high with AC

NOTE

To facilitate use, all of the subroutine addresses in this chapter specify the low-order P3 address - 1 value; for all subroutine calls, therefore, load the appropriate address into the P3 register exactly as shown.

5. Interrupts. In order for the SC/MP Microprocessor to process interrupts, it is necessary that P3 be used as an interrupt-service-routine pointer. Thus, while P3 is used as a subroutine pointer, interrupts must be disabled by programming the SC/MP Interrupt Enable (IE) Flag to the reset state.

Since an XPPC 3 Instruction is employed by each of the subroutines to reinstate the PC return address, P3 always points to the exit address of the subroutine upon return to the applications program. Subroutines that can be recalled via the exit address are considered repeatable; subroutines that cannot be recalled via the exit address are considered nonrepeatable. The repeatable subroutines are GECHO, PUTC, MESH, GHEX, and PHSEB. The remaining subroutines (GETP, GETC, PECHO, GHEXE, and PHEX) are not repeatable because their exit addresses are common to one of the repeatable subroutines (see figures 3-5 through 3-9). In the subroutine descriptions that follow, both initial and repeat calling sequences are listed for each subroutine.



Table 3-12. DEBUG TTY Utility Subroutines

Subroutine	Calling Address	Description
GETC	7A88	This subroutine <u>receives a character from the TTY keyboard without echo</u> ; that is, when a TTY key is pressed after GETC is called, the character is input to GETC but it is not printed out on the TTY. At the start of GETC, the TTY Reader Relay is deenergized to disable the TTY tape reader.
GETP	7A84	This subroutine is identical to GETC except that the TTY Reader Relay is enabled to permit <u>a character to be received either from the TTY keyboard or the TTY tape reader</u> . In either case, the character is not printed out at the TTY.
GECHO	7A90	This subroutine <u>receives a character from the TTY keyboard with echo</u> ; that is, when a TTY key is pressed after GECHO is called, the character is input to GECHO and it is also printed out on the TTY keyboard. At the start of GECHO, the TTY Reader Relay is deenergized to disable the TTY tape reader.
PECHO	7A8C	This subroutine is identical to GECHO except that the TTY Reader Relay is energized to permit <u>a character to be received either from the TTY tape reader or the TTY keyboard</u> . In either case, the character received is also printed out at the TTY.
PUTC	7AE1	This subroutine causes the TTY to print out the ASCII character that corresponds to the value present in the SC/MP Accumulator when the subroutine is called. (For reference purposes, an ASCII-to-hexadecimal conversion list is provided in table 3-13.)
MESG	7B16	This subroutine prints out an ASCII message on the TTY. The message must be stored in sequential memory as a series of ASCII characters followed by a 00 data byte that indicates completion of the message. (For reference purposes, an ASCII-to-hexadecimal conversion list is provided in table 3-13.)
GHEX	7B4F	This subroutine enables a 4-digit hexadecimal value to be received from the TTY and to be stored on a software stack. If more than four digits are typed, only the last four digits are retained; if less than four digits are typed, the missing digits are treated as leading zeroes. For example, 12345678 would be stored as 5678 and 24 would be stored as 0024.
GHEXE	7B4B	This subroutine is identical to the GHEX subroutine except that the first digit is input from the SC/MP Extension Register. Thus, the desired first digit must be loaded into the SC/MP Extension Register before the subroutine is called. (Since only the last four digits are retained, typing more than three digits on the TTY will cause the digit entered via the Extension Register to be lost.)
PHEX	7BB3	This subroutine causes the TTY to provide a 2-digit hexadecimal printout that indicates the value that was present in the SC/MP Accumulator when the subroutine was called.
PHEXB	7BAD	This subroutine is identical to the PHEX subroutine except that a trailing blank follows the 2-digit hexadecimal printout.

Table 3-13. USA Standard Code for Information Interchange (ASCII)

BITS b <sub>6</sub> b <sub>5</sub> b <sub>4</sub>					Column→				0 <sub>00</sub>	0 <sub>01</sub>	0 <sub>10</sub>	0 <sub>11</sub>	1 <sub>00</sub>	1 <sub>01</sub>	1 <sub>10</sub>	1 <sub>11</sub>
					b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	0	1	2	3	4	5	6	7
0	0	0	0	0	0	NUL	DLE	SP	0	@	P	·	p			
0	0	0	1	0	1	SOH	DC1	!	1	A	Q	a	q			
0	0	1	0	0	2	STX	DC2	"	2	B	R	b	r			
0	0	1	1	0	3	ETX	DC3	#	3	C	S	c	s			
0	1	0	0	0	4	EOT	DC4	\$	4	D	T	d	t			
0	1	0	1	0	5	ENQ	NAK	%	5	E	U	e	u			
0	1	1	0	0	6	ACK	SYN	&	6	F	V	f	v			
0	1	1	1	0	7	BEL	ETB	'	7	G	W	g	w			
1	0	0	0	0	8	BS	CAN	(	8	H	X	h	x			
1	0	0	1	0	9	HT	EM	)	9	I	Y	i	y			
1	0	1	0	0	10	LF	SUB	*	:	J	Z	j	z			
1	0	1	1	0	11	VT	ESC	+	;	K	[	k	{			
1	1	0	0	0	12	FF	FS	,	<	L	\	l	;			
1	1	0	1	0	13	CR	GS	-	=	M		m	}			
1	1	1	0	0	14	SO	RS	.	>	N	~	n	~			
1	1	1	1	0	15	SI	US	/	?	O	_	o	DEL			

Note: The hex value = Column, Row; for example, A = 41, B = 42, Q = 51, and so forth.

### 3.6.1.1 GETC Subroutine

The GETC Subroutine (figure 3-5), first, sets the SC/MP Status Register to 00 to disable the TTY tape reader, and, then, reads one character into the SC/MP Accumulator from the TTY keyboard without echo. Execution of GETC, therefore, causes the original contents of the SC/MP Accumulator and the Status Register to be lost. The contents of the remaining SC/MP registers, except for P3, are not affected.

Before GETC may be called initially, P2 must be set to the high-order address of a software stack that allows at least five memory locations for use by GETC, and P3 must be set to the GETC call address 7A88. After P2 and P3 are set to the appropriate addresses, GETC may be called via an XPPC 3 Instruction. Upon completion, GETC will return to the next sequential instruction of the applications program with the received character stored in the SC/MP Accumulator, and the SC/MP Pointer Register 3 (P3) set to the exit address GECHO. Thus, if GETC is to be recalled to permit reception of another character from the TTY keyboard, the contents of the Accumulator, first, must be stored in a memory location to permit future processing, and, then, P3 must be reloaded with the GETC call address 7A88. Following these two actions, GETC may be recalled via an XPPC 3 instruction.



### 3.6.1.2 GETP Subroutine

The GETP Subroutine (figure 3-5), first, sets the SC/MP Status Register to 02 to enable the TTY tape reader, and, then, reads one character into the SC/MP Accumulator (without echo) from either the TTY keyboard or the TTY tape reader. Execution of GETP, therefore, causes the original contents of the SC/MP Accumulator and the Status Register to be lost. The contents of the remaining SC/MP registers, except for P3, are not affected.

Before GETP may be called initially, P2 must be set to the high-order address of a software stack that allows at least five memory locations for use by GETP, and P3 must be set to the GETP call address 7A84. After P2 and P3 are set to the appropriate addresses, GETP may be called via an XPPC 3 Instruction. Upon completion, GETP will return to the next sequential instruction of the applications program with the received character stored in the SC/MP Accumulator and the SC/MP Pointer Register 3 (P3) set to the exit address GECHO. Thus, if GETP is to be recalled to permit reception of another character from the TTY keyboard or the TTY tape reader, the contents of the Accumulator, first, must be stored in a memory location to permit future processing, and, then, P3 must be reloaded with the GETP call address 7A84. Following these two actions, GETP may be recalled via an XPPC 3 Instruction.

### 3.6.1.3 GECHO Subroutine

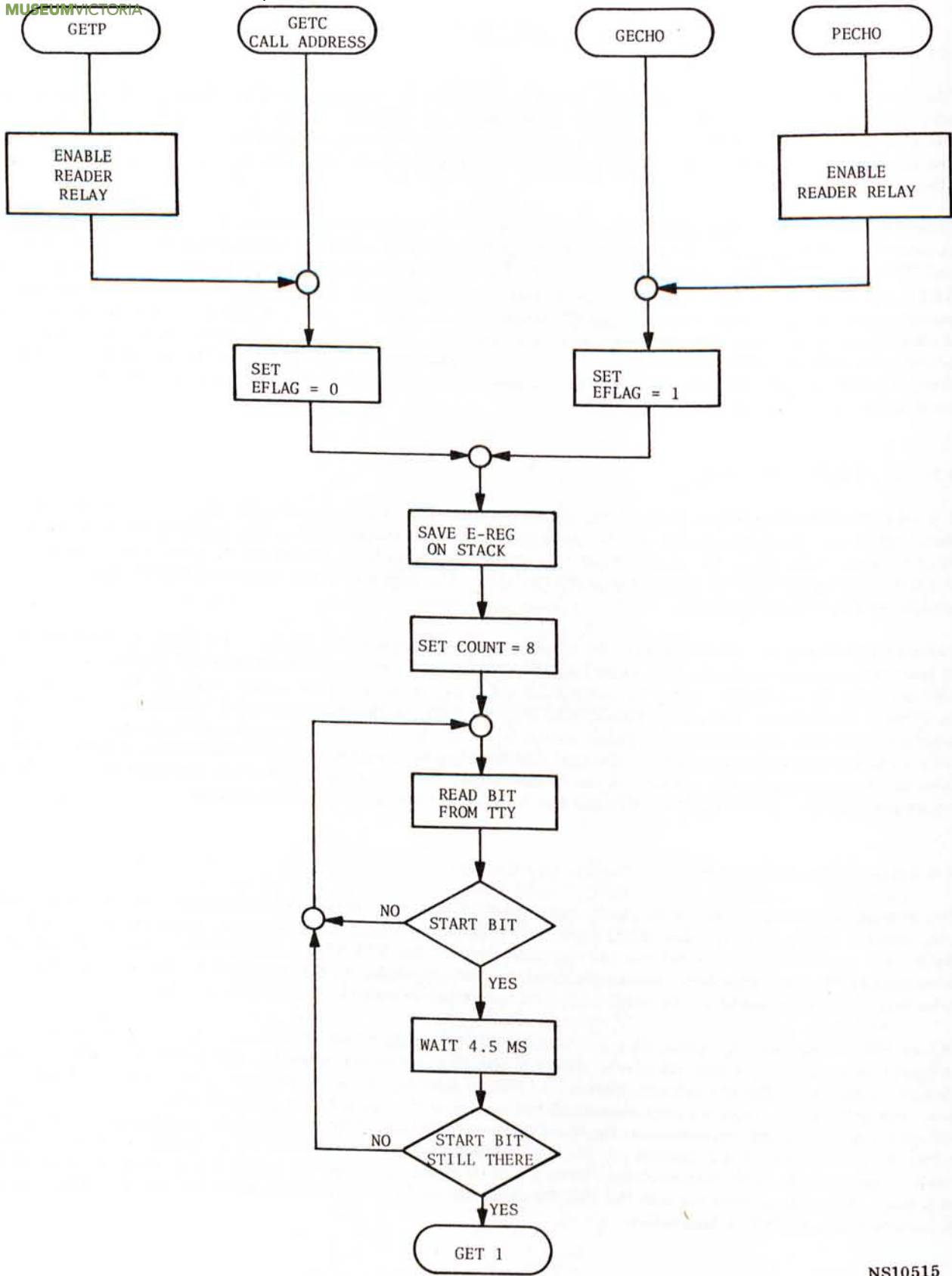
The GECHO Subroutine (figure 3-5), first, sets a software flag (EFLAG) to enable the program echo feature, and, then, reads one character into the SC/MP Accumulator while echoing back the character to the TTY via the Flag 0 output of the Status Register. Execution of GECHO, therefore, causes the original contents of the SC/MP Accumulator and the Status Register to be lost. The contents of the remaining SC/MP registers, except for P3, are not affected.

Before GECHO may be called initially, P2 must be set to the high-order address of a software stack that allows at least five memory locations for use by GECHO, and P3 must be set to the GECHO call address 7A90 or the GECHO exit address 7ADF. After P2 and P3 are set to the appropriate addresses, GECHO may be called via an XPPC 3 Instruction. Upon completion, GECHO will return to the next sequential instruction of the applications program with the received character stored in the SC/MP Accumulator and the SC/MP Pointer Register 3 (P3) set to the exit address GECHO. Thus, if GECHO is to be recalled to permit reception of another character from the TTY keyboard, the contents of the Accumulator, first, must be stored in a memory location to permit future processing. Following this, GECHO may be recalled via an XPPC 3 Instruction.

### 3.6.1.4 PECHO Subroutine

The PECHO Subroutine (figure 3-5), first, sets the SC/MP Status Register to 02 to enable the TTY tape reader, and, second, sets a software flag (EFLAG) to enable the program echo feature; then, it reads one character into the SC/MP Accumulator while echoing the character back to the TTY via the Flag 0 output of the Status Register. Execution of PECHO, therefore, causes the original contents of the SC/MP Accumulator and the Status Register to be lost. The contents of the remaining SC/MP registers, except for P3, are not affected.

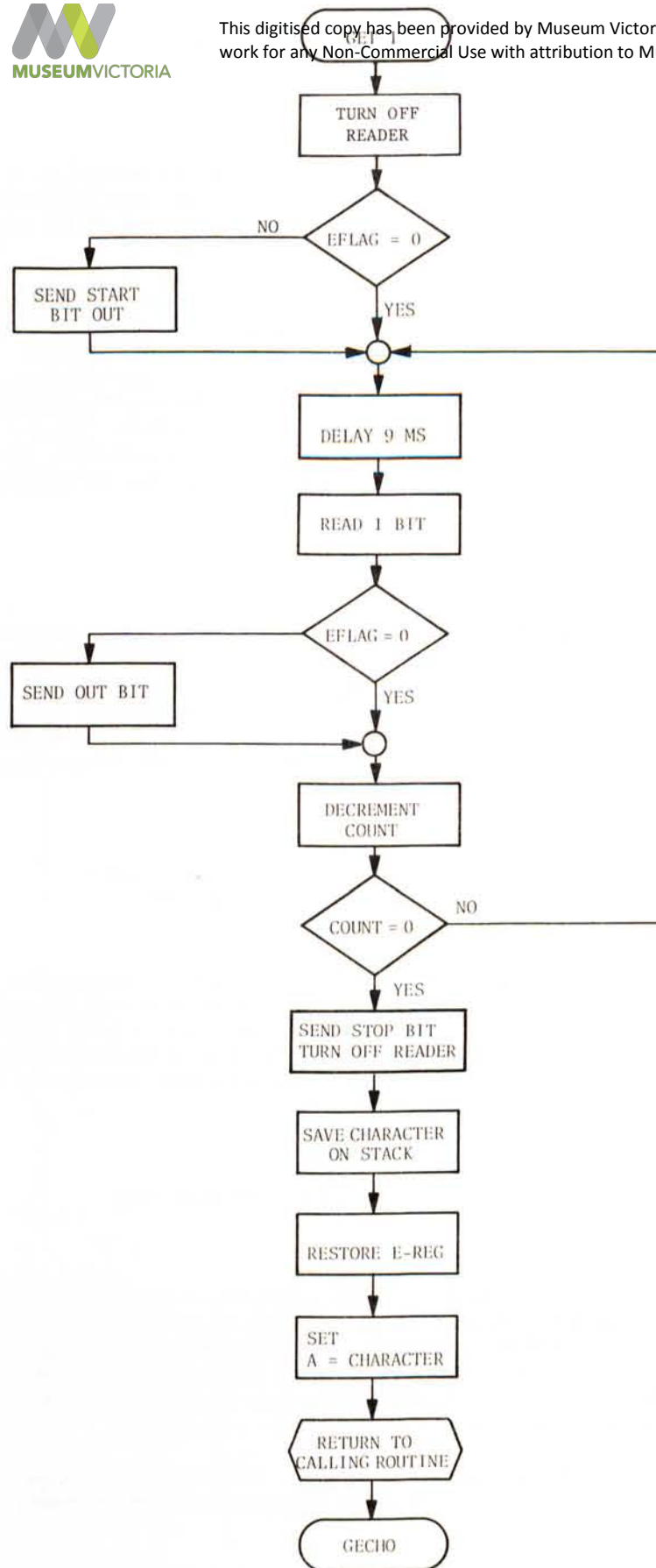
Before PECHO may be called initially, P2 must be set to the high-order address of a software stack that allows at least five memory locations for use by PECHO, and P3 must be set to the PECHO call address 7A8C. After P2 and P3 are set to the appropriate addresses, PECHO may be called via an XPPC 3 Instruction. Upon completion, PECHO will return to the next sequential instruction of the applications program with the received character stored in the SC/MP Accumulator and the SC/MP Pointer Register 3 (P3) set to the exit address GECHO. Thus, if PECHO is to be recalled to permit reception of another character from the TTY keyboard or the TTY tape reader, the contents of the Accumulator, first, must be stored in a memory location to permit future processing, and, then, P3 must be reloaded with the PECHO call address 7A8C. Following these two actions, PECHO may be recalled via an XPPC 3 Instruction.



NS10515

Figure 3-5. GETP, GETC, GECHO, and PECHO Subroutines (1 of 2)





NS10717

Figure 3-5. GETP, GETC, GECHO, and PECHO Subroutines (2 of 2)

### 3.6.1.5 PUTC Subroutine

The PUTC Subroutine (figure 3-6) transmits the value contained in the SC/MP accumulator to the TTY for printout as an ASCII character (refer to table 3-13). Execution of PUTC, therefore, causes the original contents of the SC/MP Status Register to be lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except for P3, are not affected.

Before PUTC may be called initially, P2 must be set to the high-order address of a software stack that allows at least five memory locations for use by PUTC, P3 must be set to the PUTC call address 7AE1 or the PUTC exit address 7B14, and the desired ASCII value must be loaded into the SC/MP Accumulator. After these actions are accomplished, PUTC may be called via an XPPC 3 Instruction. Upon completion, PUTC will return to the next sequential instruction of the applications program with the original ASCII value reinstated in the SC/MP Accumulator, and the SC/MP P3 register set to the exit address PUTC. Thus, if the ASCII value is to be transmitted to the TTY again, PUTC may be recalled via an XPPC 3 Instruction. If a new ASCII value is to be transmitted to the TTY, however, the new value must first be loaded into the SC/MP Accumulator before PUTC is recalled via an XPPC 3 Instruction.

### 3.6.1.6 MESH Subroutine

The MESH Subroutine (figure 3-7) enables a string of ASCII characters to be read out of sequential memory and to be transmitted to the TTY for printout; the first character is read out of the memory address specified in the call, and the remaining characters are read out of memory in sequence until either a 00 data byte is encountered (indicating completion of the message) or a TTY key is pressed to terminate manually the message printout. During execution of the MESH Subroutine, the Status Register is used for various program control functions, and the original contents of the Status Register are lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except Pointer Register 3 (P3), are not affected.

#### NOTE

The entire message must be located on a single "page" or "wraparound" will occur; refer to 3.4.1, TTY DEBUG Commands.

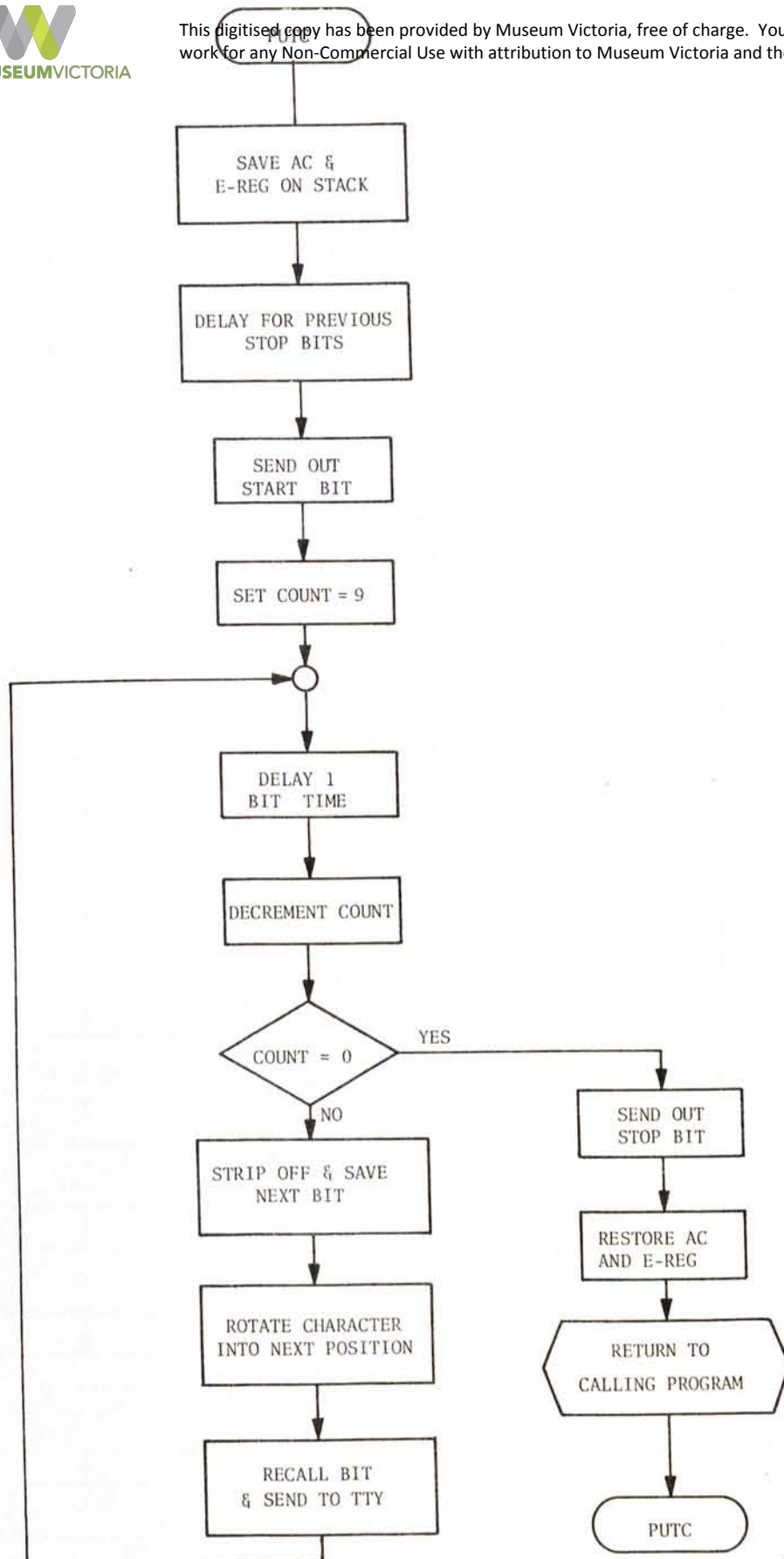
Before MESH may be called initially, P2 must be set to the high-order address of a software stack that allows at least 10 memory locations for use by MESH, and P3 must be set to the MESH call address 7B16 or the MESH exit address 7B49. Following these two actions, MESH may be called via XPPC 3 Instruction that is followed by two data bytes that contain the high-order and low-order digits of the memory address containing the first character of the message. If, for example, the first character of the message was located at memory address 4000, the calling sequence would be as follows:

<u>Memory Address</u>	<u>Contents</u>
XXXX	3F XPPC 3 INSTRUCTION
XXXX + 1	40
XXXX + 2	00

When MESH is called, it reads the first character out of the specified memory location and transmits the character to the TTY. Then MESH continues to read additional characters out of sequential memory locations and to transmit the characters to the TTY until it is terminated via a 00 data byte or by pressing a TTY key. Upon termination, MESH sets the SC/MP Pointer Register 3 (P3) to the MESH exit address 7B49 and increments the saved PC contents by 2. Thus, the memory-reference address associated with the call is skipped over and the MESH return is to the next sequential instruction of the applications program.

When additional messages are to be transmitted to the TTY for printout, MESH may be recalled after each return via an XPPC 3 Instruction followed by two data bytes that specify the memory address containing the first character of the new message.





NS10517

Figure 3-6. PUTC Subroutine

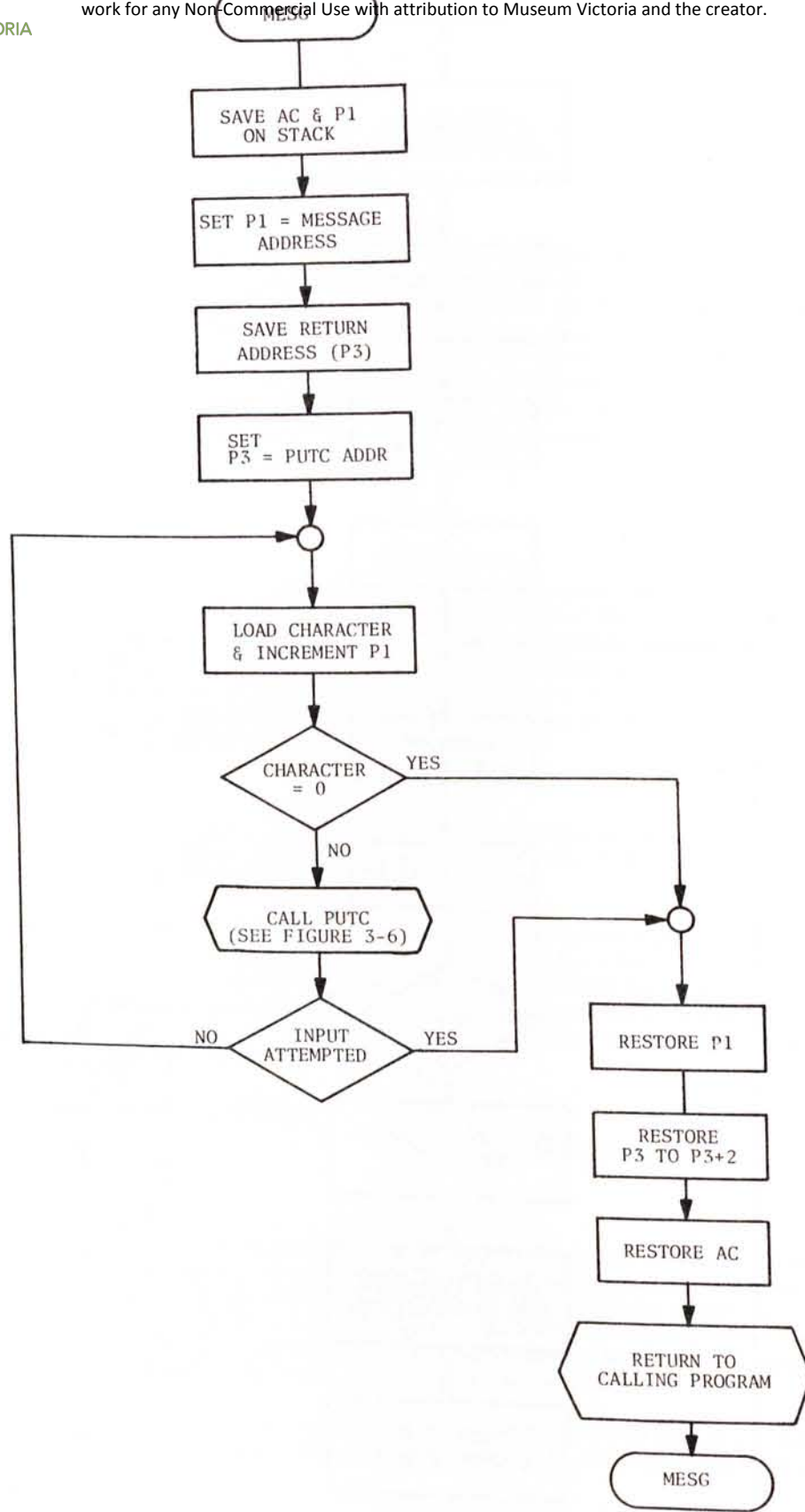


Figure 3-7. MESH Subroutine

NS10518



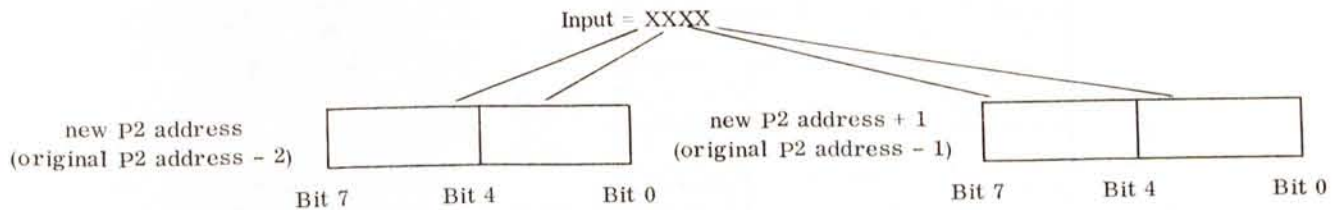
### 3.6.1.7 GHEX Subroutine

The GHEX Subroutine (figure 3-8) enables a 4-digit hexadecimal value to be received from the TTY and to be stored on a software stack. If more than four digits are typed in, only the last four digits are saved; if less than four digits are typed in, the missing digits are treated as leading zeroes. During execution of GHEX, the Status Register is used for various program control functions, and the original contents of the Status Register are lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except for P2 and P3, are not affected.

#### NOTE

The GHEX subroutine decodes ASCII characters ESC (data code 7E or 1B) and ALTMODE (data code 7D) as escape characters. If either of these characters are received, GHEX first prints out an error message (Ⓜ Ⓛ ?); then, it branches to a DEBUG firmware command loop to inhibit a return to the calling program.

Before GHEX may be called initially, P2 must be set to the high-order address of a software stack that allows at least 11 memory locations for use by GHEX, and P3 must be set to the GHEX call address 7B4F or the GHEX exit address 7BAB. After P2 and P3 are set to the appropriate addresses, GHEX may be called via an XPPC 3 Instruction. GHEX, then, will continue to accept hexadecimal digits from the TTY keyboard until it is terminated by pressing the TTY RETURN key or any other key that does not represent a hexadecimal value or an escape character (G, P, Q and so forth). Upon termination, the P2 register will be decremented by two and GHEX will return to the next sequential instruction of the applications program with the 4-digit input stored on the stack as shown below and the SC/MP Pointer Register 3 (P3) set to the exit address GHEX.



The reason that the P2 register is decremented during the GHEX return sequence is to allow the GHEX subroutine to be recalled without affecting the input data received previously. All that is necessary for repeated execution of the GHEX subroutine, therefore, is to allow sufficient memory locations in the initial stack assignment to cover all of the calls (11 memory locations for first call plus 2 memory locations for each additional call). Each repeat call can then be effected via an XPPC 3 Instruction. If, for example, the GHEX subroutine were repeated five times in succession, the stack composition would be as shown on the following page upon completion of the fifth call.

Stack address with respect to initial P2 address

Stack addresses with respect to P2 address after fifth call

P2 pointer after each call

P2	NOT USED	P2+10	
P2-1	High-order digits <u>1st</u> call	P2+9	
P2-2	Low-order digits <u>1st</u> call	P2+8	<span style="border: 1px solid black; padding: 2px;">P2</span> <u>1st</u> call
P2-3	High-order digits <u>2nd</u> call	P2+7	
P2-4	Low-order digits <u>2nd</u> call	P2+6	<span style="border: 1px solid black; padding: 2px;">P2</span> <u>2nd</u> call
P2-5	High-order digits <u>3rd</u> call	P2+5	
P2-6	Low-order digits <u>3rd</u> call	P2+4	<span style="border: 1px solid black; padding: 2px;">P2</span> <u>3rd</u> call
P2-7	High-order digits <u>4th</u> call	P2+3	
P2-8	Low-order digits <u>4th</u> call	P2+2	<span style="border: 1px solid black; padding: 2px;">P2</span> <u>4th</u> call
P2-9	High-order digits <u>5th</u> call	P2+1	
P2-10	Low-order digits <u>5th</u> call	P2	<span style="border: 1px solid black; padding: 2px;">P2</span> <u>5th</u> call
P2-11	used for temporary storage of register and working data	P2-1	
P2-12		P2-2	
P2-13		P2-3	
P2-14		P2-4	
P2-15		P2-5	
P2-16		P2-6	
P2-17		P2-7	
P2-18		P2-8	

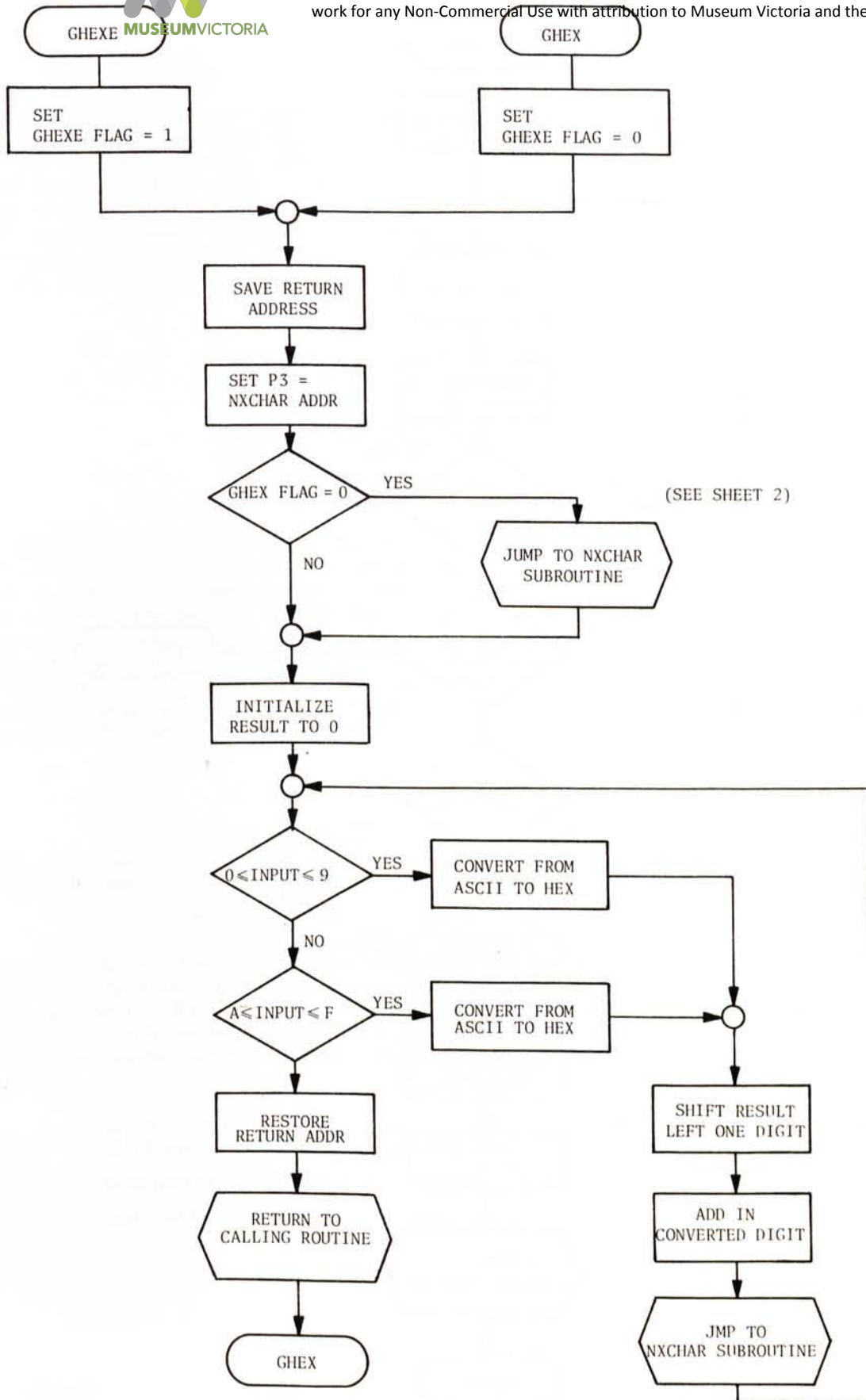
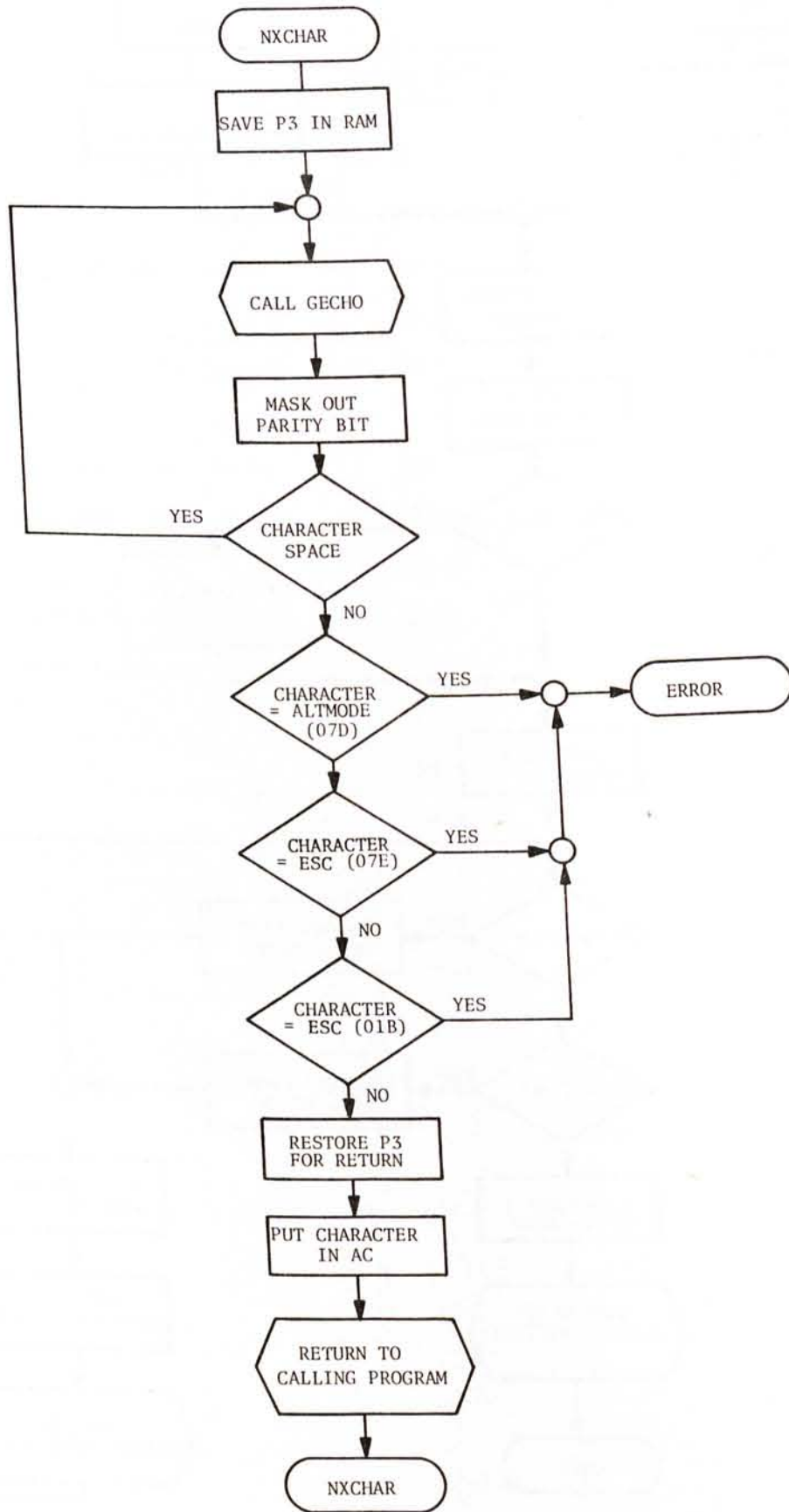


Figure 3-8. GHEX and GHEXE Subroutines (1 of 2)





NS10520

Figure 3-8. GHEX and GHEXE Subroutines (2 of 2)

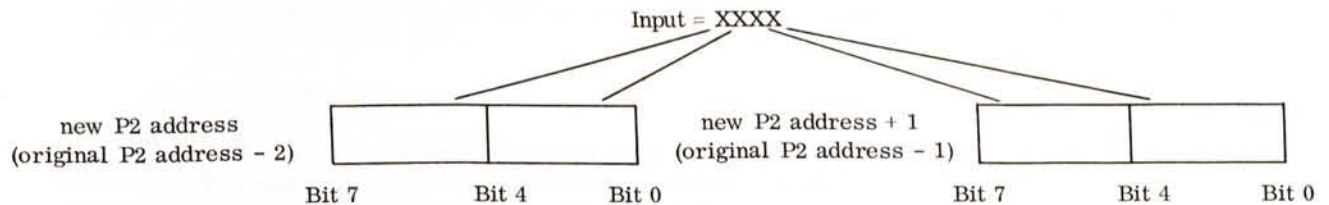
### 3.6.1.8 GHEXE Subroutine

The GHEXE Subroutine (figure 3-8 enables a 4-digit hexadecimal value to be received and stored on a software stack. The first digit is input from the SC/MP Extension Register and the remaining digits are input from the TTY keyboard. If more than three digits are typed in at the TTY keyboard, therefore, the digit that was input from the Extension Register will be lost and only the last four digits typed will be saved; if less than three digits are typed in at the TTY keyboard, the missing digits will be treated as leading zeroes. During execution of GHEXE, the Status Register is used for various program control functions and the original contents of the Status and Extension Registers are lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except for P2 and P3, are not affected.

#### NOTE

The GHEXE subroutine decodes ASCII characters ESC (data code 7E or 1B) and ALTMODE (data code 7D) as escape characters. If either of these characters are received from the TTY, GHEXE first prints out an error message (Ⓜ Ⓛ ?); then, it branches to a DEBUG firmware command loop to inhibit a return to the calling program.

Before GHEXE may be called initially, P2 must be set to the high-order address of a software stack that allows at least 11 memory locations for use by GHEXE, P3 must be set to the GHEXE call address 7B4B, and the desired first digit must be loaded into the SC/MP Extension Register. Following these actions, GHEXE may be called via an XPPC 3 Instruction. GHEXE, then, will read in the first digit from the SC/MP Extension Register and will continue to accept additional input digits from the TTY keyboard until it is terminated by pressing the TTY RETURN key or any other key that does not represent a hexadecimal value or an escape character (G, X, R, and so forth). Upon termination, the P2 register will be decremented by two and GHEXE will return to the next sequential instruction of the applications program with the 4-digit input stored on the stack as shown below, and the SC/MP P3 set to the exit address GHEX.



The reason that the P2 register is decremented during the GHEXE return sequence is to allow the GHEXE subroutine to be recalled without affecting the input data received previously. (See GHEX example of stack composition after five successive calls.) Thus, if repeated execution of the GHEXE subroutine is desired, the initial stack assignment should allow sufficient memory locations to accommodate all of the calls (11 memory locations for first call plus 2 memory locations for each additional call). The recommended sequence for recalling the GHEXE subroutine is as follows:

1. Shift first digit into Extension Register via execution of SIO Instructions, or load desired digit into Extension Register via Accumulator.
2. Load GHEXE call address 7B4B into P2.
3. Call GHEXE subroutine via XPPC 3 Instruction.



### 3.6.1.9 PHEX Subroutine

The PHEX Subroutine (figure 3-9) transmits the value contained in the SC/MP Accumulator to the TTY for printout as a 2-digit hexadecimal value. Execution of PHEX, therefore, causes the original contents of the SC/MP Status Register to be lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except for P3, are not affected.

Before PHEX may be called initially, P2 must be set to the high-order address of a software stack that allows at least 10 memory locations for use by PHEX, P3 must be set to the PHEX call address 7BB3, and the desired value must be loaded into the SC/MP Accumulator. After these actions are accomplished, PHEX may be called via an XPPC 3 Instruction. Upon completion, PHEX will return to the next sequential instruction of the applications program with the original value reinstated in the SC/MP Accumulator and the SC/MP P3 set to the exit address PHEXB. Thus, if PHEX is to be recalled to transmit another hexadecimal value to the TTY, P3 must be reloaded with the PHEX call address 7BB3; then, the new hexadecimal value must be loaded into the SC/MP Accumulator. Following these two actions, PHEX may be recalled via an XPPC 3 Instruction.

### 3.6.1.10 PHEXB Subroutine

The PHEXB Subroutine (figure 3-9) transmits the value contained in the SC/MP Accumulator to the TTY for printout as a 2-digit hexadecimal value followed by a trailing blank. Execution of PHEXB, therefore, causes the original contents of the SC/MP Status Register to be lost. The contents of the SC/MP Accumulator and the remaining SC/MP registers, except for P3, are not affected.

Before PHEXB may be called initially, P2 must be set to the high-order address of a software stack that allows at least 10 memory locations for use by PHEXB, P3 must be set to the PHEXB call address 7BAD or the PHEXB exit address 7BF3, and the desired value must be loaded into the SC/MP Accumulator. After these actions are accomplished, PHEXB may be called via an XPPC 3 Instruction. Upon completion, PHEXB will return to the next sequential instruction of the applications program with the original value reinstated in the SC/MP Accumulator and the SC/MP P3 set to the exit address PHEXB. Thus, if PHEXB is to be recalled to permit another 2-digit hexadecimal value to be printed out with trailing blank, the new hexadecimal value must be loaded into the SC/MP Accumulator before PHEXB is recalled via an XPPC 3 Instruction.

### 3.6.2 Digital Readout

The LCDS digital readout may be employed by an applications program for general display and/or trace purposes. Each digit of the readout is enabled via a unique address, and all of the digits share a common data input that controls lighting of the segments within the digits. The addresses of the digits and the data codes for hexadecimal values 0 through F are on the following page.

#### NOTE

The LCDS digital readout displays the hexadecimal values B and D in lower case (b and d) because *for easy distinction* ~~b and d are more readily distinguished~~ from the values 8 and 0.



<u>Digit</u>	<u>Address</u>	<u>Hexadecimal Value</u>	<u>Data Code</u>
1st (left-most) digit	7020	0	3F
2nd digit	7010	1	06
3rd digit	7008	2	5B
4th digit	7004	3	4F
5th digit	7002	4	66
6th (right-most) digit	7001	5	6D
		6	7D
		7	07
		8	7F
		9	67
		A	77
		B	7C
		C	39
		D	5E
		E	79
		F	71

*Handwritten notes:* 01, 02, 04, 08, 10, 20, 40, 5f, D0, D1, D2, D4, D7, BLANK, 00

Segment Assignments For Digit Indicators

LCDS address decoding is such that address bits BA00 through BA05 and data bits BD0 through BD6 are latched on the trailing edge of the SC/MP BWDS\* strobe when the digital readout is addressed. The latched address and data bits, then, are applied to the digital readout as shown in figure 3-10. This particular addressing scheme permits a hexadecimal value to be displayed on one or more digits of the digital readout simultaneously. For example, the address 7020 enables a value to be displayed on the first (left-most) digit, the address 7030 enables a value to be displayed on the first and second digits, and the address 703F enables a value to be displayed on all of the segments.

The programming sequence listed below may be used to display a hexadecimal value on the digital readout. The value then will remain displayed on the digital readout until the digital readout is addressed to permit display of a new value (data code 00 can be used to blank the digital display), the SC/MP Microprocessor is initialized, or the LCDS is placed in the DEBUG Mode.

Programming Sequence for Display of a Hexadecimal Value

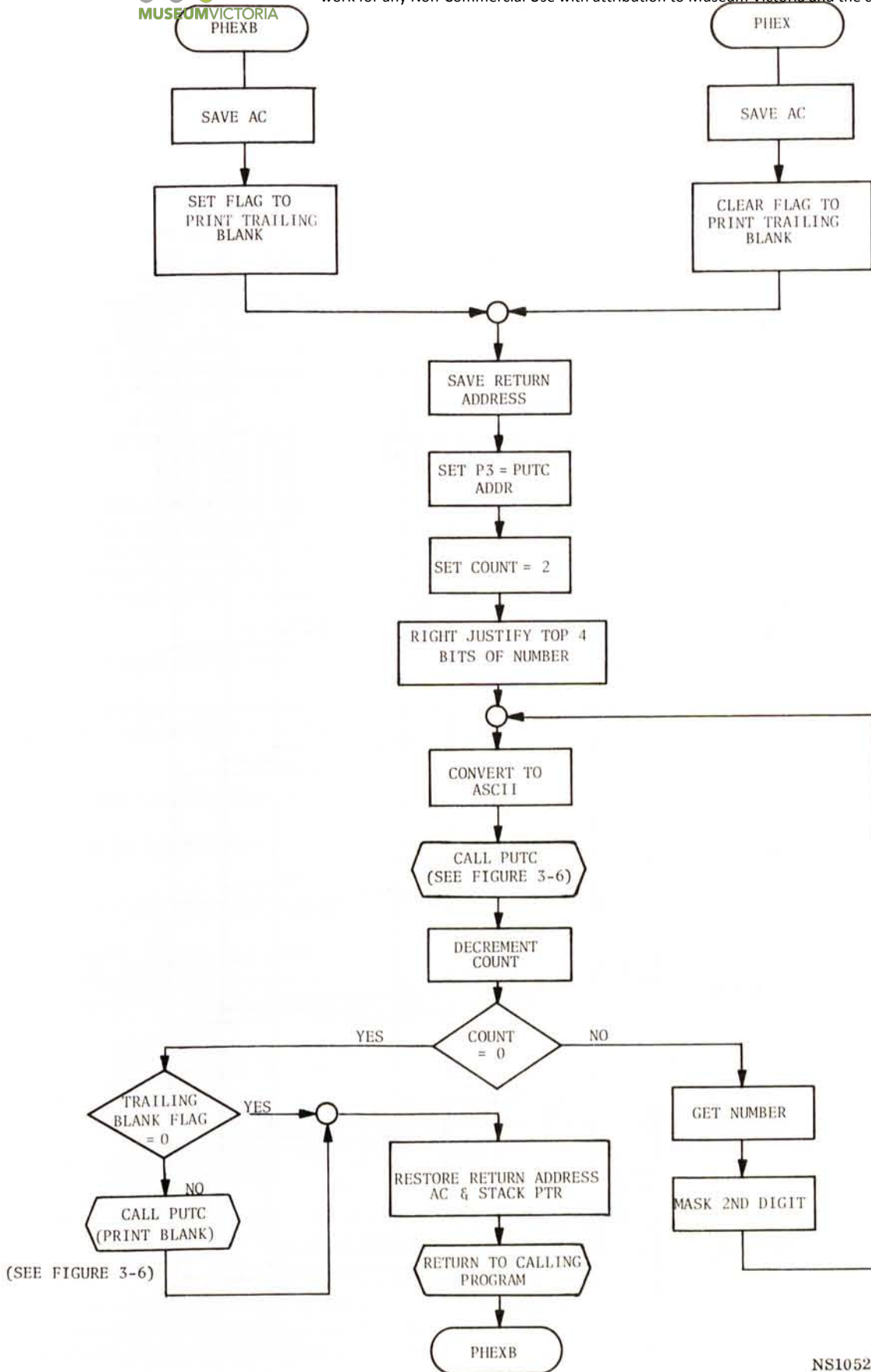
<u>Instruction</u>		<u>Comments</u>
LDI	data	Load AC with low-order digital readout address to enable desired digit(s).
XPAL	Ptn	Exchange AC with Pointer Register low (P1, 2, or 3).
LDI	70	Load AC with high-order digital readout address.
XPAH	Ptn	Exchange AC with Pointer Register high (P1, 2, or 3 — as above).
LDI	data	Load AC with data code of desired value.
ST	Ptn	Store AC at Pointer Register address (P1, 2, or 3 — as above).

3.6.3 Keyboard Pushbuttons

The LCDS 16-pushbutton keyboard matrix may be employed by an applications program to permit data to be input to the program by the pushbuttons. Each row of pushbuttons in the matrix is assigned a unique address, and each pushbutton in a row provides an output via a unique data bit. The address/data matrix assigned to the keyboard is shown on the following page.

<u>Switch</u>	<u>Address</u>	<u>Data Output</u>
0/READ PC	7010	BD0 = 1
1/READ P1	7010	BD1 = 1
2/READ P2	7010	BD2 = 1
3/READ P3	7010	BD3 = 1
4/READ AC	7008	BD0 = 1
5/READ EX	7008	BD1 = 1
6/READ ST	7008	BD2 = 1
7/READ MEM	7008	BD3 = 1
8/LOAD PC	7004	BD0 = 1
9/LOAD P1	7004	BD1 = 1
A/LOAD P2	7004	BD2 = 1
B/LOAD P3	7004	BD3 = 1
C/LOAD AC	7002	BD0 = 1
D/LOAD EX	7002	BD1 = 1
E/LOAD ST	7002	BD2 = 1
F/LOAD MEM	7002	BD3 = 1

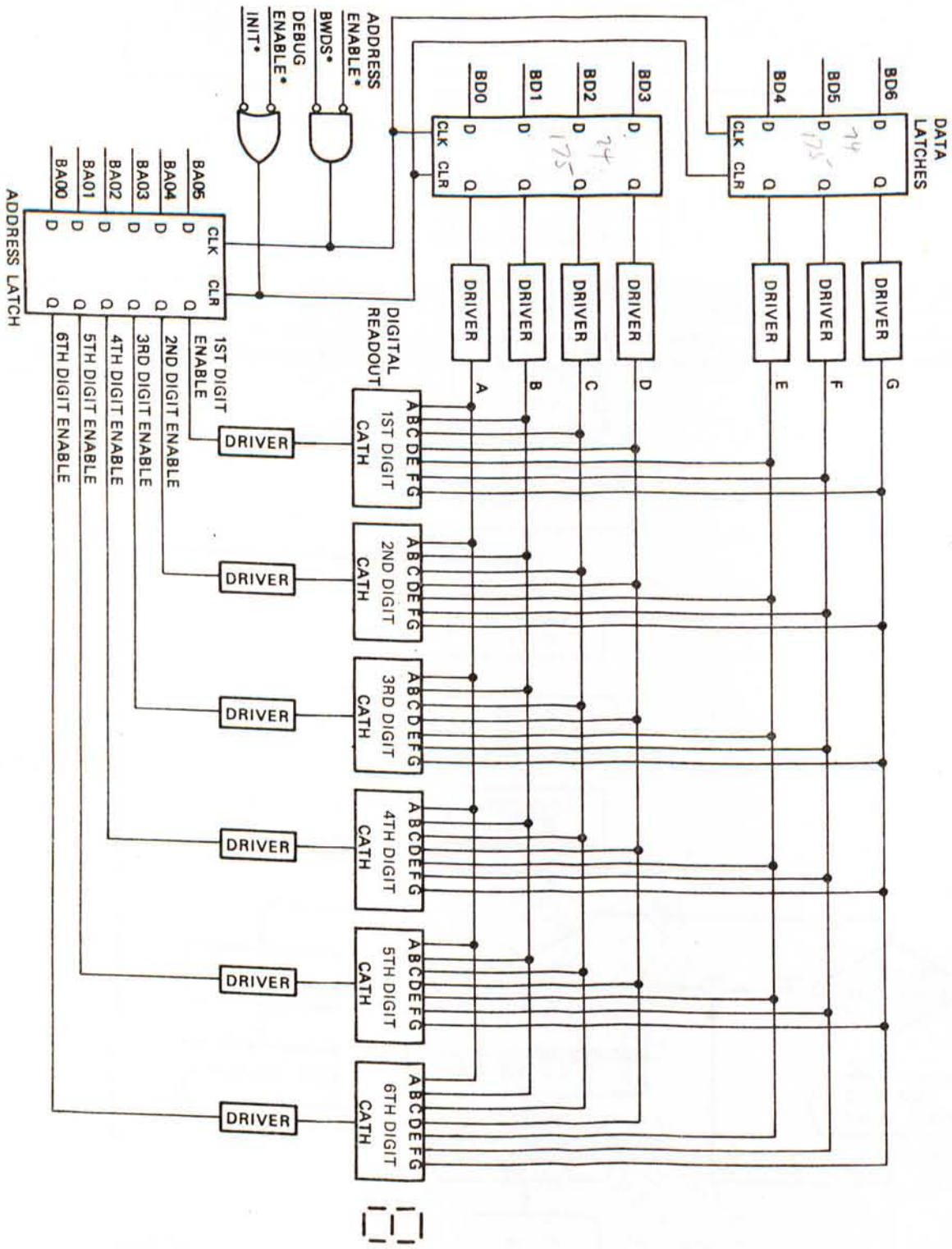
Reading a particular row of pushbuttons is effected via a Load (LD) Instruction. Upon completion of the Load Instruction the contents of the Accumulator will either be zero (indicating that none of the pushbuttons in the row was pressed) or greater than zero (indicating that one of the pushbuttons in the row was pressed). If the contents of the Accumulator are greater than zero, the position of the high bit will indicate the pushbutton that was pressed.



NS10521

Figure 3-9. PHEX and PHEXE Subroutines





NS10718

Figure 3-10. LCDS Digital Readout and Control Circuit

## Chapter 4

## PROGRAMMING EXAMPLES

## 4.1 INTRODUCTION

This chapter illustrates typical SC/MP programming techniques by providing detailed descriptions of the LCDS utility subroutines that are available to the user. Each subroutine description includes a detailed flowchart that shows the particular action being accomplished at each stage of the subroutine, and an annotated instruction listing that specifies exactly how the action is accomplished. Thus, each subroutine description provides both a conceptual analysis of an actual applications requirement, and a specific programming methodology that can be adapted to other applications requirements. On the last page of each annotated listing, which is in the form of a flowchart, is a printout of the actual listing presented on a foldout page. Thus, the actual listing may be in view and easily referred to while following either the detailed flowchart or the annotated listing of the subroutine.

Programming techniques covered in the descriptions include: subroutine calls and returns, stack development and use, program delay and synchronization, input/output serial data transmission, temporary data storage, basic data processing operations (AND, OR, Exclusive OR, and so forth), Extension and Status Register use, basic arithmetic operations (Add, Subtract, Loop Count, and so forth), ASCII decoding, ASCII/hexadecimal conversion, and multi-character message transmission.

## 4.2 PROGRAMMING CONVENTIONS AND INPUT/OUTPUT DATA ASSIGNMENTS

General programming conventions and input/output data assignments that are common to all of the LCDS utility subroutines are covered in 3.6.1, TTY Utilities.

## 4.3 SUBROUTINE DESCRIPTION

Descriptions of the LCDS utility subroutines follow.

## 4.3.1 GETC, GETP, GECHO, and PECHO Subroutines

The GETC, GETP, GECHO, and PECHO subroutines provide the capability for reception of one ASCII character from the TTY keyboard and/or paper-tape reader, and for optional printout of the ASCII character at the TTY keyboard. The specific functions of the subroutines are listed below.

- a. GETC Subroutine. This subroutine receives one ASCII character from the TTY keyboard without printout of the character at the TTY.
- b. GETP Subroutine. This subroutine receives one ASCII character from either the TTY keyboard or the TTY paper-tape reader without printout of the character at the TTY.
- c. GECHO Subroutine. This subroutine receives one ASCII character from the TTY keyboard and echoes the character back to the TTY for printout.
- d. PECHO Subroutine. This subroutine receives one ASCII character from either the TTY keyboard or the TTY paper-tape reader and echoes the character back to the TTY for printout.

Overall execution of the GETC, GETP, GECHO, and PECHO subroutines is shown in flowchart form in figure 4-1, and an annotated instruction listing is provided in figure 4-2. For detailed information on how these subroutines may be incorporated into an applications program, refer to 3.6.1.1 through 3.6.1.4.



A general summary of the register assignments and the call/return sequence for the GETP, GETC, GECHO and PECHO subroutines follows.

a. Register Assignment Prior to Call:

P2 — high order address of software stack that contains at least five memory locations

P3 — GETP           7A84  
           GETC           7A88  
           PECHO        7A8C  
           GECHO        7A90

b. Subroutine Call: XPPC 3 Instruction

c. Subroutine Return: to XPPC 3 Instruction address + 1

d. Register Assignment Upon Return:

P1 — unchanged  
 P2 — high-order address of stack  
 P3 — GECHO repeat address 7ADF  
 AC — ASCII input character  
 EX — unchanged

	CY/L	OV	SB	SA	IE	F2	F1	F0
S -	?	U	I	I	U	U	0	0
Bit	7	6	5	4	3	2	1	0

where

U - unchanged  
 I - read-only bit (logic state is controlled by external input)  
 ? - CY/L flag reflects state of last TTY output data bit

#### 4.3.2 PUTC Subroutine

The PUTC subroutine transmits the value contained in the SC/MP Accumulator to the TTY for printout. Overall execution of the PUTC subroutine is shown in figure 4-3, and an annotated program listing is provided in figure 4-4. For detailed information on how this subroutine may be incorporated into an applications program, refer to 3.6.1.5.

A general summary of the register assignments and the call/return sequence for the PUTC subroutine follows.

a. Register Assignment Prior to Call:

P2 — high-order address of software stack that contains at least five memory locations  
 P3 — PUTC call address 7AE1  
 AC — ASCII output character

b. Subroutine Call: XPPC 3 Instruction

c. Subroutine Return: to XPPC 3 Instruction address + 1

d. Register Assignment Upon Return:

P1 — unchanged  
 P2 — high-order address of stack  
 P3 — PUTC repeat address 7ADF  
 AC — unchanged (ASCII output value)  
 EX — unchanged



	CY/L	OV	S <sub>B</sub>	S <sub>A</sub>	IE	F2	F1	F0
S -	U	U	I	I	U	U	U	0
Bit	7	6	5	4	3	2	1	0

where

- U - unchanged
- I - read only bit (logic state is controlled by external input)

#### 4.3.3 MESH Subroutine

The MESH subroutine enables a string of ASCII characters to be read out of sequential memory and to be transmitted to the TTY for printout; the first character is read out of the memory address specified in the call, and the remaining characters are read out of memory in sequence until either a 00 data byte is encountered (indicating completion of the message) or a TTY key is pressed to manually terminate the message printout. Overall operation of the MESH subroutine is shown in figure 4-5, and an annotated program listing is provided in figure 4-6. For detailed information on how this subroutine may be incorporated into an applications program, refer to 3.6.1.6.

A general summary of the register assignments and the call/return sequence for the MESH subroutine is as follows.

#### NOTE

When the MESH subroutine is called, the high- and low-order address of the first message character must be located immediately following the XPPC 3 Instruction; for example:

Memory Address	Contents
XXXX	XPPC 3 Instruction
XXXX + 1	high-order byte of message address
XXXX + 2	low-order byte of message address

- a. Register Assignment Prior to Call:  
 P2 - high-order address of software stack that contains at least 10 memory locations  
 P3 - MESH call address 7B16
- b. Subroutine Call: XPPC 3 Instruction
- c. Subroutine Return: to XPPC 3 Instruction address + 3
- d. Register Assignment Upon Return:  
 P1 - unchanged  
 P2 - high-order address of stack  
 P3 - MESH repeat address 7B49  
 AC - unchanged  
 EX - unchanged

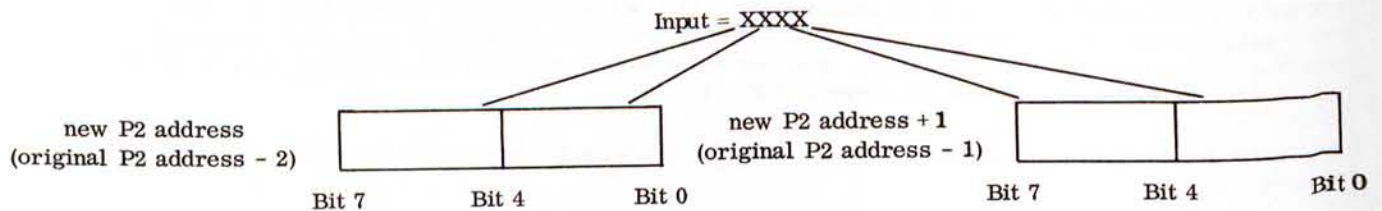
	CY/L	OV	S <sub>B</sub>	S <sub>A</sub>	IE	F2	F1	F0
S -	U	U	I	I	U	U	U	0
Bit	7	6	5	4	3	2	1	0

where

- U - unchanged
- I - read-only bit (logic state is controlled by external input)

#### 4.3.4 GHEX and GHEXE Subroutines

The GHEX and GHEXE subroutines provide an ASCII-to-hexadecimal conversion function that enables ASCII data to be read in and stored on a software stack in 4-digit hexadecimal format. The only difference in execution of the two subroutines is that the GHEXE subroutine reads in the first ASCII character from the Extension Register and subsequent ASCII characters from the TTY keyboard, whereas the GHEX subroutine reads in all of the ASCII characters from the TTY keyboard. During execution of either subroutine, reading in of ASCII data continues until escape character ESC (data code 7E or 1B) or ALTMODE (data code 7D) is detected, or until an ASCII terminator is received (any ASCII character except 0-9, A-F, ESC, and ALT MODE). When an escape character is detected, an error message is first printed out (CR LF, ?), and then a branch is made to a DEBUG firmware command loop to terminate subroutine execution and inhibit a return to the calling program. When an ASCII terminator is detected, it causes the subroutine to return to the calling program with the P2 register decremented by two and with a 4-digit hexadecimal value stored on the stack as shown below. (If less than four digits are read in before the ASCII terminator is received, the missing digits are treated as leading zeroes; if more than four digits are read in, only the last four digits are saved.)



Overall execution of the GHEX and GHEXE subroutines is shown in flowchart form in figure 4-7, an annotated program listing is provided in figure 4-8, and supplementary descriptions of the more involved subroutine operations are provided in figures 4-9 through 4-11. For detailed information on how the GHEX and GHEXE subroutines may be incorporated into an applications program, refer to 3.6.1.7 and 3.6.1.8.

A general summary of the register assignments and the call/return sequence for the GHEX and GHEXE subroutines follows:

a. Register Assignment Prior to Call:

- P2 — high order address of software stack that contains at least 11 memory locations
- P3 — GHEXE call address 7B4B or GHEX call address 7B4F
- EX — first ASCII character (GHEXE subroutine call only)

b. Subroutine Call: XPPC 3 Instruction

c. Subroutine Return: to XPPC 3 Instruction address + 1

d. Register Assignment Upon Return:

- P1 — unchanged
- P2 — high order address of stack - 2
- P3 — GHEX repeat address 7BAB
- AC — ASCII Terminator
- EX — ASCII Terminator

	CY/L	OV	SB	SA	IE	F2	F1	F0
S -	?	?	I	I	U	U	0	0
Bit	7	6	5	4	3	2	1	0

where

- U - unchanged
- I - read-only bit (logic state is controlled by external input)
- ? - unknown (varies according to ASCII input value)

#### 4.3.5 PHEXB and PHEX Subroutines

The PHEX and PHEXB subroutines provide an ASCII-to-hexadecimal conversion function that enables the contents of the SC/MP Accumulator to be transmitted to the TTY for printout, respectively, as a 2-digit hexadecimal value with or without trailing blank. Overall execution of the PHEX and PHEXB subroutines is shown in figure 4-12, an annotated program listing is provided in figure 4-13, and a supplementary description of the hexadecimal-to-ASCII conversion process is provided in figure 4-14. For detailed information on how the PHEX and PHEXB subroutines may be incorporated into an applications program, refer to 3.6.1.9 and 3.6.1.10, respectively.

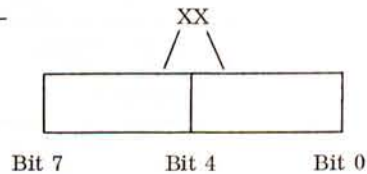
a. Register Assignment Prior to Call:

P2 – high order address of software stack that contains at least 10 memory locations

P3 – PHEXB call address 7BAD or

PHEX call address 7BB3

AC – two digit value to be transmitted to TTY –



b. Subroutine Call: XPPC 3 Instruction

c. Subroutine Return: to XPPC 3 Instruction address + 1

d. Register Assignment Upon Return:

P1 – unchanged

P2 – high order address of software stack

P3 – PHEXB repeat address 7BF3

AC – unchanged

EX – unchanged

	CY/L	OV	SB	SA	IE	F2	F1	F0
S -	?	?	I	I	U	U	U	0
Bit	7	6	5	4	3	2	1	0

where

U - unchanged

I - read-only bit (logic state is controlled by external input)

? - unknown (varies according to ASCII output value)



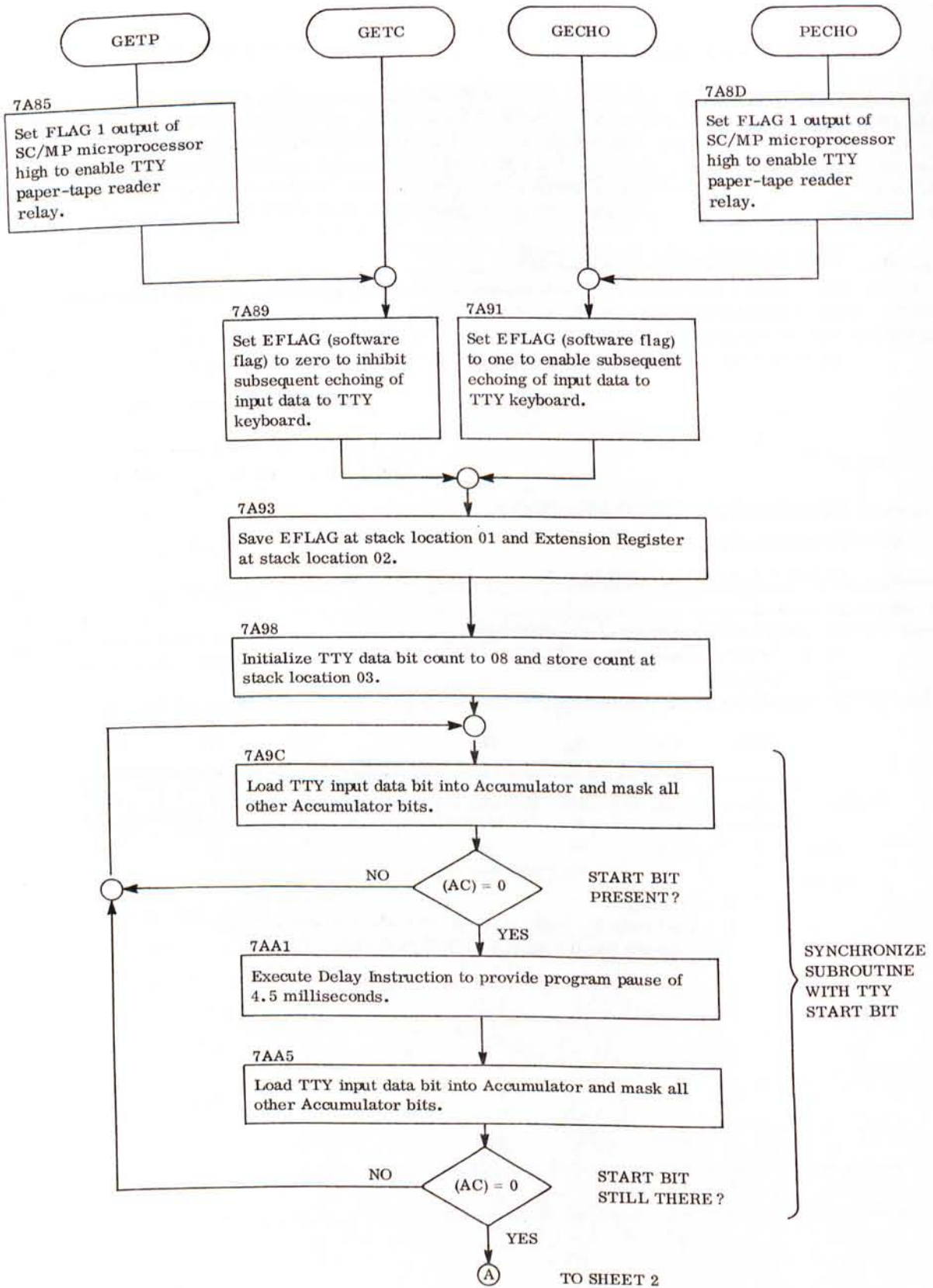
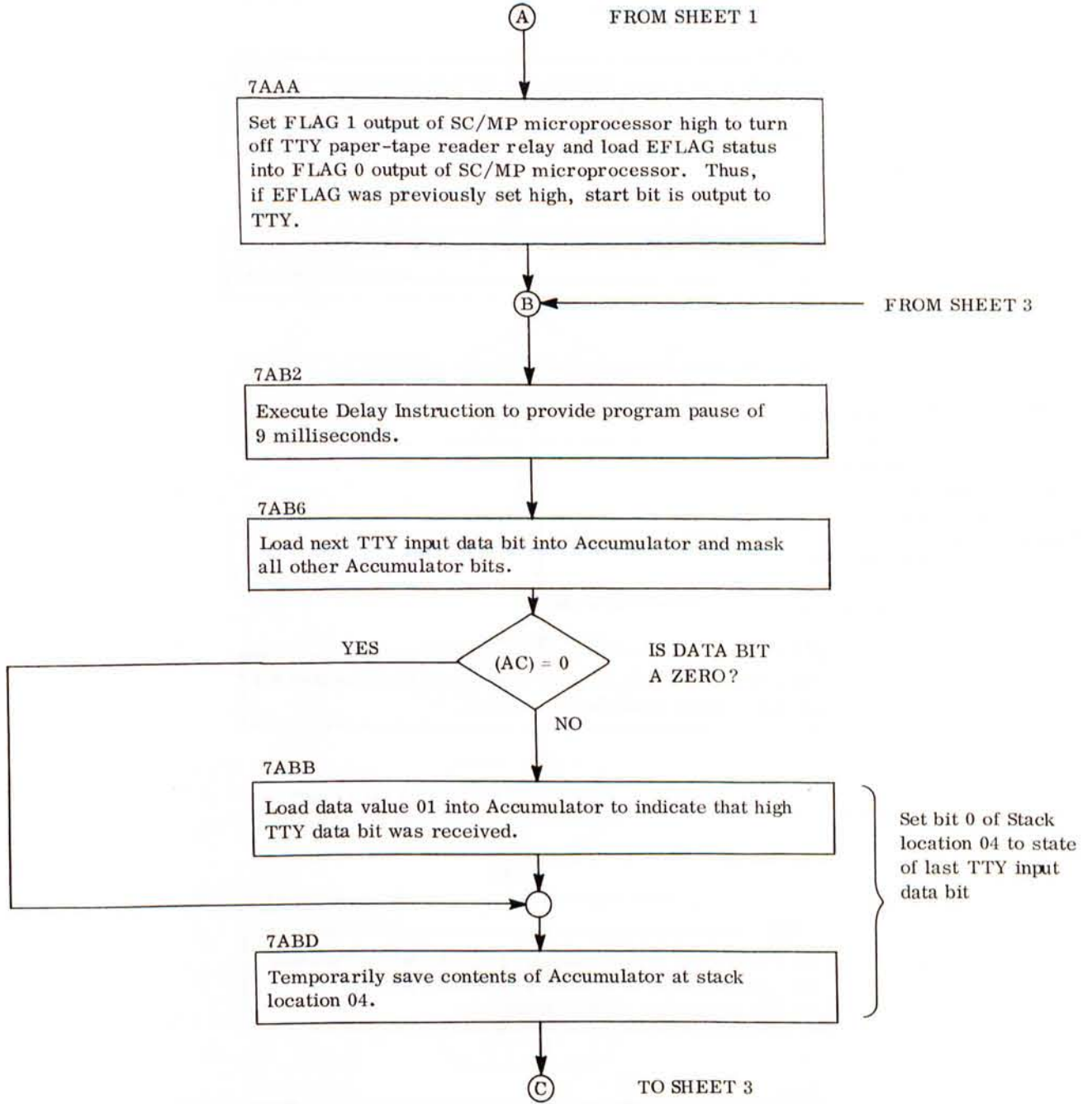


Figure 4-1. GETP, GETC, GECHO, and PECHO Subroutine Detailed Flowchart (Sheet 1 of 4)

NS10644



NS10645

Figure 4-1. GETP, GETC, GECHO, and PECHO Subroutine Detailed Flowchart (Sheet 2 of 4)

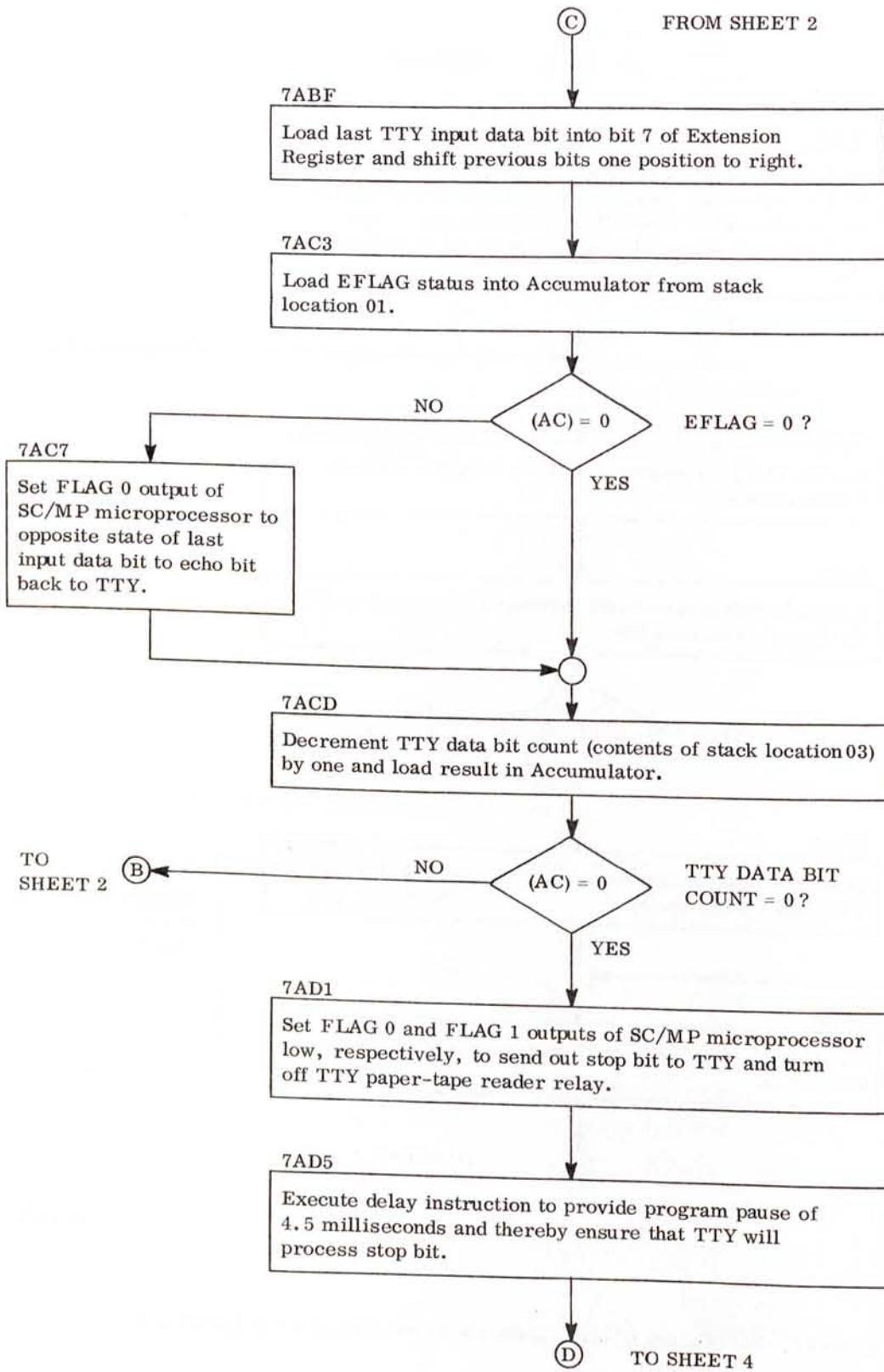
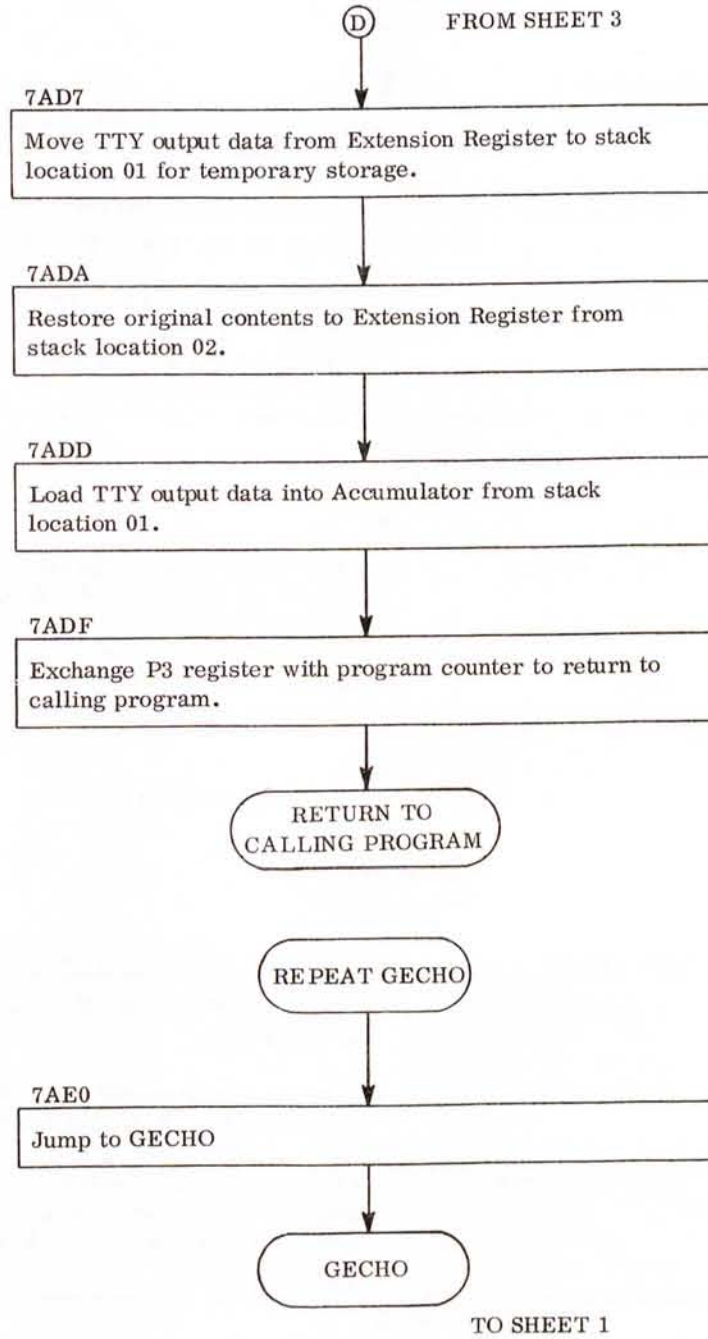


Figure 4-1. GETP, GETC, GECHO, and PECHO Subroutine Detailed Flowchart (Sheet 3 of 4)





NS10647

Figure 4-1. GETP, GETC, GECHO, and PECHO Subroutine Detailed Flowchart (Sheet 4 of 4)

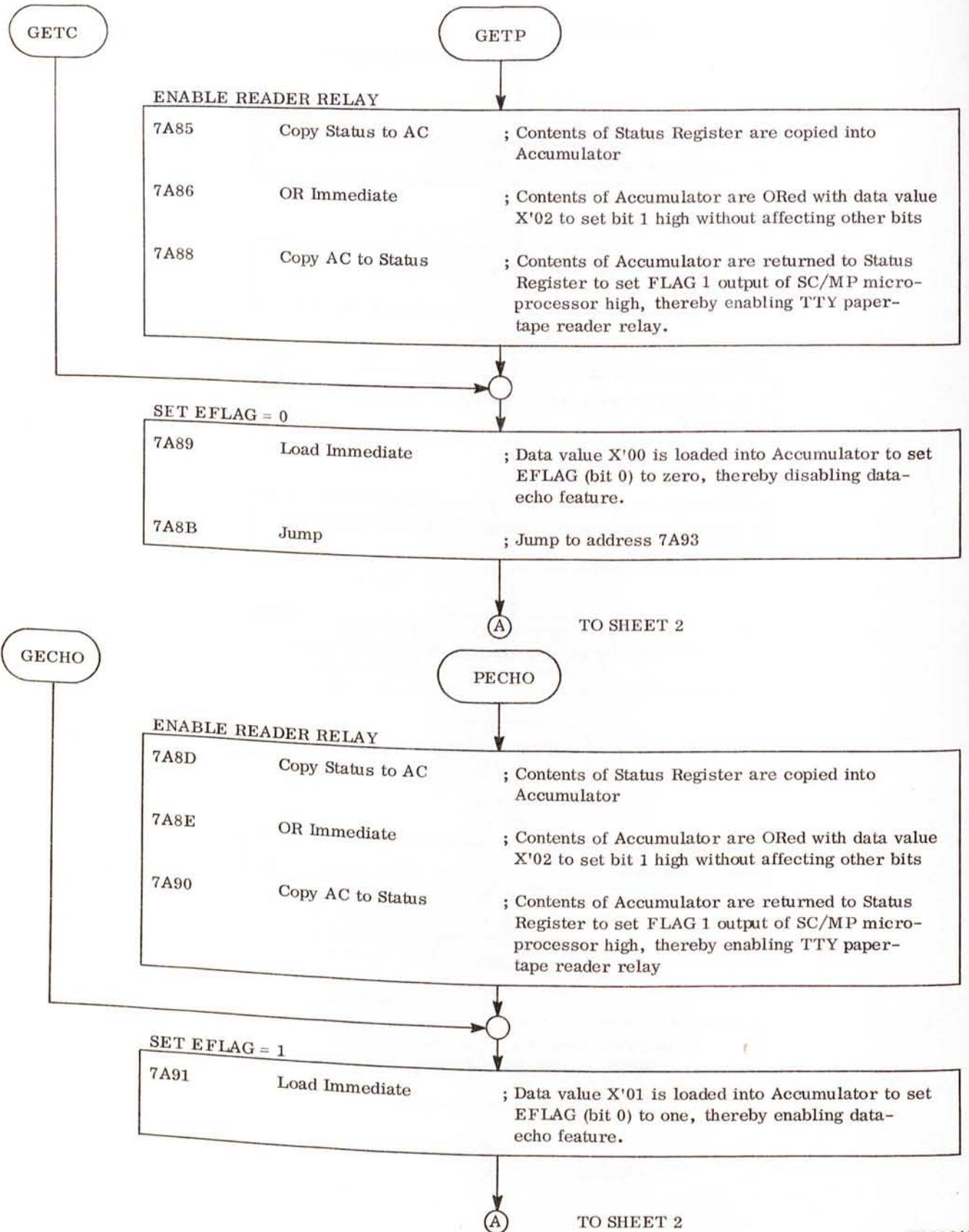
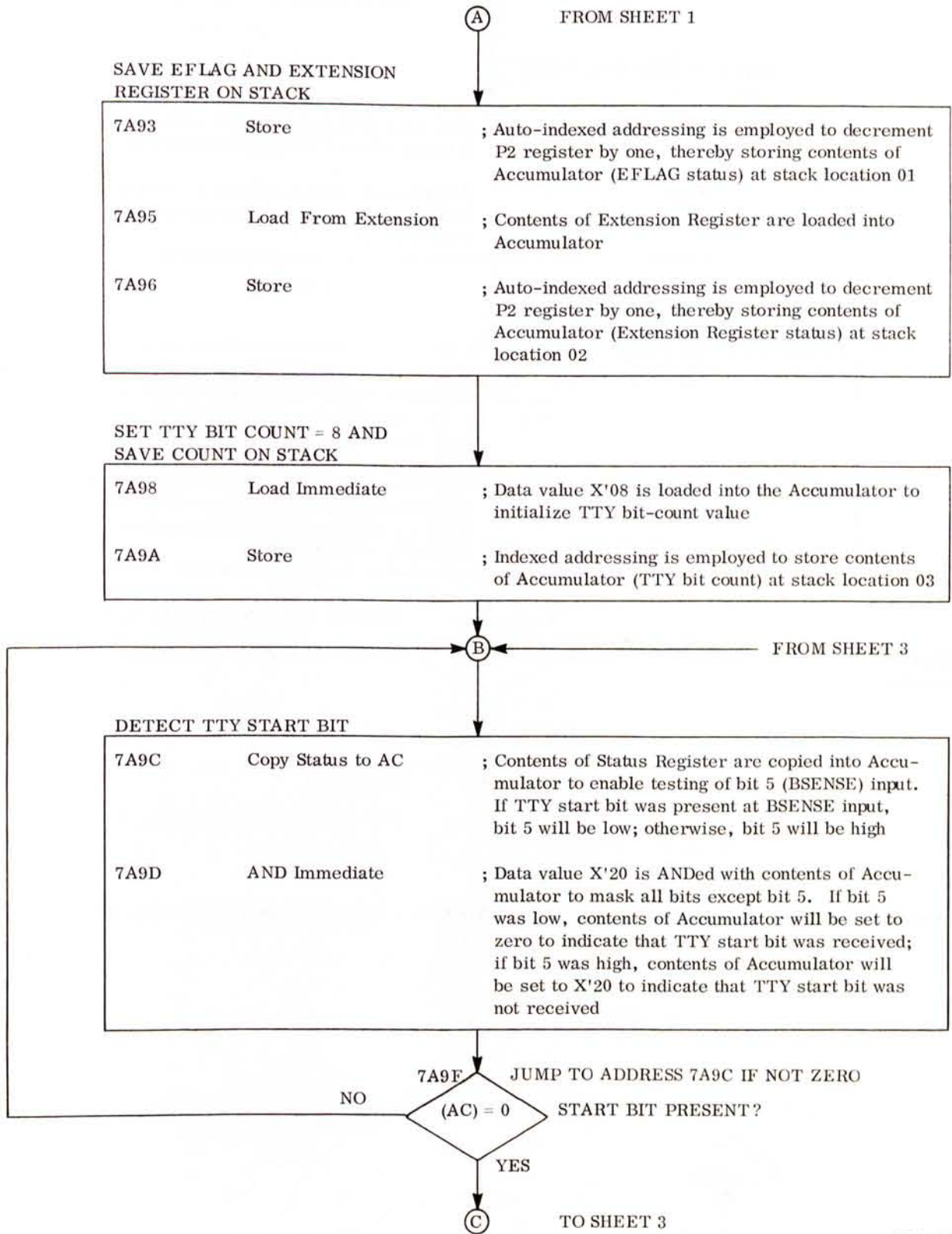


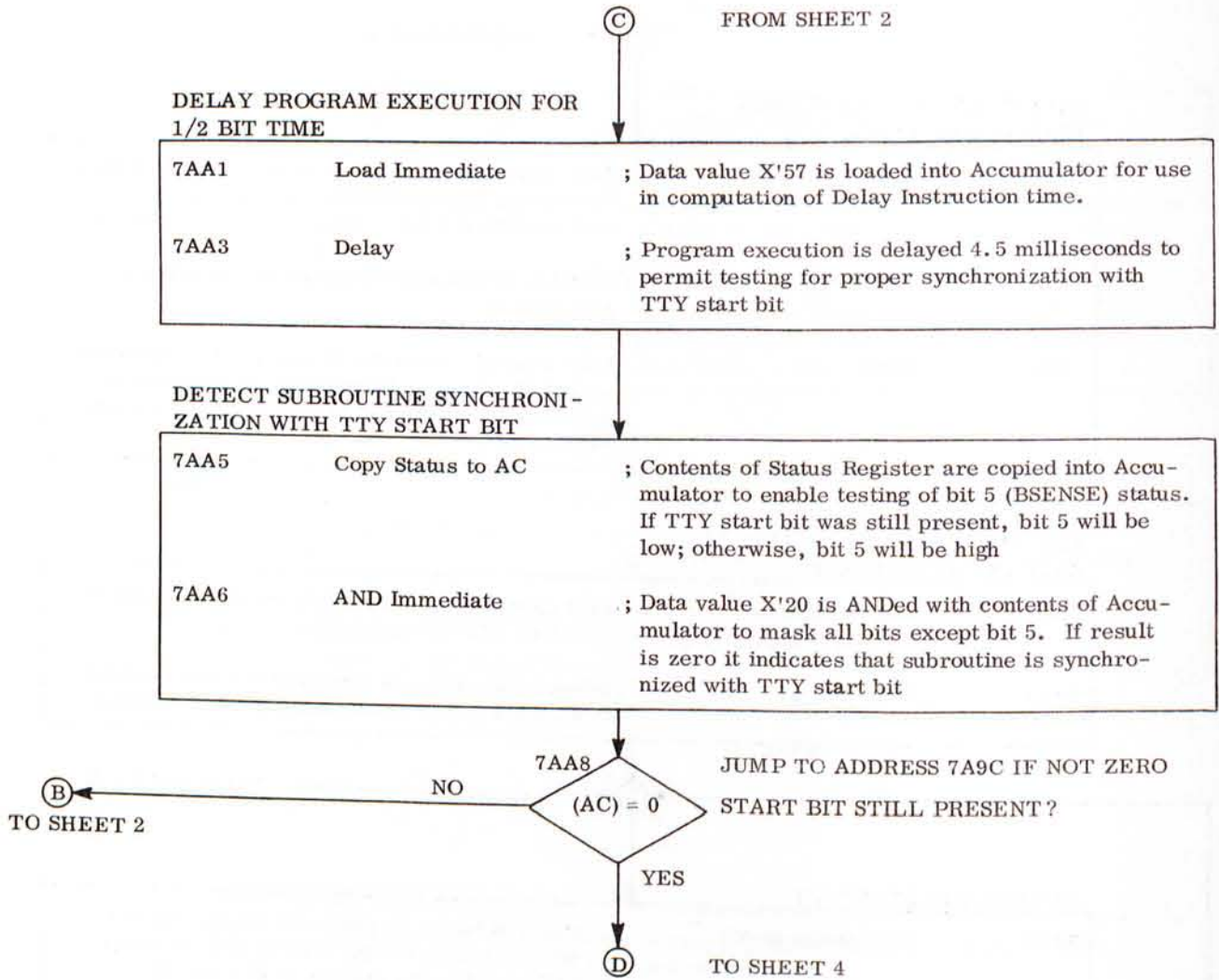
Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine – Annotated Instruction Listing (Sheet 1 of 8)



NS10649

Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine – Annotated Instruction Listing (Sheet 2 of 8)





NS10650

Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing (Sheet 3 of 8)

(D)

FROM SHEET 3

TURN OFF TTY READER RELAY  
AND SEND OUT START BIT IF  
EFLAG = 1

7AAA	Copy Status to AC	; Contents of Status Register are copied into Accumulator.
7AAB	AND Immediate	; Contents of Accumulator are ANDed with data value X'FD to set bit 1 low without affecting other bits.
7AAD	OR	; Contents of Accumulator are ORed with contents of stack location 01 to load EFLAG status into bit 0 without affecting other bits.
7AAF	Copy AC to Status	; Contents of Accumulator are returned to Status Register to set FLAG 0 output of SC/MP microprocessor low, and FLAG 2 output to state of EFLAG. Thus, TTY paper-tape reader relay is turned off via low FLAG 1 output and, if EFLAG was previously set high, start bit is sent out to TTY via FLAG 0.
NOTE		
The LCDS inverts the FLAG 0 output of the SC/MP microprocessor before routing it to the TTY; that is, a low FLAG 0 output serves as a mark, and a high FLAG 0 output serves as a space.		
7AB0	NOP	; No operation.
7AB1	NOP	; No operation.

FROM SHEET 7

(E)

DELAY 9 MILLISECONDS (1 BIT)

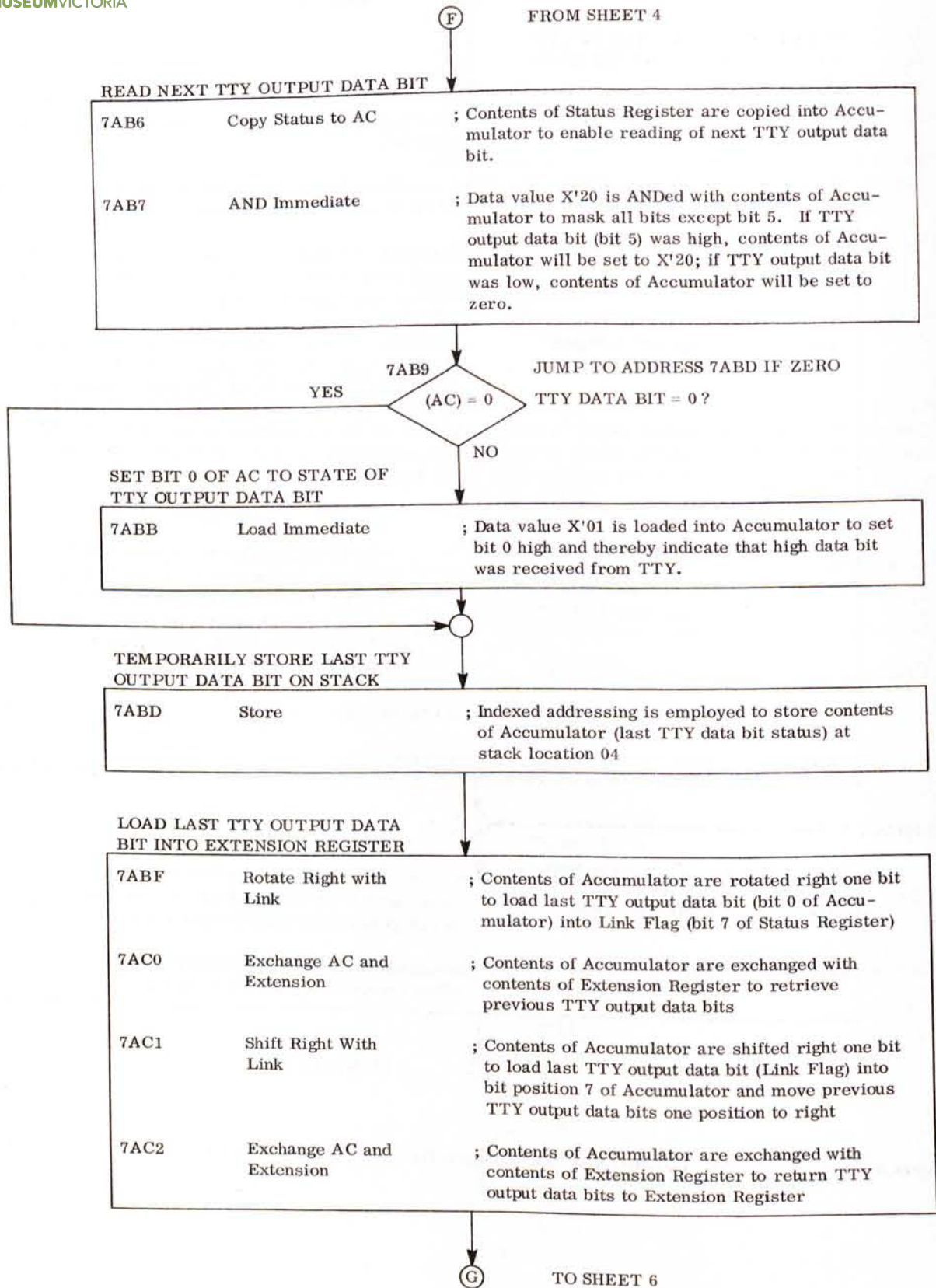
7AB2	Load Immediate	; Data value X'7E is loaded into Accumulator for use in computation of Delay Instruction time.
7AB4	Delay	; Program execution is delayed 9 milliseconds to allow reception of next TTY output data bit.

(F)

TO SHEET 5

NS10651

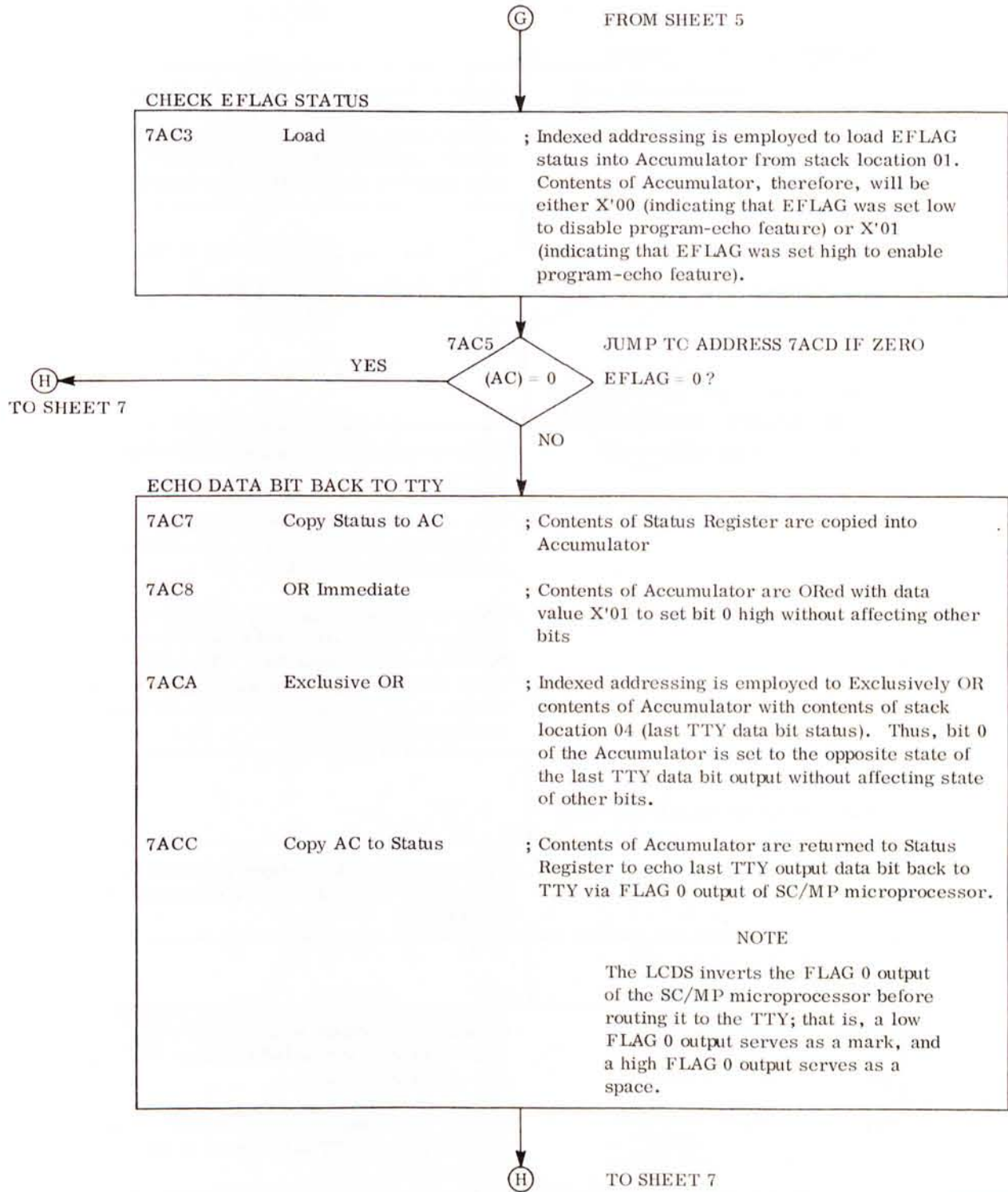
Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing (Sheet 4 of 8)



NS10652

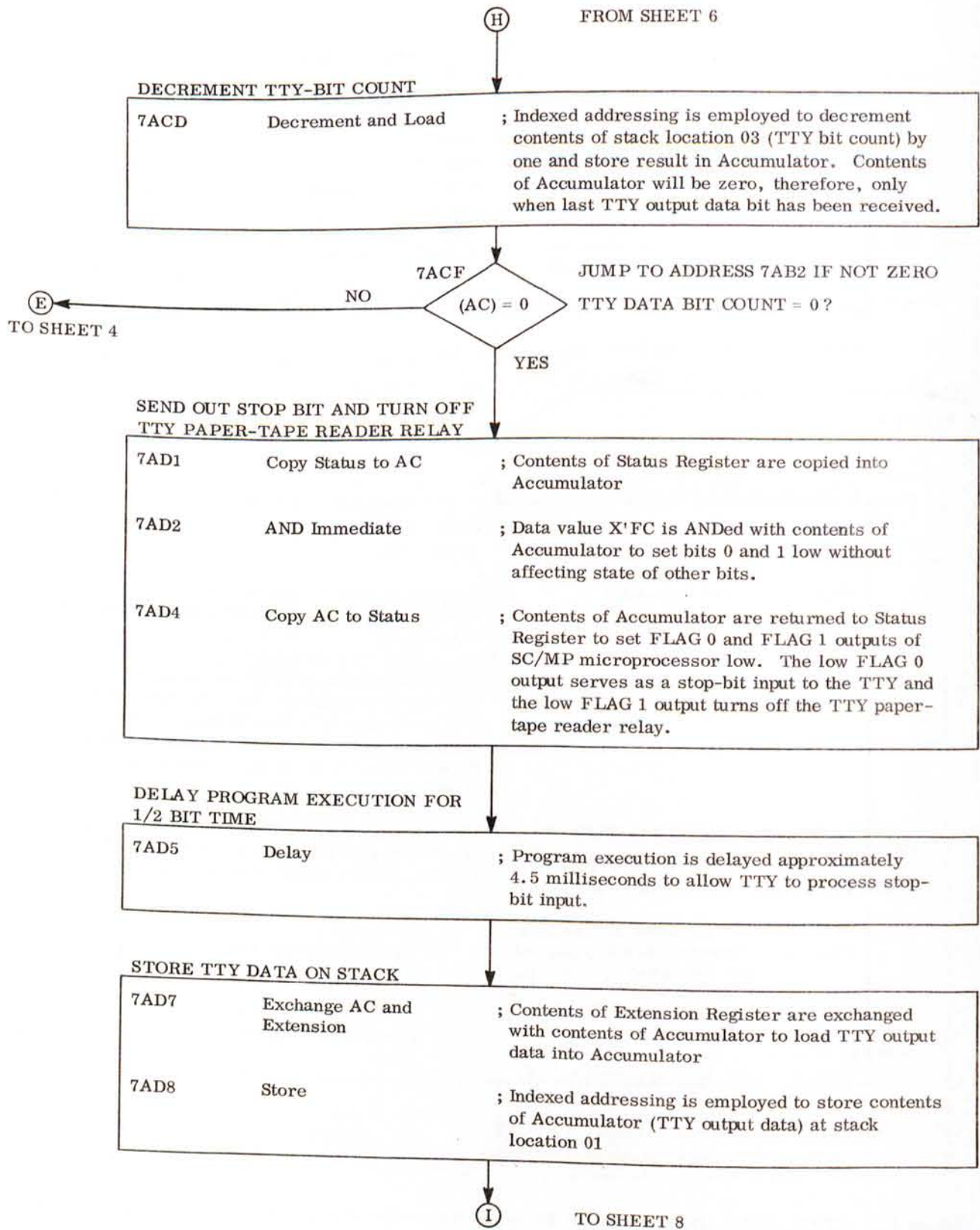
Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing (Sheet 5 of 8)





NS10653

Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing (Sheet 6 of 8)

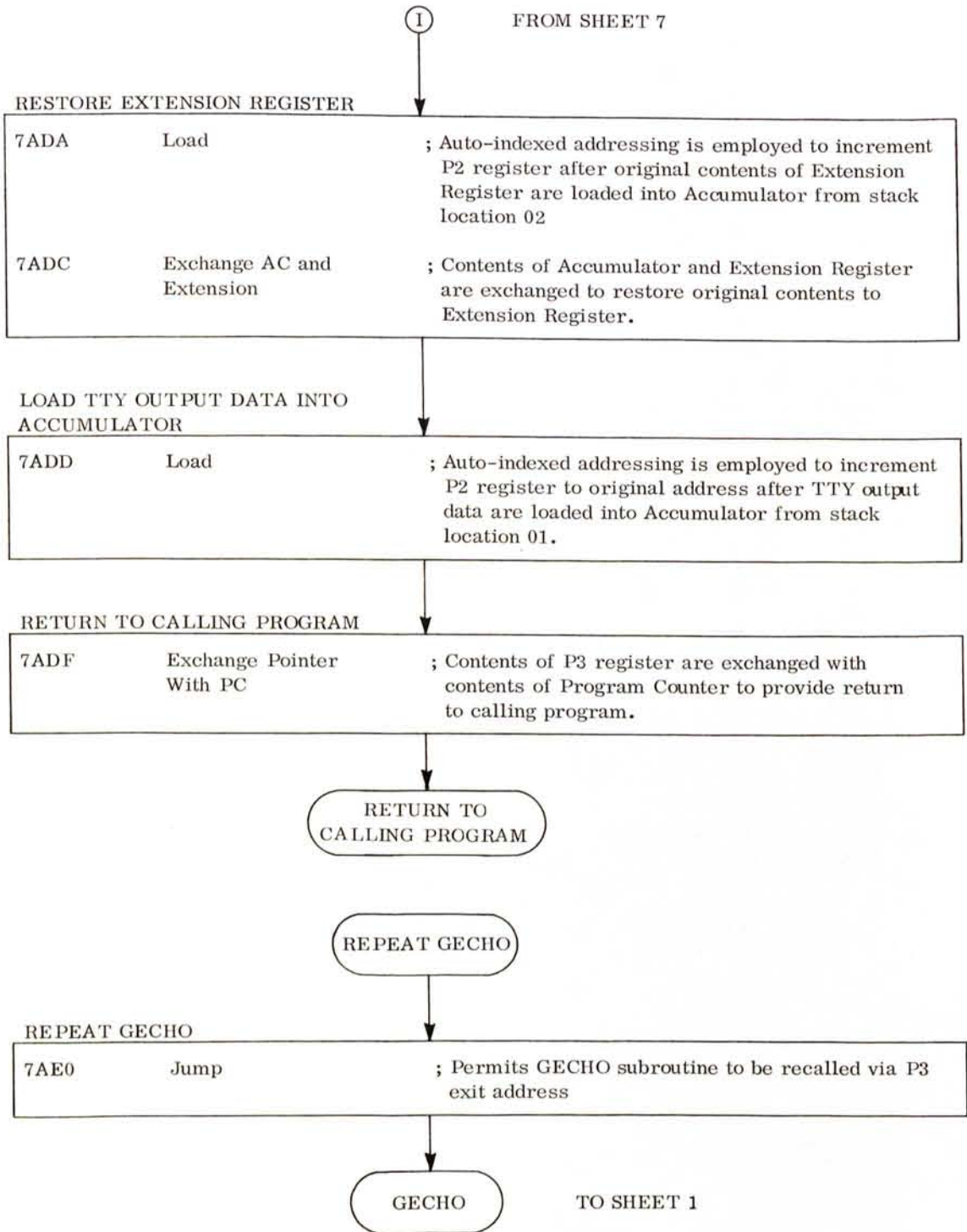


NS10654

Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine – Annotated Instruction Listing (Sheet 7 of 8)







NS10655

Figure 4-2. GETP, GETC, GECHO, and PECHO Subroutine — Annotated Instruction Listing (Sheet 8 of 8)

## GETP, GETC, GECHO, and PECHO – Actual Listing

```

560 7A85 06   GETP:   CSA           ; ENABLE READER RELAY
561 7A86 DC02  ORI           2
562 7A88 07   CAS
563 7A89 C400  GETC:   LDI           0   ; EFLAG=0 -- NO ECHO
564 7A8B 9006  JMP           $1
565 7A8D 06   PECHO:  CSA           ; PECHO - ENABLE READER
566 7A8E DC02  ORI           2   ; THEN GO ON TO GECHO
567 7A90 07   CAS
568 7A91 C401  GECHO:  LDI           1   ; EFLAG=1 -- ECHO
569 7A93 CEFF  $1:     ST           @-1(P2) ; SAVE EFLAG ON STACK
570 7A95 40   LDE           ; SAVE E REGISTER
571 7A96 CEFF  ST           @-1(P2)
572 7A98 C408  LDI           8   ; COUNT=8
573 7A9A CAFF  ST           -1(P2)
574 7A9C 06   $WAIT:  CSA           ; READ A BIT FROM TTY
575 7A9D D420  ANI           020
576 7A9F 9CFB  JNZ          $WAIT
577 7AA1 C457  LDI           87  ; TEST FOR START BIT
578 7AA3 8F04  DLY           4   ; DELAY 1/2 BIT TIME
579 7AA5 06   CSA
580 7AA6 D420  ANI           020 ; IS START BIT STILL THERE?
581 7AA8 9CF2  JNZ          $WAIT ; NO
582 7AAA C201  LD            1(P2) ; TEST FOR ECHO (EFLAG)
583 7AAC 9804  JZ           $LOOP
584 7AAE 06   CSA           ; SET START BIT INTO FLAG0
585 7AAF DC01  ORI           1
586 7AB1 07   CAS
587 7AB2 C47E  $LOOP:  LDI           126 ; DELAY 1 BIT TIME
588 7AB4 8F08  DLY           8
589 7AB6 06   CSA           ; READ 1 BIT -
590 7AB7 D420  ANI           020 ; SENSEB IS THE INPUT
591 7AB9 9802  JZ           $3
592 7ABB C401  LDI           1
593 7ABD CAFE  $3:     ST           -2(P2) ; SAVE BIT VALUE (0 OR 1)
594 7ABF 1F   RRL           ; ROTATE INTO LINK
595 7AC0 01   XAE
596 7AC1 1D   SRL           ; SHIFT INTO CHARACTER
597 7AC2 01   XAE           ; RETURN CHARACTER TO E REG
598 7AC3 C201  LD            1(P2) ; CHECK EFLAG
599 7AC5 9806  JZ           $4
600 7AC7 06   CSA           ; MASK OUT OLD FLAG
601 7AC8 DC01  ORI           1
602 7ACA E2FE  XOR           -2(P2) ; ADD IN NEW BIT
603 7ACC 07   CAS           ; PUT BACK INTO STATUS FLAGS
604 7ACD BAFF  $4:     DLD           -1(P2) ; DECREMENT BIT COUNT
605 7ACF 9CE1  JNZ          $LOOP ; AND LOOP IF NOT DONE
606 7AD1 06   CSA           ; SET STOP BIT INTO FLAG0
607 7AD2 D4FC  ANI           0FC
608 7AD4 07   CAS
609 7AD5 8F08  DLY           8
610 7AD7 01   XAE           ; TEMPORARILY SAVE CHARACTER
611 7AD8 CA01  ST            1(P2) ; ON STACK
612 7ADA C601  LD            @1(P2) ; RESTORE ORIGINAL E (SKIP COUNT)
613 7ADC 01   XAE
614 7ADD C601  LD            @1(P2) ; RESTORE A
615 7ADF 3F   XPPC          P3 ; RETURN
616 7AE0 90AF  JMP          GECHO ; FOR SUBSEQUENT CALLS - GECHO ONLY
  
```





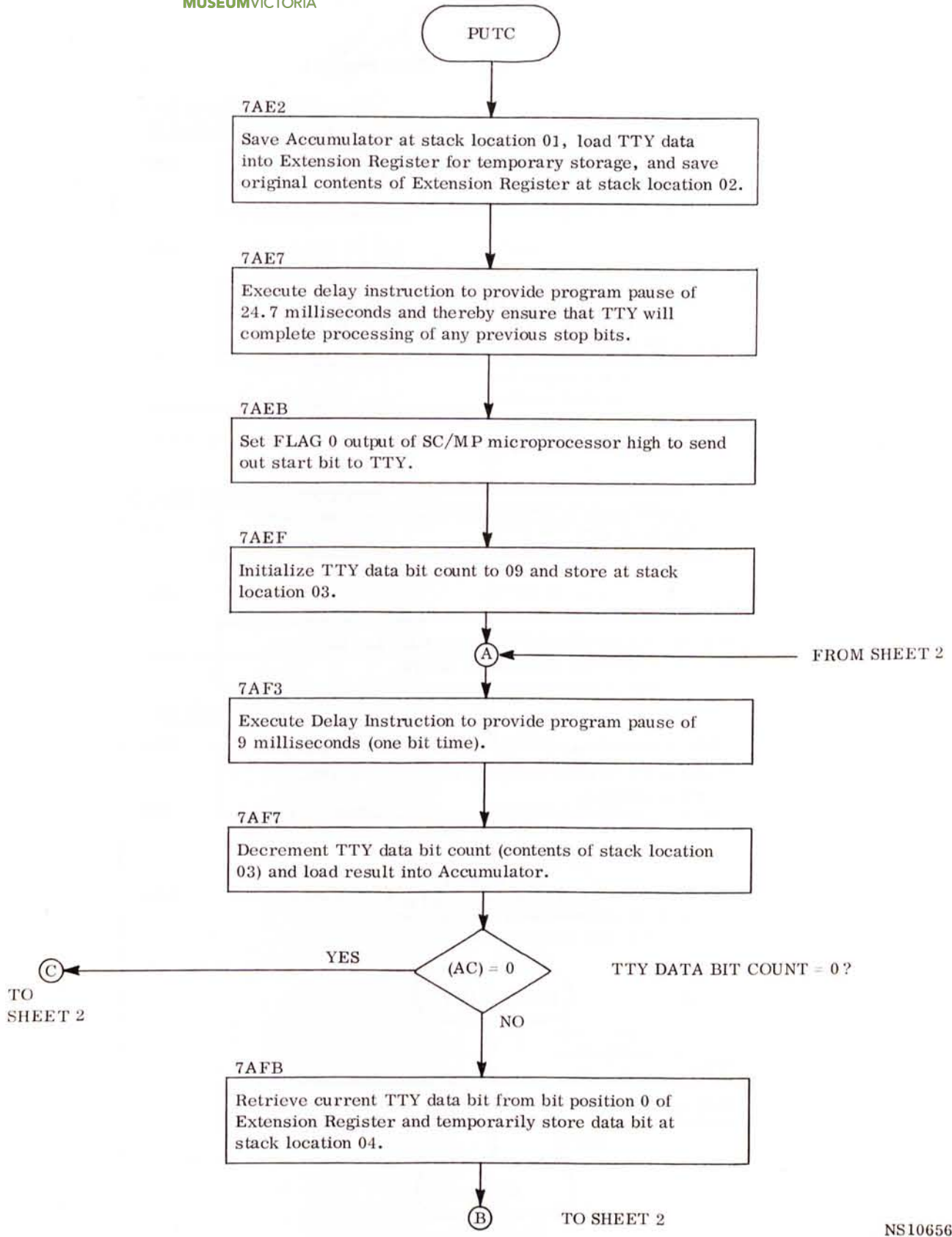


Figure 4-3. PUTC Subroutine – Detailed Flowchart (Sheet 1 of 2)

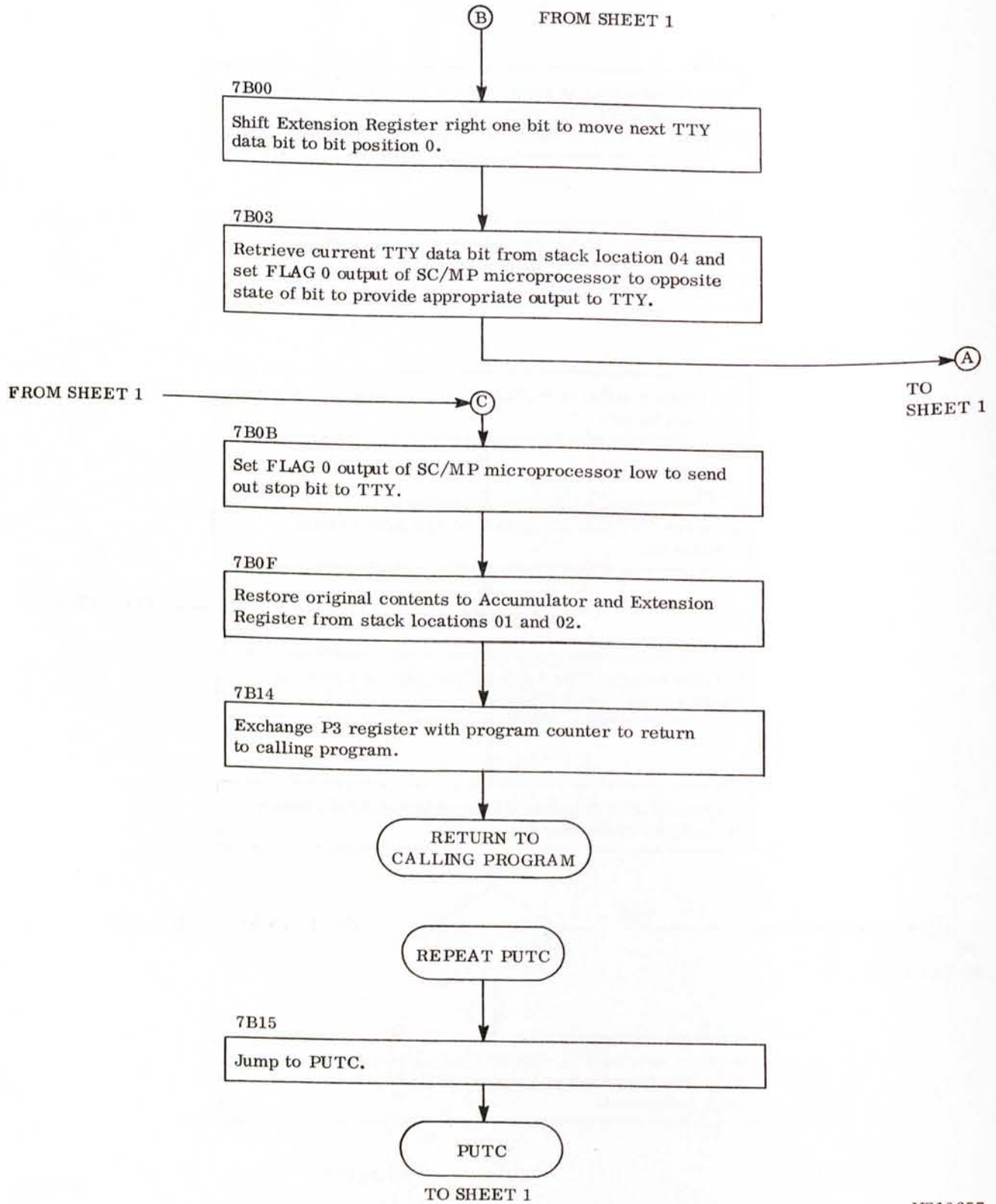


Figure 4-3. PUTC Subroutine — Detailed Flowchart (Sheet 2 of 2)

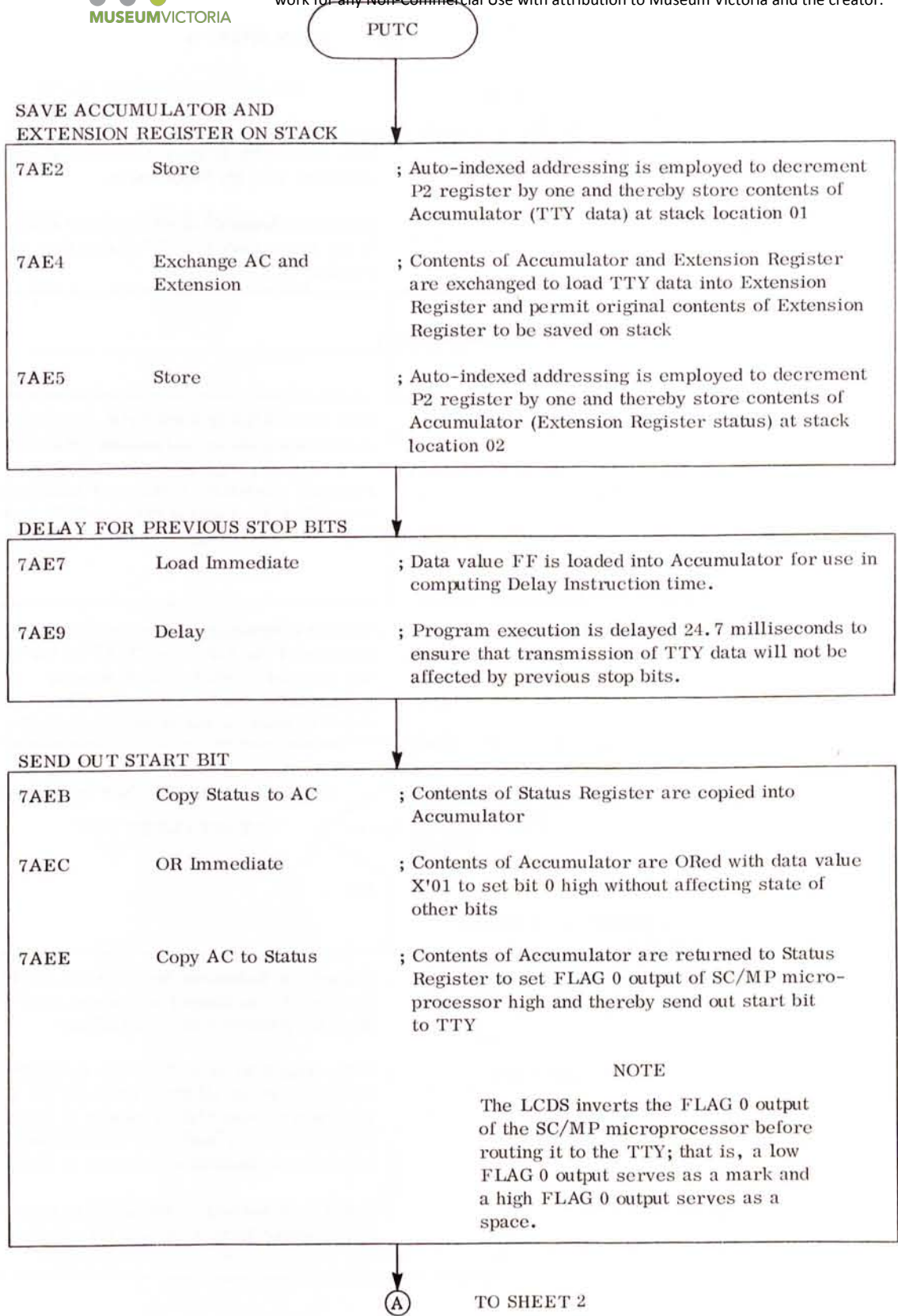
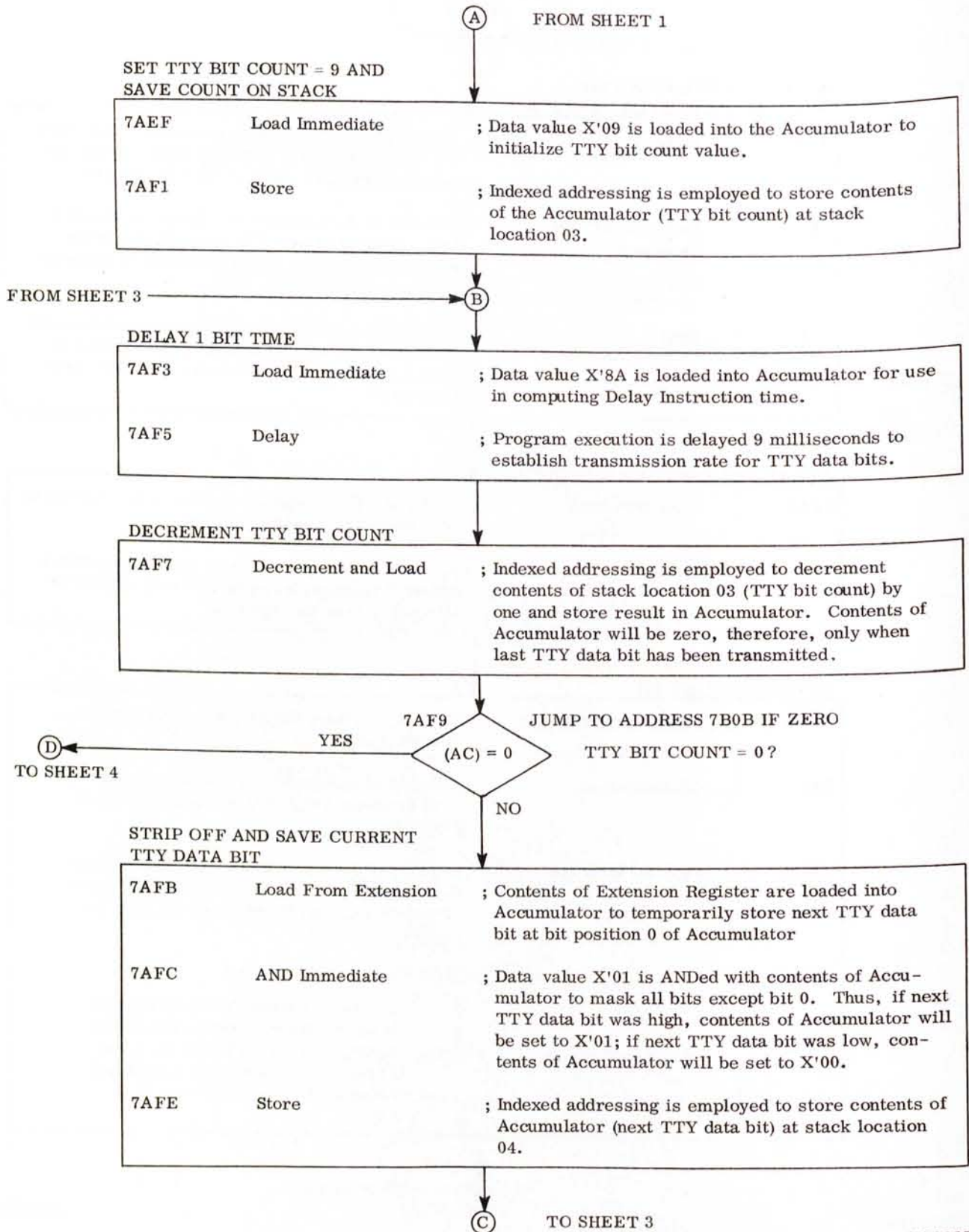


Figure 4-4. PUTC Subroutine — Annotated Instruction Listing (Sheet 1 of 4)





NS10659

Figure 4-4. PUTC Subroutine — Annotated Instruction Listing (Sheet 2 of 4)

Ⓢ

FROM SHEET 2

ROTATE NEXT TTY DATA BIT  
INTO POSITION

7B00	Exchange AC and Extension	; Contents of Accumulator and Extension Register are exchanged to load TTY data into Accumulator.
7B01	Shift Right	; Contents of Accumulator are shifted right one bit to locate next TTY data bit at bit position 0.
7B02	Exchange AC and Extension	; Contents of Extension Register and Accumulator are exchanged to return TTY data to Extension Register.

RECALL CURRENT BIT AND  
SEND TO TTY

7B03	Copy Status to AC	; Contents of Status Register are copied into Accumulator.
7B04	OR Immediate	; Contents of Accumulator are ORed with Data Value X'01 to set bit 0 high without affecting state of other bits.
7B06	Exclusive OR	; Indexed addressing is employed to Exclusively OR contents of Accumulator with contents of stack location 04 (next TTY data bit) and store result in Accumulator. Thus, bit 0 of Accumulator is set to opposite state of next TTY data bit without affecting state of other bits.
7B08	Copy AC to Status	; Contents of Accumulator are returned to Status Register to transmit data bit to TTY via FLAG 0 output of SC/MP microprocessor.

NOTE

The LCDS inverts the FLAG 0 output of the SC/MP microprocessor before routing it to the TTY; that is, a low FLAG 0 output serves as a mark and a high FLAG 0 output serves as a space.

7B09	Jump	; Jump to address 7AF3 (Delay Instruction) to continue transmission of TTY data bits.
------	------	---

Ⓢ

TO SHEET 2

NS10660

Figure 4-4. PUTC Subroutine — Annotated Instruction Listing (Sheet 3 of 4)

SEND OUT STOP BIT

7B0B	Copy Status to AC	; Contents of Status Register are copied into Accumulator
7B0C	AND Immediate	; Data value X'FE is ANDed with contents of Accumulator to set bit 0 low without affecting state of other bits.
7B0E	Copy AC to Status	; Contents of Accumulator are returned to Status Register to set FLAG 0 output of SC/MP microprocessor low and thereby transmit stop bit to TTY.

NOTE

The LCDS inverts the FLAG 0 output of the SC/MP microprocessor before routing it to the TTY; that is, a low FLAG 0 output serves as a mark and a high FLAG 0 output serves as a space.

RESTORE ACCUMULATOR AND EXTENSION REGISTER

7B0F	Load	; Auto-indexed addressing is employed to increment P2 register by one after original contents of Extension Register are loaded into Accumulator from stack location 02.
7B11	Exchange AC and Extension	; Contents of Accumulator and Extension Register are exchanged to restore original contents to Extension Register
7B12	Load	; Auto-indexed addressing is employed to increment P2 register to original address after TTY data is loaded into Accumulator from stack location 01.

RETURN TO CALLING PROGRAM

7B14	Exchange Pointer With PC	; Contents of P3 register are exchanged with contents of Program Counter to provide return to calling program
------	--------------------------	---

RETURN TO CALLING PROGRAM

REPEAT PUTC

REPEAT PUTC

7B15	Jump	; Permits PUTC subroutine to be recalled via P3 exit address.
------	------	---

PUTC

TO SHEET 1

NS10661

Figure 4-4. PUTC Subroutine — Annotated Instruction Listing (Sheet 4 of 4)



ASCII CHARACTER - PUTC

CHARACTER TO THE TELETYPE

IN A ON ENTRY

```
;)          ; SAVE REGISTERS
;)          ; COPY CHARACTER INTO E

;)          ; DELAY FOR PREVIOUS STOP BITS

;)          ; FLAG0 = 0 (START BIT)

;)          ; BIT COUNTER = 9
;)          ; DELAY 1 BIT TIME
;)          ; DECREMENT BIT COUNT
;)          ; PREPARE NEXT BIT

;)          ; SHIFT E REG RIGHT FOR NEXT

;)          ; MASK OUT OLD FLAG
;)          ; ADD IN NEW BIT
;)          ; PUT BACK INTO STATUS FLAGS

;)          ; FLAG0 = 1 (STOP BIT)

;)          ; RESTORE ORIGINAL E (SKIP COUNT)

;)          ; RESTORE A
;)          ; RETURN
;)          ; FOR SUBSEQUENT CALLS
```



## PUTC Subroutine — Actual Listing

```

617          .PAGE    'PRINT ASCII CHARACTER - PUTC'
618          .LOCAL
619          ;
620          ;      PUTC - SEND A CHARACTER TO THE TELETYPE
621          ;
622          ;      CHARACTER IS IN A ON ENTRY
623          ;
624 7AE2 CEFF  PUTC:  ST      @-1(P2)      ; SAVE REGISTERS
625 7AE4 01    XAE      ; COPY CHARACTER INTO E
626 7AE5 CEFF  ST      @-1(P2)
627 7AE7 C4FF  LDI      255      ; DELAY FOR PREVIOUS STOP BITS
628 7AE9 8F17  DLY      23
629 7AEB 06    CSA      ; FLAG0 = 0 (START BIT)
630 7AEC DC01  ORI      1
631 7AEE 07    CAS
632 7AEF C409  LDI      9      ; BIT COUNTER = 9
633 7AF1 CAFF  ST      -1(P2)
634 7AF3 C48A  $LOOP:  LDI      138     ; DELAY 1 BIT TIME
635 7AF5 8F08  DLY      8
636 7AF7 BAFF  DLD      -1(P2)     ; DECREMENT BIT COUNT
637 7AF9 9810  JZ      $DONE
638 7AFB 40    LDE
639 7AFC D401  ANI      1      ; PREPARE NEXT BIT
640 7AFE CAFE  ST      -2(P2)
641 7B00 01    XAE      ; SHIFT E REG RIGHT FOR NEXT
642 7B01 1C    SR
643 7B02 01    XAE
644 7B03 06    CSA      ; MASK OUT OLD FLAG
645 7B04 DC01  ORI      1
646 7B06 E2FE  XOR      -2(P2)     ; ADD IN NEW BIT
647 7B08 07    CAS      ; PUT BACK INTO STATUS FLAGS
648 7B09 90E8  JMP      $LOOP
649 7B0B 06    $DONE:  CSA      ; FLAG0 = 1 (STOP BIT)
650 7B0C D4FE  ANI      0FE
651 7B0E 07    CAS
652 7B0F C601  LD      @1(P2)     ; RESTORE ORIGINAL E (SKIP COUNT)
653 7B11 01    XAE
654 7B12 C601  LD      @1(P2)     ; RESTORE A
655 7B14 3F    XPPC      P3      ; RETURN
656 7B15 90CB  JMP      PUTC      ; FOR SUBSEQUENT CALLS

```





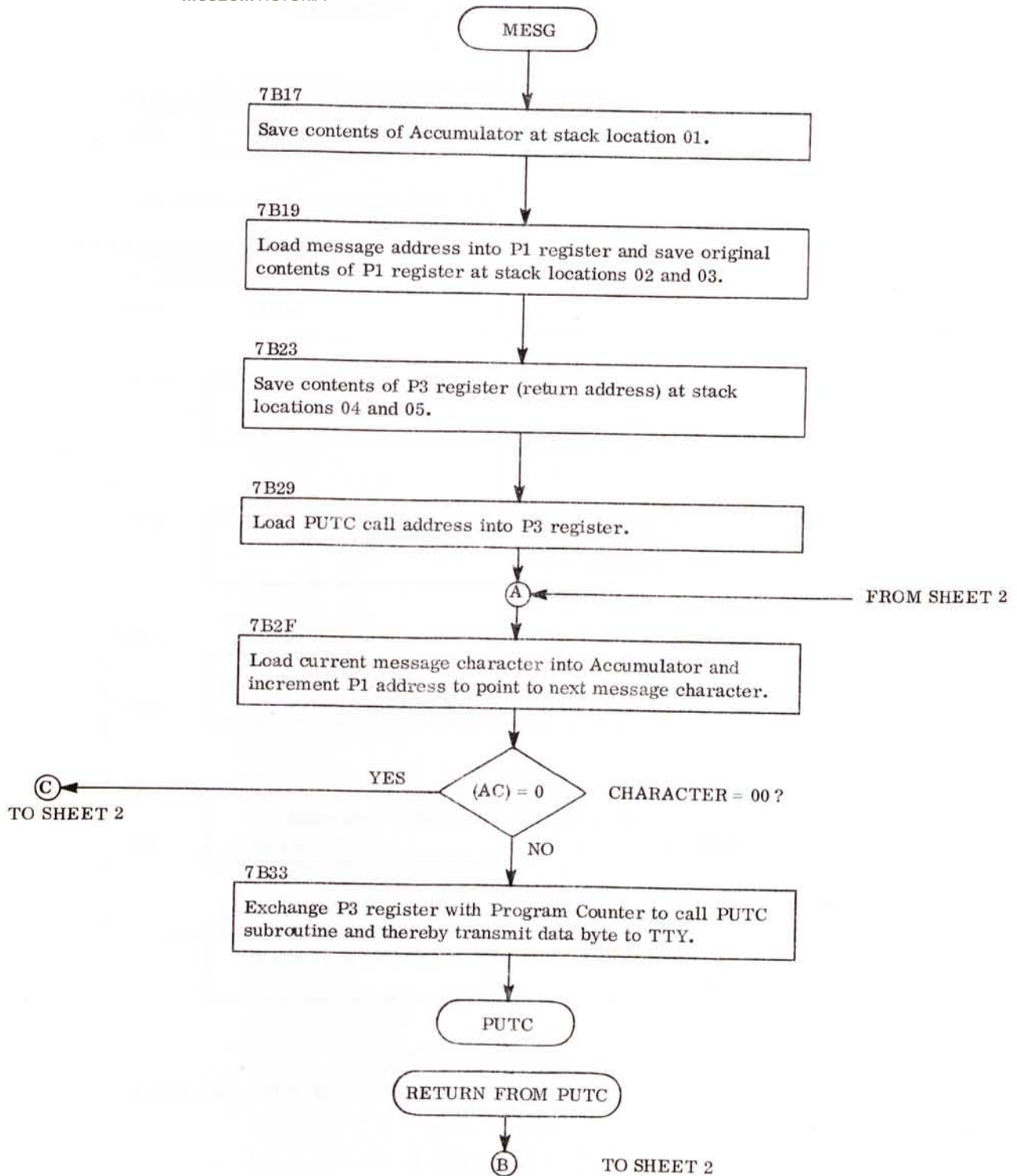


Figure 4-5. MESH Subroutine -- Detailed Flowchart (Sheet 1 of 2)

NS10662

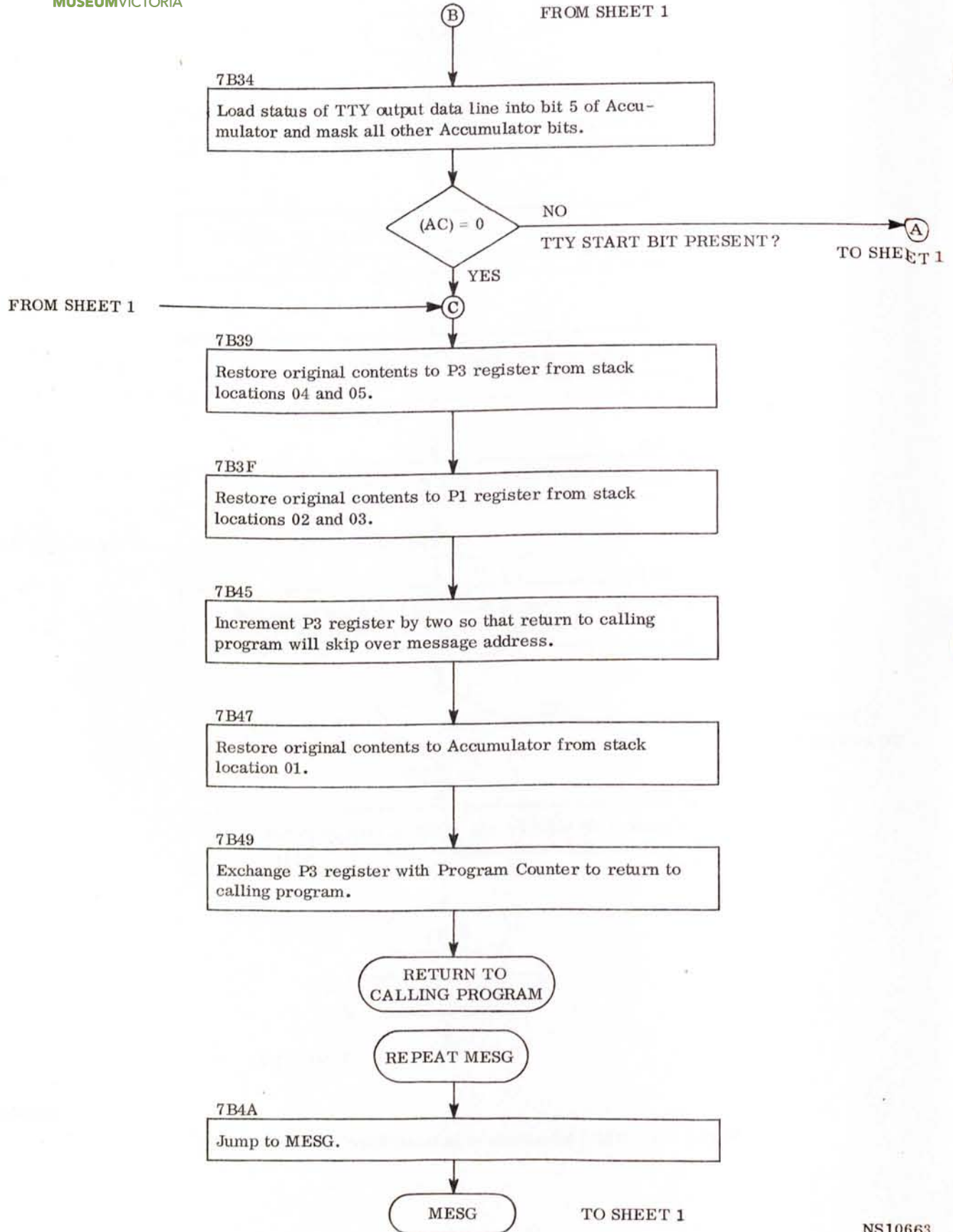
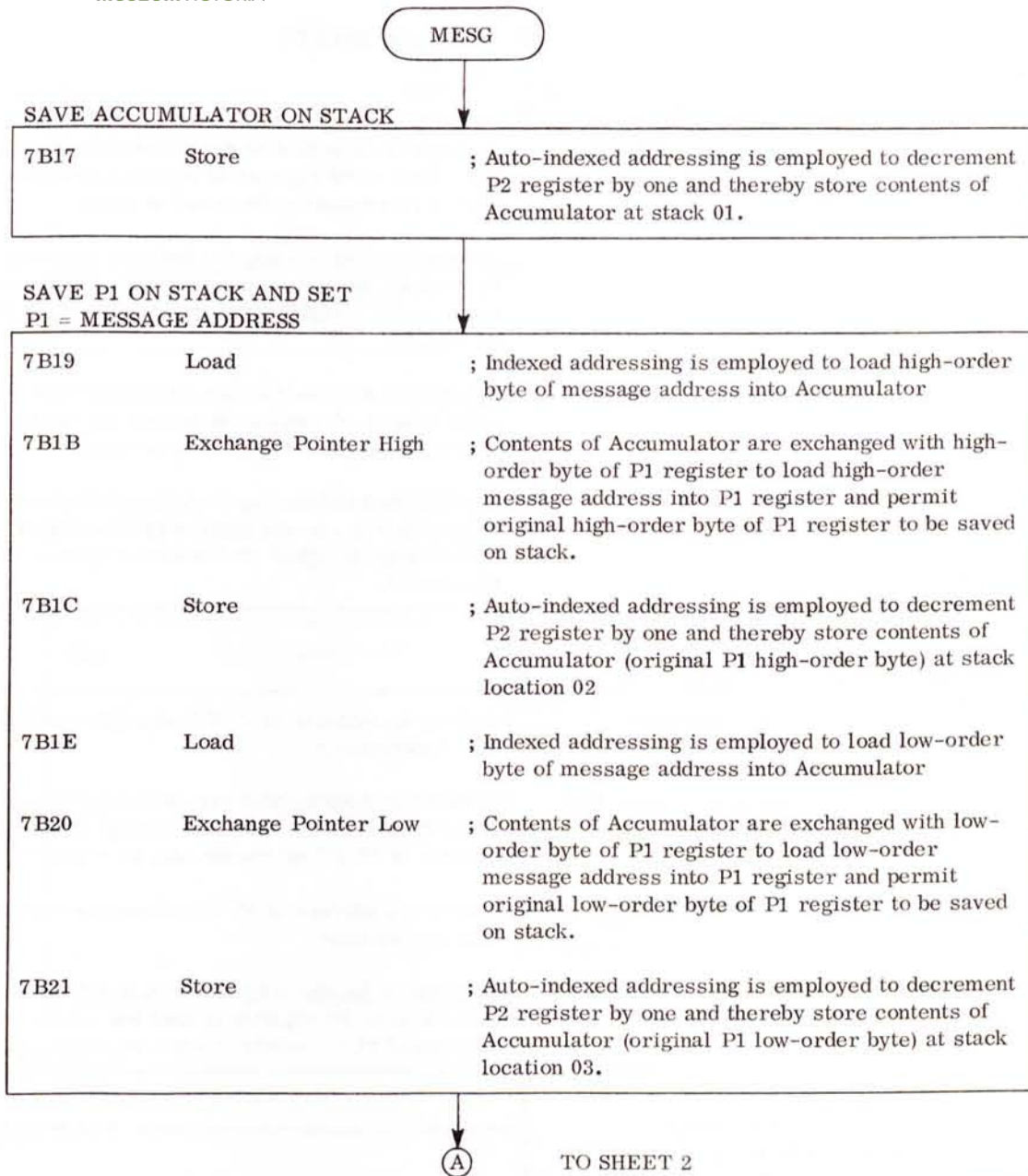


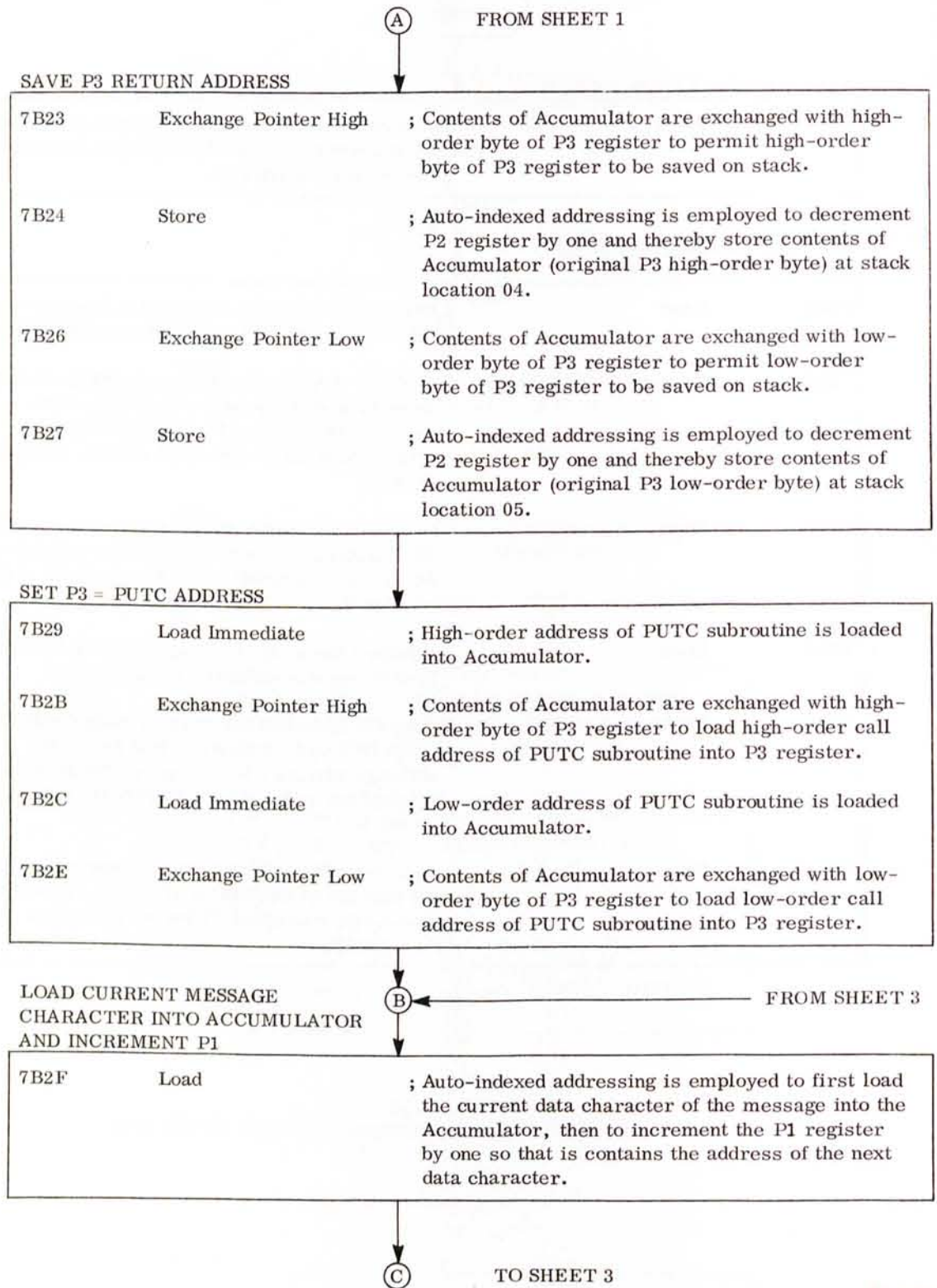
Figure 4-5. MESH Subroutine — Detailed Flowchart (Sheet 2 of 2)





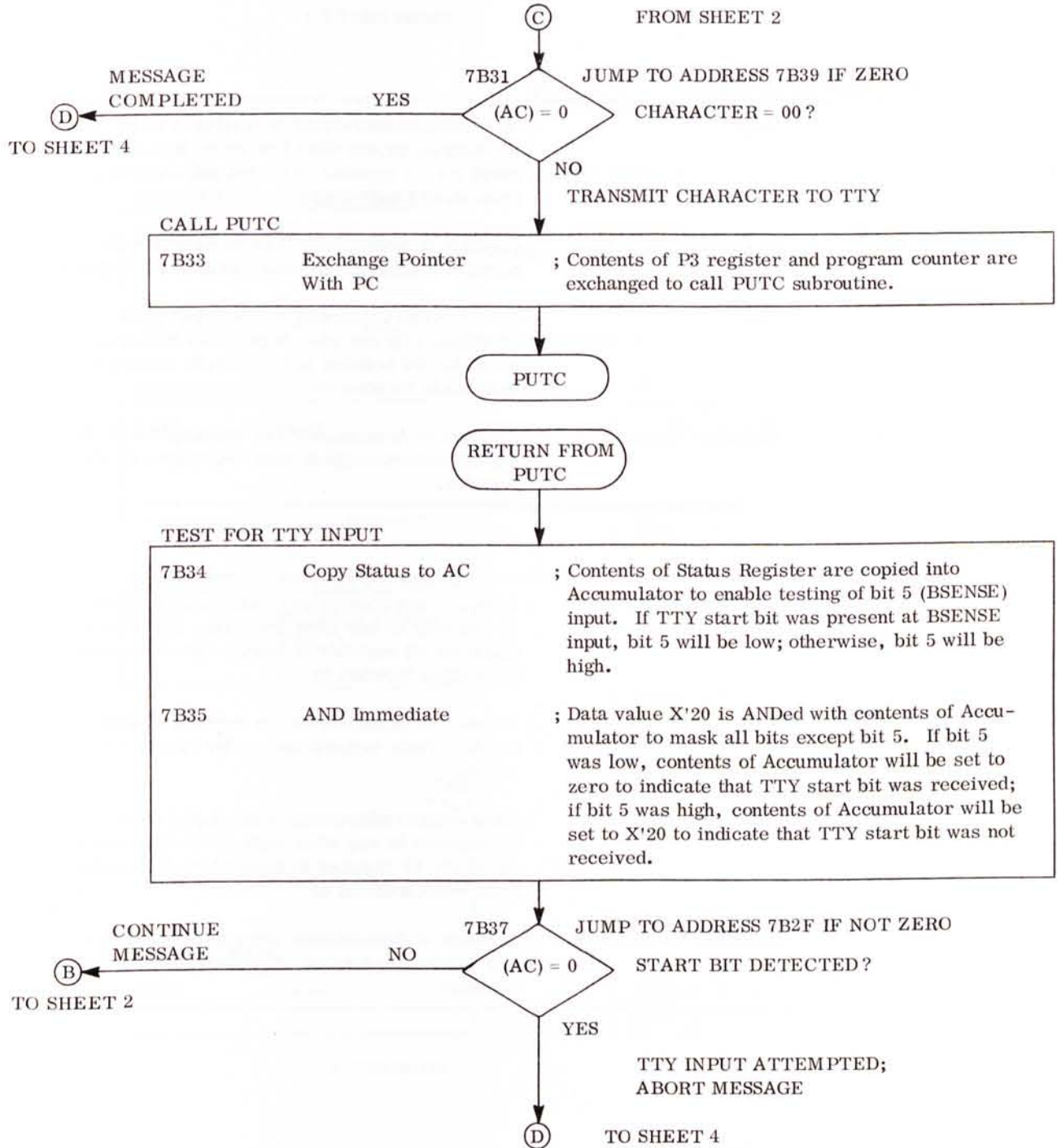
NS10664

Figure 4-6. MESH Subroutine — Annotated Instruction Listing (Sheet 1 of 5)



NS10665

Figure 4-6. MESH Subroutine — Annotated Instruction Listing (Sheet 2 of 5)



NS10666

Figure 4-6. MSG Subroutine – Annotated Instruction Listing (Sheet 3 of 5)



ⓓ

FROM SHEET 3

RESTORE P3 REGISTER

7B39	Load	; Auto-indexed addressing is employed to increment P2 register by one after low-order byte that was saved for P3 register is loaded into Accumulator from stack location 05.
7B3B	Exchange Pointer Low	; Contents of Accumulator are exchanged with P3-low to restore original low-order byte to P3 register.
7B3C	Load	; Auto-indexed addressing is employed to increment P2 register by one after high-order byte that was saved for P3 register is loaded into Accumulator from stack location 04.
7B3E	Exchange Pointer High	; Contents of Accumulator are exchanged with P3-high to restore original high-order byte to P3 register.

RESTORE P1 REGISTER

7B3F	Load	; Auto-indexed addressing is employed to increment P2 register by one after low-order byte that was saved for P1 register is loaded into Accumulator from stack location 03.
7B41	Exchange Pointer Low	; Contents of Accumulator are exchanged with P1-low to restore original low-order byte to P1 register.
7B42	Load	; Auto-indexed addressing is employed to increment P2 register by one after high-order byte that was saved for P1 register is loaded into Accumulator from stack location 02.
7B44	Exchange Pointer High	; Contents of Accumulator are exchanged with P1-high to restore original high-order byte to P1 register.

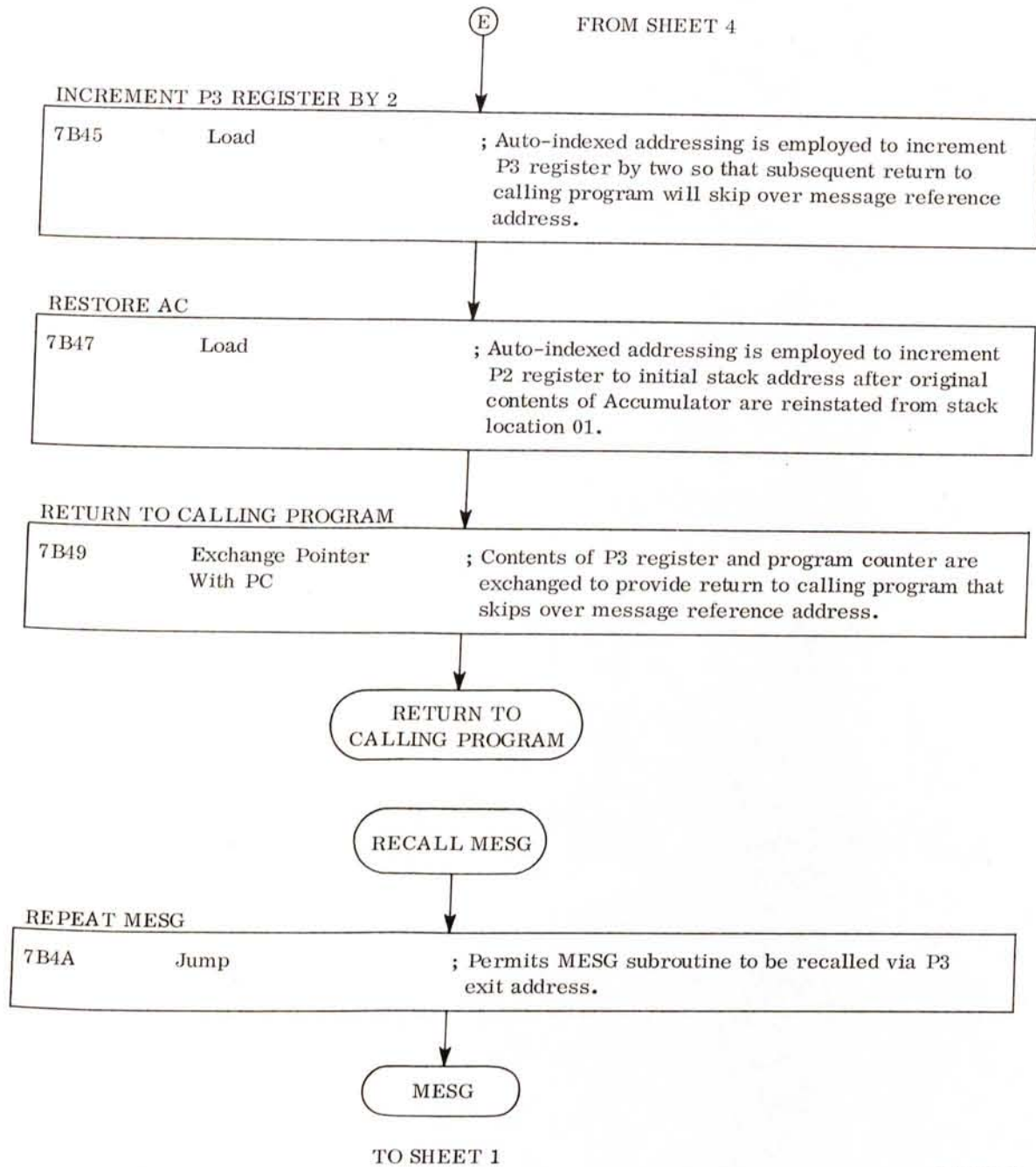
ⓔ

TO SHEET 5

NS10667

Figure 4-6. MESH Subroutine — Annotated Instruction Listing (Sheet 4 of 5)





NS10668

Figure 4-6. MESG Subroutine — Annotated Instruction Listing (Sheet 5 of 5)



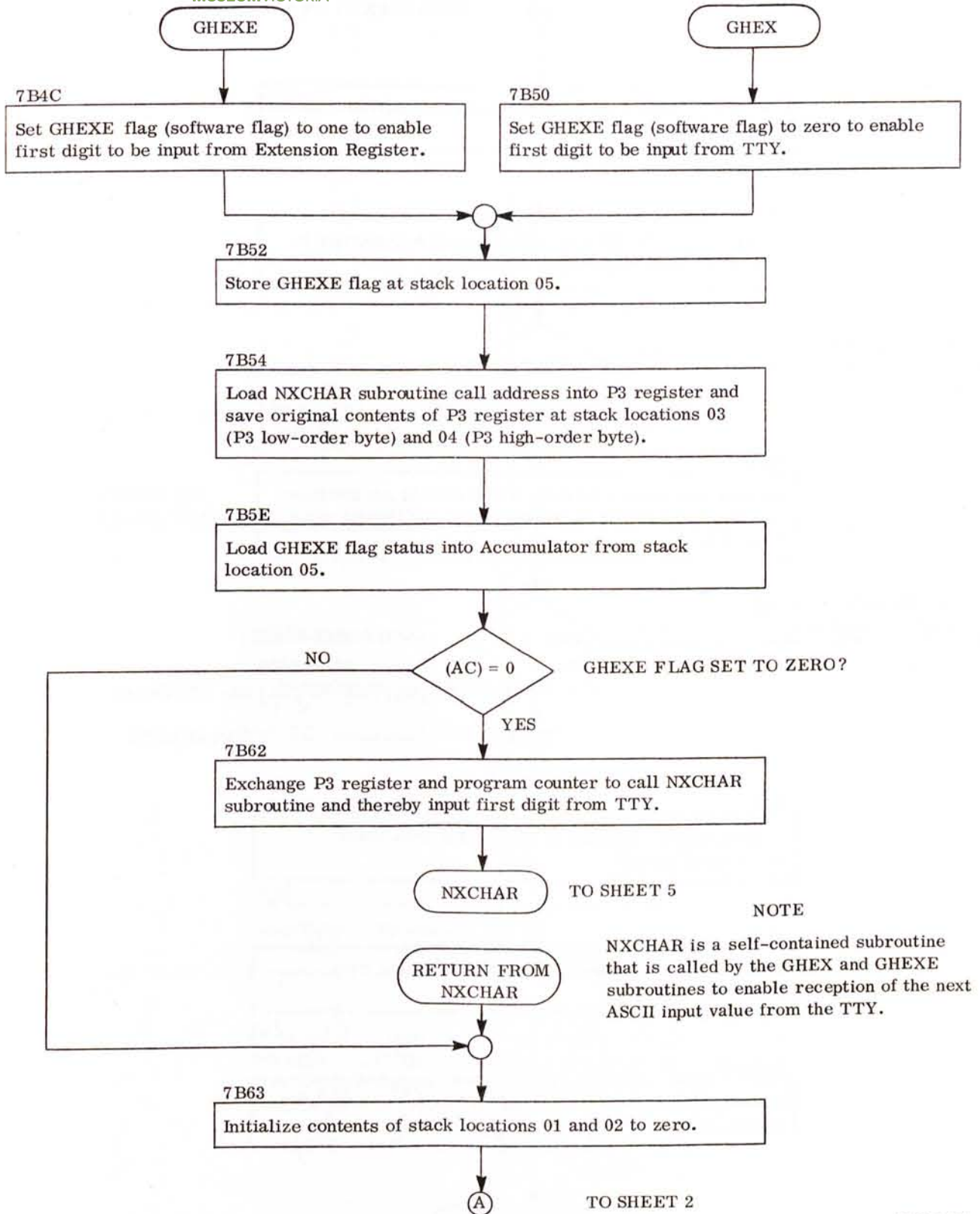
### MESG Subroutine — Actual Listing

```

657          .PAGE      'PRINT ASCII MESSAGE STRING - MESG'
658          .LOCAL
659          ;
660          ;          MSG - MESSAGE PRINTING ROUTINE
661          ;
662          ;          CALL IS:
663          ;
664          ;          (SET UP P3 TO POINT TO MSG)
665          ;          XPPC      P3          ; CALL MSG
666          ;          .DBYTE   MSG          ; MESSAGE ADDRESS
667          ;
668          ;          THE MESSAGE IS TERMINATED BY A NULL (ZERO) CHARACTER.
669          ;
670 7B17 CEFF  MSG:      ST          @-1(P2)          ; SAVE A
671 7B19 C301          LD          1(P3)          ; COPY MSG ADDRESS INTO P1
672 7B1B 35          XPAH         P1          ; WHILE SAVING OLD P1
673 7B1C CEFF          ST          @-1(P2)
674 7B1E C302          LD          2(3)
675 7B20 31          XPAL         P1
676 7B21 CEFF          ST          @-1(P2)
677 7B23 37          XPAH         P3          ; SAVE P3
678 7B24 CEFF          ST          @-1(P2)
679 7B26 33          XPAL         P3
680 7B27 CEFF          ST          @-1(P2)
681 7B29 C47A          LDI         H(PUTC)          ; SET P3 TO PUTC ADDRESS
682 7B2B 37          XPAH         P3
683 7B2C C4E1          LDI         L(PUTC)-1
684 7B2E 33          XPAL         P3
685 7B2F C501          $LOOP:     LD          @1(P1)          ; LOAD CHAR AND INCREMENT P1
686 7B31 9806          JZ          $END
687 7B33 3F          XPPC         P3          ; CALL PUTC
688 7B34 06          CSA          ; CHECK FOR TTY START BIT
689 7B35 D420          ANI         020          ; ABORT MESSAGE IF INPUT ATTEMPTED
690 7B37 9CF6          JNZ         $LOOP
691 7B39 C601          $END:     LD          @1(P2)          ; RESTORE P3 AND P1
692 7B3B 33          XPAL         P3
693 7B3C C601          LD          @1(P2)
694 7B3E 37          XPAH         P3
695 7B3F C601          LD          @1(P2)
696 7B41 31          XPAL         P1
697 7B42 C601          LD          @1(P2)
698 7B44 35          XPAH         P1
699 7B45 C702          LD          @2(3)          ; INCREMENT P3 BY 2
700 7B47 C601          LD          @1(P2)          ; RESTORE A
701 7B49 3F          XPPC         P3          ; RETURN, SKIPPING OVER PARAMETER
702 7B4A 90CB          JMP          MSG

```

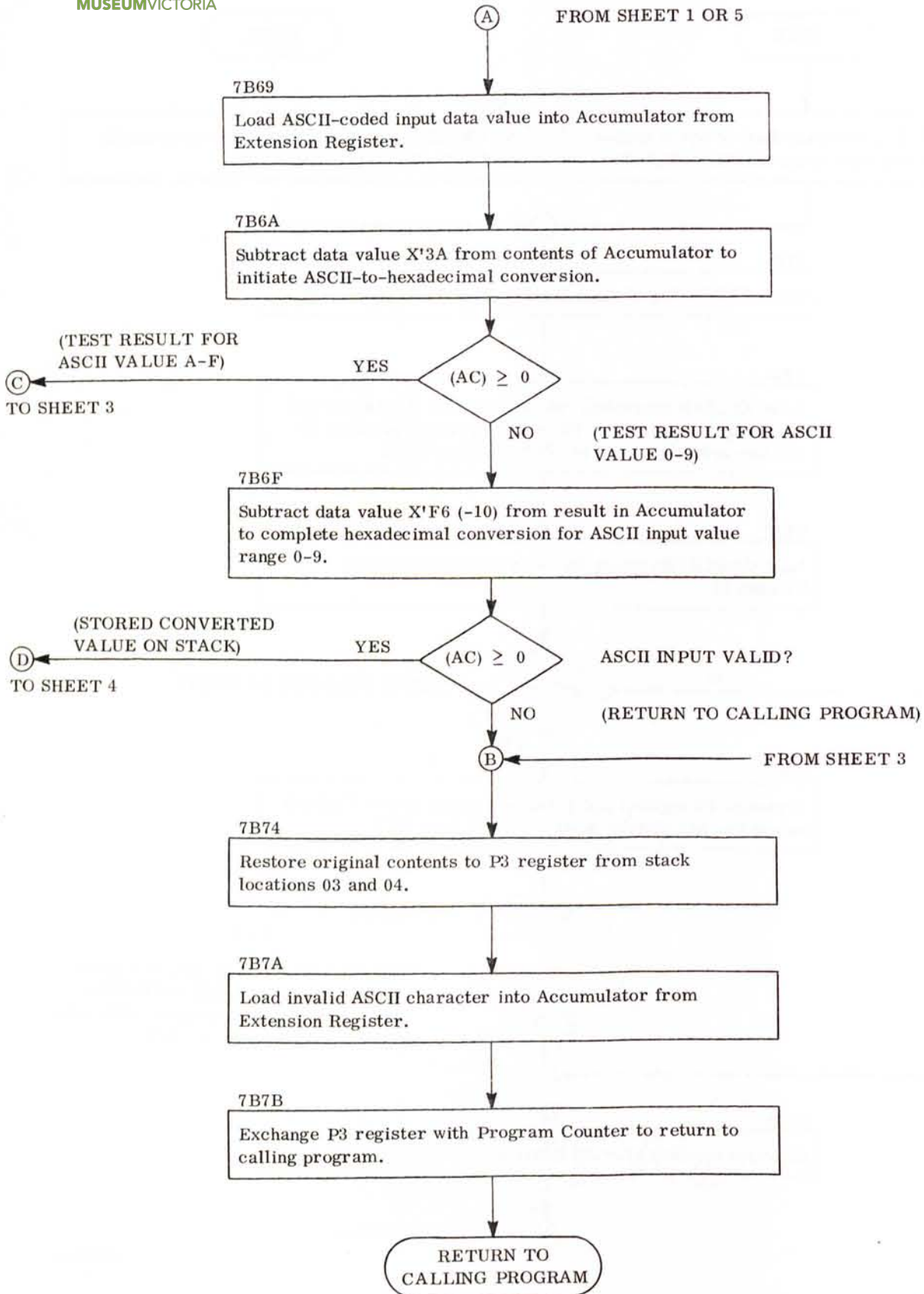




NS10669

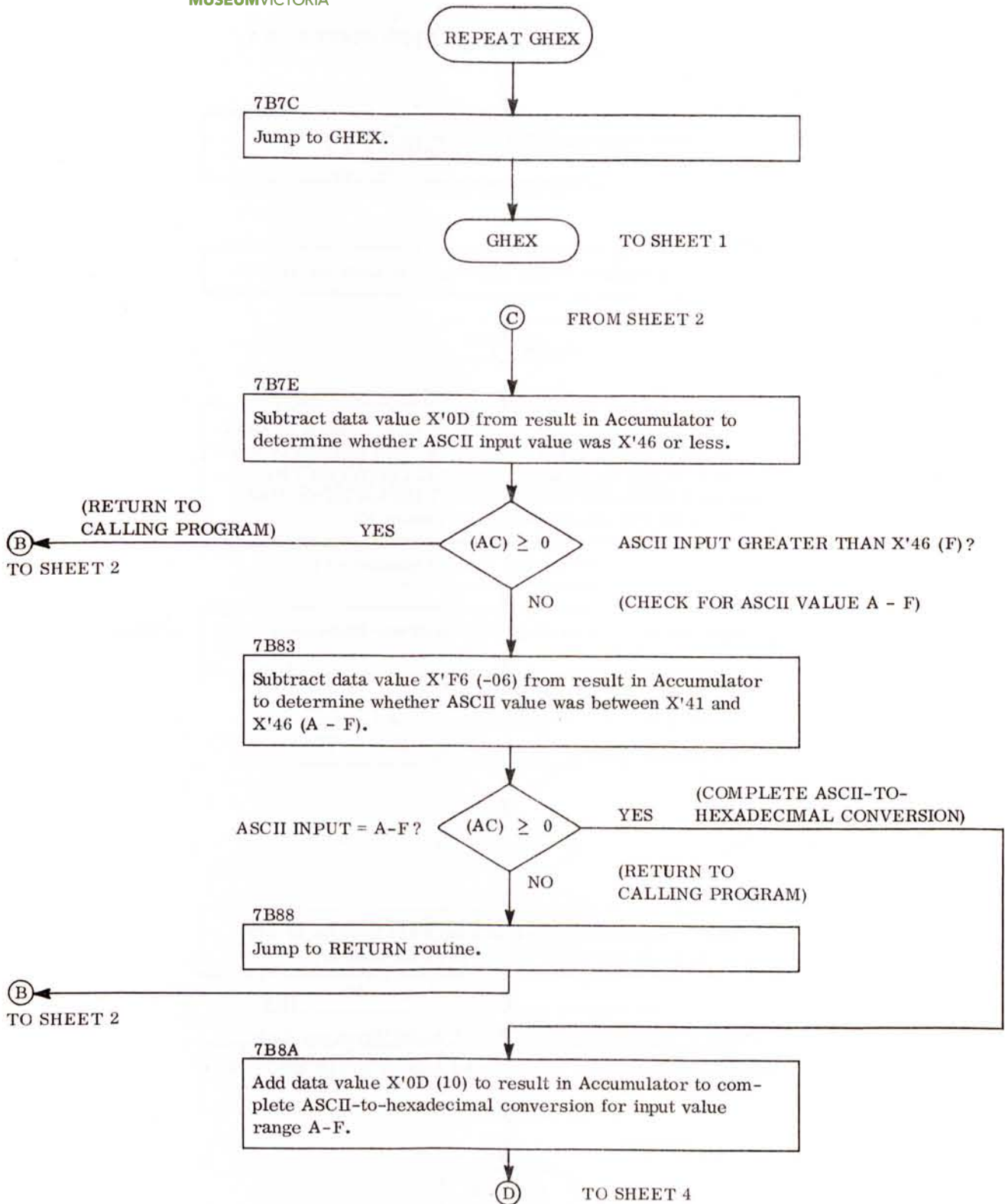
Figure 4-7. GHEX and GHEXE Subroutine – Detailed Flowchart (Sheet 1 of 7)





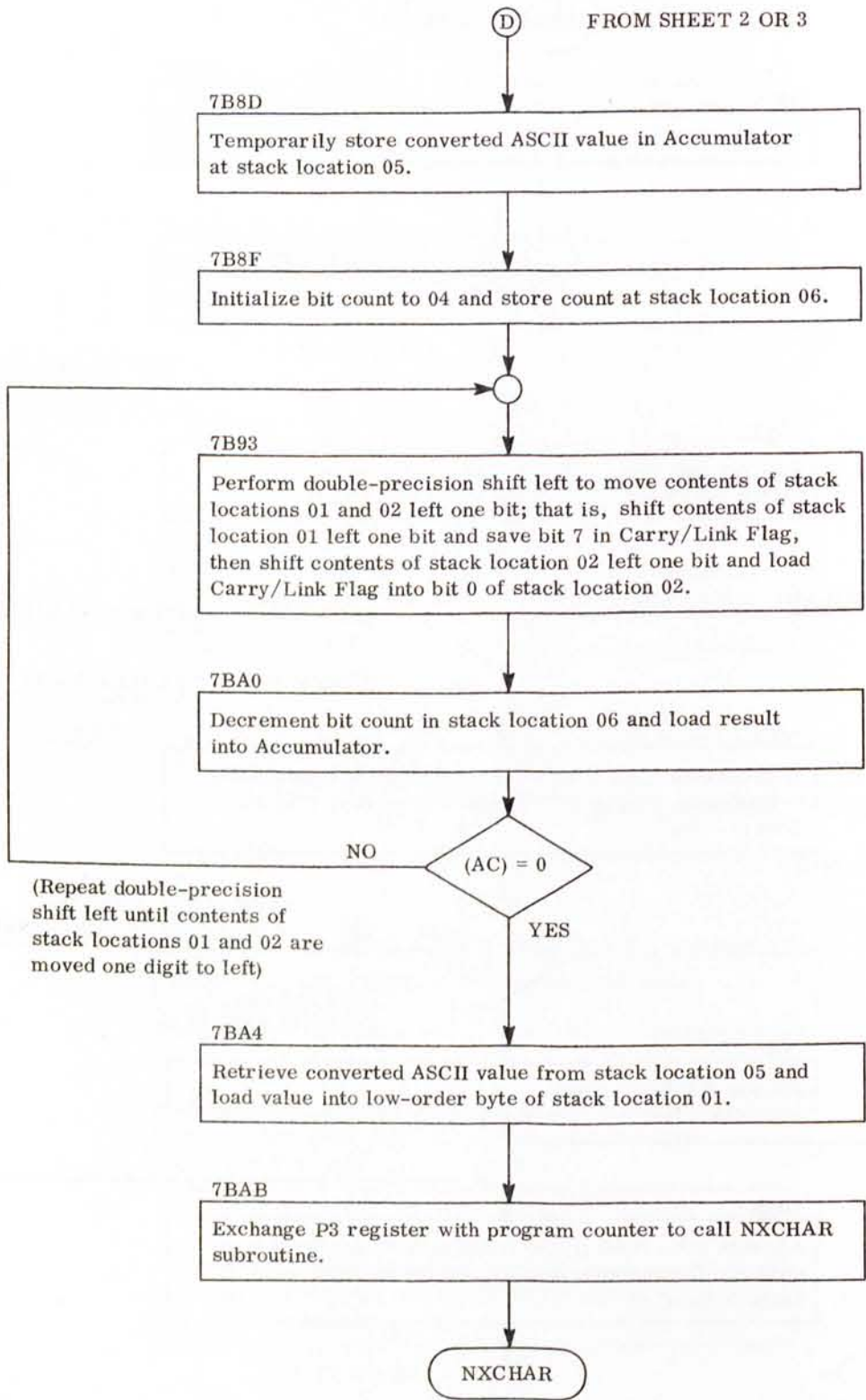
NS10670

Figure 4-7. GHEX and GHEXE Subroutine – Detailed Flowchart (Sheet 2 of 7)



NS10671

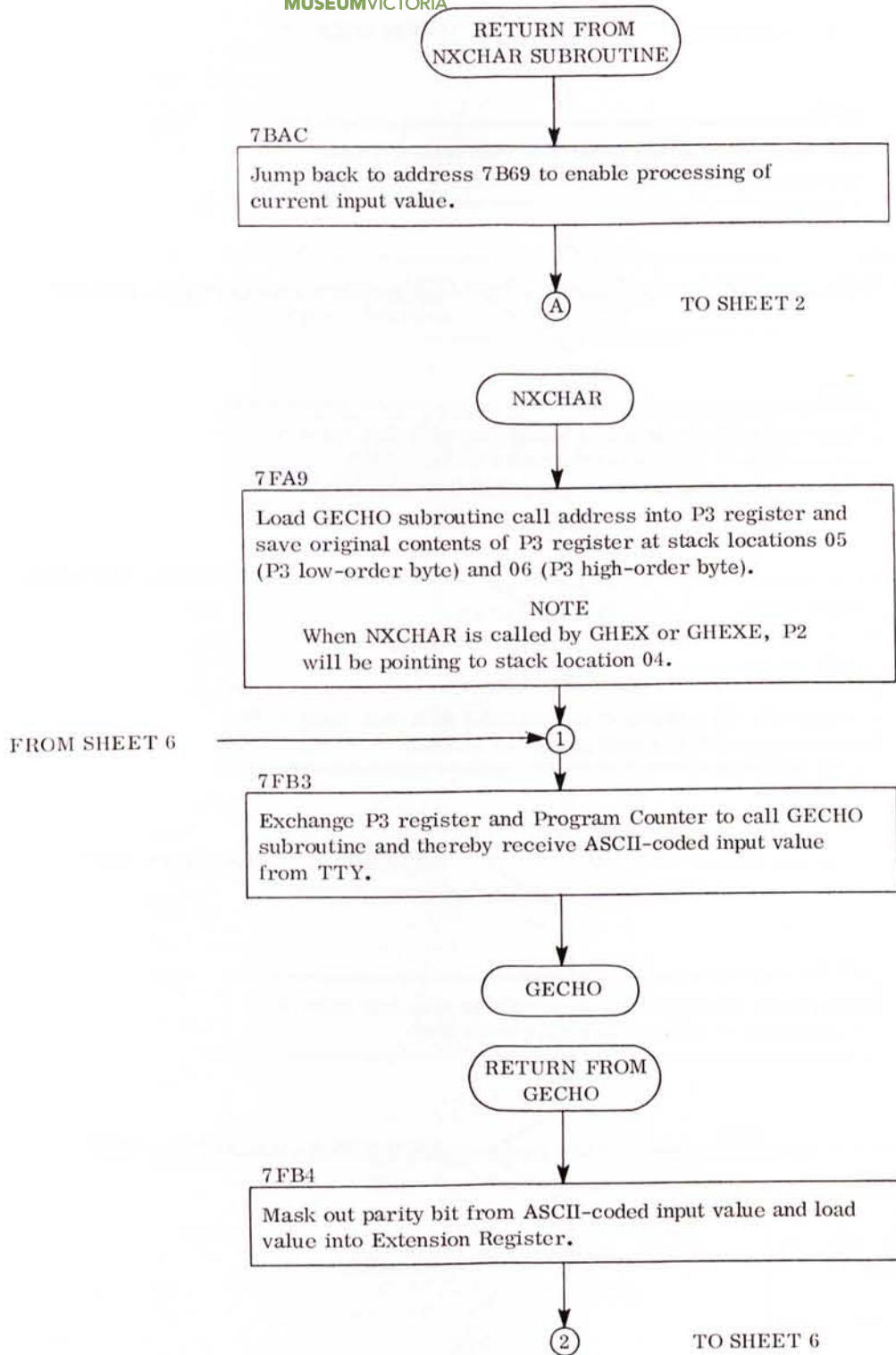
Figure 4-7. GHEX and GHEXE Subroutine – Detailed Flowchart (Sheet 3 of 7)



NS10672

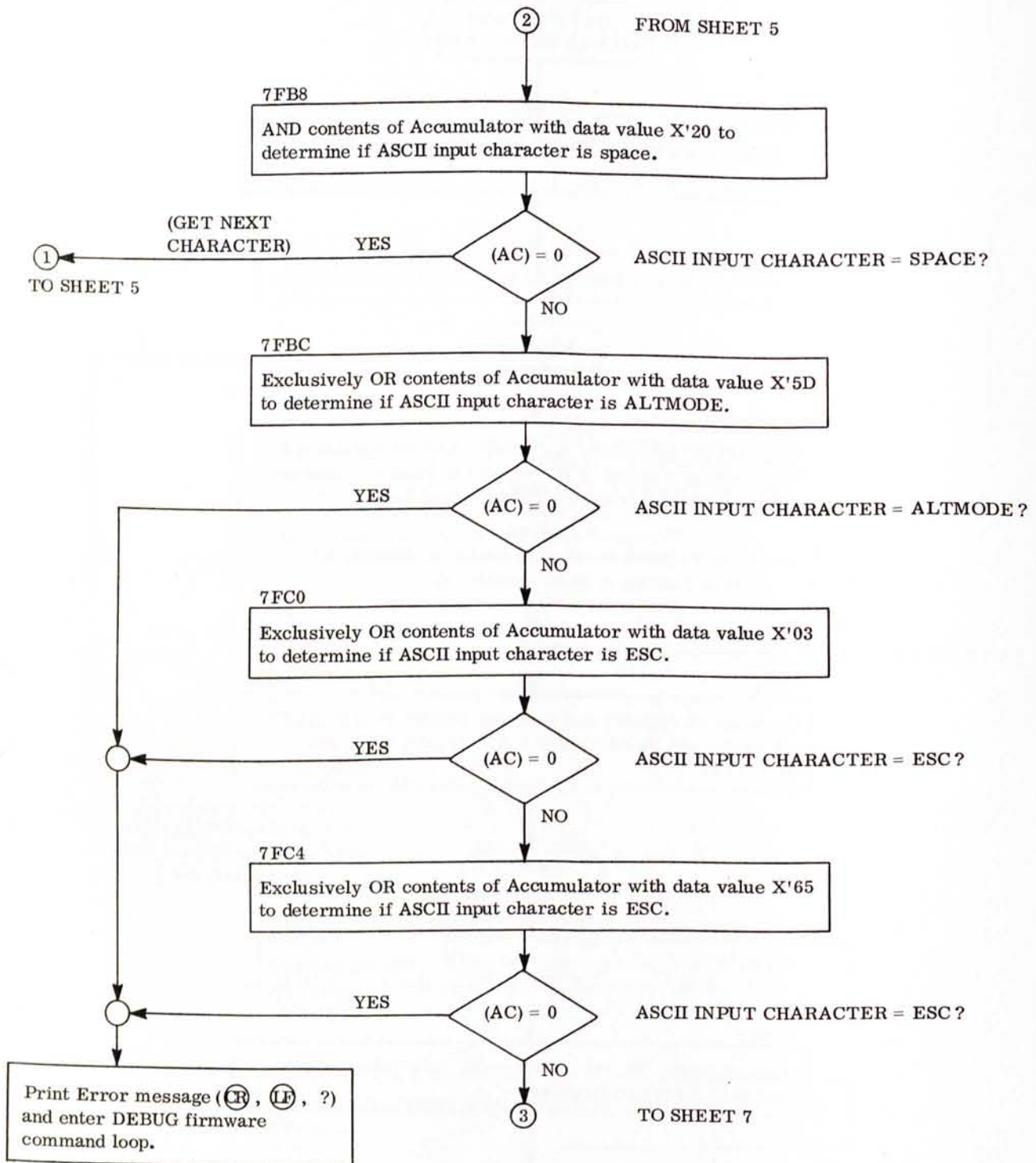
Figure 4-7. GHEX and GHEXE Subroutine — Detailed Flowchart (Sheet 4 of 7)





NS10673

Figure 4-7. GHGX and GHGXE Subroutine — Detailed Flowchart (Sheet 5 of 7)



NS10674

Figure 4-7. GHEx and GHExE Subroutine — Detailed Flowchart (Sheet 6 of 7)

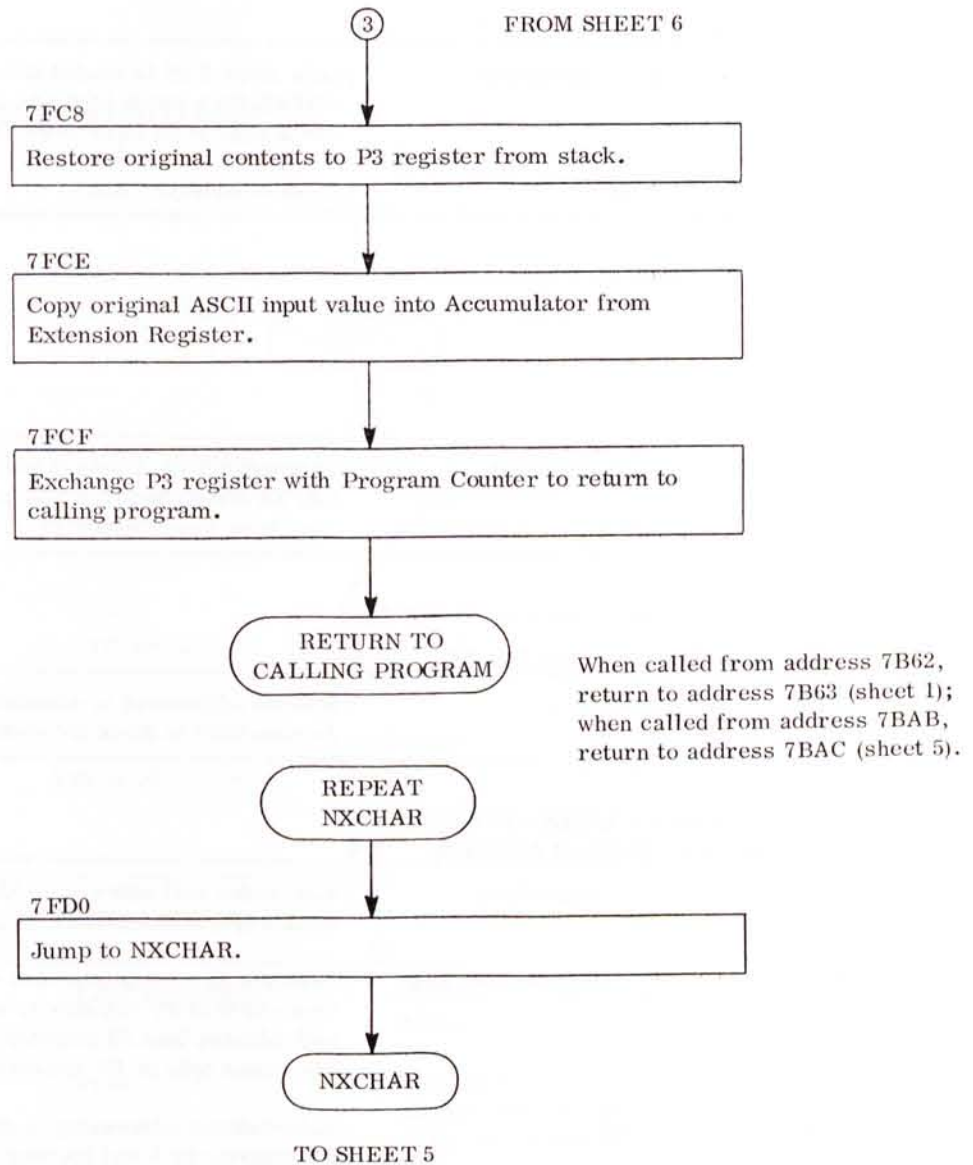


Figure 4-7. GHEX and GHEXE Subroutine – Detailed Flowchart (Sheet 7 of 7)

NS10675



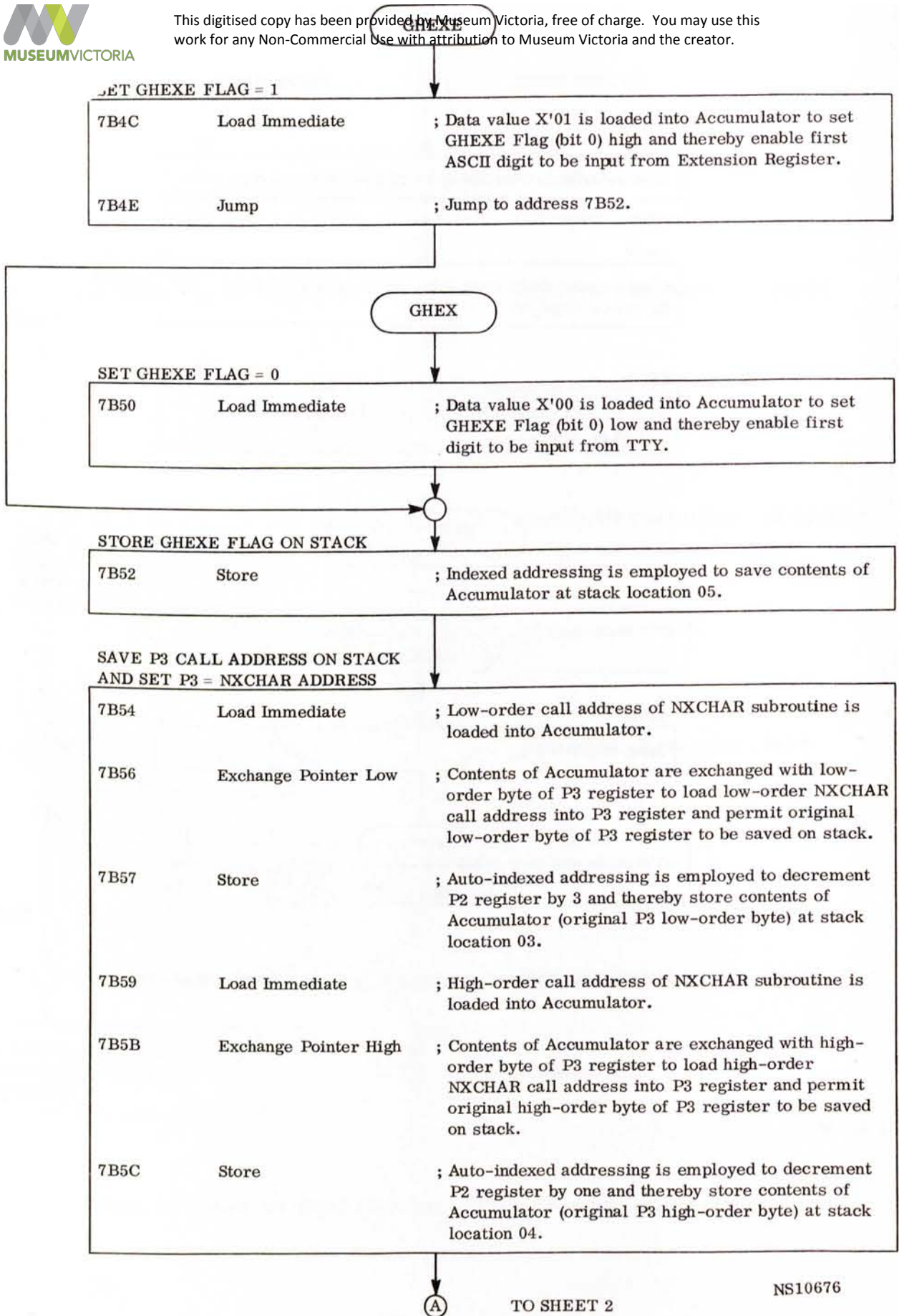
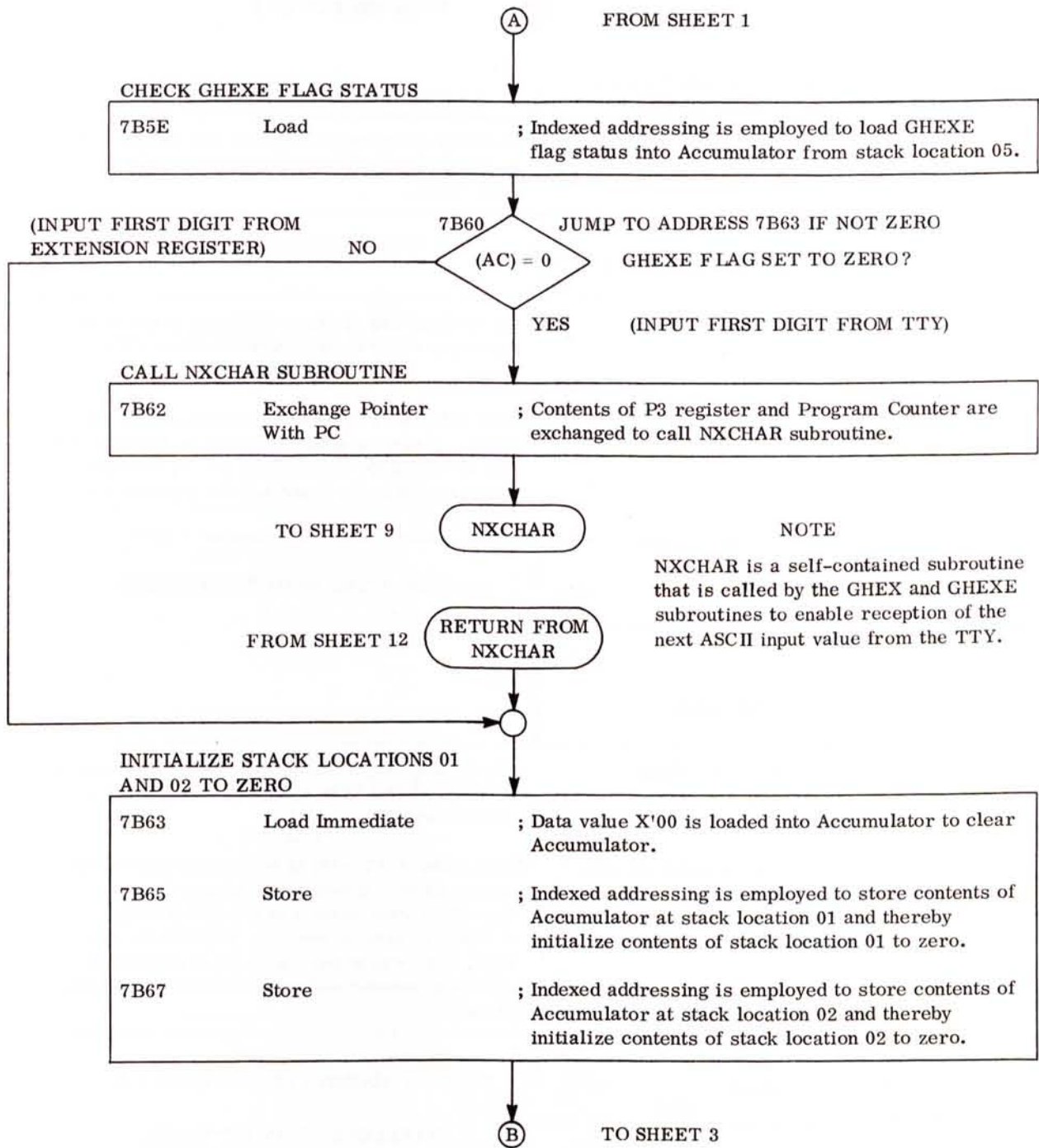
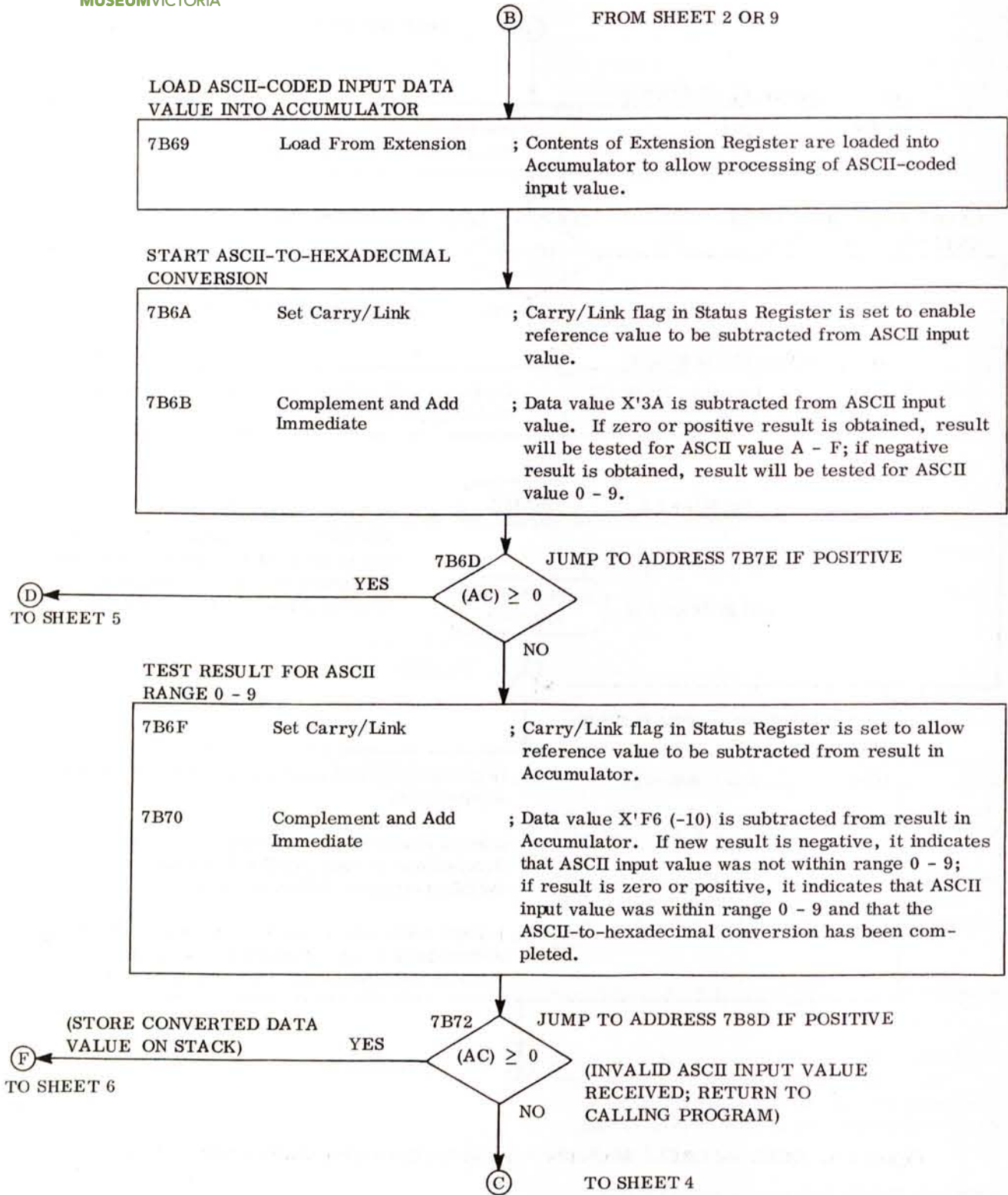


Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 1 of 12)



NS10677

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 2 of 12)

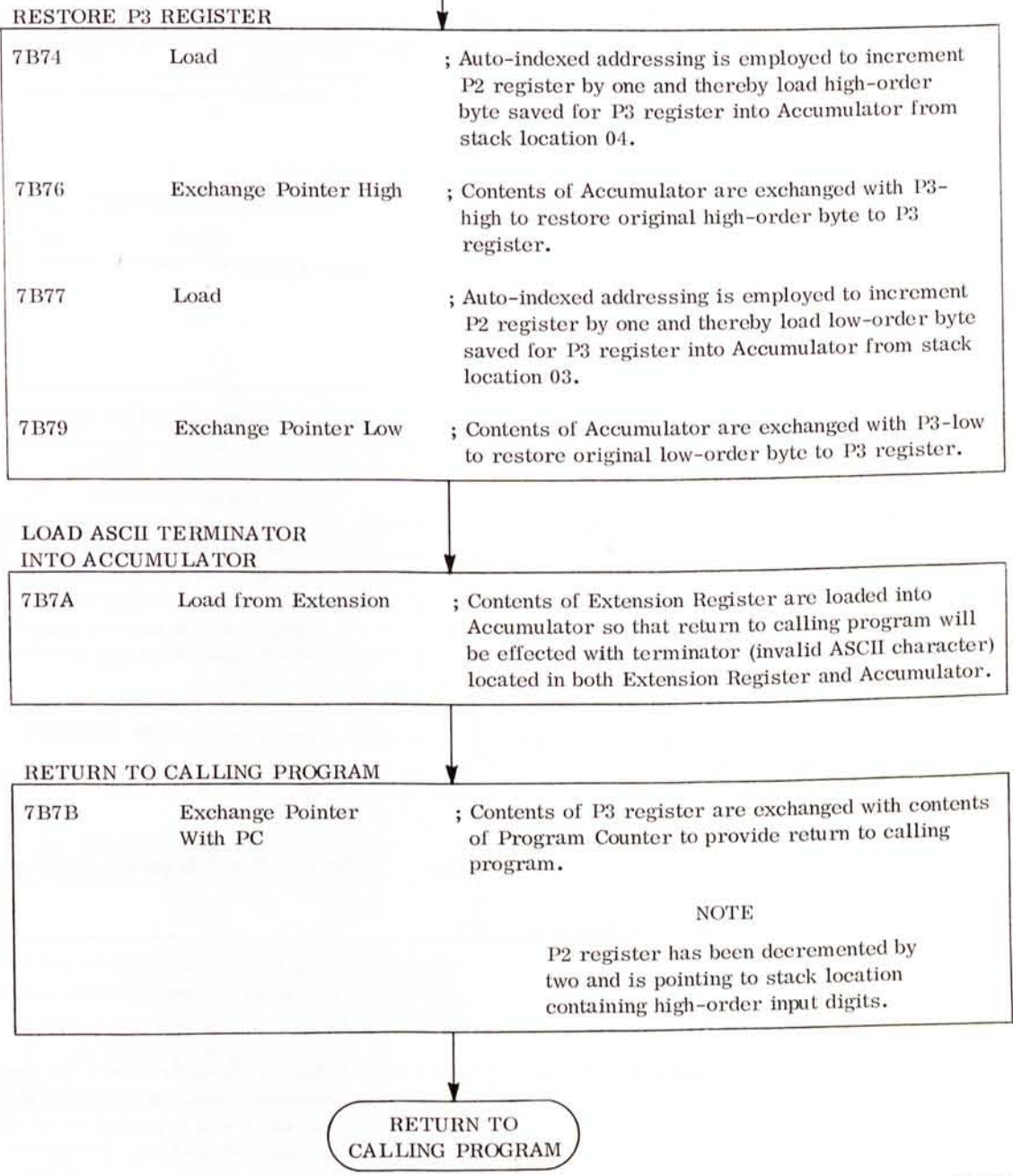


NS10678

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 3 of 12)

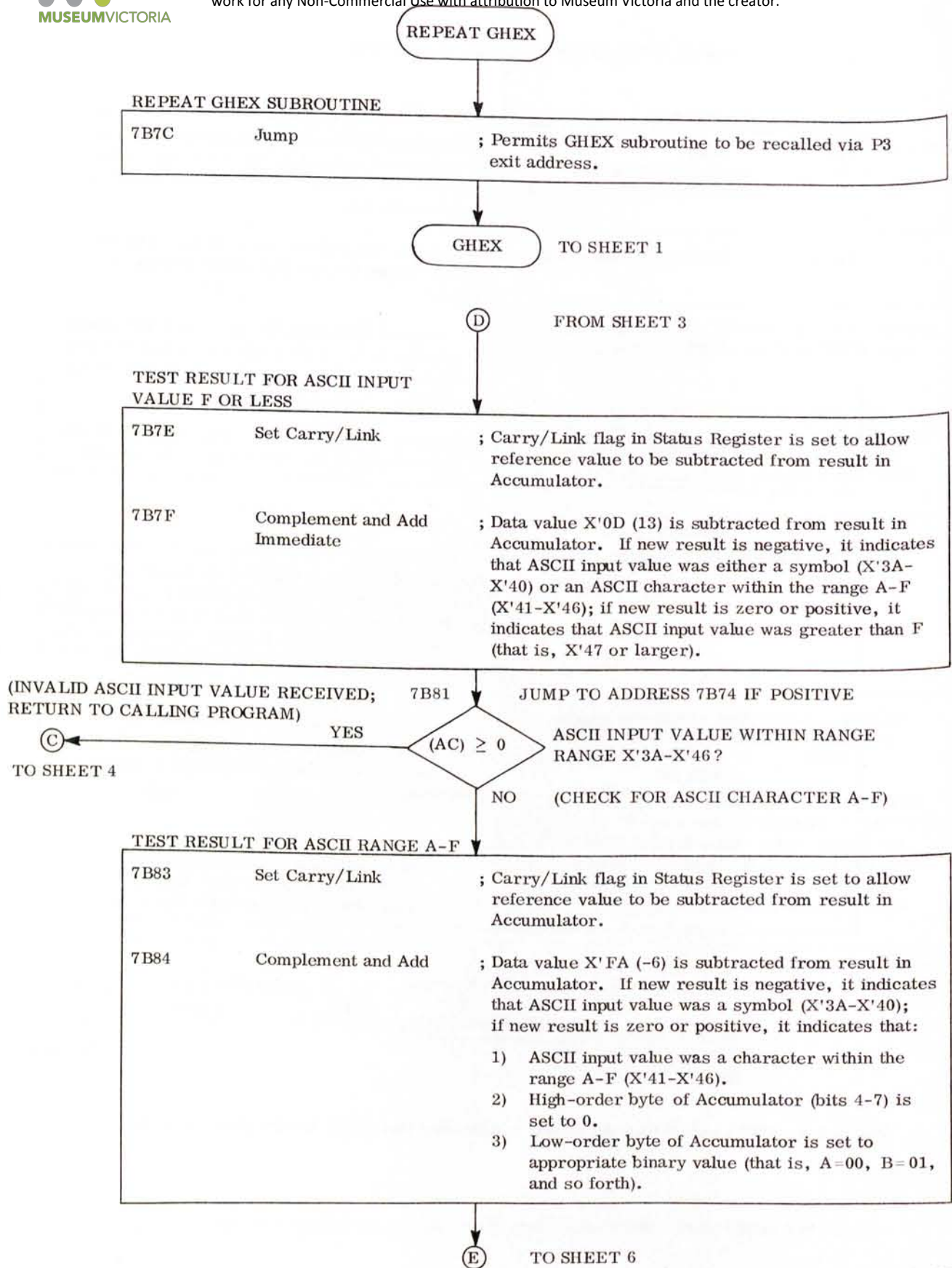


Ⓢ FROM SHEET 3 OR 5 OR 6



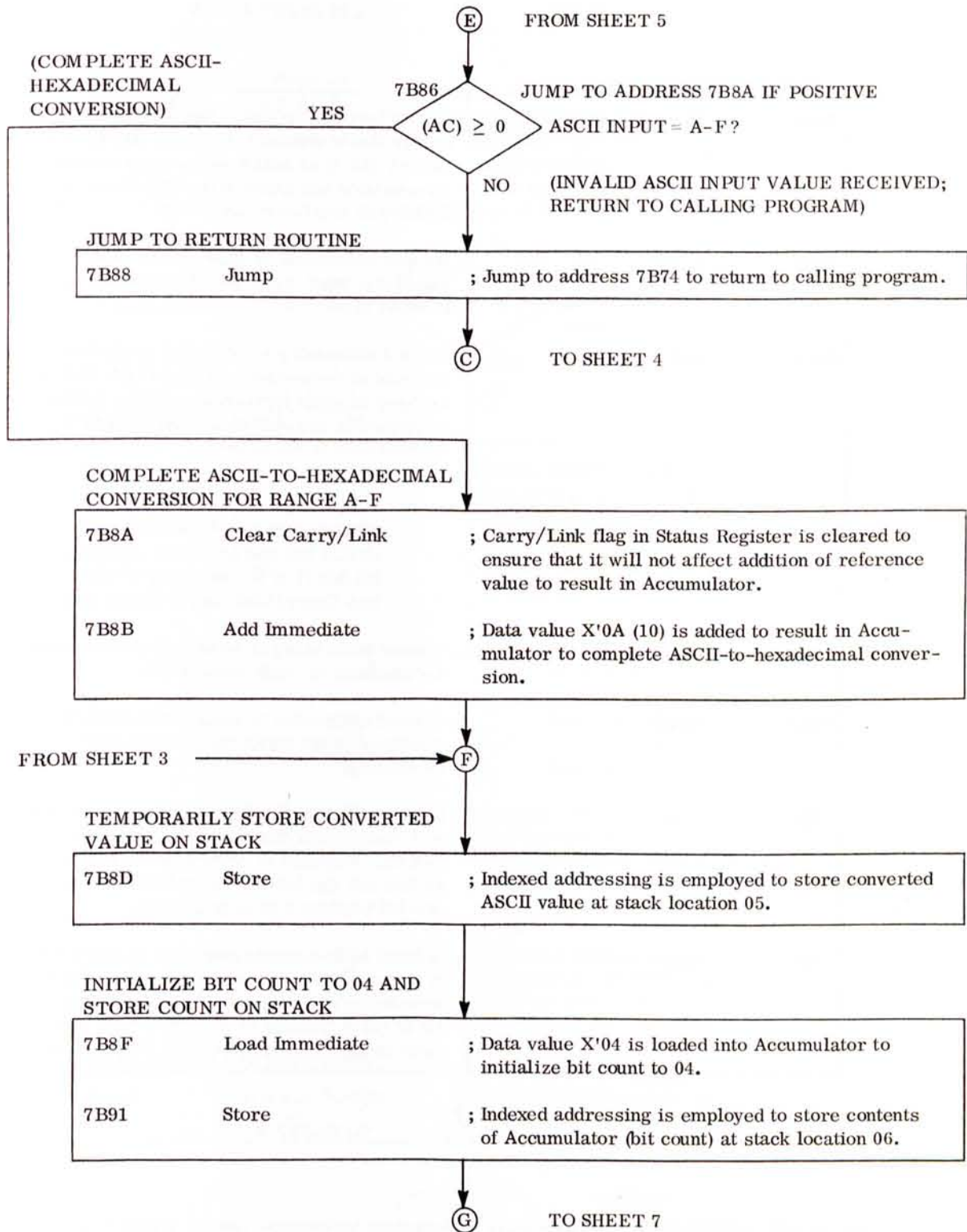
NS10679

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 4 of 12)



NS10680

Figure 4-8. GHEX and GHEXE Subroutine – Annotated Instruction Listing (Sheet 5 of 12)



NS10681

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 6 of 12)



ⓐ

FROM SHEET 6 OR 8

DOUBLE-PRECISION SHIFT LEFT

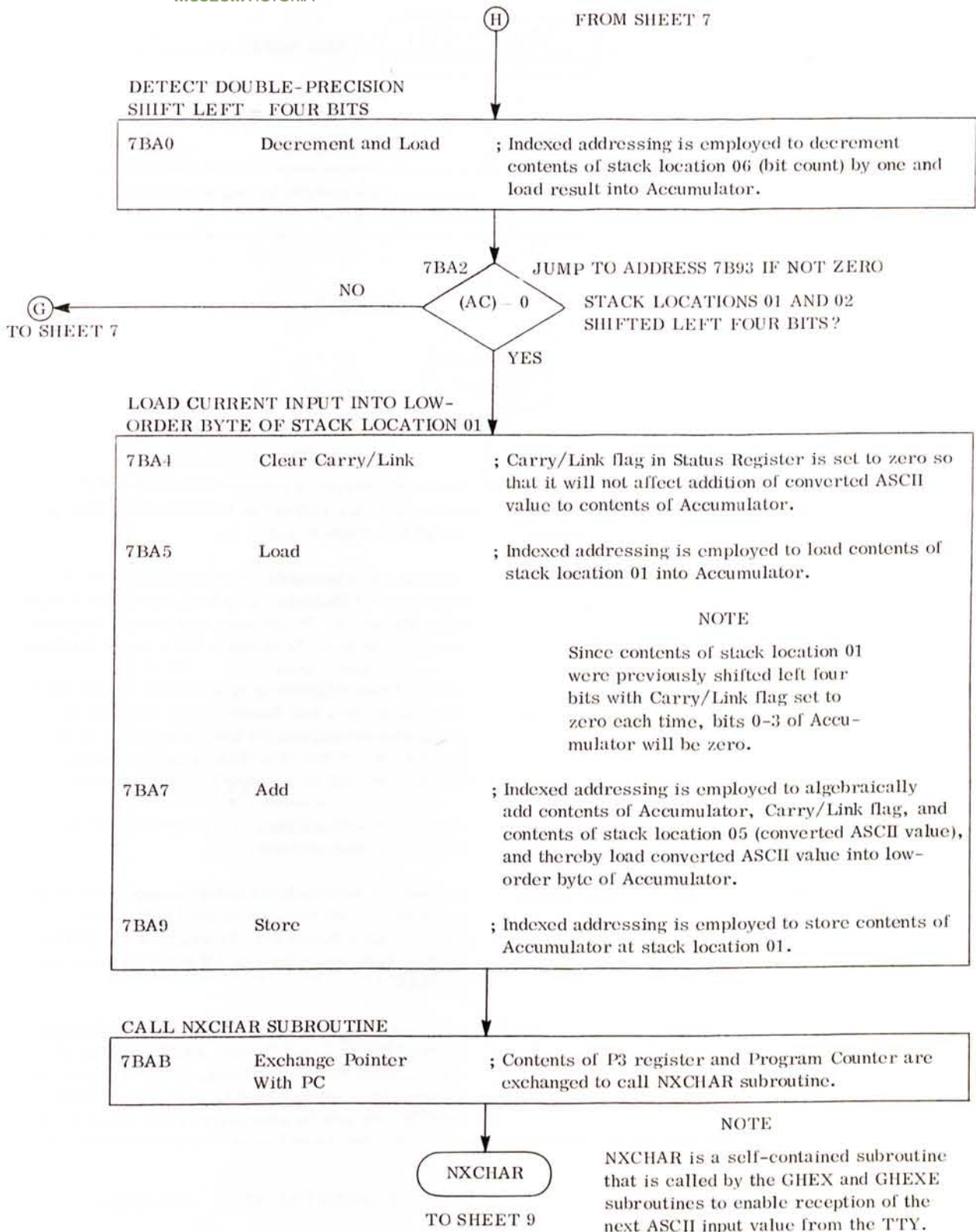
7B93	Clear Carry/Link	; Carry/Link flag in Status Register is cleared to ensure that a zero will be loaded into least significant bit (bit 0) of Accumulator when contents of Accumulator are added with contents of stack location 01 and Carry/Link flag.
7B94	Load	; Indexed addressing is employed to load two least-significant input digits into Accumulator from stack location 01.
7B96	Add	; Indexed addressing is employed to algebraically add contents of Accumulator, Carry/Link flag, and contents of stack location 01. Thus, contents of Accumulator are shifted left one bit and bit 0 of Accumulator is set to zero.
NOTE		
When contents of Accumulator are shifted left one bit, most significant bit (bit 7) of Accumulator is shifted into Carry/Link flag in Status Register.		
7B98	Store	; Indexed addressing is employed to store contents of Accumulator at stack location 01.
7B9A	Load	; Indexed addressing is employed to load two most-significant input digits into Accumulator from stack location 02.
7B9C	Add	; Contents of Accumulator are added algebraically with Carry/Link flag and contents of stack location 02. Contents of Accumulator, therefore, are shifted left one bit and Carry/Link flag is loaded into bit position 0 of Accumulator.
7B9E	Store	; Indexed addressing is employed to store contents of Accumulator at stack location 02. Thus, upon completion of this instruction, most significant bit of stack location 01 will have been moved to least significant bit position of stack location 02.

ⓑ

TO SHEET 8

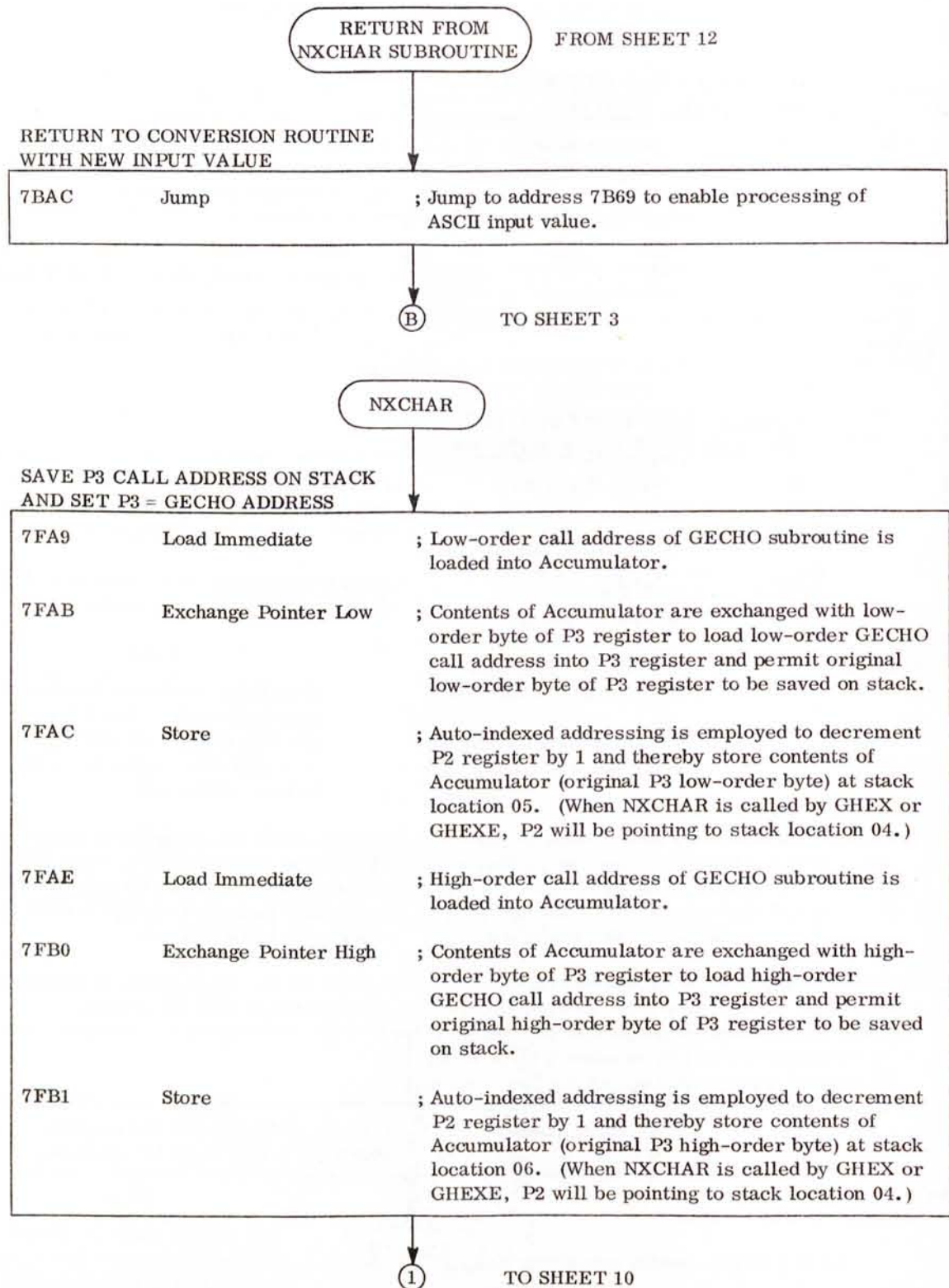
NS10682

Figure 4-8. GHEx and GHExE Subroutine — Annotated Instruction Listing (Sheet 7 of 12)



NS10683

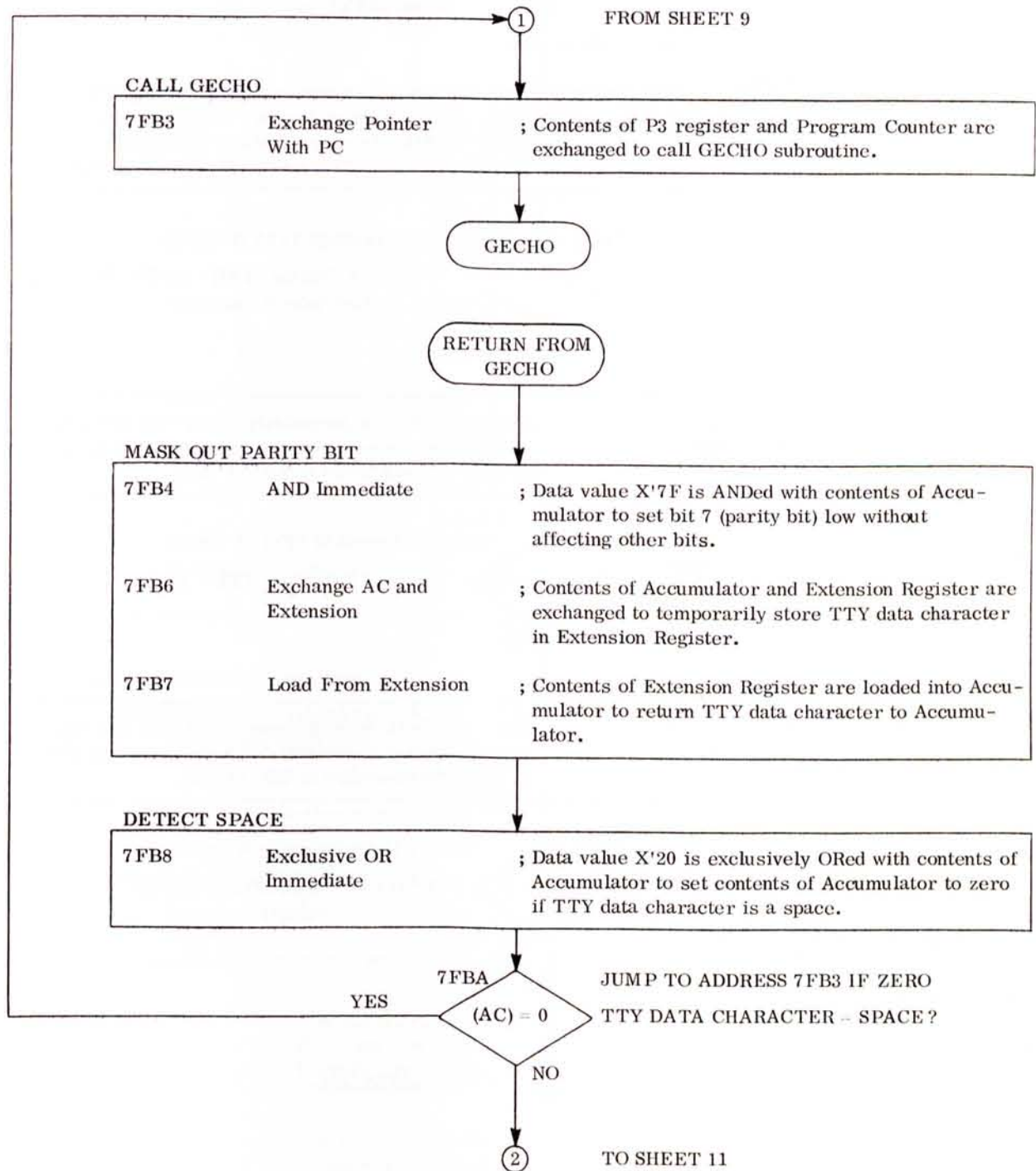
Figure 4-8. GHEX and GHEXE Subroutine - Annotated Instruction Listing (Sheet 8 of 12)



NS10684

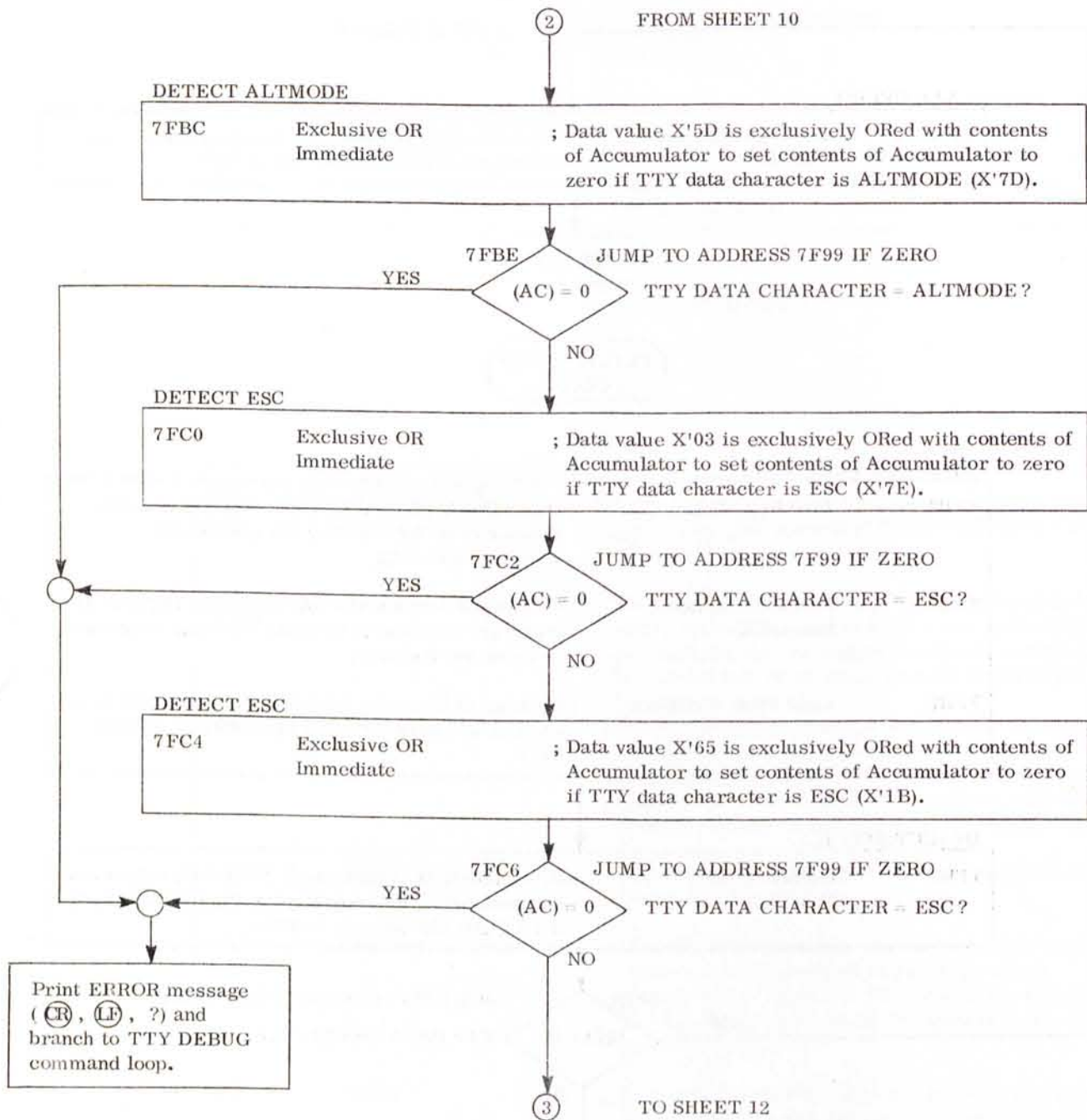
Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 9 of 12)





NS10685

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 10 of 12)



NS10686

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 11 of 12)

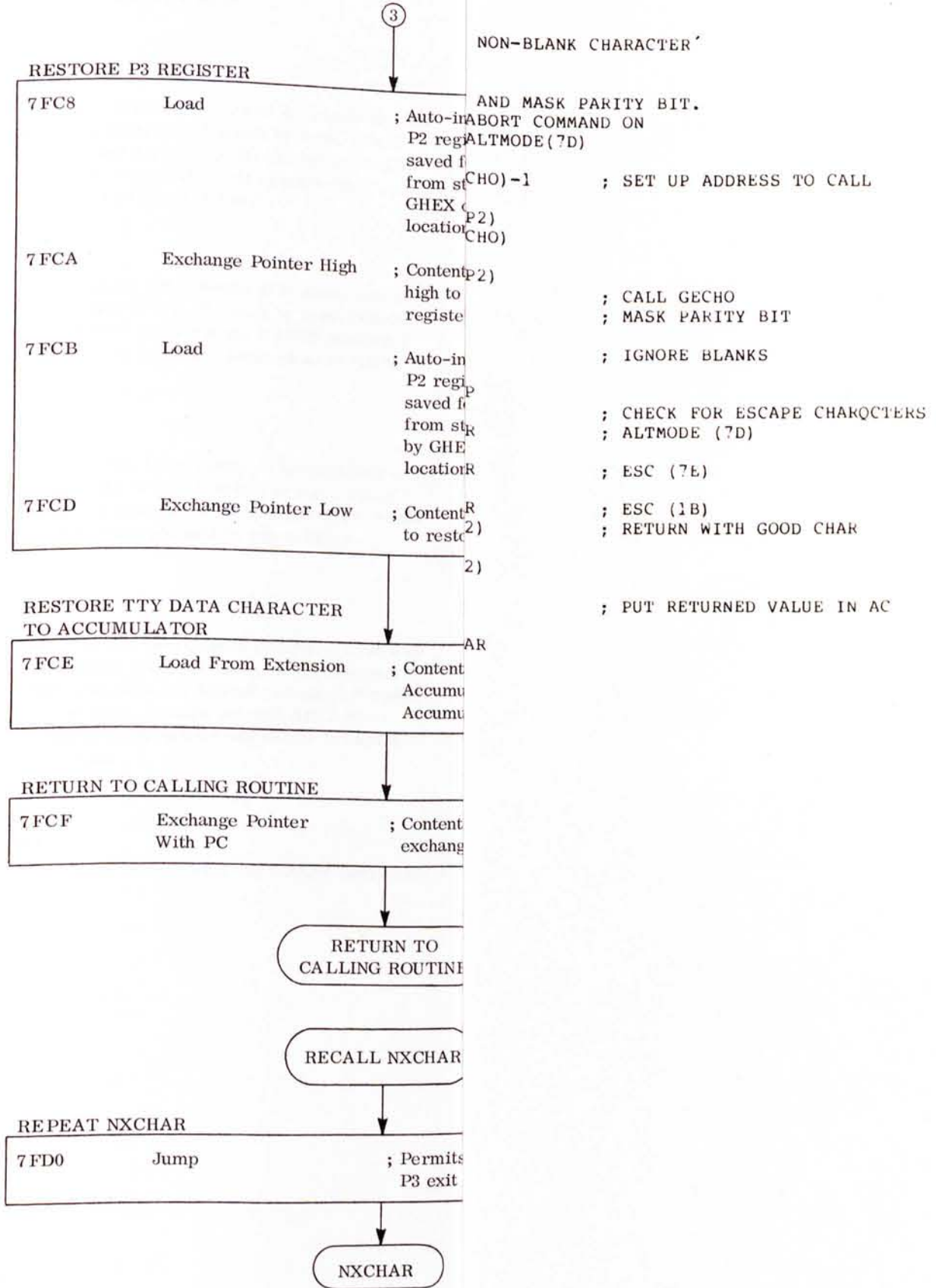
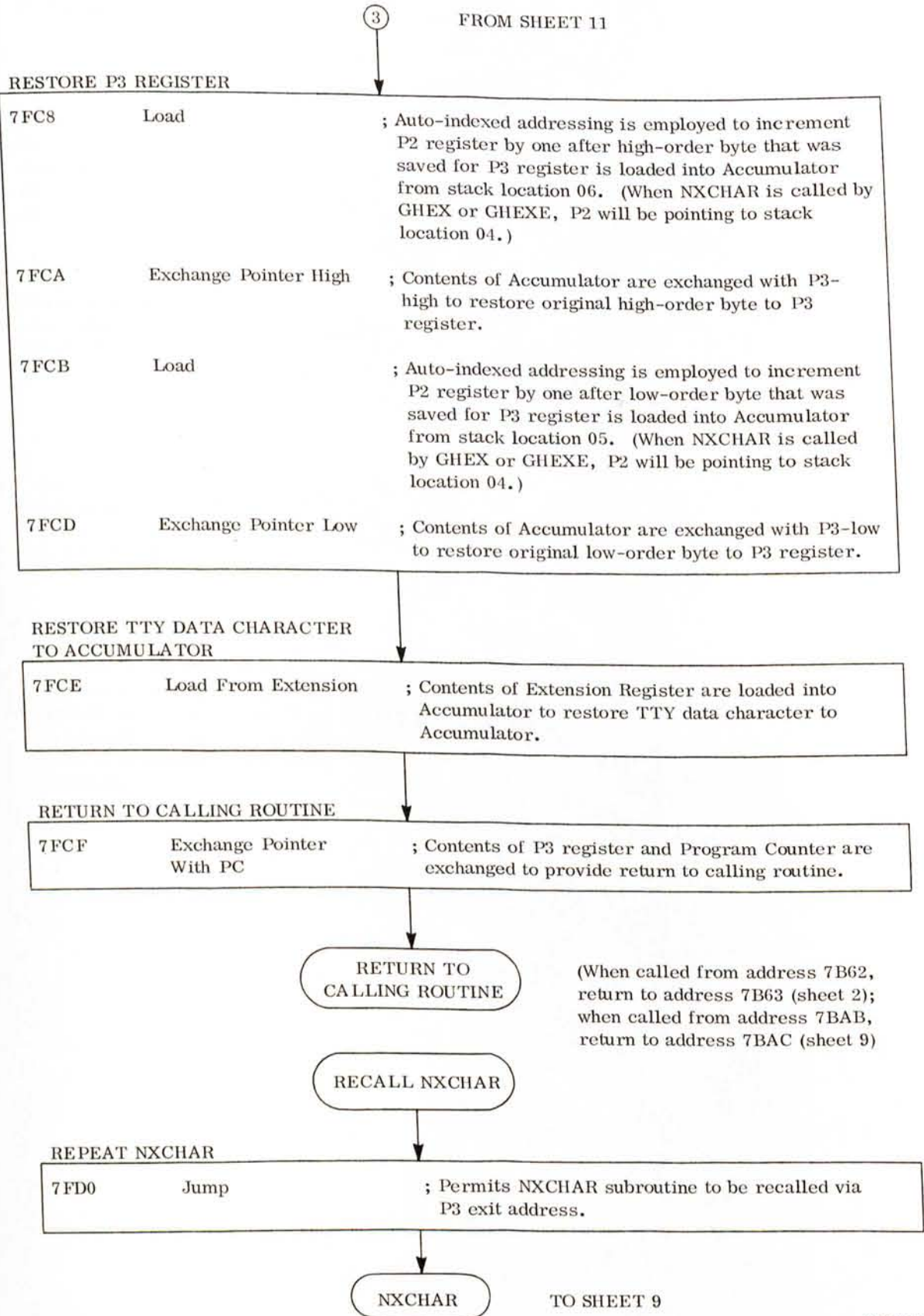


Figure 4-8. GHEX and GHEXE Subroutine — Annotated





NS10687

Figure 4-8. GHEX and GHEXE Subroutine — Annotated Instruction Listing (Sheet 12 of 12)

## GHEX and GHEXE Subroutine — Actual Listing

```

703          .PAGE      'HEX NUMBER INPUT'
704          .LOCAL
705          ;
706          ; GHEX GETS A 16-BIT VALUE AND PUSHES IT TO THE STACK.
707          ; GHEXE ASSUMES THE FIRST CHAR IS IN THE E REGISTER.
708          ; ONLY THE LAST 4 INPUT DIGITS ARE SAVED.
709          ;
710          ; RETURNS VALUE IN TOP 2 WORDS OF STACK AND TERMINATOR
711          ; IN THE AC AND EX REGISTERS.
712          ;
713 7B4C C401 GHEXE: LDI      1
714 7B4E 9002      JMP      $6
715 7B50 C400 GHEX:  LDI      0          ; RESET GHEXE FLAG
716 7B52 CAFB $6:   ST       -5(P2)
717 7B54 C4A8      LDI      L(NXCHAR)-1      ; SAVE RETURN ADDRESS AND SET UP
718 7B56 33        XPAL     P3          ; TO NXCHAR
719 7B57 CEFD      ST       @-3(P2)      ; STORE RETURN ADDRESS TO LEAVE ROOM
720 7B59 C47F      LDI      H(NXCHAR)      ; FOR RESULT
721 7B5B 37        XPAH     P3
722 7B5C CEFF      ST       @-1(P2)
723 7B5E C2FF      LD        -1(P2)
724 7B60 9C01      JNZ      $1
725 7B62 3F        XPPC     P3
726 7B63 C400 $1:   LDI      0          ; INITIALIZE RESULT TO 0
727 7B65 CA03      ST       3(P2)
728 7B67 CA02      ST       2(P2)
729 7B69 40       $LOOP: LDE
730 7B6A 03        SCL
731 7B6B FC3A      CAI      '9'+1      ; CHECK FOR 0-9
732 7B6D 940F      JP        $2          ; NOT 0-9, TOO LARGE
733 7B6F 03        SCL
734 7B70 FCF6      CAI      '0'-'9'-1      ; CHECK FOR 0-9
735 7B72 9419      JP        $3          ; IF POSITIVE, NUMBER IS
736              ; IN RANGE AND CONVERTED.
737 7B74 C601 $RET:  LD        @1(P2)      ; NUMBER IS NOT A HEX DIGIT,
738 7B76 37        XPAH     P3          ; RETURN
739 7B77 C601      LD        @1(P2)
740 7B79 33        XPAL     P3
741 7B7A 40       LDE
742 7B7B 3F        XPPC     P3
743 7B7C 90D2      JMP      GHEX
744 7B7E 03       $2:   SCL
745 7B7F FC0D      CAI      'F'+1-'9'-1      ; CHECK FOR DIGITS A-F.
746 7B81 94F1      JP        $RET        ; NUMBER TOO LARGE
747 7B83 03        SCL
748 7B84 FCFA      CAI      'A'-'F'-1
749 7B86 9402      JP        $4          ; DIGIT BETWEEN A&F
750 7B88 90EA      JMP      $RET
751 7B8A 02       $4:   CCL
752 7B8B F40A      ADI      10          ; ADJUST DIGIT VALUE FOR 10-16
753 7B8D CAFF      $3:   ST       -1(P2)      ; SAVE ADJUSTED DIGIT
754 7B8F C404      LDI      4          ; SET UP BIT COUNTER FOR
755 7B91 CAFE      ST       -2(P2)      ; SHIFT.
756 7B93 02       $5:   CCL          ; SHIFT HEX DIGIT LEFT ONE
757 7B94 C203      LD        3(P2)      ; DIGIT, ONE BIT EACH
758 7B96 F203      ADD     3(P2)      ; TIME THROUGH LOOP.
759 7B98 CA03      ST       3(P2)
760 7B9A C202      LD        2(P2)
761 7B9C F202      ADD     2(P2)
762 7B9E CA02      ST       2(P2)
763 7BA0 BAFE      DLD     -2(P2)
764 7BA2 9CEF      JNZ     $5
765 7BA4 02        CCL
766 7BA5 C203      LD        3(P2)      ; ADD CURRENT DIGIT INTO
767 7BA7 F2FF      ADD     -1(P2)      ; NUMBER
768 7BA9 CA03      ST       3(P2)
769 7BAB 3F        XPPC     P3          ; GET NEXT CHAR
770 7BAC 90BB      JMP     $LOOP        ; AND LOOP

```



### GHEX and GHEXE Subroutine — Actual Listing

```

703          .PAGE   'HEX NUMBER INPUT'
704          .LOCAL
705          ;
706          ; GHEX GETS A 16-BIT VALUE AND PUSHES IT TO THE STACK.
707          ; GHEXE ASSUMES THE FIRST CHAR IS IN THE E REGISTER.
708          ; ONLY THE LAST 4 INPUT DIGITS ARE SAVED.
709          ;
710          ; RETURNS VALUE IN TOP 2 WORDS OF STACK AND TERMINATOR
711          ; IN THE AC AND EX REGISTERS.
712          ;
713 7B4C C401 GHEXE: LDI      1
714 7B4E 9002      JMP      $6
715 7B50 C400 GHEX:  LDI      0          ; RESET GHEXE FLAG
716 7B52 CAFB $6:   ST        -5(P2)
717 7B54 C4A8      LDI      L(NXCHAR)-1      ; SAVE RETURN ADDRESS AND SET UP
718 7B56 33        XPAL     P3          ; TO NXCHAR
719 7B57 CEFD      ST        @-3(P2)      ; STORE RETURN ADDRESS TO LEAVE ROOM
720 7B59 C47F      LDI      H(NXCHAR)      ; FOR RESULT
721 7B5B 37        XPAH     P3
722 7B5C CEFF      ST        @-1(P2)
723 7B5E C2FF      LD        -1(P2)
724 7B60 9C01      JNZ      $1
725 7B62 3F        XPPC     P3
726 7B63 C400 $1:   LDI      0          ; INITIALIZE RESULT TO 0
727 7B65 CA03      ST        3(P2)
728 7B67 CA02      ST        2(P2)
729 7B69 40        $LOOP: LDE
730 7B6A 03        SCL
731 7B6B FC3A      CAI      '9'+1      ; CHECK FOR 0-9
732 7B6D 940F      JP        $2          ; NOT 0-9, TOO LARGE
733 7B6F 03        SCL
734 7B70 FCF6      CAI      '0'-'9'-1      ; CHECK FOR 0-9
735 7B72 9419      JP        $3          ; IF POSITIVE, NUMBER IS
736                          ; IN RANGE AND CONVERTED.
737 7B74 C601 $RET:  LD        @1(P2)      ; NUMBER IS NOT A HEX DIGIT,
738 7B76 37        XPAH     P3          ; RETURN
739 7B77 C601      LD        @1(P2)
740 7B79 33        XPAL     P3
741 7B7A 40        LDE
742 7B7B 3F        XPPC     P3
743 7B7C 90D2      JMP      GHEX
744 7B7E 03        $2:   SCL
745 7B7F FC0D      CAI      'F'+1-'9'-1      ; CHECK FOR DIGITS A-F.
746 7B81 94F1      JP        $RET          ; NUMBER TOO LARGE
747 7B83 03        SCL
748 7B84 FCFA      CAI      'A'-'F'-1      ; DIGIT BETWEEN A&F
749 7B86 9402      JP        $4
750 7B88 90EA      JMP      $RET
751 7B8A 02        $4:   CCL
752 7B8B F40A      ADI      10          ; ADJUST DIGIT VALUE FOR 10-16
753 7B8D CAFF      $3:   ST        -1(P2)      ; SAVE ADJUSTED DIGIT
754 7B8F C404      LDI      4          ; SET UP BIT COUNTER FOR
755 7B91 CAFE      ST        -2(P2)      ; SHIFT.
756 7B93 02        $5:   CCL          ; SHIFT HEX DIGIT LEFT ONE
757 7B94 C203      LD        3(P2)      ; DIGIT, ONE BIT EACH
758 7B96 F203      ADD     3(P2)      ; TIME THROUGH LOOP.
759 7B98 CA03      ST        3(P2)
760 7B9A C202      LD        2(P2)
761 7B9C F202      ADD     2(P2)
762 7B9E CA02      ST        2(P2)
763 7BA0 BAFE      DLD     -2(P2)
764 7BA2 9CEF      JNZ      $5
765 7BA4 02        CCL
766 7BA5 C203      LD        3(P2)      ; ADD CURRENT DIGIT INTO
767 7BA7 F2FF      ADD     -1(P2)      ; NUMBER
768 7BA9 CA03      ST        3(P2)
769 7BAB 3F        XPPC     P3          ; GET NEXT CHAR
770 7BAC 90BB      JMP      $LOOP      ; AND LOOP

```



```

626          .PAGE      'GET NON-BLANK CHARACTER'
627          .LOCAL
628          ;
629          ; GET NEXT CHARACTER AND MASK PARITY BIT.
630          ; IGNORE BLANKS AND ABORT COMMAND ON
631          ; ESC (7E OR 1B) OR ALTMODE(7D)
632          ;
633 7FA9 C490 NXCHAR: LDI      L(GECHO)-1      ; SET UP ADDRESS TO CALL
634 7FAB 33      XPAL      P3
635 7FAC CEFF      ST        @-1(P2)
636 7FAE C47A      LDI      H(GECHO)
637 7FB0 37      XPAH      P3
638 7FB1 CEFF      ST        @-1(P2)
639 7FB3 3F      $LOOP: XPPC      P3          ; CALL GECHO
640 7FB4 D47F      ANI      07F          ; MASK PARITY BIT
641 7FB6 01      XAE
642 7FB7 40      LDE          ; IGNORE BLANKS
643 7FB8 E420      XRI      020
644 7FBA 98F7      JZ        $LOOP
645 7FBC E45D      XRI      05D          ; CHECK FOR ESCAPE CHARQCTERS
646 7FBE 98D9      JZ        ERROR      ; ALTMODE (7D)
647 7FC0 E403      XRI      03
648 7FC2 98D5      JZ        ERROR      ; ESC (7E)
649 7FC4 E465      XRI      065
650 7FC6 98D1      JZ        ERROR      ; ESC (1B)
651 7FC8 C601      LD        @1(P2)      ; RETURN WITH GOOD CHAR
652 7FCA 37      XPAH      P3
653 7FCB C601      LD        @1(P2)
654 7FCD 33      XPAL      P3
655 7FCE 40      LDE          ; PUT RETURNED VALUE IN AC
656 7FCF 3F      XPPC      P3
657 7FD0 90D7      JMP        NXCHAR

```



- A. AND contents of Accumulator with data value X'7F to mask parity bit.

AC	0XXX	XXXX	(X indicates don't care)
Data	<u>0111 1111</u>		
Result	0XXX	XXXX	

- B. Exclusively OR contents of Accumulator with data value X'20. If result is zero, detect space and repeat GECHO subroutine to get next character; if result is not zero, proceed as shown below.

	SPACE	ALTMODE	ESC	ESC
AC	0010 0000	0111 1101	0111 1110	0001 1011
Data	0010 0000	0010 0000	0010 0000	0010 0000
Result	0000 0000	0101 1101	0101 1110	0011 1011

- C. Exclusively OR contents of Accumulator with data value X'5D. If result is zero, detect ALTMODE and print out ERROR message; if result is not zero, proceed as shown below.

	ALTMODE	ESC	ESC
AC	0101 1101	0101 1110	0011 1011
Data	0101 1101	0101 1101	0101 1101
Result	0000 0000	0000 0011	0110 0110

- D. Exclusively OR contents of Accumulator with data value X'03. If result is zero, detect ESC and print out ERROR message; if result is not zero, proceed as shown below.

	ESC	ESC
AC	0000 0011	0110 0110
Data	0000 0011	0000 0011
Result	0000 0000	0110 0101

- E. Exclusively OR contents of Accumulator with data value X'65. If result is zero, detect ESC and print out ERROR message; if result is not zero, restore original ASCII input value to Accumulator and return to calling program.

	ESC
AC	0110 0101
Data	0110 0101
Result	0000 0000

NS10688

Figure 4-9. GHEX and GHEXE Subroutine — Detection of Space, ALTMODE, and ESC



A. Set Carry/Link (C/L) flag in Status Register to enable reference value to be subtracted from ASCII input in Accumulator.

B. Subtract data value X'3A (58) from ASCII input in Accumulator via Complement-And-Add Instruction. If result is negative, proceed to C below to test result for ASCII values 0-9; if result is zero or positive, proceed to D below to test that result indicates ASCII value F or less.

	/ (X'2F)	0 (X'30)	9 (X'39)	: X'(3A)
AC	0010 1111	0011 0000	0011 1001	0011 1010
Data	1100 0101	1100 0101	1100 0101	1100 0101
C/L	1	1	1	1
Result	1111 0101	1111 0110	1111 1111	0000 0000
C/L	0	0	0	1

	@ (X'40)	A (X'41)	F (X'46)	G (X'47)
AC	0100 0000	0100 0001	0100 0110	0100 0111
Data	1100 0101	1100 0101	1100 0101	1100 0101
C/L	1	1	1	1
Result	0000 0110	0000 0111	0000 1100	0000 1101
C/L	1	1	1	1

C. Set Carry/Link flag in Status Register, then, subtract data value X'F6 (-10) from contents of Accumulator via Complement-And-Add Instruction. If result is negative (indicating that ASCII input was X'2F or less), detect invalid ASCII input and terminate ASCII-to-hexadecimal conversion; if result is zero or positive, ASCII character was within range 0-9 and conversion is completed.

	/ (X'2F)	0 (X'30)	9 (X'39)
AC	1111 0101	1111 0110	1111 1111
Data	0000 1001	0000 1001	0000 1001
C/L	1	1	1
Result	1111 1111	0000 0000	0000 1001
C/L	0	1	1

D. Set Carry/Link flag in Status Register, then subtract data value X'0D (13) from contents of Accumulator via Complement-And-Add Instruction. If result is zero or positive (indicating that ASCII value was X'47 or greater), detect invalid ASCII input and terminate ASCII-to-hexadecimal conversion; if result is negative, proceed to E (sheet 2) to test that result indicates ASCII value A or greater.

	: (X'3A)	@ (X'40)	A (X'41)
AC	0000 0000	0000 0110	0000 0111
Data	1111 0010	1111 0010	1111 0010
C/L	1	1	1
Result	1111 0011	1111 1001	1111 1010
C/L	0	0	0

	F (X'46)	G (X'47)
AC	0000 1100	0000 1101
Data	1111 0010	1111 0010
C/L	1	1
Result	1111 1111	0000 0000
C/L	0	1

NS10689

Figure 4-10. ASCII-to-Hexadecimal Conversion Routine for GHEX and GHEXE Subroutines (Sheet 1 of 2)

- E. Set Carry/Link flag in Status Register; then, subtract data value X'FA (-6) from contents of Accumulator via Complement-And-Add Instruction. If result is negative (indicating that ASCII value was X'40 or less), detect invalid ASCII input and terminate ASCII-to-hexadecimal conversion; if result is zero or positive (indicating that ASCII value is within range for A-F), proceed to F below to complete ASCII-to-hexadecimal conversion.

	: (X'3A)	@ (X'40)	A (X'41)	F (X'46)
AC	1111 0011	1111 1001	1111 1010	1111 1111
Data	0000 0101	0000 0101	0000 0101	0000 0101
C/L	1	1	1	1
Result	1111 1001	1111 1111	0000 0000	0000 0101
C/L	0	0	1	1

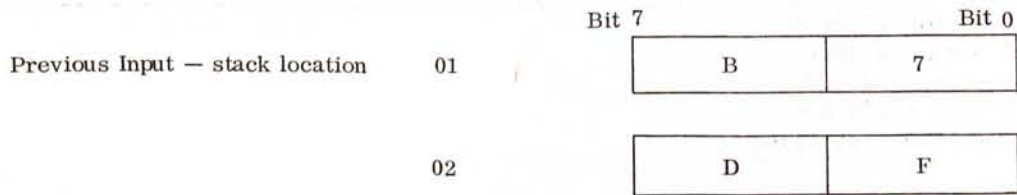
- F. Clear Carry/Link flag in Status Register; then, add data value X'0A (10) to contents of Accumulator to complete ASCII-to-hexadecimal conversion.

	A (X'41)	F (X'46)
AC	0000 0000	0000 0101
Data	0000 1010	0000 1010
C/L	0	0
Result	0000 1010	0000 1111
C/L	0	0

NS10690

Figure 4-10. ASCII-to-Hexadecimal Conversion Routine for GHEX and GHEXE Subroutines (Sheet 2 of 2)

The GHEX and GHEXE subroutines store each valid input at the least-significant-digit stack location after shifting the previous input one digit to the left. To illustrate execution of the 4-digit save routine, the values DFB7 are arbitrarily assumed to have been previously received and stored on the stack as shown below, and the current input value is arbitrarily assumed to be F.



- A. Temporarily store current input (X'0F) at stack location 05.
- B. Initialize Bit Count to 04 and store bit count at stack location 06.

C. Clear Carry/Link flag in Status Register, load contents of stack location 01 into Accumulator, and then add contents of stack location 01 with contents of Accumulator and Carry/Link flag. In effect, this operation multiplies the contents of stack location 01 by 2, thereby shifting the result in the Accumulator left one bit and retaining the most significant bit in the Carry/Link flag.

		B	7
AC		1011	0111
Stack location 01		1011	0111
C/L			0
Result		0110	1110
C/L			1

- D. Store result in Stack location 01      Stack location 01 = 0110 1110  
C/L = 1

E. Load contents of stack location 02 into Accumulator, then add contents of stack location 02 with contents of Accumulator and Carry/Link flag. In effect, this operation multiplies the contents of stack location 02 by 2, thereby shifting the result in the Accumulator left one bit and loading bit 7 of stack location 01 into bit 0 of stack location 02 via the Carry/Link flag. (Since only the last four digits are saved, the stack location 02 bit 7 value that is loaded into the Carry/Link flag is not retained.)

		D	F
AC		1101	1111
Stack location 02		1101	1111
C/L			1
Result		1011	1111
C/L			1

- F. Store result in stack location 02      Stack location 02 = 1011 1111
- G. DECREMENT bit count in stack location 06.

NS10691

Figure 4-11. Four-Digit Save Routine for GHEX and GHEXE Subroutines (Sheet 1 of 2)



- H. Test bit count for zero and repeat C, D, and E (sheet 1) until bit count is decremented to zero. Intermediate results stored in stack locations 01 and 02 will be as follows.

	Stack Location 01		Stack Location 02	
Bit Count = 03	AC	0110 1110	AC	1011 1111
	Stack location 01	0110 1110	Stack location 02	1011 1111
	C/L	0	C/L	0
	Result	1101 1100	Result	0111 1110
	C/L	0	C/L	1
Bit Count = 02	AC	1101 1100	AC	0111 1110
	Stack location 01	1101 1100	Stack location 02	0111 1110
	C/L	0	C/L	1
	Result	1011 1000	Result	1111 1101
	C/L	1	C/L	0
Bit Count = 01	AC	7 0 1011 1000	AC	F B 1111 1101
	Stack location 01	1011 1000	Stack location 02	1111 1101
	C/L	0	C/L	1
	Result	0111 0000	Result	1111 1011
	C/L	1	C/L	1

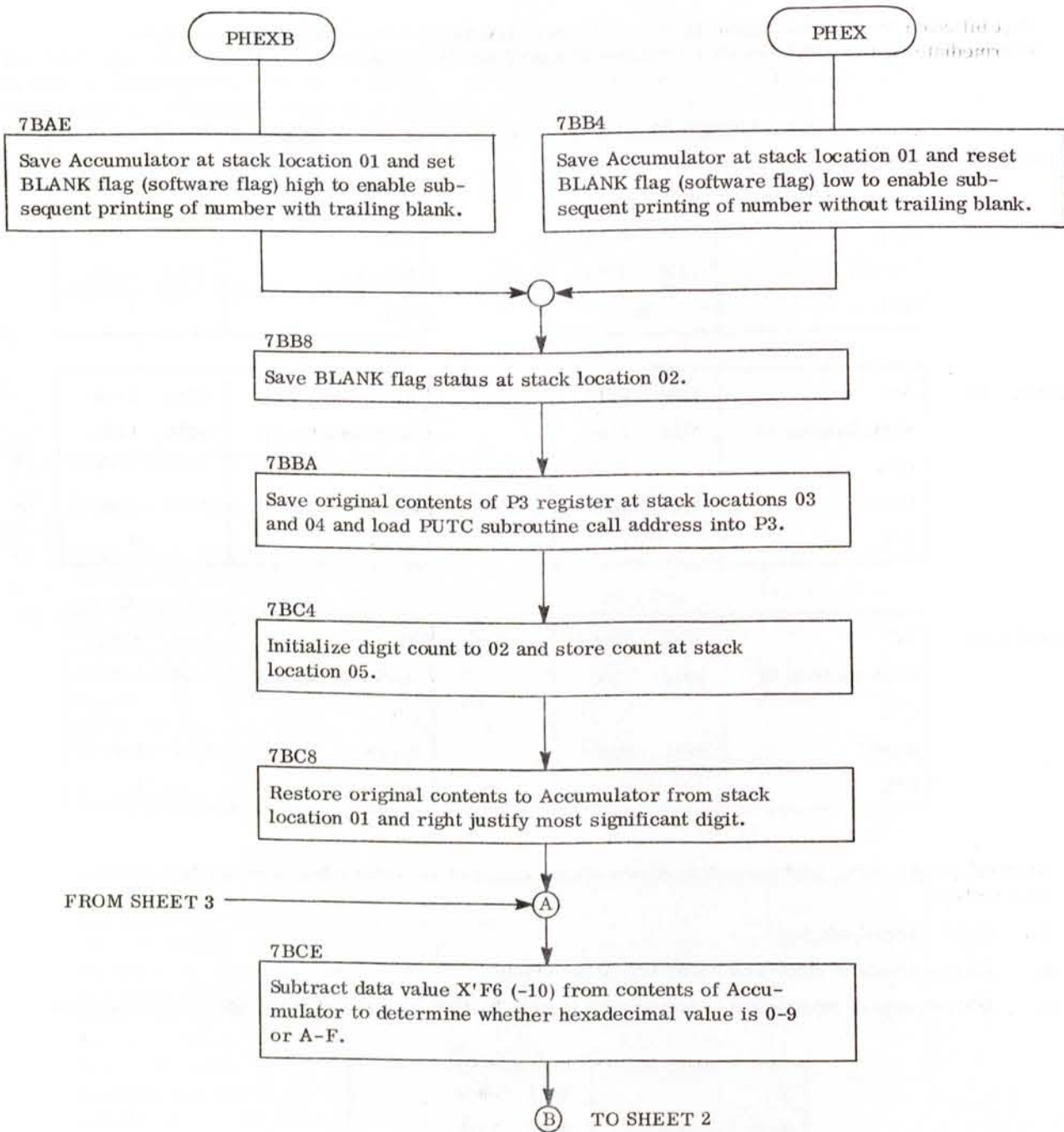
- I. When bit count = zero, load current input into stack location 01 as shown below to complete 4-digit save routine.
1. Clear Carry/Link flag.
  2. Load contents of stack location 01 into Accumulator.
  3. Add contents of Accumulator with contents of stack location 05 (current input) and Carry/Link flag.

AC	0111 0000
Stack Location 05	0000 1111
C/L	0
Result	0111 1111 (X'7F)
C/L	0

4. Store result in stack location 01 to complete 4-digit save routine.

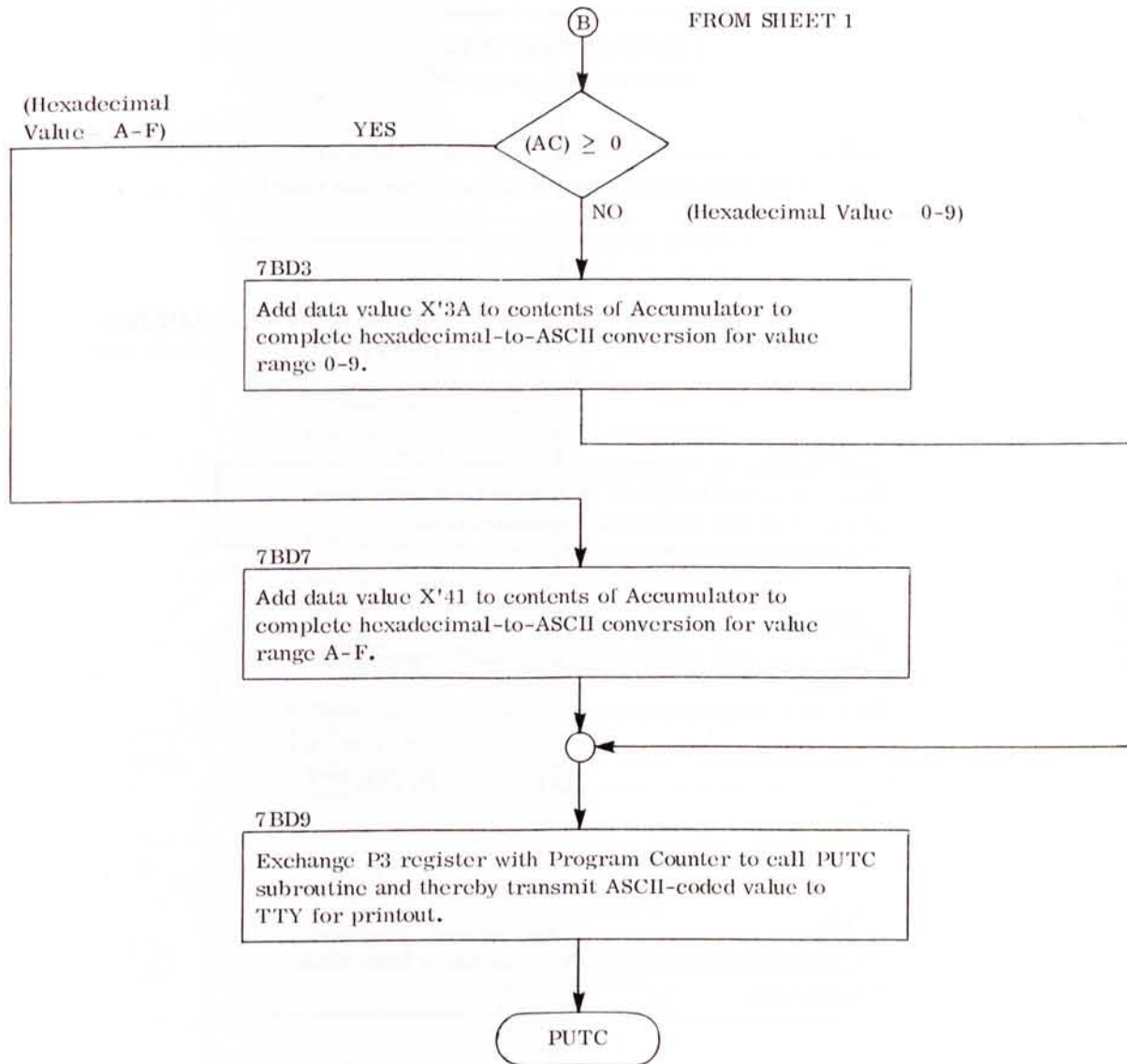
NS10692

Figure 4-11. Four-Digit Save Routine for GHEX and GHEXE Subroutines (Sheet 2 of 2)



NS10693

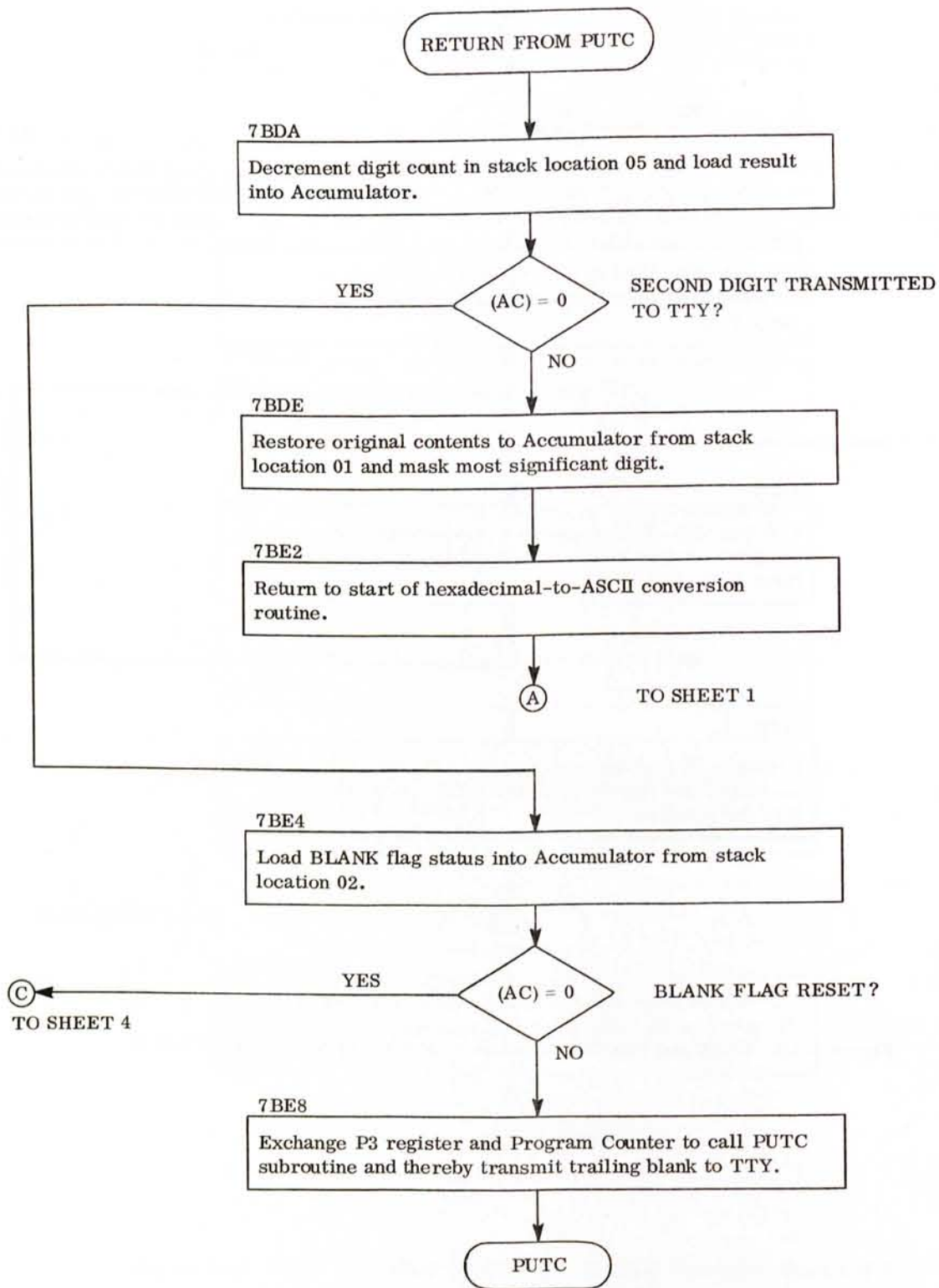
Figure 4-12. PHEX and PHEXB Subroutine -- Detailed Flowchart (Sheet 1 of 4)



NS10694

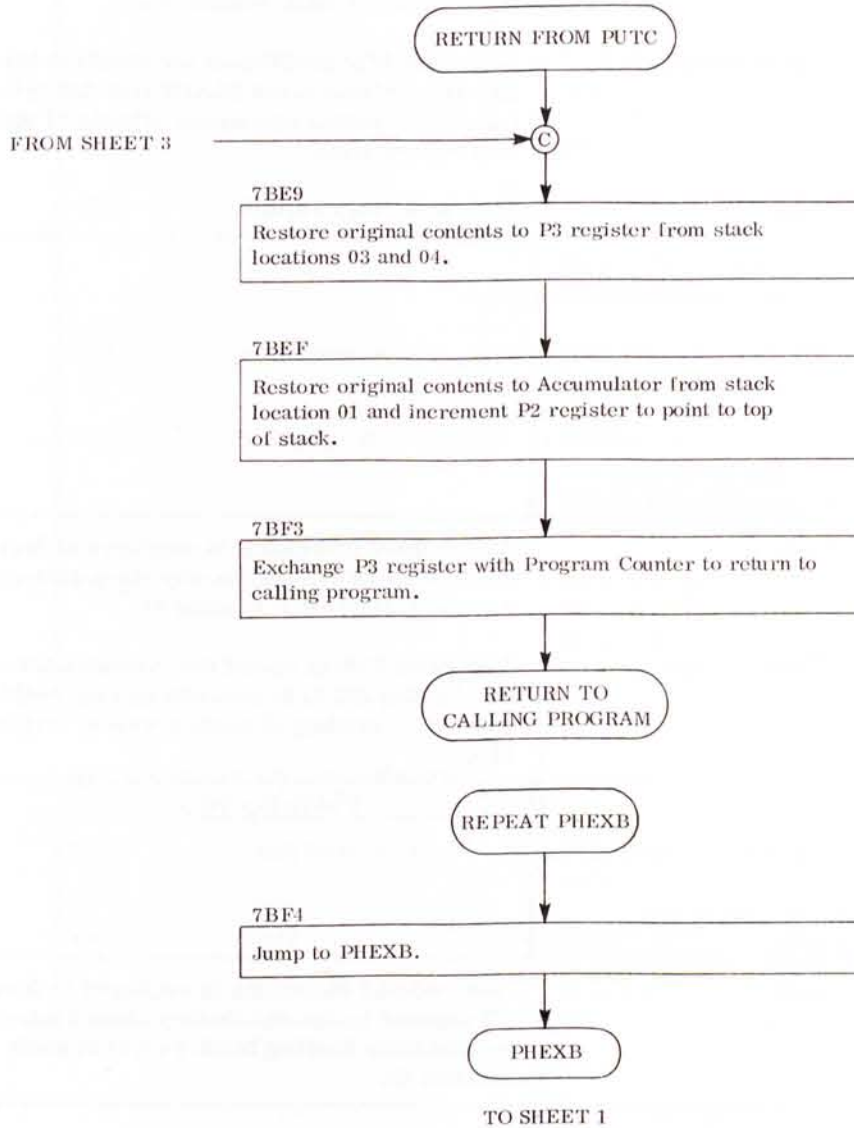
Figure 4-12. PHEX and PHEXB Subroutine — Detailed Flowchart (Sheet 2 of 4)





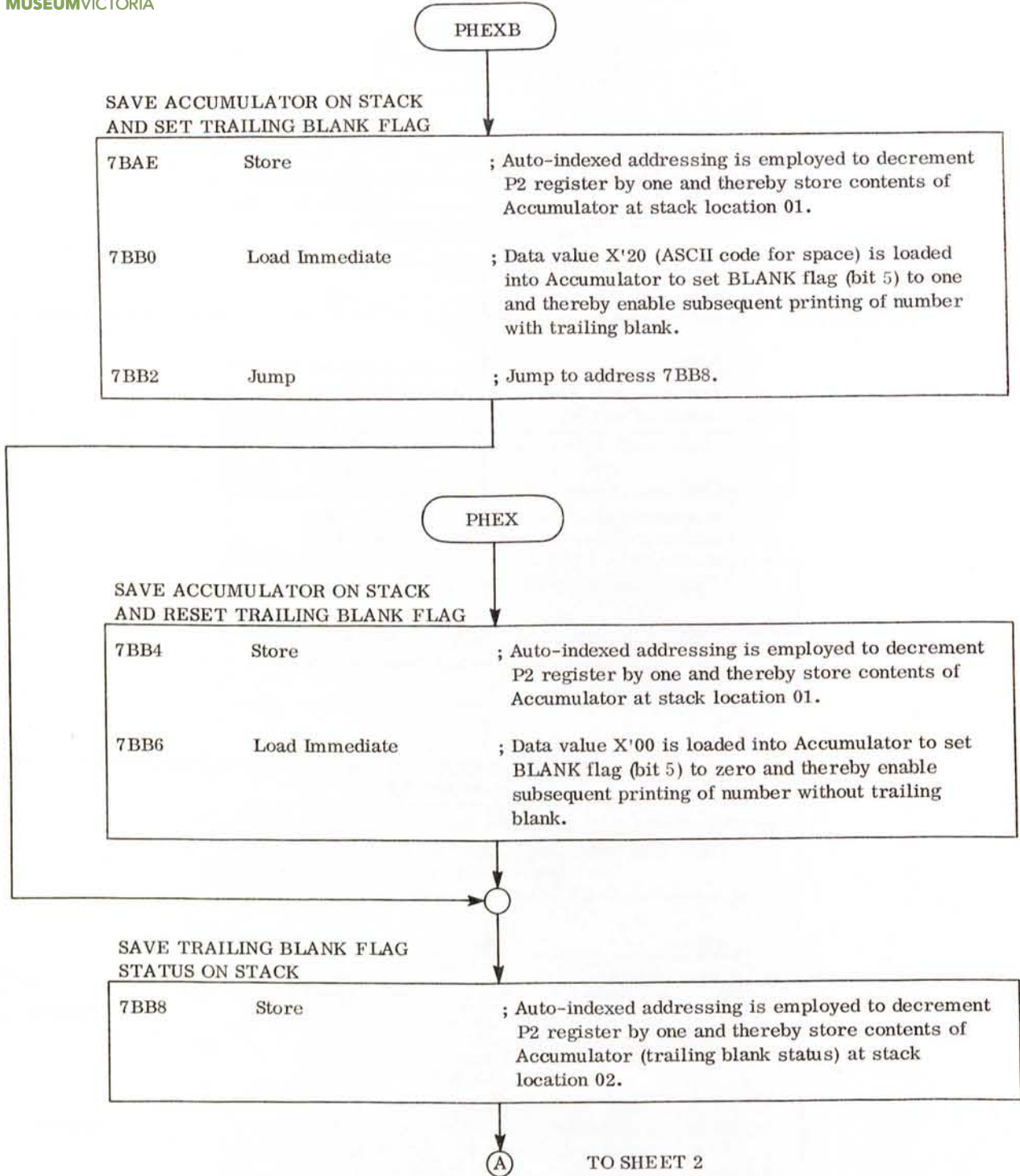
NS10695

Figure 4-12. PHEX and PHEXB Subroutine – Detailed Flowchart (Sheet 3 of 4)



NS10696

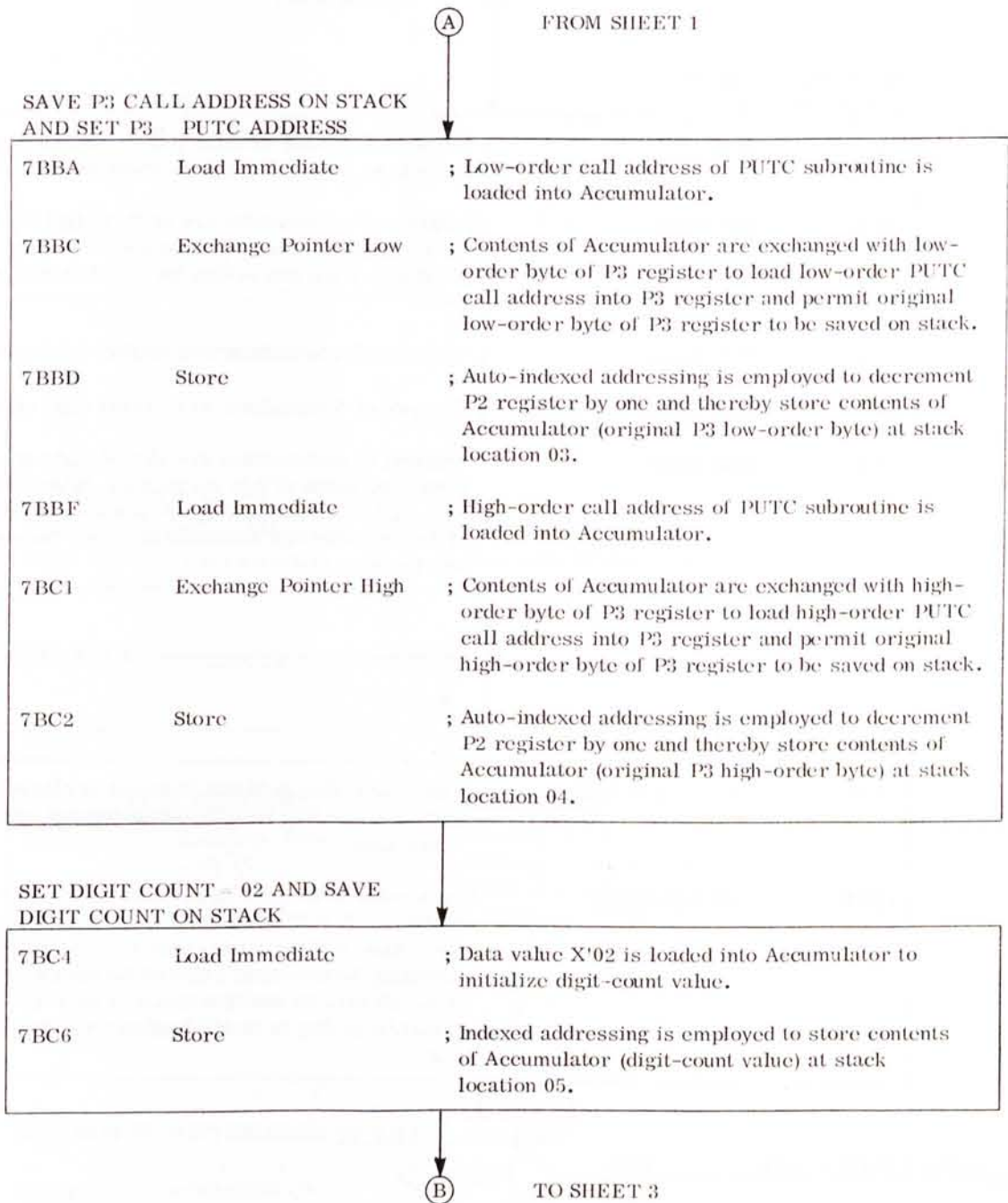
Figure 4-12. PHEX and PHEXB Subroutine – Detailed Flowchart (Sheet 4 of 4)



NS10697

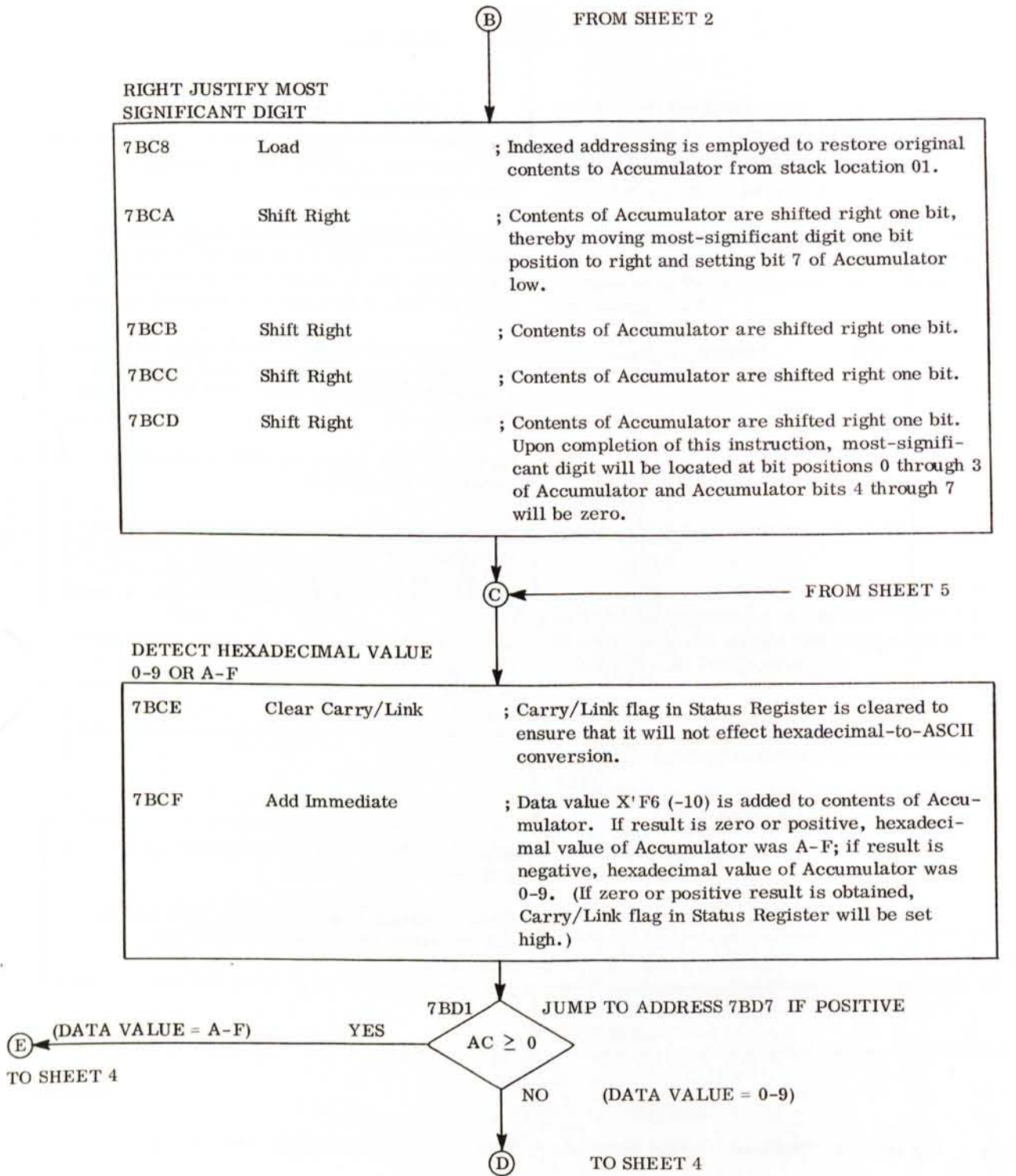
Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 1 of 7)





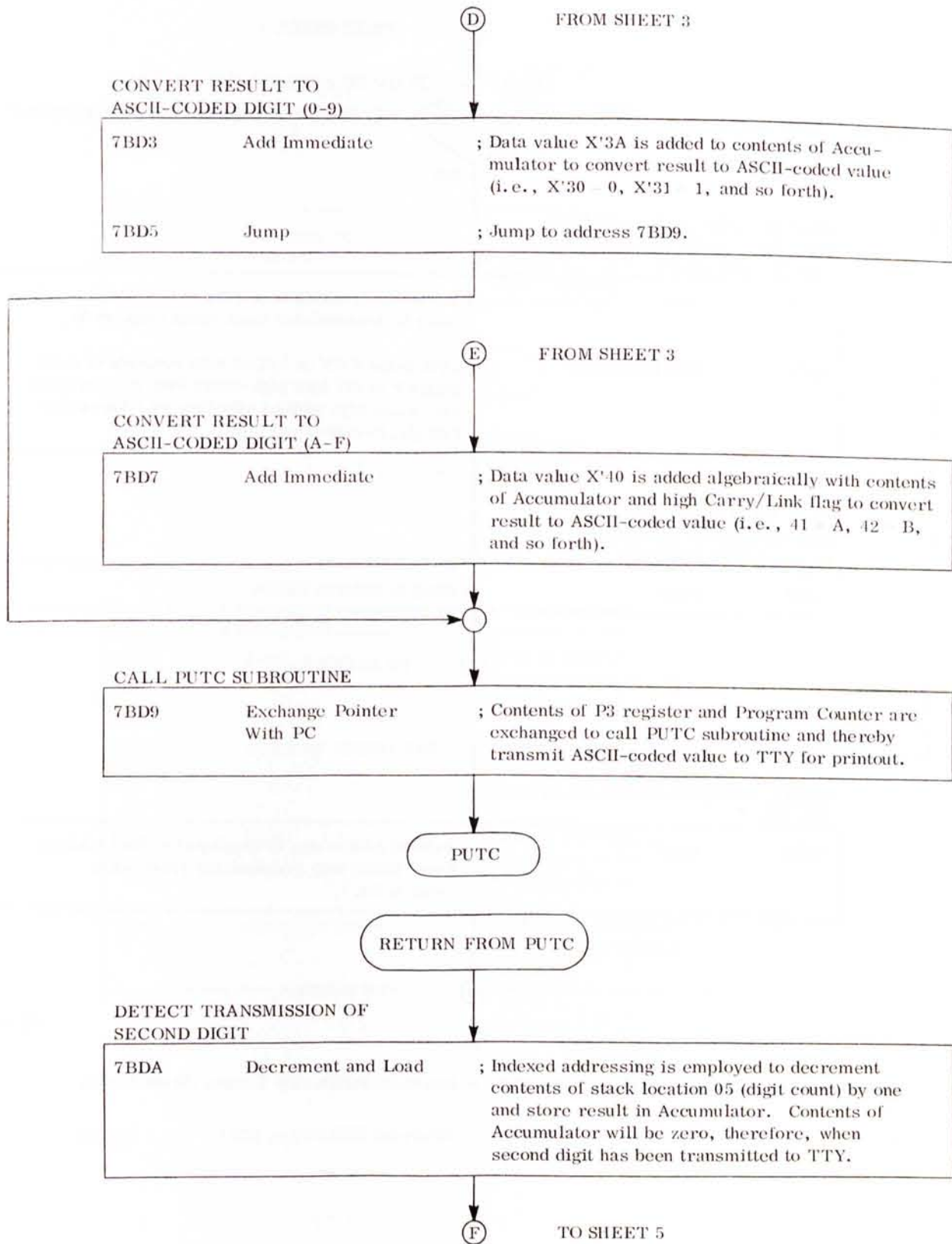
NS10698

Figure 4-13. PHEX and PHEXB Subroutine – Annotated Instruction Listing (Sheet 2 of 7)



NS10699

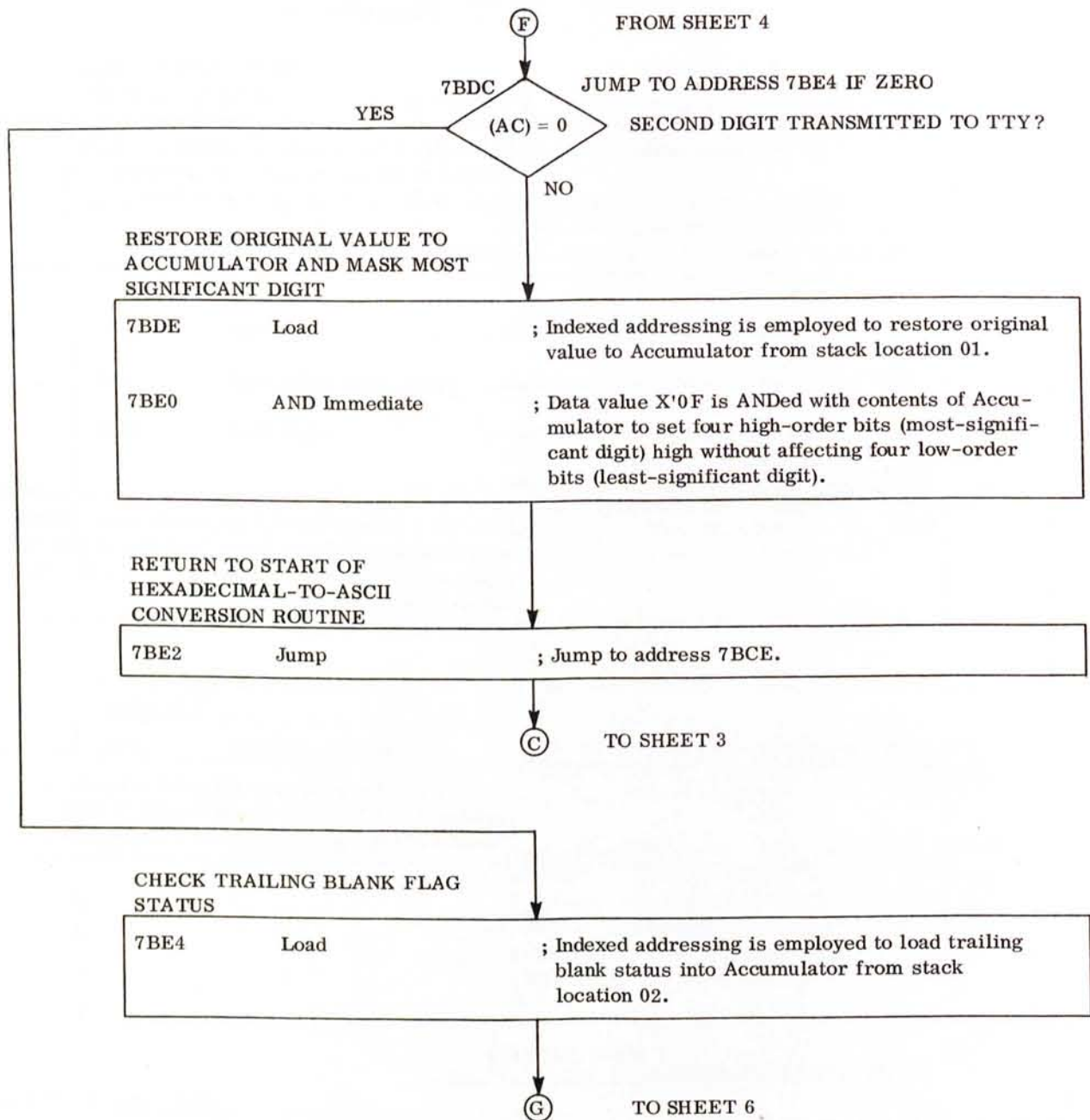
Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 3 of 7)



NS10700

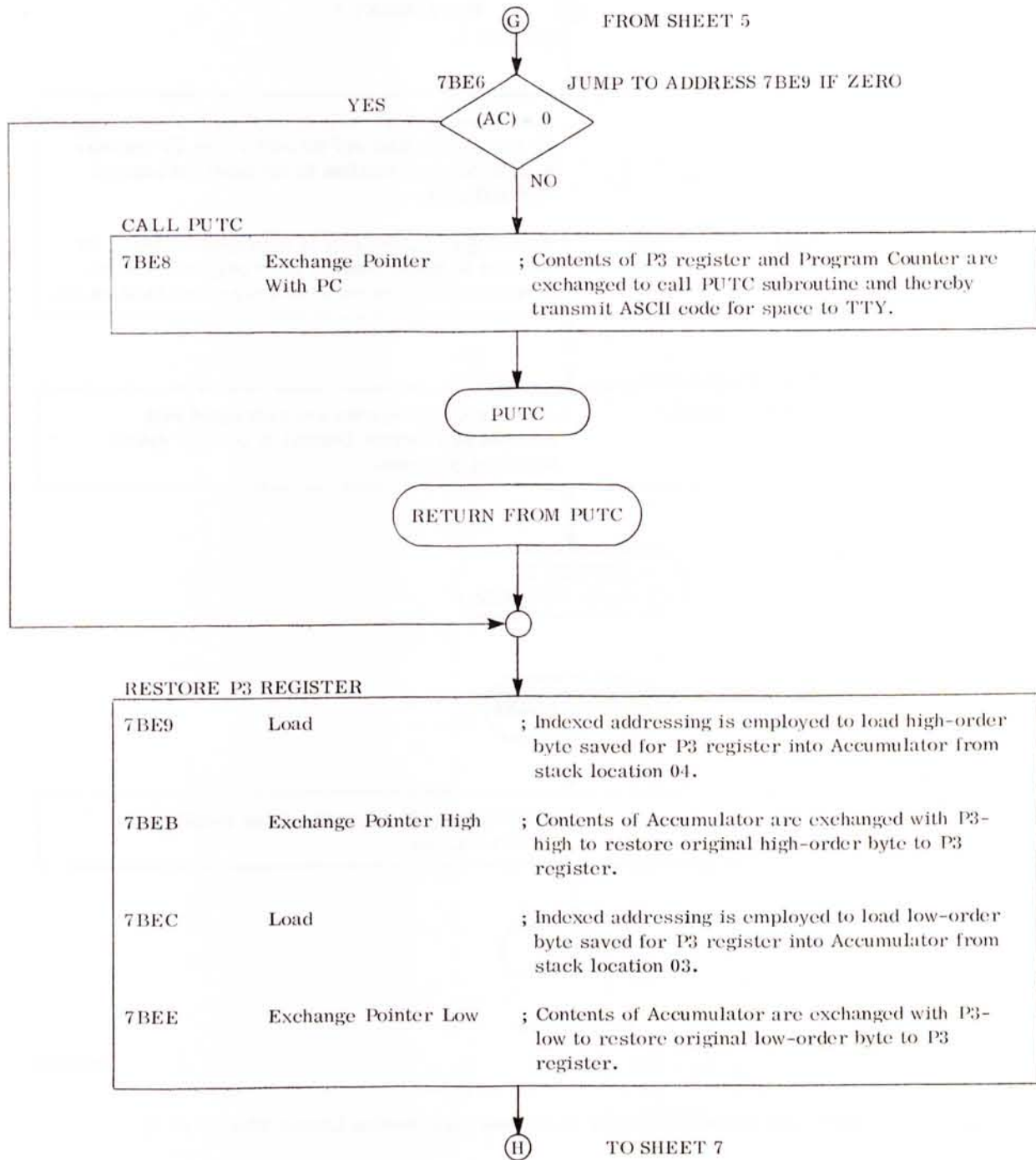
Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 4 of 7)





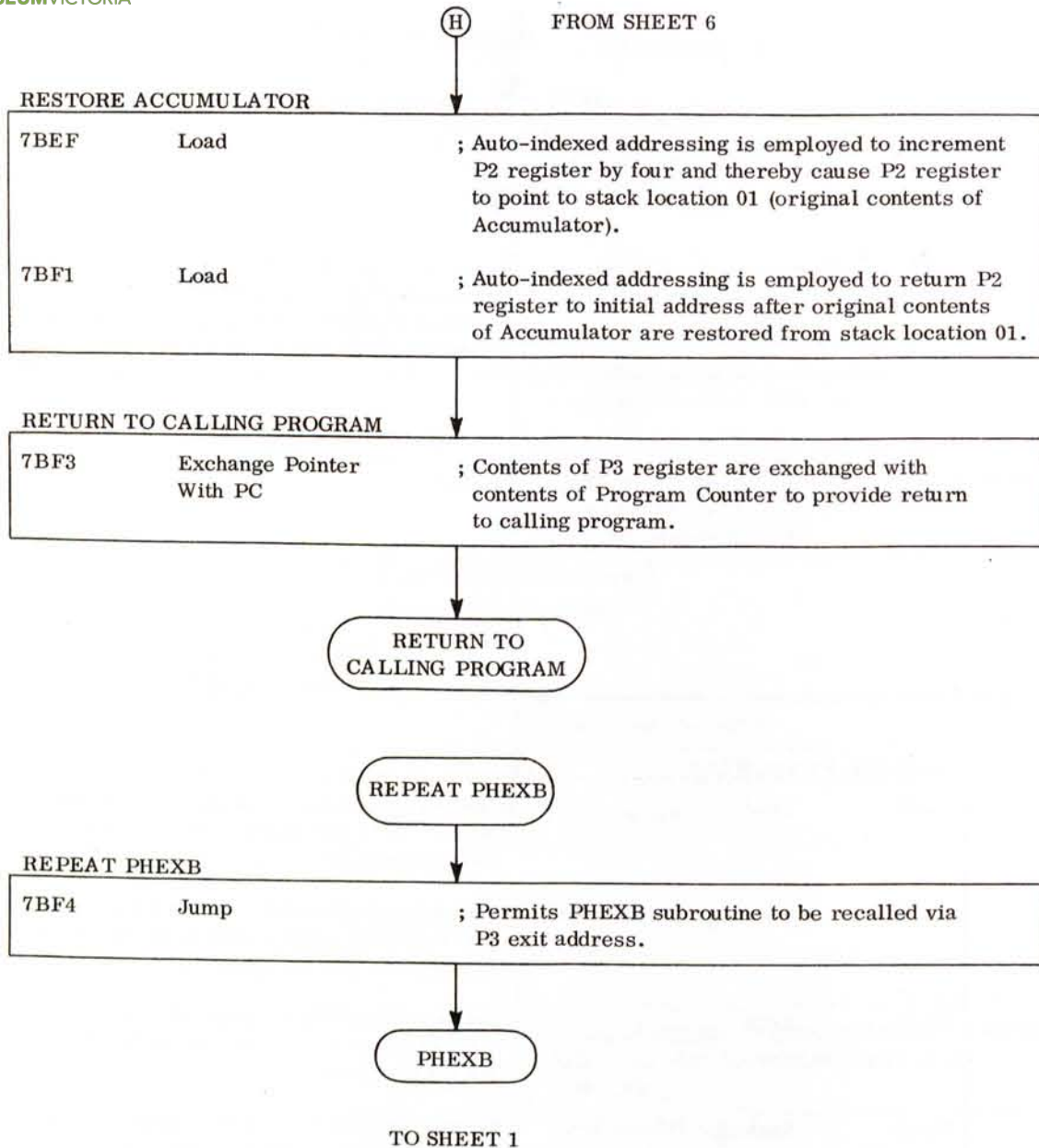
NS10701

Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 5 of 7)



NS10702

Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 6 of 7)



NS10703

Figure 4-13. PHEX and PHEXB Subroutine — Annotated Instruction Listing (Sheet 7 of 7)



## Listing

NUMBER OUTPUT

WITH TRAILING BLANK (PHEXB) OR  
NUMBER TO BE PRINTED IS

```

P2)          ; SAVE AC
              ; SET FLAG TO PRINT BLANK AFTER
              ;   NUMBER
P2)          ; SAVE AC
              ; CLEAR FLAG TO PRINT BLANK
P2)          ;   AFTER NUMBER
TC)-1        ; LOAD ADDRESS OF PUTC TO P3
              ;   AND SAVE RETURN ADDRESS

P2)
TC)

P2)          ; SET FLAG FOR 1ST NUMBER

P2)
)           ; GET ORIGINAL VALUE
           ; SHIFT TO LOW 4 BITS

           ; CONVERT TO ASCII
           ; NUMBER IS A THRU F

10
)           ; THE -1 TAKES CARE OF CARRY IN
           ; PRINT NUMBER

)           ; GET ORIGINAL NUMBER
           ; MASK 2ND DIGIT

)           ; CHECK FOR PRINTING BLANK

)           ; IF NOT 0, PRINT BLANK
           ; RESTORE RETURN ADDRESS

)
)           ; RESTORE STACK AND AC
)           ; RETURN
3

```



## PHEX and PHEXB Subroutine — Actual Listing

```

771                                     .PAGE      'HEX NUMBER OUTPUT'
772                                     .LOCAL
773                                     ;
774                                     ; PRINT HEX NUMBER WITH TRAILING BLANK (PHEXB) OR
775                                     ; WITHOUT IT (PHEX).  NUMBER TO BE PRINTED IS
776                                     ; IN AC.
777                                     ;
778 7BAE CEFF PHEXB: ST      @-1(P2)          ; SAVE AC
779 7BB0 C420          LDI      020          ; SET FLAG TO PRINT BLANK AFTER
780 7BB2 9004          JMP      $1          ; NUMBER
781 7BB4 CEFF PHEX:  ST      @-1(P2)          ; SAVE AC
782 7BB6 C400          LDI      0          ; CLEAR FLAG TO PRINT BLANK
783 7BB8 CEFF $1:    ST      @-1(P2)          ; AFTER NUMBER
784 7BBA C4E1          LDI      L(PUTC)-1    ; LOAD ADDRESS OF PUTC TO P3
785 7BBC 33           XPAL     P3          ; AND SAVE RETURN ADDRESS
786 7BBD CEFF          ST      @-1(P2)
787 7BBF C47A          LDI      H(PUTC)
788 7BC1 37           XPAH     P3
789 7BC2 CEFF          ST      @-1(P2)
790 7BC4 C402          LDI      2          ; SET FLAG FOR 1ST NUMBER
791 7BC6 CEFF          ST      @-1(P2)
792 7BC8 C204          LD       4(P2)          ; GET ORIGINAL VALUE
793 7BCA 1C           SR          ; SHIFT TO LOW 4 BITS
794 7BCB 1C           SR
795 7BCC 1C           SR
796 7BCD 1C           SR
797 7BCE 02          $5:    CCL          ; CONVERT TO ASCII
798 7BCF F4F6          ADI      -10
799 7BD1 9404          JP       $2          ; NUMBER IS A THRU F
800 7BD3 F43A          ADI      '0'+10
801 7BD5 9002          JMP      $3
802 7BD7 F440          $2:    ADI      'A'-1    ; THE -1 TAKES CARE OF CARRY IN
803 7BD9 3F          $3:    XPPC     P3          ; PRINT NUMBER
804 7BDA BA00          DLD      (P2)
805 7BDC 9806          JZ       $4
806 7BDE C204          LD       4(P2)          ; GET ORIGINAL NUMBER
807 7BE0 D40F          ANI      0F          ; MASK 2ND DIGIT
808 7BE2 90EA          JMP      $5
809 7BE4 C203          $4:    LD       3(P2)          ; CHECK FOR PRINTING BLANK
810 7BE6 9801          JZ       $6
811 7BE8 3F          XPPC     P3          ; IF NOT 0, PRINT BLANK
812 7BE9 C201          $6:    LD       1(P2)          ; RESTORE RETURN ADDRESS
813 7BEB 37           XPAH     P3
814 7BEC C202          LD       2(P2)
815 7BEE 33           XPAL     P3
816 7BEF C604          LD       @4(P2)          ; RESTORE STACK AND AC
817 7BF1 C601          LD       @1(P2)
818 7BF3 3F          XPPC     P3          ; RETURN
819 7BF4 90B8          JMP      PHEXB

```





A. Clear Carry/Link (C/L) flag in Status Register to ensure that it will not affect hexadecimal-to-ASCII conversion.

B. Add data value X'F6 (-10) with contents of Accumulator and Carry/Link flag. If result is negative, convert result to appropriate ASCII value as shown in C below; if result is zero or positive, convert result to appropriate ASCII value as shown in D below.

	0	9	A	F
AC	0000 0000	0000 1001	0000 1010	0000 1111
Data	1111 0110	1111 0110	1111 0110	1111 0110
C/L	0	0	0	0
Result	1111 0110	1111 1111	0000 0000	0000 0101
C/L	0	0	1	1

C. Add data value X'3A with contents of Accumulator and Carry/Link flag to complete hexadecimal-to-ASCII conversion.

	0	9
AC	1111 0110	1111 1111
Data	0011 1010	0011 1010
C/L	0	0
Result	0011 0000	0011 1001
C/L	1	1

D. Add data value X'40 with contents of Accumulator and Carry/Link flag to complete hexadecimal-to-ASCII conversion.

	A	F
AC	0000 0000	0000 0101
Data	0100 0000	0100 0000
C/L	1	1
Result	0100 0001	0100 0110
C/L	0	0

NS10704

Figure 4-14. Hexadecimal-to-ASCII Conversion Routine for PH1EX and PH1EXB Subroutines

## Chapter 5

### FUNCTIONAL DESCRIPTION

#### 5.1 GENERAL DESCRIPTION

The LCDS (see figure 5-1) provides control and display functions that permit an SC/MP CPU Application Card to be used to develop, test, and debug SC/MP hardware and software applications designs. Functionally, the LCDS consists of a resident firmware program, a keyboard and display panel (Panel), and all of the logic and buffer circuits required to transfer control back and forth between the resident firmware program (DEBUG Mode) and a user-entered applications program (RUN Mode). From a software standpoint, DEBUG operations encompass the address range 7000 through 7FFF; the remainder of the 64K address range (0000 through 6FFF and 8000 through FFFF) is available for storage of the user's applications program and assignment of RUN Mode functions to the user's applications hardware.

When the DEBUG Mode is selected, it is entered via a hardware-forced Save Routine that is executed when the current RUN Mode instruction is completed. During the Save Routine, the contents of the SC/MP accumulator, status and P3 registers are stored in LCDS memory and the P3 register is loaded with address 7808. Then, as the last step of the Save Routine, the contents of the P3 register and program counter are exchanged so that execution of the resident DEBUG firmware program can be initiated at starting address 7809.

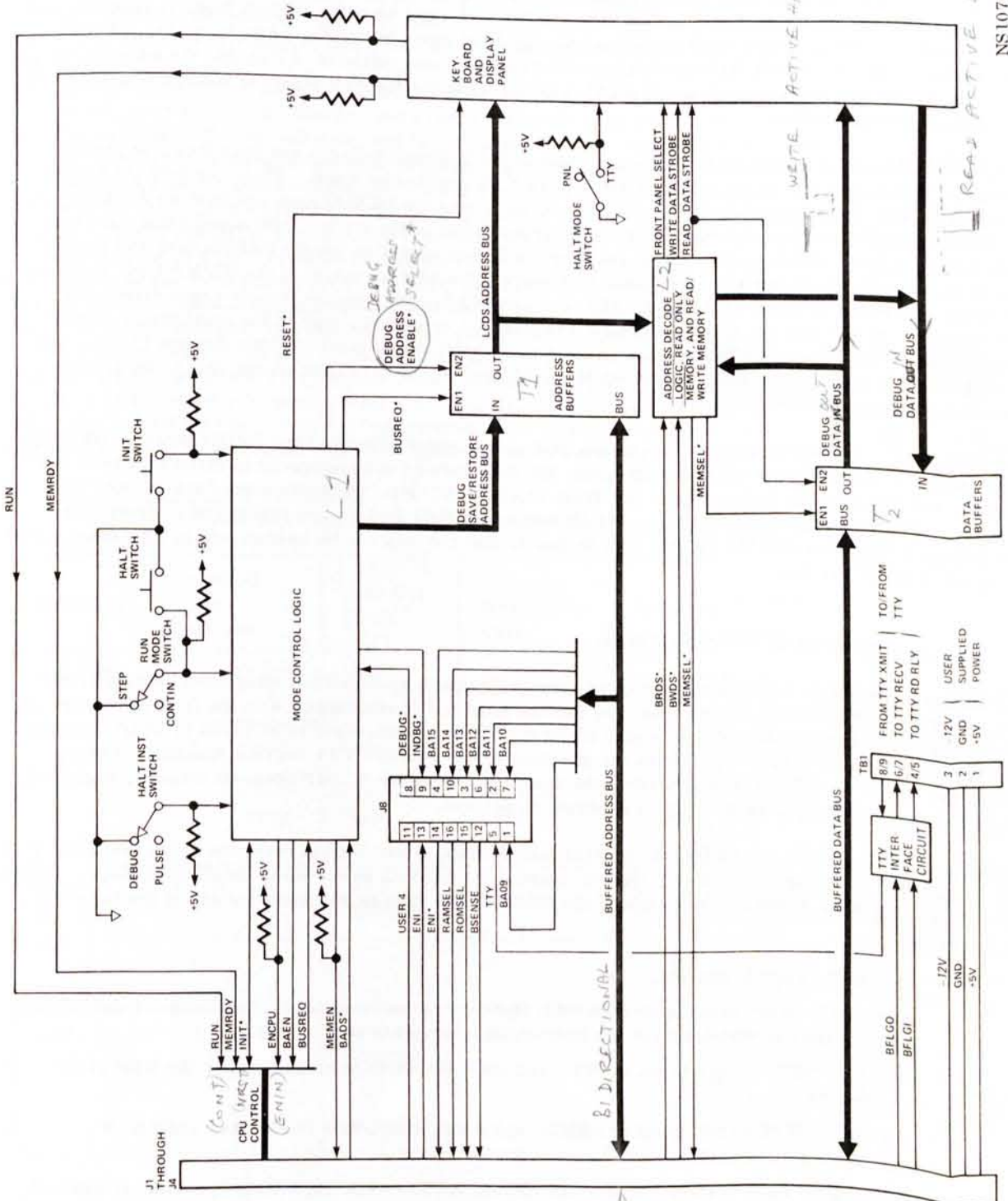
At the start of the DEBUG firmware program, the contents of the remaining SC/MP registers are saved in LCDS memory, then the setting of the HALT MODE switch is checked to determine whether control and display capability is to be provided at the Panel or at the optional TTY. When the HALT MODE switch is set to PNL, the DEBUG firmware branches to the Panel Service Routines and displays the saved PC value on the digital readout to indicate that DEBUG commands may be entered via the panel keyboard; similarly, when the HALT MODE switch is set to TTY, the DEBUG firmware branches to the TTY Service Routines and prints out the saved PC contents on the TTY to indicate that DEBUG commands may be entered via the TTY keyboard. After displaying or printing out the saved PC value, the DEBUG firmware then continually cycles through the appropriate service routines to permit processing of ensuing DEBUG commands as described in chapter 3, Operation.

Termination of the DEBUG Mode occurs when the RUN Mode is subsequently selected via the RUN pushbutton (HALT MODE switch set to PNL) or the TTY GO command (HALT MODE switch set to TTY). After the RUN Mode is selected, the DEBUG firmware branches to an Exit Subroutine that restores part of the saved status to the SC/MP microprocessor; then, a hardware-forced DEBUG Return Routine is executed to restore the remaining status to the SC/MP microprocessor. Upon completion of the DEBUG Return Routine, an IFETCH data input/output cycle occurs to initiate RUN Mode operation at the restored PC address and thereby provide a return to the user's application program without loss of SC/MP microprocessor status. (Users familiar with operation of the SC/MP CPU Application Card should note that the DEBUG Return Routine decrements the value contained in memory locations 77F5 and 77F6 before restoring it to the PC. This action compensates for the automatic PC increment which occurs during the IFETCH data input/output cycle that initiates RUN Mode operation.)

#### NOTE

The following paragraphs assume a thorough understanding of the operation of the SC/MP CPU Application Card. For detailed information on the operation of the SC/MP CPU Application Card, refer to the SC/MP CPU Application Card Specification.





NS10705

Figure 5-1. LCDS Overall Block Diagram

### 5.1.1 CPU and Data I/O Control

As shown in figure 5-1, the LCDS provides a prewired interface bus at connectors J1 through J4 to permit plug-in interconnection of an SC/MP CPU Application Card and other SC/MP family application cards. Wiring of the bus is such that the basic CPU control signals (MEMRDY, RUN, and ENCPU) are pulled to +5V via resistors on the LCDS to permit continuous operation of the CPU card when the DEBUG Mode is enabled, and application-system control of CPU card operation when the RUN Mode is enabled. Similarly, the MEMEN (Address Enable) input for the SC/MP family application cards is also pulled to +5V so that the address compare circuit on the cards will remain continuously enabled except when the MEMEN signal is externally driven low for special-purpose applications.

TRI-STATE<sup>®</sup> buffers<sup>T14T2</sup> are employed for interconnection of the Buffered Address and Data Buses<sup>Buffered from CPU card</sup> to permit program-controlled operation of the LCDS via the SC/MP CPU Application Card. Thus, the DEBUG Address Select\* output of the Mode Control Logic is set high, except during the DEBUG Save and Restore Routines, so that the Address Buffers will continuously drive the address output of the SC/MP CPU Application Card onto the LCDS Address Bus. (Mode Control Logic and Address Buffer operation for the DEBUG Save and Restore Routines is covered in the paragraphs that follow.) Whenever the address output of the SC/MP CPU Application Card is within the range assigned to the LCDS (7000 through 7FFF), the Address Decode Logic drives the MEMSEL\* output low for address acknowledgement and provides an enable signal to the appropriate memory device (addresses 7600 through 7FFF) or the Keyboard and Display Panel (address 7000 through 71FF). The ensuing BRDS\* or BWDS\* strobe output of the SC/MP CPU Application Card then causes data to be written into or read out of the addressed location.

GATING  
Gating of read/write data to/from the Buffered Data Bus occurs under control of the BRDS\* strobe. When the BRDS\* strobe goes low while the LCDS is addressed, the Data Buffers are configured to drive data from the DEBUG Data Out Bus onto the Buffered Data Bus from which the SC/MP CPU Application Card accepts the data on the trailing edge of the BRDS\* strobe. At all times, the Data Buffers are configured to drive data from the Buffered Data Bus onto the DEBUG Data In Bus so that the data can be written into an addressed LCDS location by the BWDS\* strobe.

BRDS  
READ ON TRAILING EDGE

### 5.1.2 Hardware-Forced DEBUG Save Routine

The DEBUG Save Routine is a hardware-controlled subroutine that is executed following initialization of the LCDS or selection of the DEBUG Mode. The Save Routine functions in conjunction with the first subroutine of the DEBUG firmware to cause the internal status of the SC/MP microprocessor to be saved in LCDS memory locations 77F5 through 77FF before the firmware branches to the Panel or TTY Service Routines. During execution of the Panel or TTY Service Routines, the values saved for the SC/MP program counter, registers, and accumulator can then be examined and/or modified as desired.

When power is first applied to the LCDS, the DEBUG Save Routine is executed in response to the low-going INIT\* output of the SC/MP CPU Application Card, thereby causing the LCDS to power-up in the DEBUG Mode. Following the power-up initialization sequence, execution of the DEBUG Save Routine reoccurs for any of the following specific conditions:

1. The INIT switch is pressed.
2. The HALT switch is pressed while RUN Mode operation is enabled. (Execution of the DEBUG Save Routine is initiated after the instruction in progress is completed).
3. The RUN/STEP switch is set to STEP and one instruction is executed after the RUN Mode is selected.
4. The HALT INST switch is set to DEBUG and a Halt Instruction is executed while RUN Mode operation is enabled.
5. The DEBUG input to the LCDS is externally driven low while RUN Mode operation is enabled.



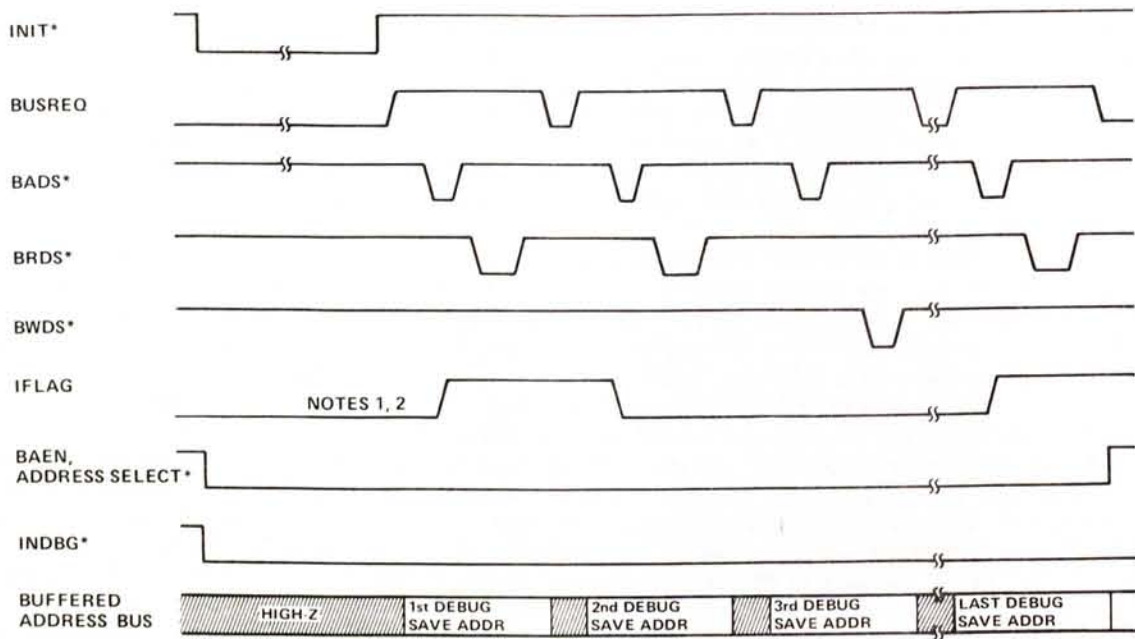
The Mode Control Logic enables execution of the DEBUG Save Routine by holding the BAEN and Address Select\* outputs low while processing the BUSREQ outputs of the SC/MP CPU Application Card to force a series of DEBUG Save Routine addresses onto the Buffered Address Bus. (While the BAEN output is low, the address output of the SC/MP CPU Application Card is held in the high-impedance state, but operation of the remaining circuitry on the card is not affected.) As shown in figure 5-2, driving the BAEN and Address Select\* outputs low occurs either on the leading edge of an INIT\* pulse, or on the leading edge of the BADS\* strobe associated with the first IFETCH data input/output cycle that follows selection of the DEBUG Mode.

While the address output of the SC/MP CPU Application Card is held in the high-impedance state by the low BAEN signal, the DEBUG Save Routine Addresses provide the functions necessary for SC/MP instruction execution (refer to table 5-1). For example, the first address is always output during an IFETCH data input/output cycle, so it causes the data value C8 to be read into the SC/MP microprocessor and treated as an instruction opcode. This opcode specifies a Store (ST) Instruction which requires two additional data input/output cycles for completion (see figure 4 of the SC/MP CPU Application Card Data Sheet). Normally, the second data input/output cycle would enable the displacement byte of the instruction to be read into the SC/MP microprocessor for computation of an Effective Address, and the third data input/output cycle would enable the contents of the SC/MP accumulator to be stored at the Effective Address. With the address output of the SC/MP CPU Application Card disabled, however, the Effective Address associated with the Store Instruction is specified by the third DEBUG Save Address, thereby causing the contents of the SC/MP accumulator to be saved in LCDS memory location 77FD. The next 16 DEBUG Save Routine Addresses are then processed in a like manner to 1) load the DEBUG program start address (7808) into the P3 register, 2) save the original contents of the P3 register in LCDS memory locations 77FB and 77FC, and 3) save the status register contents in memory location 77FF. The last DEBUG Save Address then references an XPPC P3 instruction to cause the contents of the program counter to be exchanged with the contents of the P3 register.

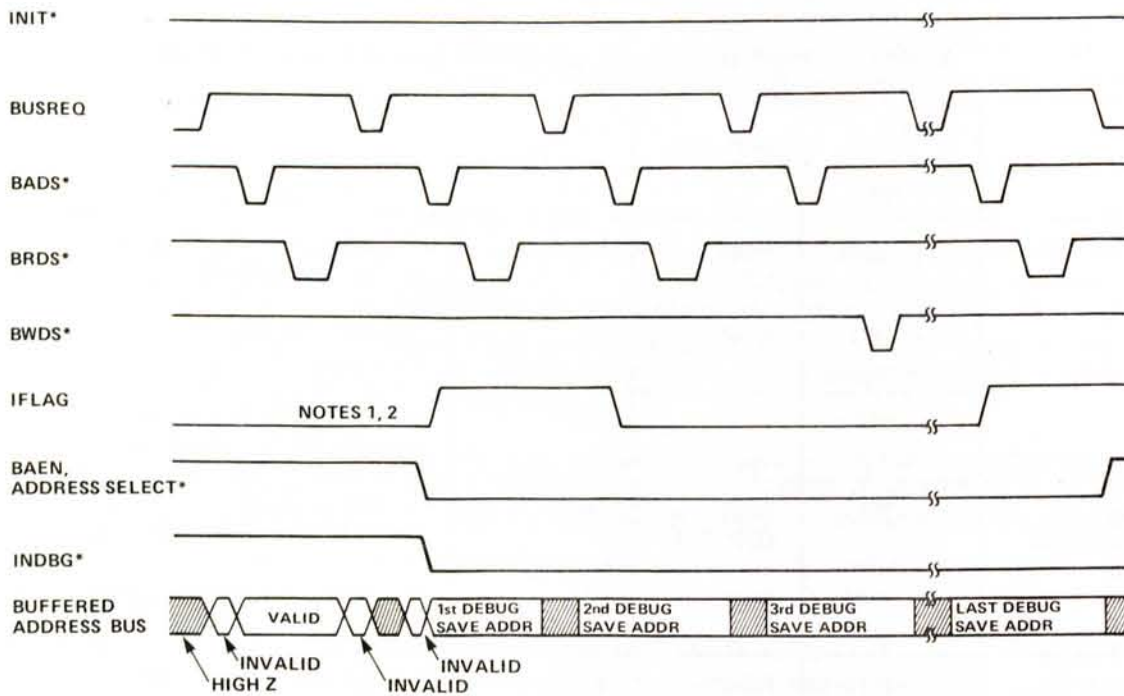
Table 5-1. Hardware-Forced DEBUG Save Routine Address Functions

Instruction	DEBUG Save Address	Data I/O Cycle	Buffered Read/Write Data	Comments
Store (ST)	1. 7800	Read (IFETCH)	C8	SC/MP microprocessor reads in opcode for Store (ST) Instruction.
	2. 7800	Read	C8	SC/MP microprocessor reads in dummy displacement value.
	3. 77FD	Write	(AC)	Contents of SC/MP accumulator are saved saved in LCDS memory location 77FD.
Load Immediate (LI)	4. 7801	Read (IFETCH)	C4	SC/MP microprocessor reads in opcode for Load Immediate (LI) Instruction.
	5. 7802	Read	08	SC/MP accumulator is loaded with low-order DEBUG program start address.
XPAL P3	6. 7803	Read (IFETCH)	33	SC/MP microprocessor reads in opcode for XPAL P3 Instruction; contents of SC/MP accumulator (low-order DEBUG start address) are exchanged with low-order byte of Pointer Register 3.





A. Debug Save Routine Initiated by Low-Going INIT\* Pulse



B. Debug Save Routine Not Initiated by Low-Going INIT\* Pulse

- NOTES:
1. The IFLAG output of the SC/MP CPU Application Card is shown only as a general timing reference; it is not used by the LCDS.
  2. The Mode Control Logic detects an IFETCH data I/O cycle by processing the DI5 data-bit input under control of the BADS\* strobe.

NS10706

Figure 5-2. DEBUG Save Routine Timing

Table 5-1. Hardware-Forced DEBUG Save Routine Address Functions (Continued)

Instruction	DEBUG Save Address	Data I/O Cycle	Buffered Read/Write Data	Comments
Store (ST)	7. 7800	Read (IFETCH)	C8	SC/MP microprocessor reads in opcode for Store (ST) Instruction.
	8. 7800	Read	C8	SC/MP microprocessor reads in dummy displacement value.
	9. 77FC	Write	(AC)	Pointer Register 3 low-order byte is saved in LCDS memory location 77FC.
Load Immediate (LI)	10. 7801	Read (IFETCH)	C4	SC/MP microprocessor reads in opcode for Load Immediate (LI) Instruction.
	11. 7804	Read	78	SC/MP accumulator is loaded with high-order DEBUG program start address.
XPAH P3	12. 7805	Read (IFETCH)	37	SC/MP microprocessor reads in opcode for XPAH P3 Instruction; contents of SC/MP accumulator (high-order DEBUG Start Address) are exchanged with high-order byte of Pointer Register 3.
Store (ST)	13. 7800	Read (IFETCH)	C8	SC/MP microprocessor reads in opcode for Store (ST) Instruction.
	14. 7800	Read	C8	SC/MP microprocessor reads in dummy displacement value.
	15. 77FB	Write	(AC)	Pointer Register 3 high-order byte is saved in LCDS memory location 77FB.
Copy Status to AC (CSA)	16. 7806	Read (IFETCH)	06	SC/MP microprocessor reads in opcode for Copy Status to AC (CSA) Instruction; contents of Status Register are copied into accumulator.
Store (ST)	17. 7800	Read (IFETCH)	C8	SC/MP microprocessor reads in opcode for Store (ST) Instruction.
	18. 7800	Read	C8	SC/MP microprocessor reads in dummy displacement value.
	19. 77FF	Write	(AC)	Contents of Status Register are saved in LCDS memory location 77FF.
XPPC P3	20. 7808	Read (IFETCH)	3F	SC/MP microprocessor reads in opcode for XPPC P3 Instruction; contents of SC/MP Program Counter are exchanged with contents of Pointer Register 3.



After the last DEBUG Save Routine Address is output, the BAEN and DEBUG Address Enable\* outputs of the Mode Control Logic are reset high on the trailing edge of the BUSREQ signal to reenable the address outputs of the SC/MP CPU Application Card. On the leading edge of the next BUSREQ signal, the SC/MP CPU Application Card then outputs address 7809 to initiate execution of the DEBUG firmware program. The first routine of the DEBUG firmware program, in turn, saves the contents of the remaining SC/MP registers (PC, P1, P2, and EX) in the designated LCDS memory locations, reads the setting of the TTY HALT MODE switch, and effects a branch to the Panel or TTY Service Routines as appropriate. The Panel or TTY Service Routines are then executed continuously to provide control and display capabilities at the Panel or TTY — as described in chapter 3, Operation.

Users familiar with SC/MP microprocessor operation should note that saving of the program counter contents is effected mathematically rather than directly. When the DEBUG Save Routine is initiated, the SC/MP program counter will have already output the address of the next instruction that would have been fetched had the DEBUG Mode not been selected. Incrementing of the program counter then continues normally during the DEBUG Save Routine and, when the XPPC P3 instruction is executed to complete the DEBUG Save Routine, the program counter will have been incremented 15 times. (As shown in figure 4 of the SC/MP CPU Application Card Data Sheet, incrementing of the program counter occurs at the start of each data input/output cycle except when an Effective Address is output during a data input/output cycle.) During the first DEBUG firmware routine, therefore, the value 15 is subtracted from the P3 register, and the result is stored in LCDS memory locations 77F5 and 77F6. Thus, the value saved for the program counter is the address that was present on the Buffered Address Bus when the DEBUG Save Routine was initiated.

#### 5.1.3 DEBUG Mode Termination

In addition to providing control and display functions, the DEBUG firmware Panel and TTY Service Routines also check for selection of the RUN Mode via the RUN switch or the TTY GO command. When the RUN Mode is selected, it causes the DEBUG firmware to branch to an Exit Routine that initiates restoration of the saved values to the SC/MP microprocessor. During the Exit Routine, the following specific actions occur:

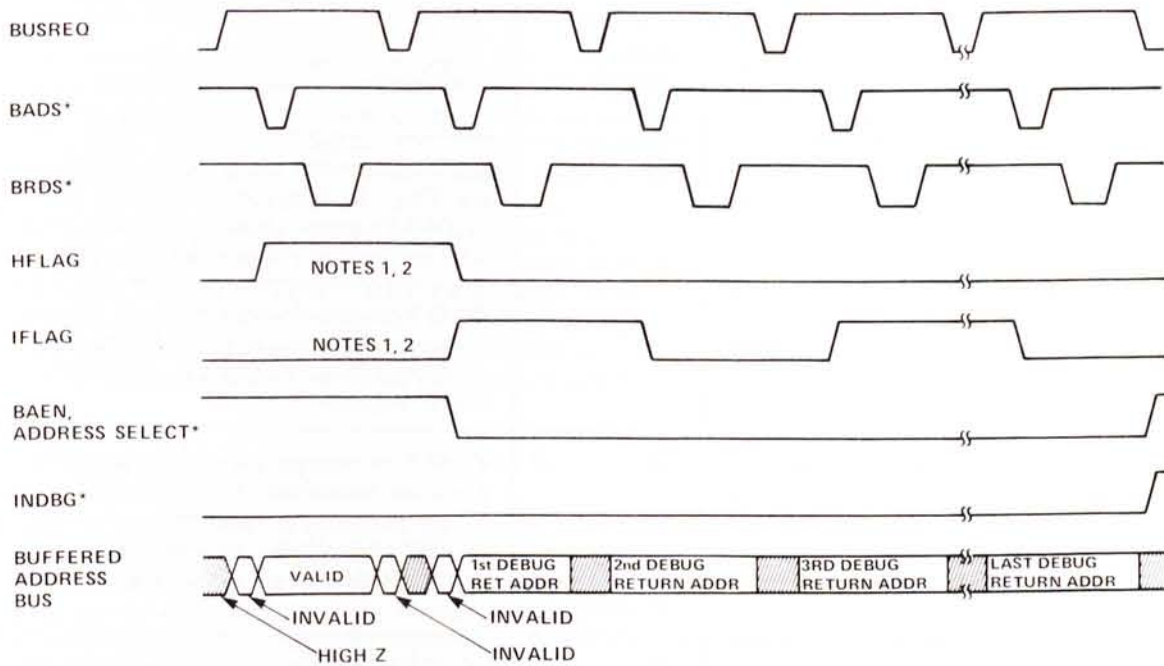
1. The values saved for the extension, P1 and P2 registers are restored to the SC/MP microprocessor.
2. The value saved for the program counter is loaded into the P3 register for temporary storage, then it is decremented by ten to compensate for the automatic incrementing that will occur during the hardware-forced DEBUG Return Routine.
3. The value saved for the status register is loaded into the accumulator for temporary storage and a Halt Instruction is executed to transfer program control to the hardware-forced DEBUG Return Routine.

#### 5.1.4 Hardware-Forced DEBUG Return Routine

The DEBUG Return Routine is a hardware-controlled subroutine that is executed by the Mode Control Logic in response to the Halt Instruction that occurs at the end of the DEBUG firmware Exit Routine. As shown in figure 5-3, execution of the DEBUG Return Routine is similar to execution of the DEBUG Save Routine described previously; that is, the BAEN and DEBUG Address Enable\* outputs of the Mode Control Logic are held low while the BUSREQ output of the SC/MP CPU Application Card is processed to force a series of DEBUG Return Routine Addresses onto the Buffered Address Bus (refer to table 5-2). A general summary of the functions provided by the DEBUG Return Routine is as follows:

1. The decremented program counter value that was temporarily stored in pointer register 3 is returned to the program counter so that it will be automatically incremented to the appropriate RUN Mode starting value during further execution of the hardware-forced DEBUG Return Routine.
2. The contents of the accumulator are copied into the status register to restore the saved value to the status register.
3. Pointer Register 3 and the accumulator are loaded from LCDS memory to complete restoration of SC/MP microprocessor internal status.





- NOTE: 1. The IFLAG and HFLAG outputs of the SC/MP CPU Application Card are shown only as general timing references; they are not used by the LCDS.
2. The Mode Control Logic detects an IFETCH data I/O cycle and execution of a Halt Instruction by, respectively, processing the DI5 and DI7 data-bit inputs under control of the BADS\* strobe.

NS10707

Figure 5-3. DEBUG Return Routine Timing

After the last hardware-forced DEBUG Return Routine address is output, the BAEN and DEBUG Address Enable\* outputs of the Mode Control Logic are reset high on the trailing edge of the BUSREQ signal to reenable the address output of the SC/MP CPU Application Card. Thus, the SC/MP CPU Application Card outputs the restored program counter address on the leading edge of the next BUSREQ signal and executes an IFETCH data input/output cycle to read the first RUN Mode instruction opcode from the referenced memory location. Execution of the user-entered applications program then continues normally until the RUN Mode is subsequently terminated as described in the DEBUG Save Routine description.

#### 5.1.5 TTY Interface Circuit

The TTY Interface Circuit provides the signal level conversion functions required for bidirectional transmission of data between the SC/MP CPU Application Card and the optional TTY. As described in chapter 3, Operation, the DEBUG firmware always functions such that the SC/MP CPU Application Card transmits data to the optional TTY via the BFLG0 output and receives data from the TTY via the BSENSE input. When data are to be received from the optional TTY paper-tape reader, the BFLG1 output of the SC/MP CPU Application Card is set high for the duration of the data transmission to enable the TTY paper-tape reader relay; at all other times the BFLG1 signal is set low by the DEBUG firmware to disable the paper-tape reader relay and thereby ensure that the paper-tape reader will not affect reception of data from the TTY keyboard.

Table 5-2. Hardware-Forced DEBUG Return Routine Address Functions

Instruction	DEBUG Return Address	Data I/O Cycle	Buffered Read/Write Data	Comments
XPPC P3	1. 7808	Read (IFETCH)	3F	SC/MP microprocessor reads in opcode for XPPC P3 Instruction; contents of SC/MP Program Counter are exchanged with contents of Pointer Register 3.
Copy AC to Status (CAS)	2. 7807	Read (IFETCH)	07	SC/MP microprocessor reads in opcode for Copy AC to Status (CAS) Instruction; contents of accumulator are copied into Status Register.
Load Immediate (LI)	3. 7801	Read (IFETCH)	C4	SC/MP microprocessor reads in opcode for Load Immediate (LI) Instruction.
	4. 77FC	Read	Value Saved for P3-low	SC/MP Accumulator is loaded with Pointer Register 3 low-order byte.
XPAL P3	5. 7803	Read (IFETCH)	33	SC/MP microprocessor reads in opcode for XPAL P3 Instruction; contents of SC/MP Accumulator are exchanged with low-order byte of Pointer Register 3.
Load Immediate (LI)	6. 7801	Read (IFETCH)	C4	SC/MP microprocessor reads in opcode for Load Immediate (LI) Instruction.
	7. 77FB	Read	Value Saved for P3 high	SC/MP Accumulator is loaded with Pointer Register 3 high-order byte.
XPAH P3	8. 7805	Read (IFETCH)	37	SC/MP microprocessor reads in opcode for XPAH P3 Instruction; contents of SC/MP Accumulator are exchanged with high-order byte of Pointer Register 3.
Load Immediate (LI)	9. 7801	Read (IFETCH)	C4	SC/MP microprocessor reads in opcode for Load Immediate (LI) Instruction.
	10. 77FD	Read	(AC)	SC/MP Accumulator is loaded with saved value.

## 5.2 DETAILED DESCRIPTION

A detailed block diagram of the LCDS is provided in figure 5-4 and equivalent-level circuit descriptions follow.



## 5.2.1 Mode Control Logic

The Mode Control Logic (see figure 5-4, sheet 1) provides the control functions required to save the internal status of the SC/MP microprocessor when the DEBUG Mode is selected, and to restore the saved status when the RUN Mode is selected. Selection of either mode causes the Halt output of the RUN/DEBUG Mode Enable Logic to go high, thereby enabling the Sync Latch to be clocked reset when the DI5 data-bit input also goes high during BADS time of the IFETCH input/output cycle that occurs after the current instruction is completed. (Clocking of the Sync Latch occurs on the leading edge of the BADS\* strobe associated with the high DI5 input.) When the Sync Latch is clocked reset, the low-going Q output sets the DEBUG Save/Return Routine Enable Latch and resets the DEBUG Status Latch to enable execution of the DEBUG Save or Return Routine as appropriate. The Halt output of the RUN/DEBUG Mode Enable Logic is then reset by the resulting outputs of the DEBUG latches and the Sync Latch is clocked set on the leading edge of the next BADS\* strobe to permit further operation of the DEBUG latches to be controlled by the Address Generator. A timing diagram of Mode Control Logic operation for the DEBUG Save and Return Routines is provided in figure 5-5.

### 5.2.1.1 DEBUG Mode Selection & Save Routines

Whenever the DEBUG Mode is selected, the Address Generator will be in the reset state. When the DEBUG Address Enable signal goes high, therefore, the resulting Clock input to the Address Generator causes the first hardware-forced DEBUG Save Routine address to be output. The DEBUG Address Enable signal then stays high for the duration of the Save Routine and the Address Generator outputs the remainder of the Save Routine addresses in response to the subsequent BUSREQ outputs of the SC/MP CPU Application Card.

While the DEBUG Address Enable signal is high, the BAEN and DEBUG Address Enable\* signals remain low (see figure 5-4, sheet 1). The address output of the SC/MP CPU Application Card, therefore, is held in the high-impedance state for the duration of the hardware-forced DEBUG Save Routine, and the BUSREQ\* inputs to the Address Buffers enable the Save Routine addresses to be driven onto the Buffered and LCDS Address Buses. When the last Save Routine Address is output, the Y7 control bit goes high, thereby enabling the DEBUG Save/Return Routine Enable latch to be clocked reset on the trailing edge of the associated BUSREQ\* signal. After the DEBUG Save/Return Routine Enable Latch is reset, the low DEBUG Address Enable signal inhibits further clocking of the Address Generator, the high BAEN signal reenables the address output of the SC/MP CPU Application Card, and the high DEBUG Address Enable\* signal enables the CPU card addresses to be driven onto the LCDS Address Bus. Thus, the Address Generator is "frozen" at the last hardware-forced DEBUG Save Routine address, and the DEBUG firmware program is executed continuously until the RUN Mode is selected, *scanning keyboard, display, awaiting commands.*

### 5.2.1.2 RUN Mode Selection & Return Routine

After the DEBUG Status Latch is clocked reset at the start of the DEBUG Save Routine, it remains reset until the RUN Mode is subsequently selected. While the DEBUG firmware program is being executed, therefore, a low INDBG\* status signal is applied continuously to the RUN/DEBUG Mode Enable Logic to inhibit the circuit from operating except in response to a high DI7 (HFLAG) input. Thus, during execution of the DEBUG firmware program, the RUN/DEBUG Mode Enable Logic functions only to detect execution of the Halt instruction that occurs after the RUN Mode is selected, *to reload registers of the saved values.*

When the DI7 (HFLAG) input goes high during execution of the DEBUG firmware Halt instruction, the RUN/DEBUG Mode Enable Logic outputs a high Halt signal and clocking of the Sync and DEBUG Save/Return Routine Enable Latches occurs as described in 5.2.1. The ensuing outputs of the DEBUG Save/Return Routine Enable Latch then reset the Halt signal and enable execution of the hardware-forced DEBUG Return Routine as described previously for the DEBUG Save Routine (that is, the high DEBUG Address Enable signal enables clocking of the Address Generator, the low BAEN signal disables the address outputs of the SC/MP CPU Application Card, and the low DEBUG Address Enable\* signal enables the Address Buffers to drive the Return Routine addresses onto the Buffered and LCDS Address Buses while the BUSREQ\* input is low). When the last Return Routine address is output, the Y7 output of the Address Generator goes high and the DEBUG Save/Return Routine Enable Latch is clocked reset on the trailing edge of the associated BUSREQ\* signal. At the same time, the Address Generator provides a Clear pulse that resets the internal Address Counter and also sets the DEBUG Status Latch. Thus, the Address Generator is returned to the reset state, then RUN Mode operation is initiated at the restored program counter address on the leading edge of the next BUSREQ signal; *and the user program takes control of the CPU.*





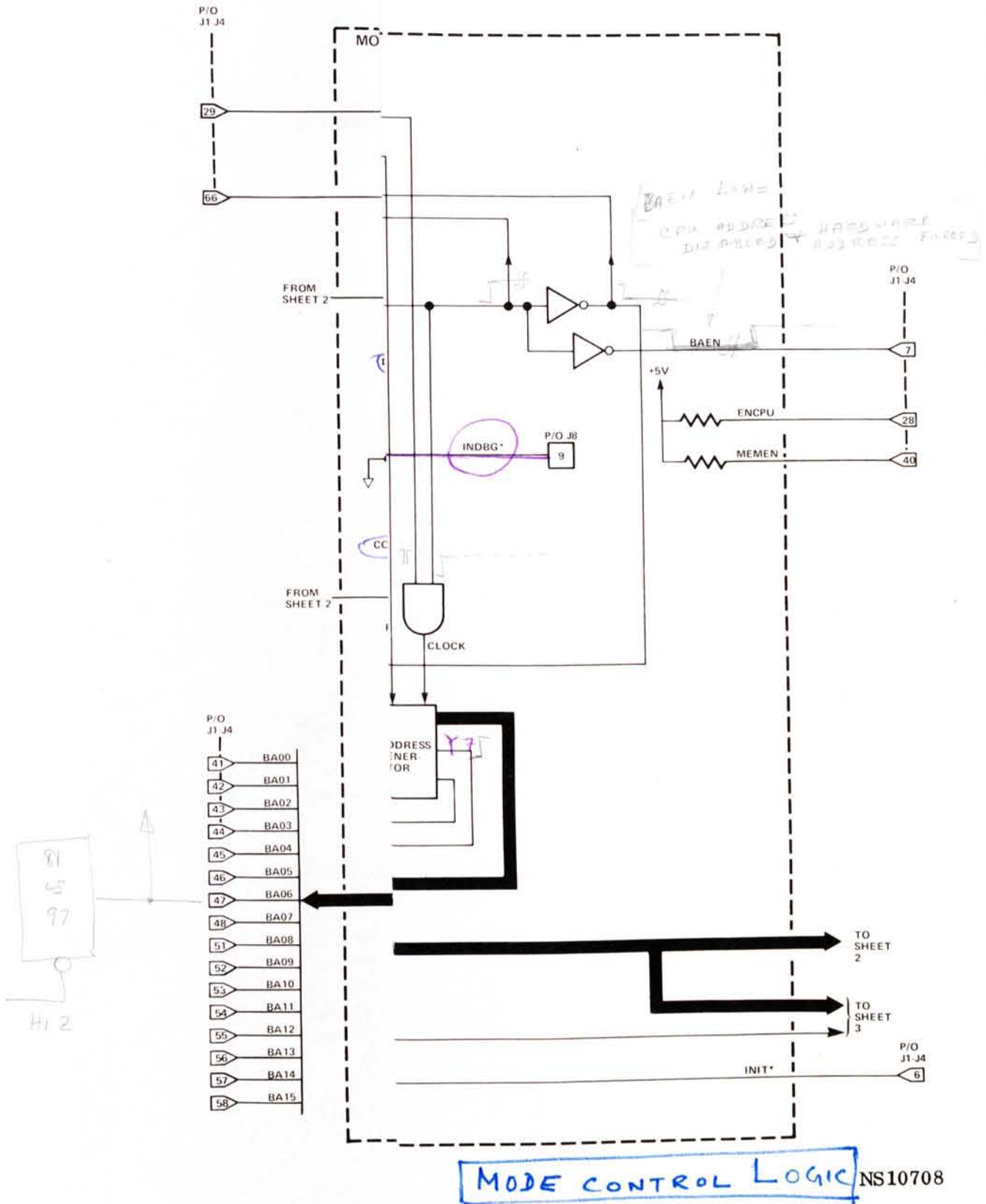
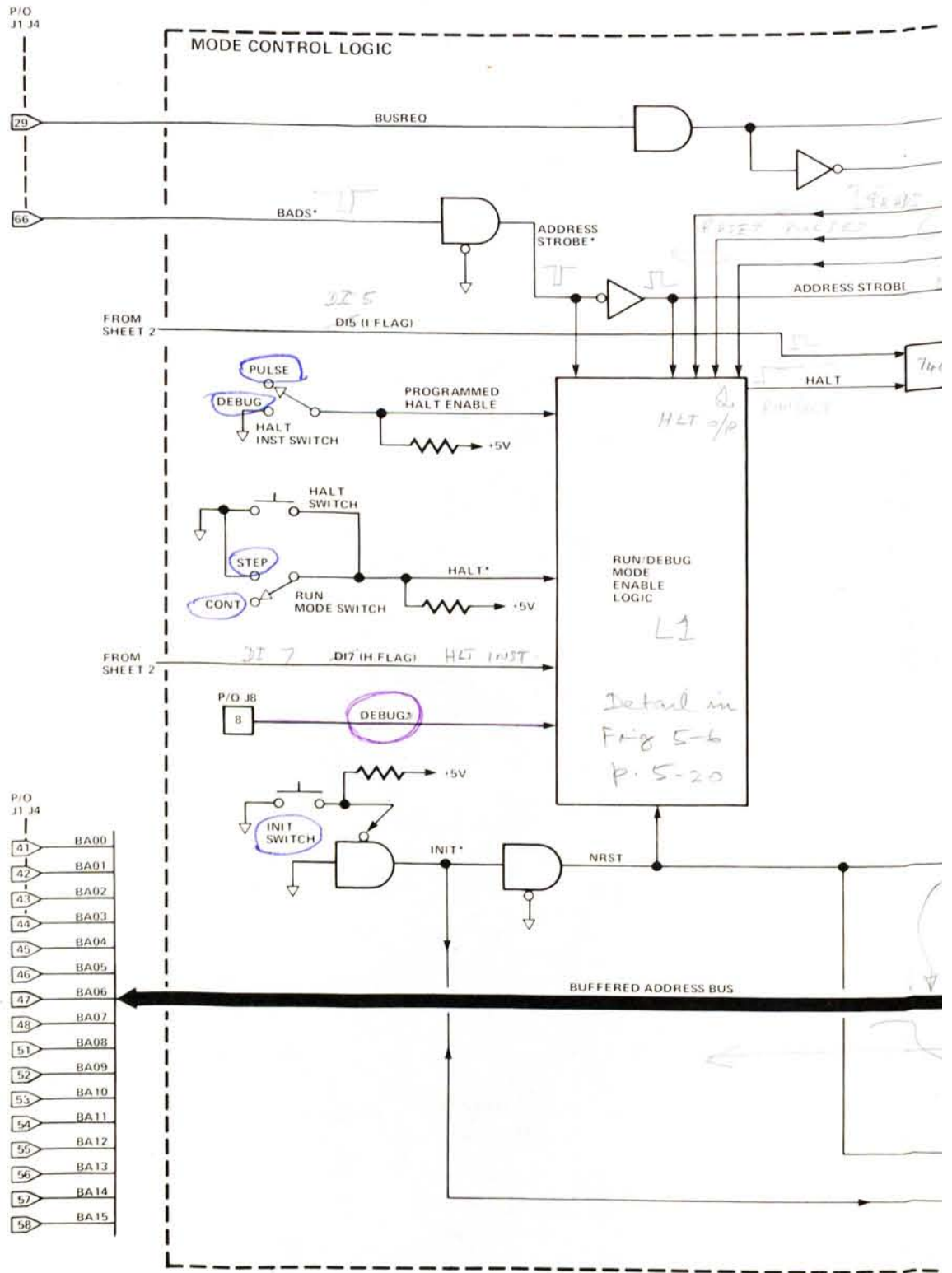


Figure 5-4. LCDS Block Diagram (Sheet 1 of 3)

①





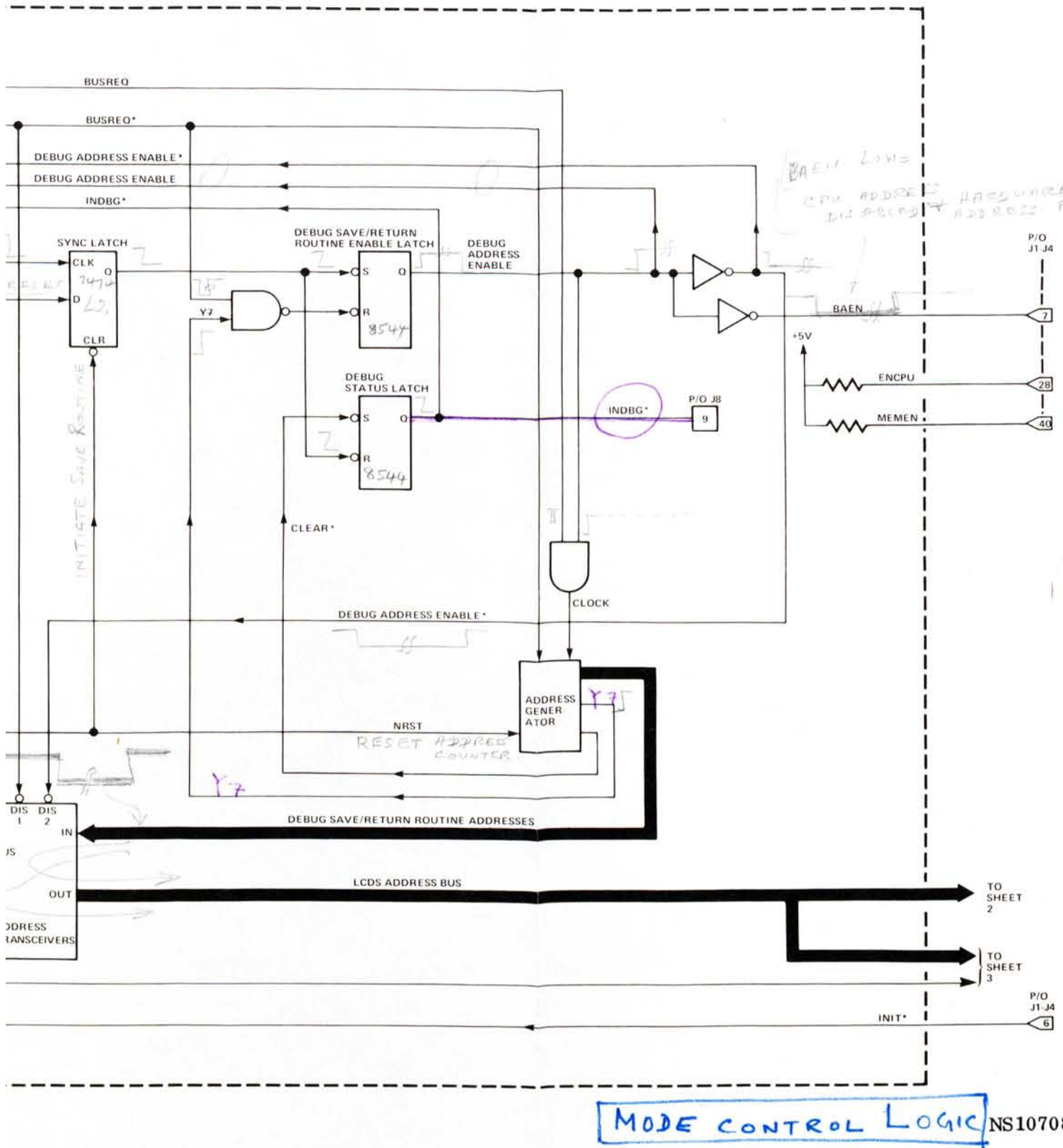
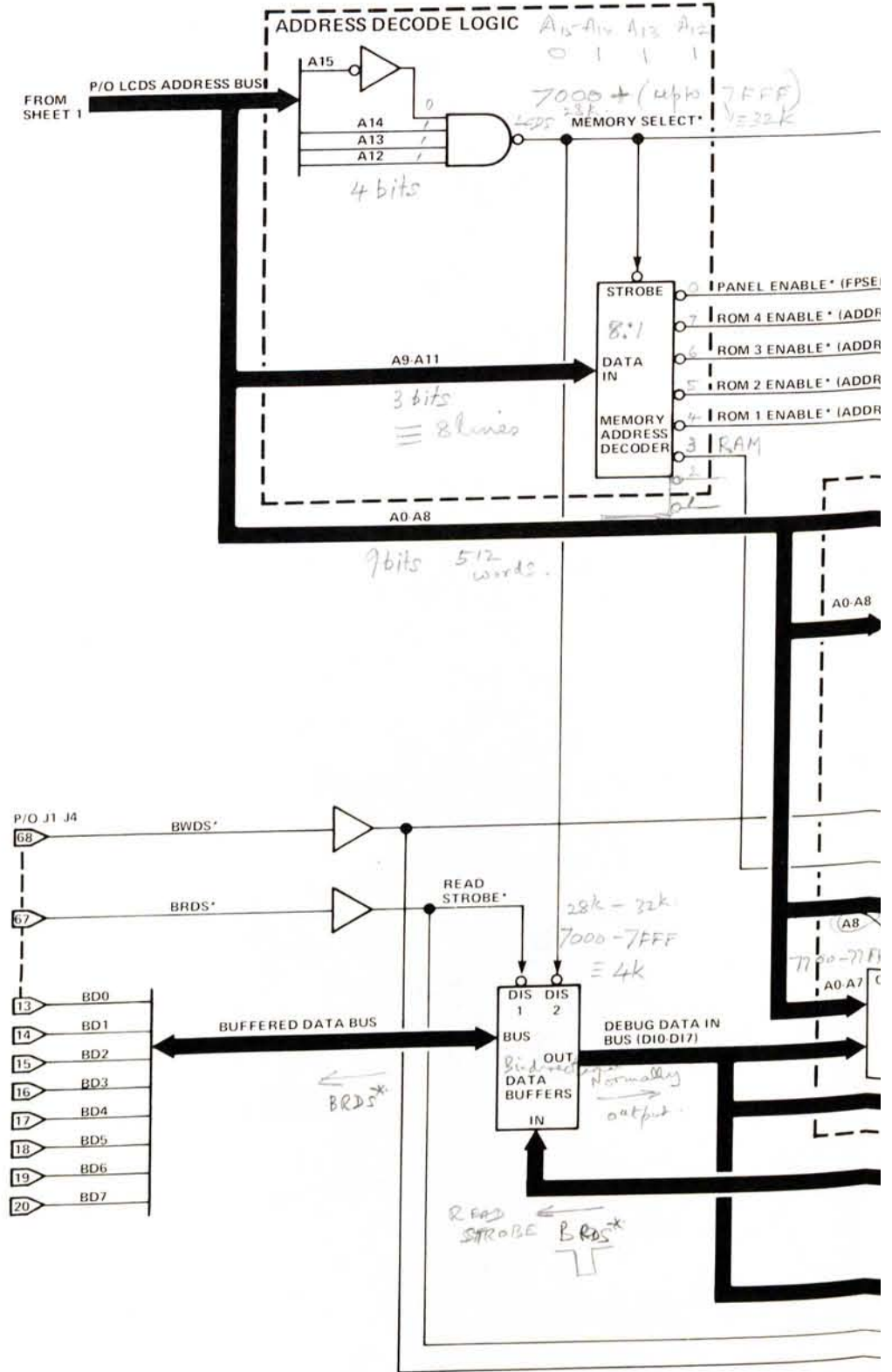
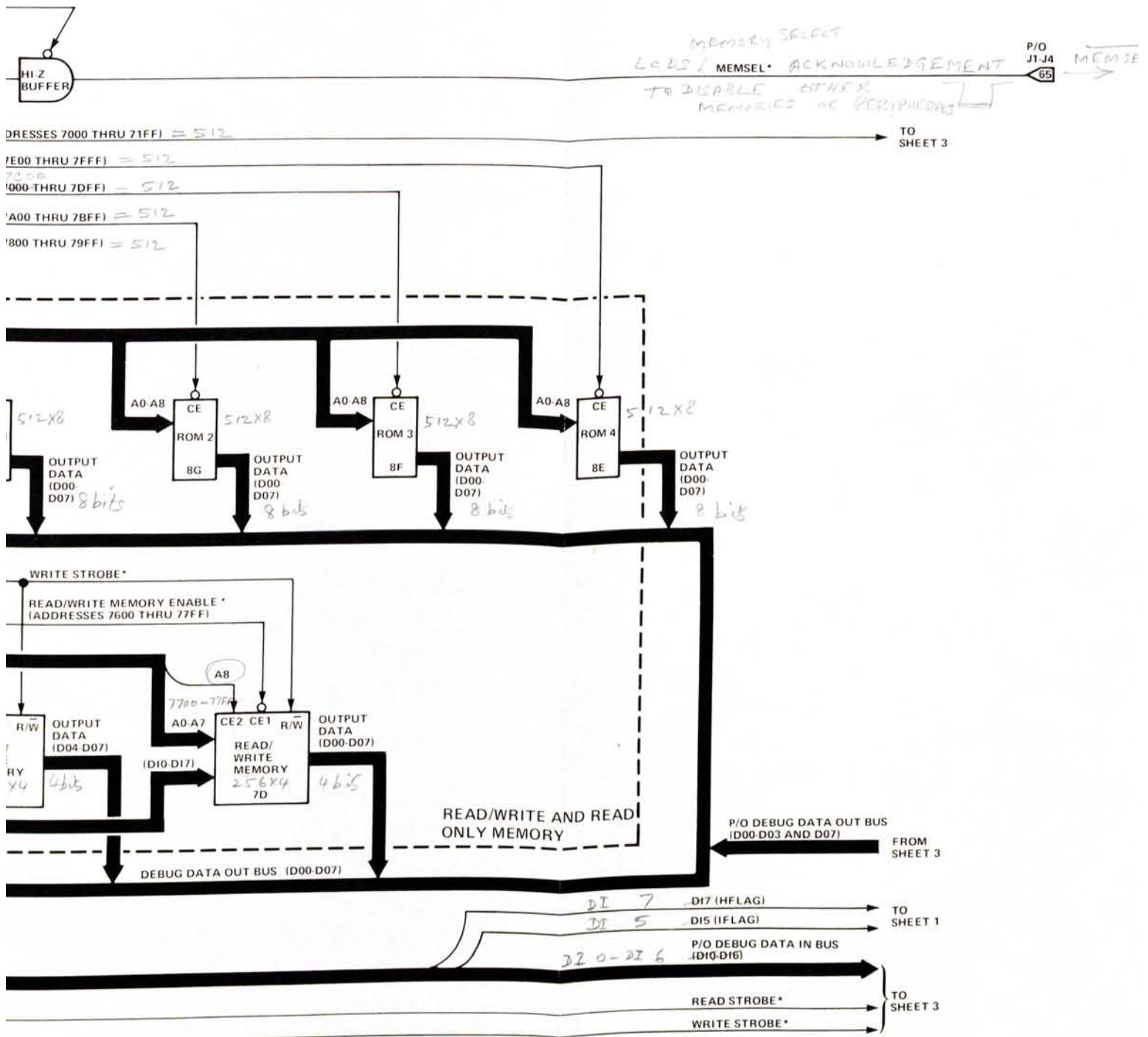


Figure 5-4. LCDS Block Diagram (Sheet 1 of 3)

1





NS10709

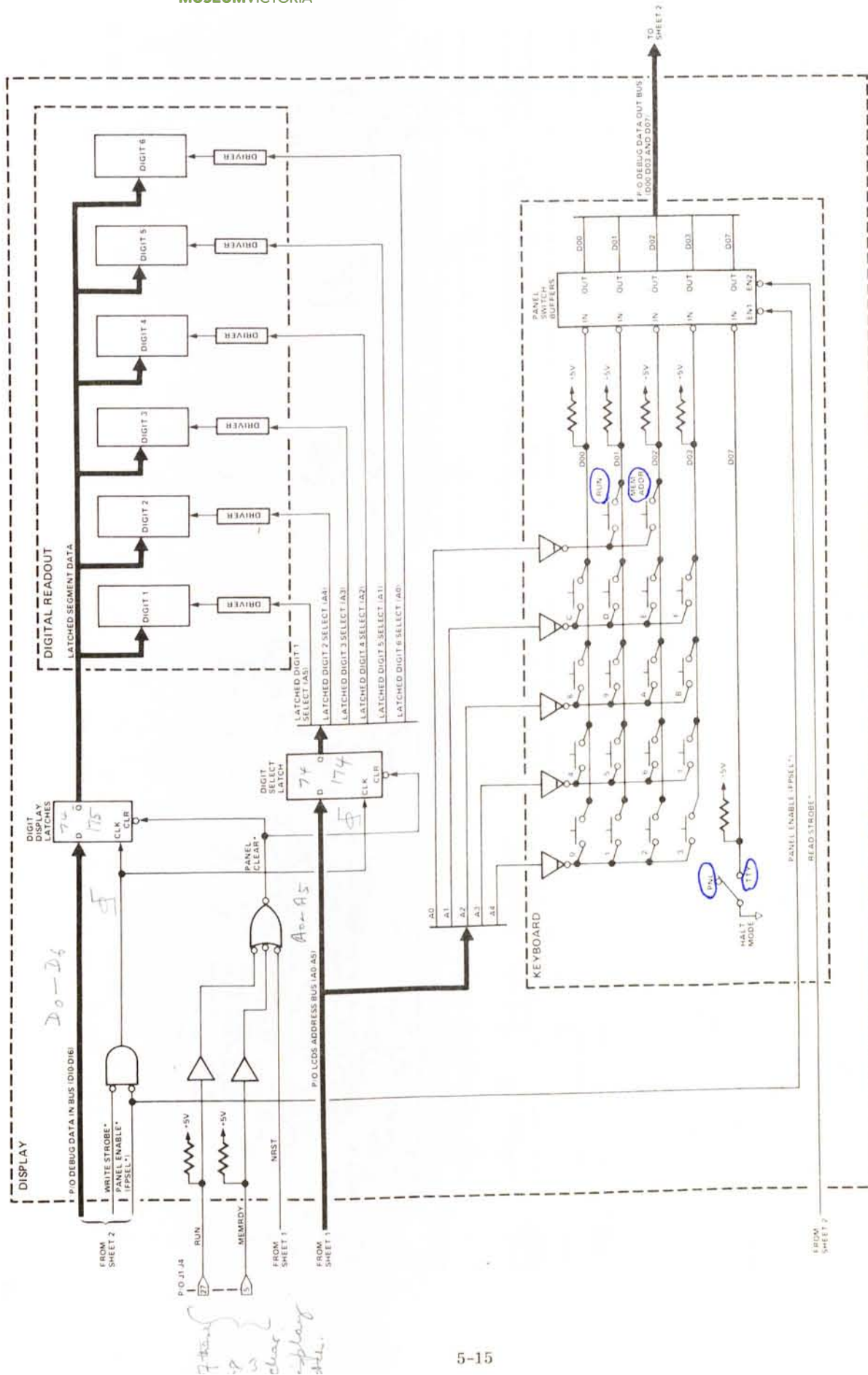
Figure 5-4. LCDS Block Diagram (Sheet 2 of 3)

2

5-13/5-14



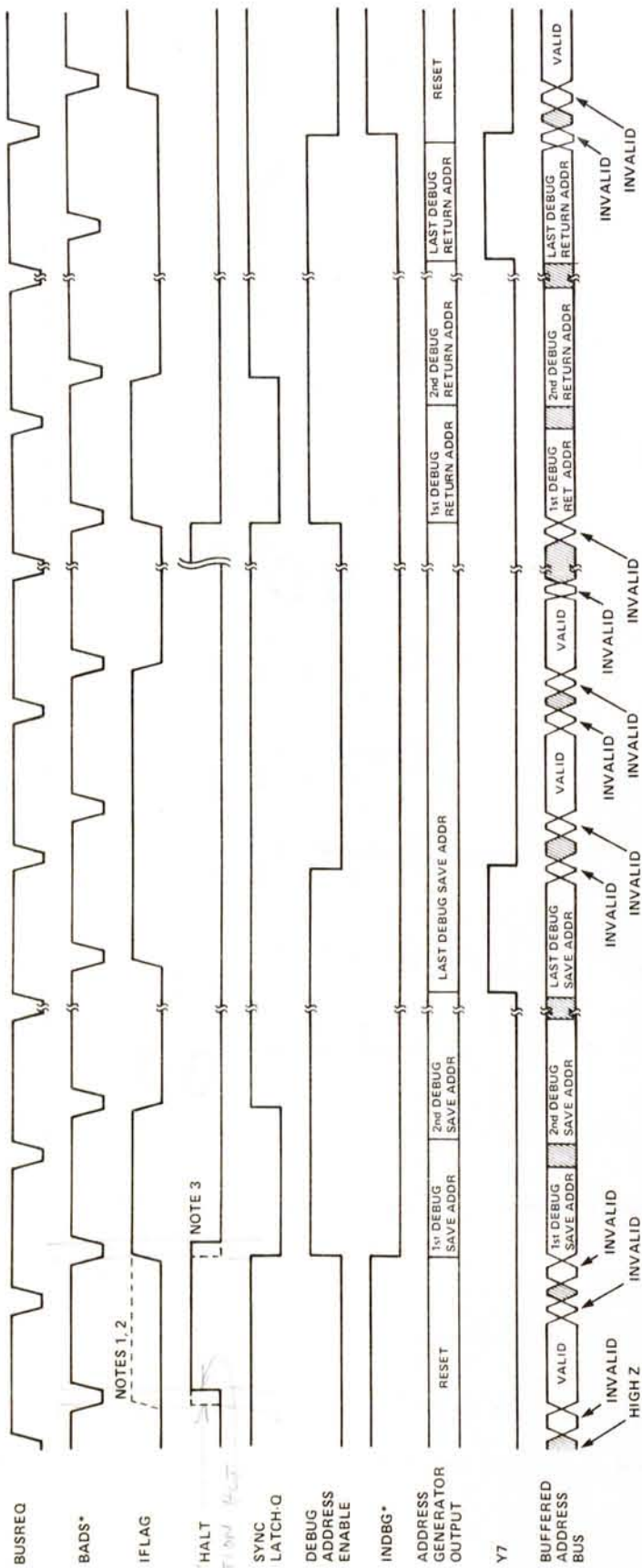




NS10710

3

Figure 5-4. LCDS Detailed Block Diagram (Sheet 3 of 3)



- NOTES:
1. The IFLAG output of the SC/MP CPU Application Card is shown only as a general timing reference; it is not used by the LCDS.
  2. The IFLAG output of the SC/MP CPU Application Card is set high whenever an IFETCH data I/O cycle is enabled (that is, signal byte instruction or first byte of double-byte instruction is being fetched from memory).
  3. When the RUN Mode is selected or the DEBUG Mode is selected via execution of a Halt instruction, the Halt signal is set high and returned low on the leading edge of the BADS\* strobe; when the DEBUG Mode is selected by externally driving the User DEBUG\* input low, pressing the Halt switch, or setting the RUN MODE switch to STEP, the Halt signal is set high and returned low on the trailing edge of the BADS\* strobe.

NS10711

Figure 5-5. Mode Control Logic Timing for DEBUG Save and Restore Routines



RAM  
7700  
  
F  
R  
E  
E  
116<sub>10</sub>  
  
7774  
256  
777F  
  
116  
12  
88  
40  
256  
  
F  
R  
E  
E  
88<sub>10</sub>  
  
77D8  
  
40<sub>10</sub>  
  
77FF

### 5.2.1.3 Initialization

Whenever the INIT\* signal is set low to initialize the SC/MP microprocessor, it also serves as an asynchronous reset input to the Sync Latch and the Address Generator. Thus, when the INIT\* signal is returned high, the DEBUG Save Routine is executed as described previously.

### 5.2.2 Address Decode Logic

The Address Decode Logic (see figure 5-4, sheet 2) continuously monitors the A12 through A15 address-bit input to provide a Memory Select\* (MEMSEL\*) output when the LCDS memory, keyboard, or digital readout is addressed (addresses 7000 through 7FFF). While the Memory Select\* signal is low, the Memory Address Decoder is enabled and address bits A9 through A11 are further decoded to provide an Enable output to the appropriate circuit.

### 5.2.3 Read-Only Memory

ROM

The LCDS Read-Only Memory (see figure 5-4, sheet 2) consists of four 512-by-8 bit ROM chips that function under control of their individual ROM Enable inputs. Thus, when the Read-Only Memory is addressed, the appropriate ROM Enable signal goes low and the selected ROM outputs data from the memory location specified by address bits A0 through A8.

### 5.2.4 Read/Write Memory

RAM

The LCDS Read/Write Memory (see figure 5-4, sheet 2) consists of two 256-by-4-bit read/write memory chips that are addressed in parallel to provide an 8-bit input/output data format. When the Read/Write Memory Enable signal is low and address bit A8 is high, the memory chips are enabled and reading and writing of data occur under control of the Write Strobe\* input. For a read data input/output cycle, the Write Strobe\* remains high and data is output from the memory location specified by address bits A0 through A7. For a write data input/output cycle, data are written into the memory location specified by address bits A0 through A7 on the trailing edge of the Write Strobe\*. The address range for the read/write memory is X'7700 through X'77FF; the locations used by the DEBUG firmware are X'7774 through X'777F and X'77D8 through X'77FF.

12 locations RAM 40 locations

### 5.2.5 Data Buffers

The Data Buffers are bidirectional buffers that function under control of the MEMSEL\* and Read Strobe\* inputs. When both the MEMSEL\* and Read Strobe\* inputs are low, the Data Buffers are configured to drive data from the DEBUG Data Out Bus onto the Buffered Data Bus. At all times, the Data Buffers are continually driving data from the Buffered Data Bus onto the DEBUG Data In Bus.

ie. Normally data output from CPU. During Read data input to CPU.

### 5.2.6 Digital Readout

The Digital Readout (see figure 5-4, sheet 3) consists of six hexadecimal indicators that can be addressed individually or in parallel. (A listing of the addresses assigned to the Digital Readout is provided in 3.6.2 along with a hexadecimal value/data code cross-reference list.) Addressing of the Digital Readout occurs when the Panel Enable\* and Write Strobe\* inputs go low (during a Store at Address X'7000 through X'71FF Instruction), thereby causing address bits A0 through A5 to be clocked into the Digit Select Latches, and data bits D10 through D16 to be clocked into the Digit Display Latches. The latched address and data then are applied continuously to the Digital Readout until new information is clocked into the latches or the latches are reset via the Panel Clear\* signal. Resetting of the latches via the Panel Clear\* signal occurs under one of the following conditions:

1. The INIT\* signal is set low to initialize the SC/MP microprocessor, thereby causing a low NRST input to be applied to the Digital Readout.



2. The RUN input to the SC/MP CPU Application Card is externally driven low to suspend SC/MP microprocessor operation.
3. The MEMRDY input to the SC/MP CPU Application Card is externally driven low to extend a read or write data input/output cycle.

### 5.2.7 Keyboard

The keyboard consists of an 18-pushbutton matrix that is enabled under program control via the A0 through A4 address-bit inputs. Thus, when any of these bits goes high, a switch closure in the corresponding row causes a low output to be applied to the Panel Switch Buffers. The Panel Switch Buffers, however, remain disabled until both the Panel Enable\* and Read Strobe\* inputs go low during the read data input/output cycle of a Load from Address 7000-71FF Instruction. When both the Panel Enable\* and Read Strobe\* signals are low, the Panel Switch Buffers then invert the data present at the IN terminals and drive the inverted data onto the DEBUG Data Out Bus. For a description of how the Keyboard pushbutton matrix may be employed by a user's application program, refer to 3.6.3. *page 3-53*

### 5.2.8 HALT MODE Switch

In addition to the inputs received from the Keyboard pushbutton matrix, the Panel Switch Buffers also receive a continuous input from the HALT MODE switch. This input is periodically checked by the DEBUG firmware program so that DEBUG command capability will be provided at the selected location (Panel or optional TTY) as described in chapter 3, Operation.

## 5.3 CIRCUIT DESCRIPTION

Operation of the RUN/DEBUG Mode Enable Logic and the Address Generator circuits is described in the paragraphs that follow.

### 5.3.1 RUN/DEBUG Mode Enable Logic

The function of the RUN/DEBUG Mode Enable Logic (see figure 5-6) is to indicate a valid RUN or DEBUG Mode selection by providing a high Halt output to the Sync Latch. Thus, when power is first applied to the LCDS, flip-flop 8B is reset by the low NRST signal during the power-up initialization cycle, and the resultant Halt output of gate 6B3 ensures that the LCDS powers up in the DEBUG Mode. Termination of the Halt output then occurs on the trailing edge of the first Address Strobe\* (BADS\*) that is generated by the SC/MP CPU Application Card after the power-up initialization cycle is completed. (When the first Address Strobe\* input is received, the INDBG\* status signal already will have been set low, thereby causing a high level to be present at the D input of flip-flop 8B; flip-flop 8B, therefore, will be clocked to the set state on the low-to-high transition of the Address Strobe\*.)

After the power-up initialization cycle is completed, latch 9B functions to detect selection of the RUN or DEBUG Mode via execution of a Halt Instruction, and flip-flop 8B functions to detect selection of the DEBUG Mode via the INIT, HALT or RUN MODE switches, or via the external User DEBUG\* Input. The specific conditions under which flip-flops 8B and 9B are set and reset are described below.

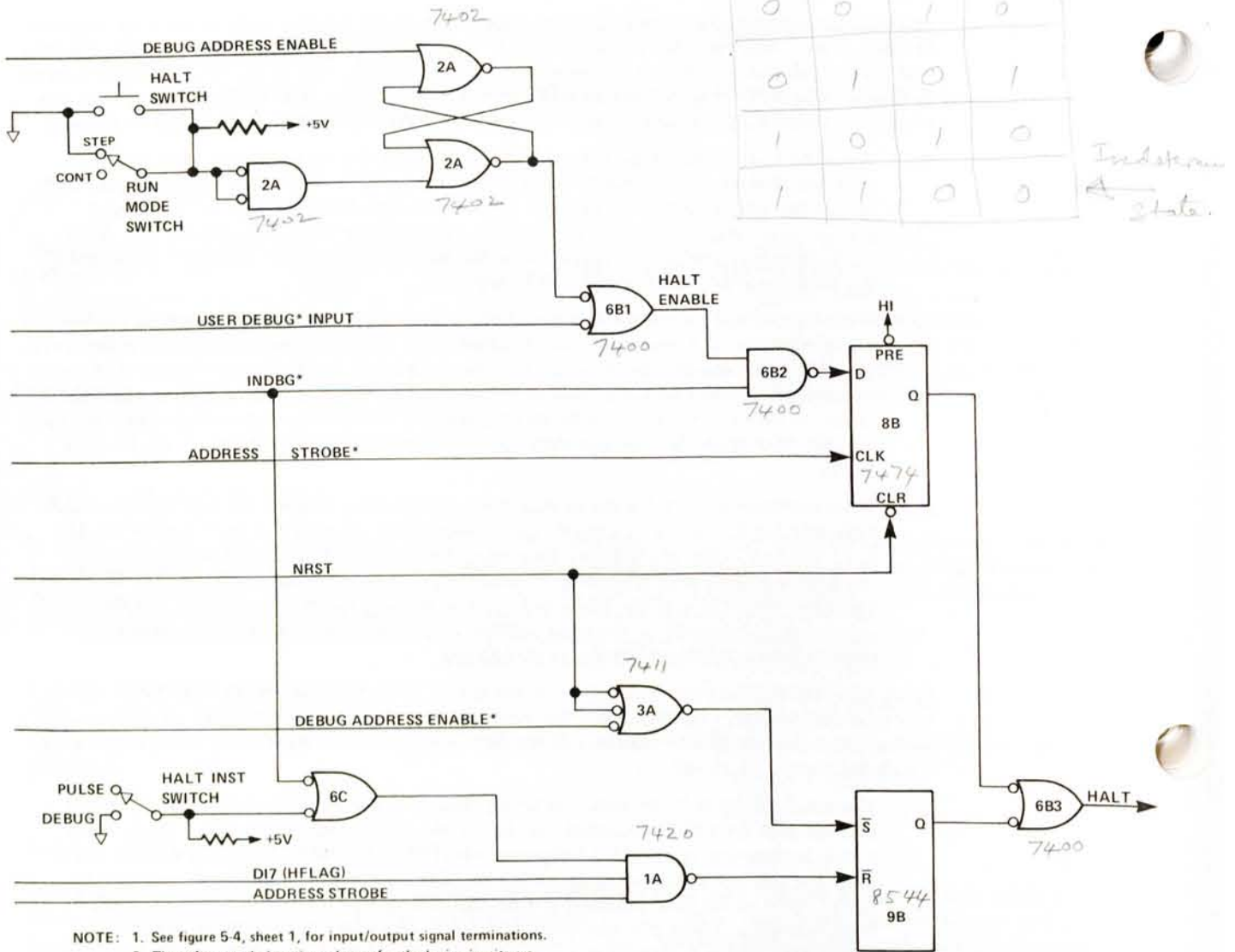
1. Flip-Flop 8B. After the power-up initialization cycle is completed, flip-flop 8B is clocked to the reset state each time that the DEBUG Mode is selected by pressing the HALT switch, externally driving the User DEBUG\* Input low, or setting the RUN MODE switch to STEP. Any of these actions causes a high Halt Enable output to be provided by gate 6B1 and, if the INDBG\* status signal is also high, a low output to be provided by gate 6B2. (Gating of the Halt Enable and INDBG\* signals together ensures that any of the specified actions will not affect execution of the DEBUG firmware program.) When the output of gate 6B2 goes low, flip-flop 8B is clocked to the reset state on the next low-to-high transition of the Address Strobe\*. After flip-flop 8B is clocked to the reset state, the INDBG\* status signal is set low in response to the Halt output of gate 6B3 (refer to 5.2.1), then the Halt



output is terminated when flip-flop 8B is clocked set on the trailing edge of the next Address Strobe\* input. Note that the manner in which flip-flop 8B is returned to the set state ensures that the Halt output will always be terminated on the trailing edge of the first Address Strobe\* received after INDBG\* goes low, regardless of the state of the Halt Enable signal. For the conditions specified previously, the state of the Halt Enable signal is controlled as follows:

- a. When the Halt Enable signal is set high by pressing the HALT switch, the DEBUG Address Enable input typically will go high (in response to the Halt output of gate 6B3) before the HALT switch is released. Thus, the Halt Enable signal normally will remain high until the HALT switch is released to allow resetting of latch 2A. With latch 2A reset, continuous execution of the application program will be enabled when the RUN Mode is subsequently reselected.
  - b. When the RUN Mode switch is set to STEP, latch 2A is held set and the Halt Enable signal remains high continuously. Whenever the RUN Mode is selected, therefore, flip-flop 8B is clocked reset on the trailing edge of the first Address Strobe\* that is received after the INDBG\* status signal goes high to indicate a resumption of RUN Mode operation. Thus, one application program instruction is executed each time that the RUN Mode is selected; then, the LCDS automatically returns to the DEBUG Mode.
  - c. When the User DEBUG\* Input is externally driven low, it holds the Halt Enable signal high only for the length of time that it remains low. In order to permit flip-flop 8B to be clocked to the reset state, therefore, the User DEBUG\* Input must remain low until a low-to-high transition of the Address Strobe\* occurs. Similarly, if continuous execution of the applications program is desired when the RUN Mode is subsequently reselected, the User DEBUG\* Input must be returned high before the ensuing DEBUG Return Routine is completed.
2. Latch 9B. During the power-up initialization cycle, latch 9B is set by the low NRST input received via gate 3A. Latch 9B then remains set until the DEBUG firmware executes a Halt Instruction following initial selection of the RUN Mode. Operation of latch 9B in response to a Halt Instruction is as follows:
- a. Whenever the DEBUG firmware is being executed, the INDBG\* status signal will be low and gate 1A will be partially enabled by the high output of gate 6C. Thus, when a Halt Instruction is executed following selection of the RUN Mode, latch 9B is reset when the DI7 (HFLAG) and Address Strobe inputs go high during the second input/output cycle of the Halt Instruction (see figure 4 of SC/MP CPU Application Card Specification). Latch 9B then remains reset until the DEBUG Address Enable\* signal goes low in response to the resultant Halt output of gate 6B3. When the DEBUG Address Enable\* signal goes low, latch 9B is set by the low output of gate 3A and the Halt output is thereby terminated.
  - b. While the RUN Mode is enabled, the setting of the HALT INST switch determines whether or not execution of an applications program Halt Instruction will cause termination of the RUN Mode and entry into the DEBUG Mode. When the HALT INST switch is set to PULSE, gate 1A is disabled by the low output of gate 6C during RUN Mode operation (INDBG\* high), thereby preventing latch 9B from being reset by execution of a Halt Instruction. When the HALT INST switch is set to DEBUG, however, gate 1A remains continuously enabled for both RUN and DEBUG Mode operation. Execution of an applications program Halt Instruction, therefore, causes latch 9B to generate a Halt output (via gate 6B3) as described for 'a' above.





NOTE: 1. See figure 5-4, sheet 1, for input/output signal terminations.  
 2. The reference designations shown for the logic circuits are subject to change at any time; for parts identification purposes, refer to the Engineering Schematic and Assembly Diagrams received with the LCDS.

NS10712

Figure 5-6. RUN/DEBUG Mode Enable Logic Circuit Diagram

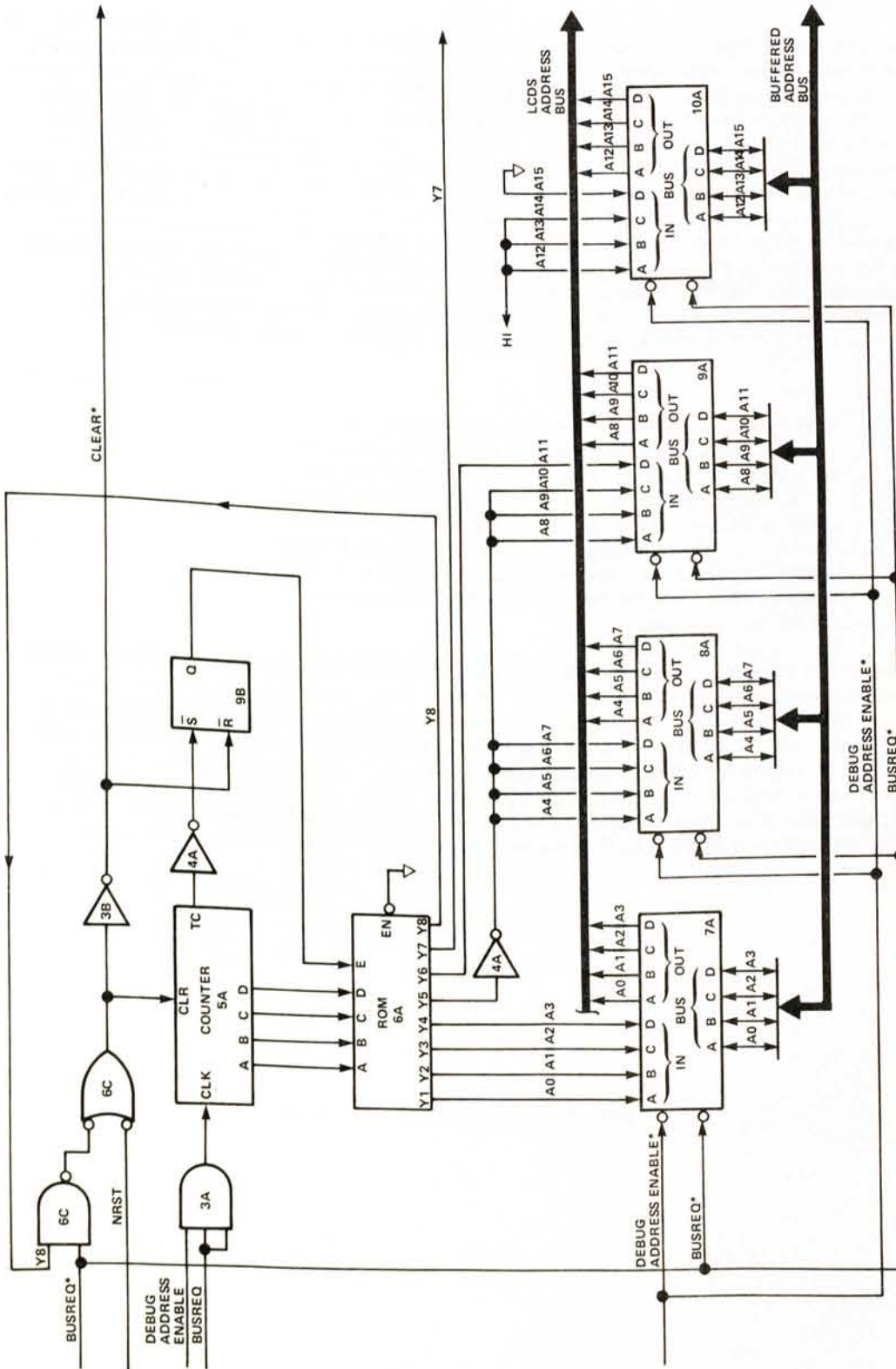
### 5.3.2 DEBUG Address Generator

The DEBUG Address Generator (see figure 5-7) consists of a 32-by-8-bit ROM that contains the hardware-forced DEBUG Save and Return Routine addresses, a binary counter, and all of the associated gating circuitry required to enable the counter to sequence the ROM through the DEBUG Save and Return Routines. When power is first applied to the LCDS, counter 5A and counter-extension latch 9B are both reset by the low NRST input, thereby causing the ROM to output reset address 00. At the same time, the DEBUG Address Enable input is also set high so that counting will be initiated when the power-up initialization cycle is completed. Upon completion of the power-up initialization cycle, the BUSREQ outputs of the SC/MP CPU Application Card then cause the counter and ROM to sequence through the DEBUG Save Routine as listed in table 5-3. (Note that the counting sequence is straightforward except that address 31 is output rather than address 15 when counter-extension latch 9B is reset by the terminal count (TC) output of counter 5A.) When the last DEBUG Save Routine address is output, ROM bit Y7 goes high, the DEBUG Address Enable signal is set low on the trailing edge of the BUSREQ input, and further counting is inhibited. Thus, after the DEBUG Save Routine is completed, the output of the counter causes the ROM to continuously output the last DEBUG Save Routine address.

#### NOTE

During the DEBUG Save and Return Routines, the DEBUG Address Enable\* input to the Address Buffers is low and the Address Buffers function under control of the BUSREQ\* inputs to drive the DEBUG Save/Return Routine addresses onto the Buffered and LCDS Address Buses. At all other times, the DEBUG Address Enable\* signal is high and the Address Buffers continuously drive address data from the Buffered Address Bus onto the LCDS Address Bus.

After the DEBUG Save Routine is completed, counting remains inhibited until the DEBUG Address Enable signal is returned high following selection of the RUN Mode (refer to 2.b of 5.3.1). When this occurs, the counter is incremented by one and the ROM outputs the first DEBUG Return Routine address. Counting then continues in response to the BUSREQ inputs until the ROM outputs the last DEBUG Return Routine address, thereby setting bits Y7 and Y8 high. After bits Y7 and Y8 go high, termination of the DEBUG Address Enable signal and re-setting of the counter occur on the trailing edge of the BUSREQ signal. Thus, upon completion of the DEBUG Return Routine, the ROM and counter are held in the reset state so that the first DEBUG Save Routine address will be output when the DEBUG Address Enable signal is subsequently returned high following selection of the DEBUG Mode.



NOTES: 1. See figure 5-4, sheet 1, for input/output signal terminations.  
 2. The reference designations shown for the logic circuits are subject to change at any time; for parts identification purposes, refer to the Engineering Schematic and Assembly Diagrams received with the LCDS.

Figure 5-7. DEBUG Address Generator and Address Buffer Circuit Diagram

NS10713



Table 5-3. DEBUG Save and Return Routine Address Generation

Counter Output					ROM Output								Comments
E	D	C	B	A	Y8	Y7	Y6	Y5	Y4	Y3	Y2	Y1	
0	0	0	0	0	0	0	0	0	0	0	0	0	Reset Address
0	0	0	0	1	0	0	1	1	0	0	0	0	1st DEBUG Save Routine Address (7800)
0	0	0	1	0	0	0	1	1	0	0	0	0	2nd DEBUG Save Routine Address (7800)
0	0	0	1	1	0	0	0	0	1	1	0	1	3rd DEBUG Save Routine Address (77FD)
0	0	1	0	0	0	0	1	1	0	0	0	1	4th DEBUG Save Routine Address (7801)
0	0	1	0	1	0	0	1	1	0	0	1	0	5th DEBUG Save Routine Address (7802)
0	0	1	1	0	0	0	1	1	0	0	1	1	6th DEBUG Save Routine Address (7803)
0	0	1	1	1	0	0	1	1	0	0	0	0	7th DEBUG Save Routine Address (7800)
0	1	0	0	0	0	0	1	1	0	0	0	0	8th DEBUG Save Routine Address (7800)
0	1	0	0	1	0	0	0	0	1	1	0	0	9th DEBUG Save Routine Address (77FC)
0	1	0	1	0	0	0	1	1	0	0	0	1	10th DEBUG Save Routine Address (7801)
0	1	0	1	1	0	0	1	1	0	1	0	0	11th DEBUG Save Routine Address (7804)
0	1	1	0	0	0	0	1	1	0	1	0	1	12th DEBUG Save Routine Address (7805)
0	1	1	0	1	0	0	1	1	0	0	0	0	13th DEBUG Save Routine Address (7800)
0	1	1	1	0	0	0	1	1	0	0	0	0	14th DEBUG Save Routine Address (7800)
1	1	1	1	1	0	0	0	0	1	0	1	1	15th DEBUG Save Routine Address (77FB)
1	0	0	0	0	0	0	1	1	0	1	1	0	16th DEBUG Save Routine Address (7806)
1	0	0	0	1	0	0	1	1	0	0	0	0	17th DEBUG Save Routine Address (7800)
1	0	0	1	0	0	0	1	1	0	0	0	0	18th DEBUG Save Routine Address (7800)
1	0	0	1	1	0	0	0	0	1	1	1	1	19th DEBUG Save Routine Address (77FF)
1	0	1	0	0	0	1	1	1	1	0	0	0	20th (last) DEBUG Save Routine Address (7808)
1	0	1	0	1	0	0	1	1	1	0	0	0	1st DEBUG Return Routine Address (7808)
1	0	1	1	0	0	0	1	1	0	1	1	1	2nd DEBUG Return Routine Address (7807)
1	0	1	1	1	0	0	1	1	0	0	0	1	3rd DEBUG Return Routine Address (7801)
1	1	0	0	0	0	0	0	0	1	1	0	0	4th DEBUG Return Routine Address (77FC)
1	1	0	0	1	0	0	1	1	0	0	1	1	5th DEBUG Return Routine Address (7803)
1	1	0	1	0	0	0	1	1	0	0	0	1	6th DEBUG Return Routine Address (7801)
1	1	0	1	1	0	0	0	0	1	0	1	1	7th DEBUG Return Routine Address (77FB)
1	1	1	0	0	0	0	1	1	0	1	0	1	8th DEBUG Return Routine Address (7805)
1	1	1	0	1	0	0	1	1	0	0	0	1	9th DEBUG Return Routine Address (7801)
1	1	1	1	0	1	1	0	0	1	1	0	1	10th (last) DEBUG Return Routine Address (77FD)

NOTE: ROM address 15 is not used.

## Chapter 6

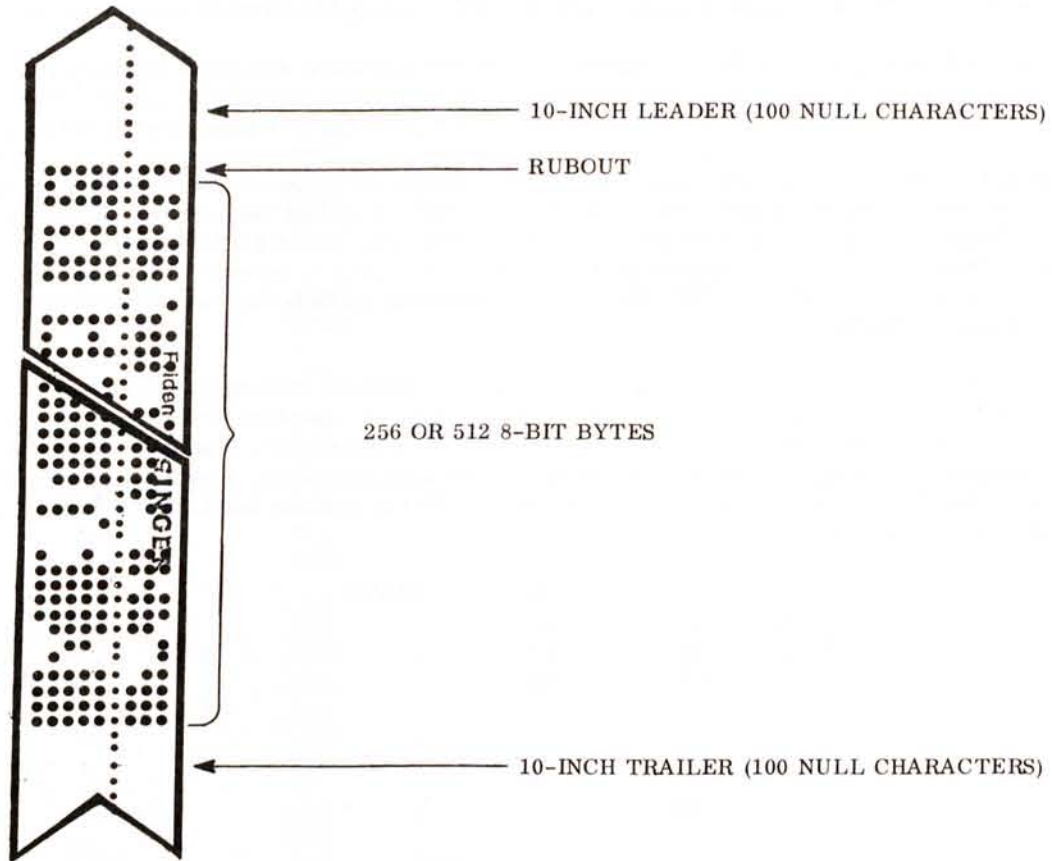
### SERVICE INFORMATION

Since the LCDS is a sophisticated software-controlled instrument, the types of indications observed for a given malfunction may appear to be totally unrelated to the actual cause of the malfunction. Thus, if an erroneous indication is detected while operating the LCDS, disconnect all applications system circuitry and verify the basic operation of the LCDS according to the Panel and TTY checkout procedures provided in chapter 2, Installation. If all of the indications can be obtained as specified for the Panel and TTY checkout procedures, it can be safely assumed that the LCDS is operating properly and that the cause of the malfunction is either in the application system hardware/software interface or in the circuits on the SC/MP CPU Application Card not associated with LCDS operation. If any indication cannot be obtained as specified for the Panel or TTY checkout procedures, return the LCDS to the factory for repair under the WARRANTY or SERVICE policy.

Appendix A

PROGRAM FOR PUNCHING COMPLEMENTED-BINARY PAPER TAPE

The program provided in this appendix enables the LCDS to generate paper tapes that can be used for programming PROMs. When the program is loaded into the LCDS Read/Write Memory locations specified in the listing that follows, it causes the contents of a selected 256- or 512-location memory range to be punched into paper tape in a complemented binary form that is suitable for use with standard PROM programmers. The overall format of the resulting paper tape is shown below.





The Read/Write Memory locations specified in the program listing were chosen because they are not used by the LCDS DEBUG firmware. Thus, after the program is initially loaded via the Panel keyboard switches or the TTY, it will be retained in LCDS Read/Write Memory until power is turned off to the LCDS. If frequent use of the program is anticipated, it also may be desirable to punch the program onto paper tape as described in 3.4.7. Reloading of the program into LCDS Read/Write Memory then can be accomplished as described in 3.4.8.

Before the program may be called, the HALT INST switch on the LCDS must be set to DEBUG to enable normal termination of the program, and the following registers must be loaded with the specified values:

- P1 - starting address of the data to be punched
- P2 - high-order address of a software stack that allows at least 10 memory locations for use by the program; LCDS Read/Write Memory address X'77D0 may be used for this purpose
- E - X'00 to select 256-location memory range or X'FF to select 512 location memory range

After the HALT INST switch is set to DEBUG and the P1, P2, and Extension Registers are loaded with the specified values, the program may be called by initiating LCDS RUN Mode operation at address X'7700 (refer to 3.3.1 and 3.4.2). The program then enters a wait state until the TTY tape punch is turned on and a (any) TTY key is pressed. Following these two actions, the program causes a complemented-binary paper tape to be punched out in the format shown previously. At the end of the trailer, the program automatically branches to the GETC routine of the LCDS DEBUG firmware so that the user may terminate program execution by turning off the TTY tape punch and pressing any TTY key. (When a TTY key is pressed after the tape punch is turned off, GETC returns to the Halt Instruction located at the end of the program and, since the HALT INST switch will be set to DEBUG, the LCDS will enter the DEBUG Mode via the hardware-forced DEBUG Save Routine; refer to 3.4.2.)

The length of the leader and trailer is determined by the program constant LEN contained in memory location 7704, and the complemented binary data is obtained via the instructions contained in memory locations 771B through 771E. All that is necessary to alter the length of the leader and trailer, therefore, is to load an appropriate new value into memory location 7704 ( $10_{10} = 1$ -inch leader/trailer). Similarly, if straight-binary rather than complemented-binary data is required, the instructions in memory locations 771B through 771F can be changed to the following:

771B	C101	LD	@1(P1)
771D	08	NOP	
771E	08	NOP	
771F	08	NOP	

```

PUNCH  PUNCH COMP BIN TAPE.
1
2
3      0001  P1    =    1
4      0002  P2    =    2
5      0003  P3    =    3
6      7700  RAM   =    077D0
7      7B4F  GHEX  =    07B4F
8      7B16  MESH  =    07B16
9      7AE1  PUTC  =    07AE1
10     7A88  GETC  =    07A88
11     0064  LEN   =    100
12     7700  .    =    07700
13
14     ;      P1 MUST HAVE ADDRESS OF BUFFER.
15     ;      P2 MUST HAVE RAM ADDRESS. 077D0 IS OK.
16     ;      E REG = 0 PUNCH 256 WORDS.
17     ;      E REG = -1 PUNCH 512 WORDS.
18
19
20     START:
21  7700  C47A  LDI    H(GETC)
22  7702  37    XPAH   P3
23  7703  C488  LDI    L(GETC)
24  7705  33    XPAL   P3
25  7706  3F    XPPC   P3      ; WAIT FOR KEYBD HIT.
26  7707  C464  LDI    LEN
27  7709  CA01  ST     1(P2)      ; SET COUNTER FOR LEADER.
28  770B  C47A  LDI    H(PUTC)      ; SET ADDRESS OF PUTC.
29  770D  37    XPAH   P3
30  770E  C4E1  LDI    L(PUTC)
31  7710  33    XPAL   P3
32
33     LDRLP:
34  7711  C400  LDI    0
35  7713  3F    XPPC   P3      ; PUNCH LEADER.
36  7714  BA01  DLD    1(P2)      ; UP DATE COUNTER.
37  7716  9CF9  JNZ    LDRLP      ; KEEP GOING.
38  7718  C4FF  LDI    -1      ; PUT RUBOUT.
39  771A  3F    XPPC   P3
40
41     LOOP:
42  771B  C400  LDI    0      ; SET A TO 0.
43  771D  02    CCL    0      ; CLEAR CARRY.
44  771E  FD01  CAD    @1(P1)      ; GET COMPLEMENTED DATA. **
45  7720  3F    XPPC   P3      ; PUNCH DATA.
46  7721  AA01  ILD    1(P2)      ; UP DATE COUNT.
47  7723  9CF6  JNZ    LOOP
48  7725  40    LDE
49  7726  9805  JZ     OUT      ; FIND WHICH.
50  7728  C400  LDI    0      ; CHECK IF DONE.
51  772A  01    XAE
52  772B  90EE  JMP    LOOP      ; RESET E REG.
53
54     OUT:
55  772D  C464  LDI    LEN      ; SET LENGTH OF TRAILER.
56  772F  CA01  ST     1(P2)
57
58     TRAIL:
59  7731  C400  LDI    0      ; PUNCH TRAILER.
60  7733  3F    XPPC   P3
61  7734  BA01  DLD    1(P2)
62  7736  9CF9  JNZ    TRAIL
63  7738  C488  LDI    L(GETC)
64  773A  33    XPAL   P3
65  773B  C47A  LDI    H(GETC)
66  773D  37    XPAH   P3
67  773E  3F    XPPC   P3      ; WAIT FOR TAPE OFF.
68  773F  00    HALT
69  0000  .END      ; RETURN TO DEBUG.

```



This digitised copy has been provided by Museum Victoria, free of charge. You may use this work for any Non-Commercial Use with attribution to Museum Victoria and the creator.

**National Semiconductor Corporation**

2900 Semiconductor Drive  
Santa Clara, California 95051  
(408) 737-5000  
TWX: 910-339-9240

**National Semiconductor GmbH**

808 Fuerstenfeldbruck  
Industriestrasse 10  
West Germany  
Telephone: (08141) 1371  
Telex: 05-27649

**NS Electronics (HK) Ltd.**

4 Hing Yip Street, 11th Floor  
Kwun Tong  
Kowloon, Hong Kong  
Telephone: 3-411241-8  
Telex: 73866 NSE HK HX

**NS International Inc.**

Miyake Bldg. 6F, 1-9 Yotsuya  
Shinjuku-Ku  
Tokyo 160, Japan  
Telephone: 03-355-3711  
Telex: J28592

**NS Electronics Pty. Ltd.**

CNR-Stud Road & Mountain Highway  
Bayswater, Victoria 3153, Australia  
Telephone: 03-729-6333  
Telex: 32096

HT29826