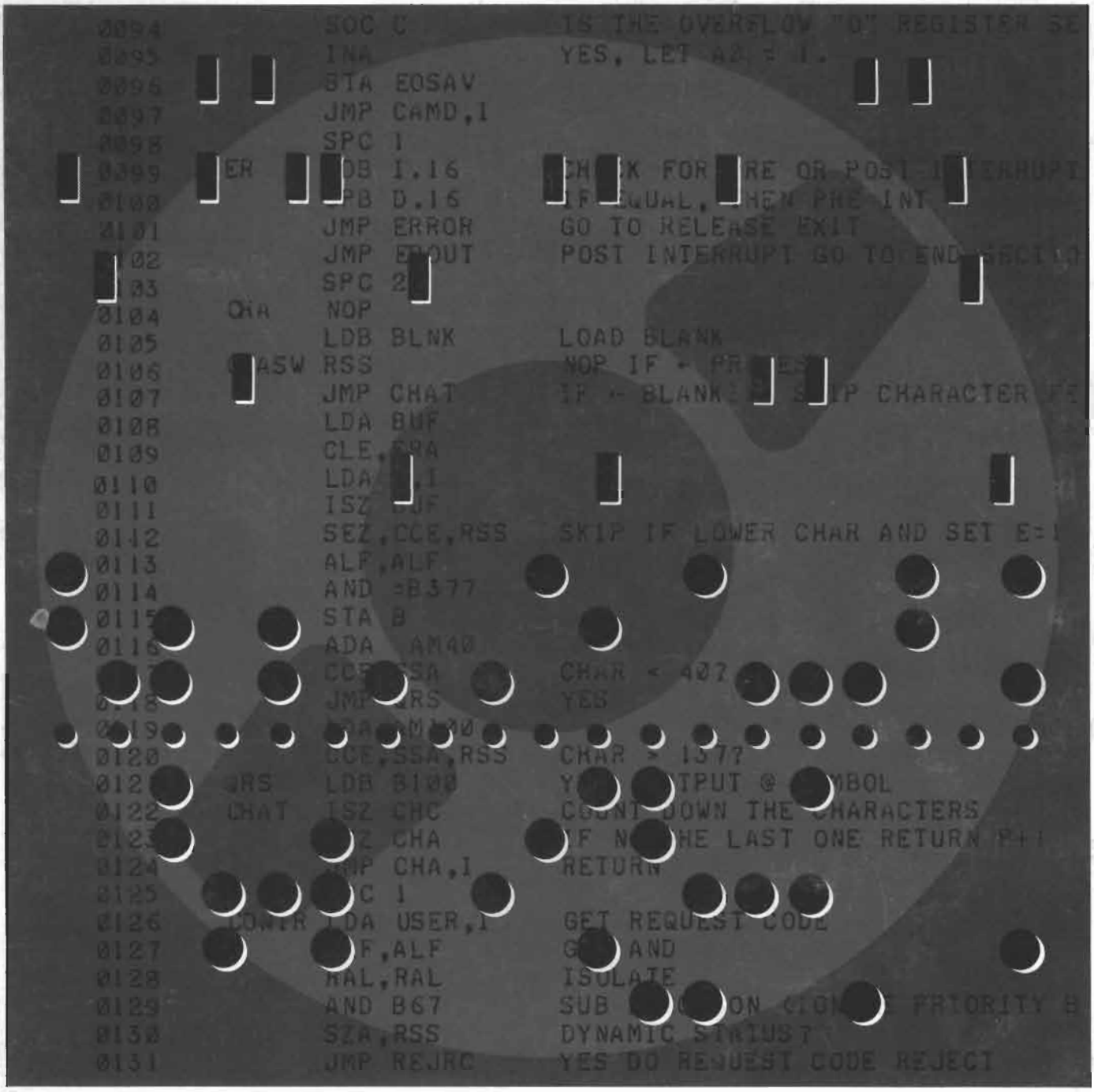


Assembler Programming Course

HEWLETT hp PACKARD

HEWLETT-PACKARD
ASSEMBLER PROGRAMMING COURSE
STUDENT ASSIGNMENT MANUAL
(STOCK NO. 5951-3029)

– NOTICE –

The information contained in this manual is for training purposes only. Consult the Hewlett-Packard documentation supplied with the computer for current information concerning the specific computer system furnished.

11000 Wolfe Road, Cupertino, California 95014 Area Code 408 257-7000 TWX 910-338-0221

Printed in U.S.A. 6/72

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

TABLE OF CONTENTS

FOREWORD

This manual contains the classroom exercises, laboratory exercises and homework assignments used in the Hewlett-Packard assembler programming course. Each exercise and assignment in this manual is designed to give the student practice in a particular technique or to re-enforce a concept or procedure covered in the lecture phase. The student who rigorously pursues a solution to these exercises both in class and in the computer laboratory will gain the most benefit from the programming course.

The answers to all the exercises presented in the programming course are provided in Appendix A. The answers should be used primarily to check your results, they may also be used in the learning process when you feel that a look at the answer will provide a path of progress when all else has failed.

We sincerely hope that you enjoy attending our assembler programming course.

Training Staff
Cupertino Division

Laboratory Exercise Guide

Lesson: **INTRODUCTION TO HP COMPUTER HARDWARE**

Exercise: 1-1

A. OBJECTIVE

1. To provide an exercise in using the front panel

B. PROBLEM

1. Hand load the program on page 1-14 of the course manual into memory.

C. PROCEDURE

1. Load the program (instructions and data) into memory starting at address 100B.

2. Display the contents of locations 100B through 105B and 200B through 202B and check for errors.

3. Load address 100B and press run. The computer should come to a halt with 102077B in the display register. Look in location 500B and also display the A and P registers.

4. Set the P register to 100B and press the memory data button. Single cycle through the program.

D. RESULTS

1. Location 500B should contain an octal 7.

DONE

Classroom Exercise Guide

Lesson: **INTRODUCTION TO THE HP ASSEMBLER**

Exercise: 2-1

A. OBJECTIVE

1. To provide an exercise in bit manipulation.

B. PROBLEM

1. Read a value from the switch register and depending upon its value either leave a variable (Z) equal to -1 or set it to Ø or 1. Sounds as a basic IF statement in rudimentary machine lingo

C. PROCEDURE

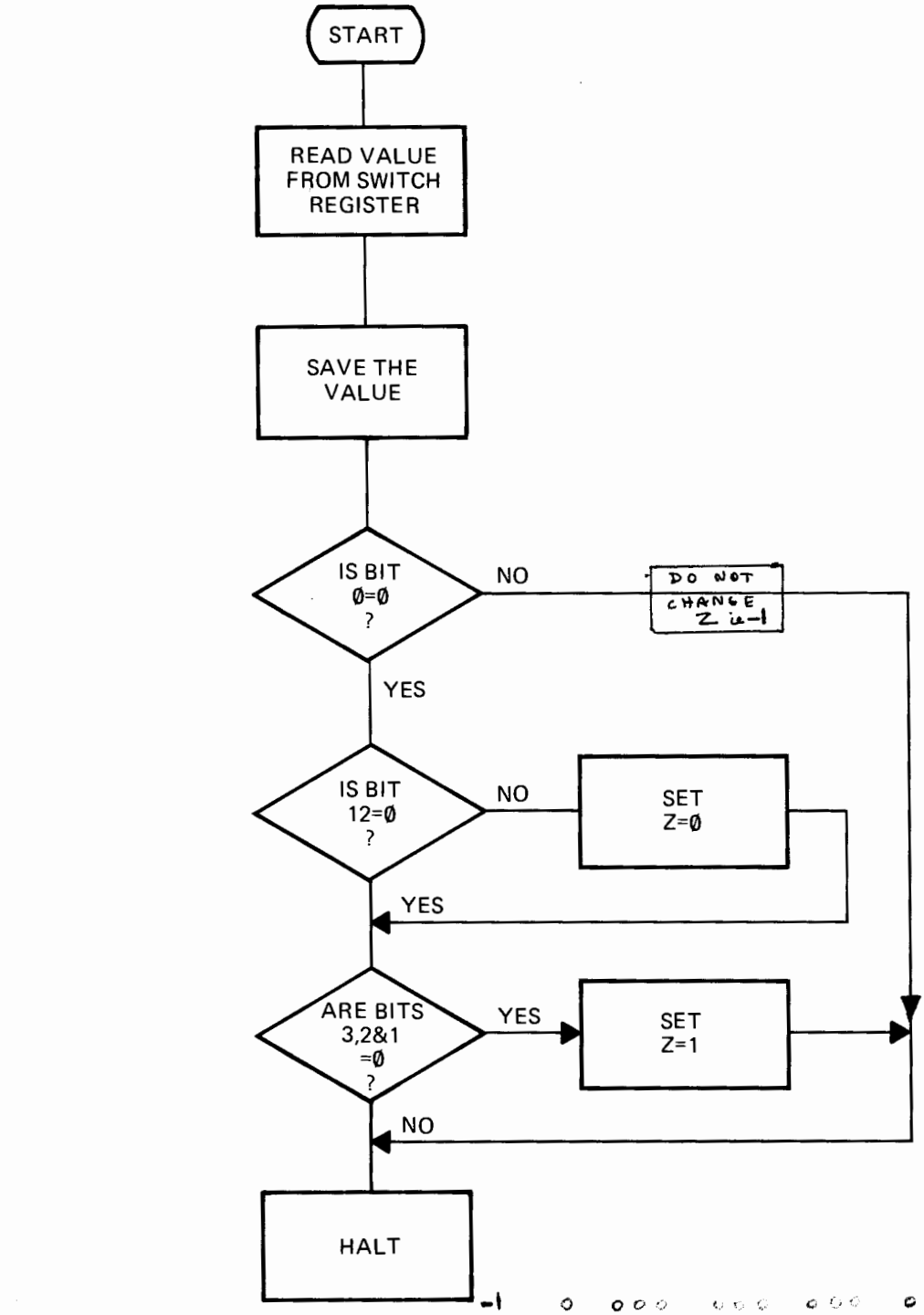
1. Specific program requirements are outlined in the flowchart on page 2-2. Use the simulated coding form on page 2-3 for your solution.

D. RESULTS

1. The results of this exercise will be discussed in class. You may check your solution by comparing it to the answer shown on page A-2-1. If your solution does not agree with the one given, do not assume you are wrong. There are many different correct solutions to this problem.

TEST VALUES:

Switch Register Value	Value of Z
1_8	-1
10002_8	Ø
0_8	+1


START
READ VALUE FROM SWITCH REGISTER
SAVE THE VALUE
IS BIT 0=0 ?
NO
DO NOT CHANGE Z i-1
YES
IS BIT 12=0 ?
NO
SET Z=0
YES
ARE BITS 3,2&1 =0 ?
YES
SET Z=1
NO
HALT


```
          LABEL     OPCODE    OPERAND     REMARKS

PAGE 0001


0001                        ASMB,A,B,L,T
Z         002026
NOT       002021
CONT      002011
FINIS     002027
MASK      002025
START     002000
SWICH     002024
**  NO ERRORS*
 PAGE 0002 #01


0001                        ASMB,A,B,L,T
0002*                                A BENDELI 25 JULY 1973
0003   02000                        ORG 2000B
0004   02000 102501  START LIA 1                LOAD A FROM SR
0005   02001 072024        STA SWICH             STORE A IN SWICH
0006   02002 007400        CCB
0007   02003 076026        STB Z                 SET Z TO -1
0008   02004 000010        SLA                   SKIP IF BIT 0=0
0009   02005 026027        JMP FINIS             NOT 0
0010   02006 001700        ALF                   ROTATE 4 LEFT
0011   02007 000010        SLA                   SKIP IF LSB A IS 0,BIT 12=0
0012   02010 026021        JMP NOT               BIT 12 NOT 0
0013   02011 062024  CONT  LDA SWICH            RELOAD A AGAIN
0014   02012 012025        AND MASK              LEAVES ONLY BITS 3,2,1
0015   02013 002002        SZA                   SKIP IF BITS 3,2,1 =0
0016   02014 026027        JMP FINIS             BITS 3,2,1 NOT 0
0017   02015 002400        CLA
0018   02016 002004        INA
0019   02017 072026        STA Z                 STORE 1 IN Z
0020   02020 026027        JMP FINIS
0021   02021 002400  NOT   CLA
0022   02022 072026        STA Z                 STORE 0 IN Z
0023   02023 026011        JMP CONT
0024   02024 000000  SWICH NOP
0025   02025 000016  MASK  OCT 000016
0026   02026 177777  Z     OCT 177777
0027   02027 000000  FINIS NOP
0028   02030 066026        LDB Z
0029   02031 102077        HLT 77B
0030   02032 026000        JMP START
0031                        END
**  NO ERRORS*
```

```
*LOAD


*L08
 PAGE 0001


0001                    ASMB,A,B,L,T
DEC      002034
AGAIN    002004
BLANK    002033
COUNT    002032
FINIS    002035
KARTR    002007
PUNCH    002024
READ     002016
START    002000
**  NO ERRORS*
 PAGE 0002 #01


0001                    ASMB,A,B,L,T
0002*                          A BENDELI 26 JULY 1973
0003  02000                    ORG 2000B
0004  02000 002400   START CLA
0005  02001 016016           JSB READ ·'    READ CHARACTER IN A
0006  02002 002003           SZA,RSS        SKIP ON NON BLANK (IE CHAR.)
0007  02003 026001           JMP *-2
0008  02004 006400   AGAIN CLB
0009  02005 066033           LDB BLANK
0010  02006 076032           STB COUNT
0011  02007 016024   KARTR JSB PUNCH
0012  02010 016016           JSB READ
0013  02011 002002           SZA              TEST & SKIP IF BLANK
0014  02012 026004           JMP AGAIN        IF A CHARACTER
0015  02013 036032           ISZ COUNT
0016  02014 026007           JMP KARTR         PUNCH(CHAR & UP TO 99 BLANK)
0017  02015 026035           JMP FINIS
0018  02016 000000   READ  NOP
0019  02017 103713           STC 13B,C         SET CONTROL & CLEAR FLAG
0020  02020 102313           SFS 13B
0021  02021 026020           JMP *-1
0022  02022 102513           LIA 13B           LOAD A FROM INTERFACE 13
0023  02023 126016           JMP READ,I
0024  02024 000000   PUNCH NOP
0025  02025 102617           OTA 17B           LOAD INTERFACE 17 FROM A
0026  02026 103717           STC 17B,C
0027  02027 102317           SFS 17B
0028  02030 026027           JMP *-1
0029  02031 126024           JMP PUNCH,I
0030  02032 000000   COUNT NOP
0031  02033 177634   BLANK DEC -100
0032  02034 000000   DEC   NOP
0033  02035 102077   FINIS HLT 77B
0034  02036 026000           JMP START
0035                           END
**  NO ERRORS*
```

OUR SELECT CODE FOR READER = 10_8 (PHOTOREADER)

TTY = 11_8

Laboratory Exercise Guide

Lesson: **ASSEMBLER PSEUDO INSTRUCTIONS**

Exercise: 3-1

A. OBJECTIVE

1. To give the student practice in using some of the Assembly language programming techniques discussed in the classroom.

B. PROBLEM

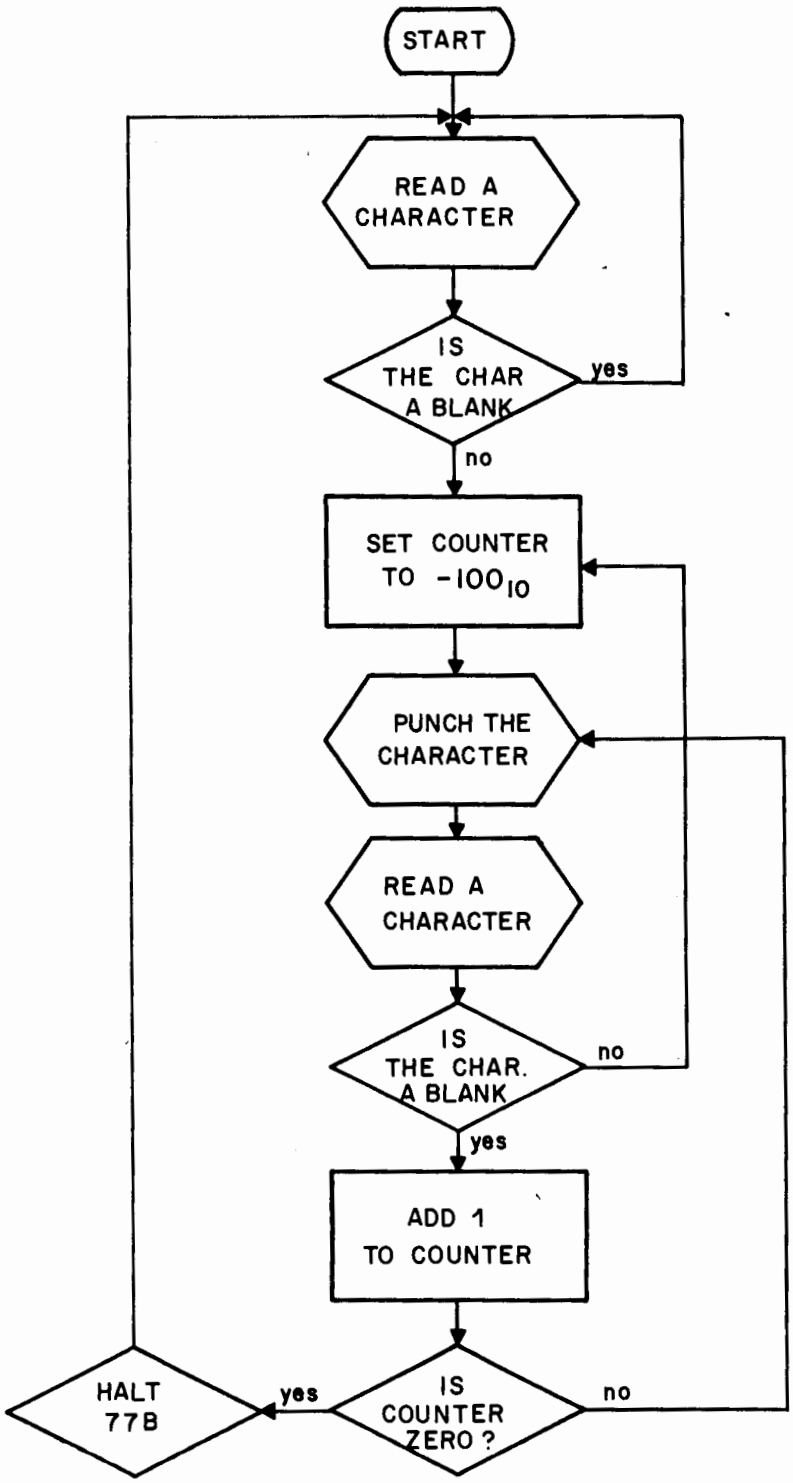
1. Using Assembly language programming techniques write a program that will read data on the Photoreader and reproduce the data on the Punch. This program will be useful for reproducing paper tape. The program will reproduce any 8 level tape exactly as it appears.

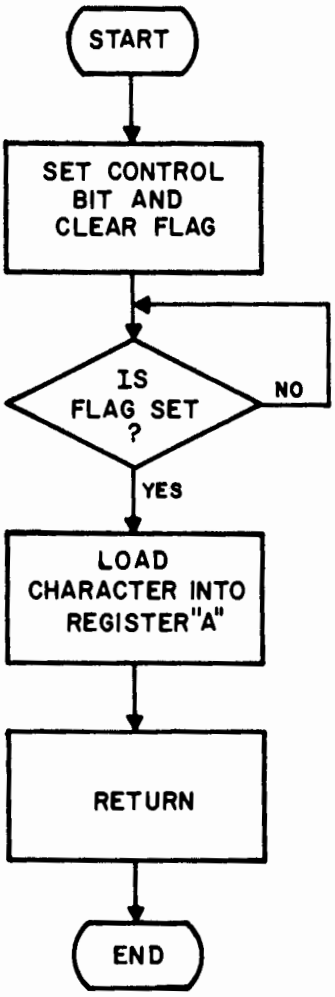
C. PROCEDURE

1. Write an absolute assembly language program that will read any 8 level paper tape code on the Photoreader and reproduce this code by punching the same code on the Punch. The program must stop processing data after processing 100_{10} consecutive blank (feed hole only) paper tape frames(ie anywhere before or after ?? and during)
answer : consecutive 100_{10} inside the message or at the end. Normally no 100_{10} blanks inside the message
2. The specific details of the problem are described in the flowchart. Notice that the READ and PUNCH operations are Subroutines.

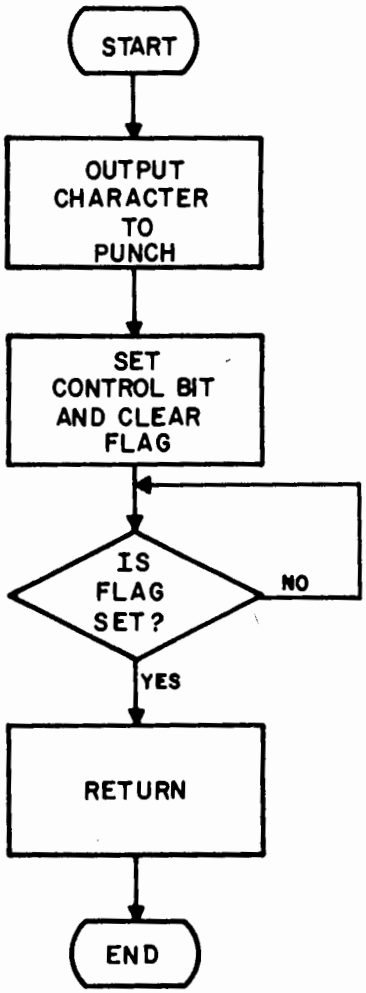
D. RESULTS

1. The results of this exercise will be discussed in class. One of many possible solutions is shown on page A-3-1 in the Answer section of this manual.


START
READ A CHARACTER
IS THE CHAR A BLANK
yes
no
SET COUNTER TO -100₁₀
PUNCH THE CHARACTER
READ A CHARACTER
IS THE CHAR. A BLANK
no
yes
ADD 1 TO COUNTER
IS COUNTER ZERO ?
yes
no
HALT 77B



SUBROUTINE "READ"



SUBROUTINE "PUNCH"

Laboratory Exercise Guide

Lesson: **ASSEMBLER PROGRAMMING TECHNIQUES**

Exercise: 4-1

A. OBJECTIVE

1. To provide the student with practice in the creation of a FORTRAN/Assembler callable assembly language subroutine.

B. PROBLEM

1. Using Assembly language, write a FORTRAN/Assembler callable function which will compare a test value against upper and lower limits and set the A register to:

   +1 if the test value is => upper limit
   0 if the test value is within limits
   -1 if the test value is <= lower limit

C. PROCEDURE

1. The calling sequence generated by FORTRAN is outlined on page 4-11 of the course manual. Three parameter addresses are passed and the actual parameters (upper, lower, test) are referenced indirectly by the function. A FORTRAN main program will be provided by the instructor; a listing of which is on page 4-3.

2. The specific requirements of the subroutine are outlined in the flowchart.

D. RESULTS

ENTER VALUES FOR UPPER, LOWER AND TEST

10, -10, 0

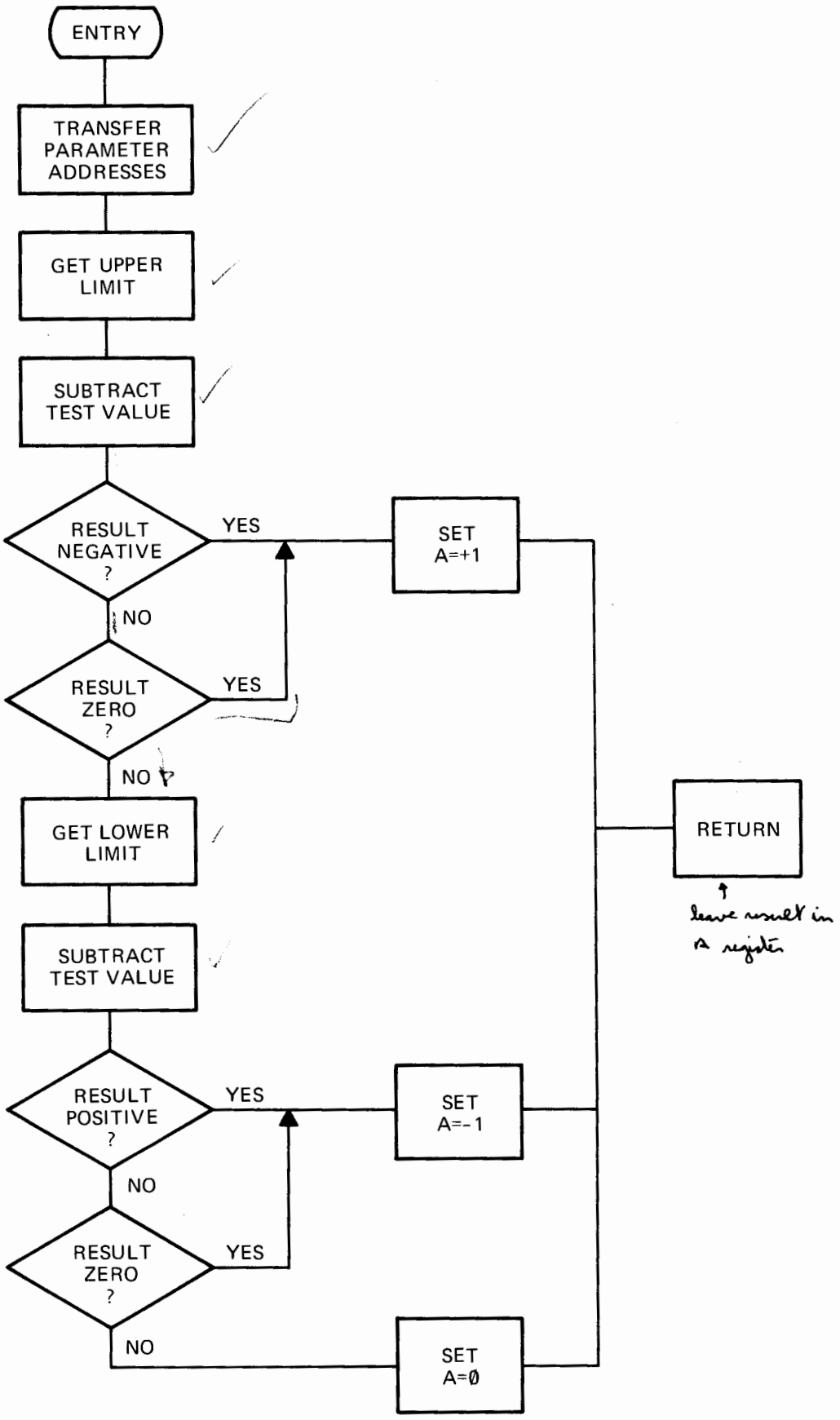
WITHIN LIMITS

ENTER VALUES FOR UPPER, LOWER AND TEST

36.57, 36.44, 36.43

<= LOWER LIMIT

ENTER VALUES FOR UPPER, LOWER AND TEST


ENTRY
TRANSFER PARAMETER ADDRESSES
GET UPPER LIMIT
SUBTRACT TEST VALUE
RESULT NEGATIVE ?
YES
SET A=+1
NO
RESULT ZERO ?
YES
NO
GET LOWER LIMIT
SUBTRACT TEST VALUE
RESULT POSITIVE ?
YES
SET A=-1
NO
RESULT ZERO ?
YES
NO
SET A=Ø
RETURN
leave result in A register

FORTRAN Calling Program:

```
FTN,L,B
C
C      MAIN PROGRAM
C
       PROGRAM LMTST
     5 WRITE (2,100)
   100 FORMAT ("ENTER VALUES FOR UPPER, LOWER AND TEST"//)
       READ (1,*) X,Y,Z
       IF(X-Y) 6,7,7
     6 WRITE (2,150)
   150 FORMAT ("LIMIT ERROR")
       GO TO 5
     7 IF (ILIMT(X,Y,Z)) 10,20,30
    10 WRITE (2,200)
   200 FORMAT (//" <= LOWER LIMIT"//)
       GO TO 5
    20 WRITE (2,300)
   300 FORMAT (//" WITHIN LIMITS"//)
       GO TO 5
    30 WRITE (2,400)
   400 FORMAT (//" => UPPER LIMIT"//)
       GO TO 5
       END
       END$
```

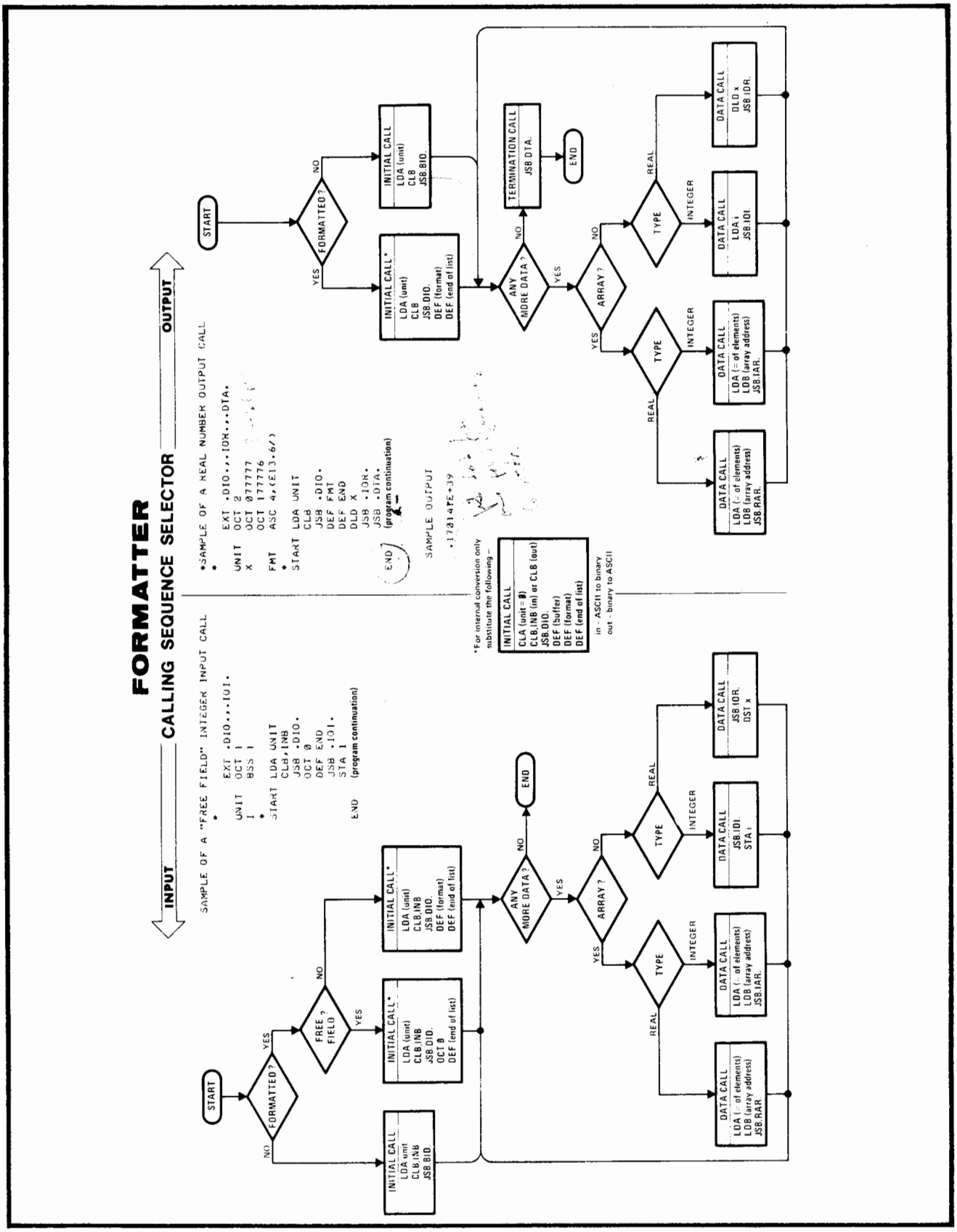
X,Y,Z define real numbers, so 1Φ, -1Φ, Φ should be an error because must be 1Φ., -1Φ., Φ.

ILIMIT is an integer and result of subroutine must be an integer XYZ wrong!.

x,y,z define real numbers, so 1ϕ, -1ϕ, ϕ should be an error because must be 1ϕ., -1ϕ., ϕ.

```
LUES FOR UPPER, LOWER AND TEST"//)


ROR")                   ASMB,R,B,L,T
                        *                   A BENDELI 27 JULY 1973
 10,20,30                      NAM EXERC
                               NOP
WER LIMIT"//)                  ENT ILIMT    DEFINES ENTRY POINT IN SUBROUTINE
                        X      BSS 2
                        Y      BSS 2        RESERVE 2 WORDS FOR VARIABLES
N LIMITS"//)            Z      BSS 2
                        ILIMT  NOP          START HERE
                               DEF X
PER LIMIT"//)                  DEF Y        LET ASSEMBLER DEFINE ADDRESSES
                               DEF Z
                               DLD X        LOAD UPPER LIMIT
                               FSB Z
                               SSA,RSS      SKIP FOR -VE
                               JMP NONEG    +VE
                        NEGTV  CLA,INA      -VE, SET A=+1
                               JMP ILIMT,I
                        NONEG  SZA          TEST IF RESULT IS Ø
                               JMP LOWER
                               RRL 8        ROTATE FOR SECOND HALF OF WORD
                               SZA,RSS      SKIP FOR A NOT Ø
                               JMP NEGTV    JUMP TO SET A=+1
                        LOWER  DLD Y
                               FSB Z
                               SSA          TEST RESULT
                               JMP NOPOS
                        SETM1  LDA FUNM1    SET A=-1
                               JMP ILIMT,I
                        NOPOS  SZA          TEST IF RESULT IS =Ø
                               JMP *+4      NOT Ø
                               RRL 8        ROTATE FOR SECOND HALF OF WORD
                               SZA,RSS      SKIP IF NOT Ø
                               JMP SETM1    USED TO SET A=-1
                               CLA          SET A=Ø
                               JMP ILIMT,I
                        FUNM1  DEC -1       RESERVE LOCATION FOR CONSTANT
                               END ILIMT
```


FORMATTER
CALLING SEQUENCE SELECTOR
INPUT
OUTPUT
SAMPLE OF A "FREE FIELD" INTEGER INPUT CALL
EXT .DIO.,.IOR.
UNIT OCT 1
I BSS 1
START LDA UNIT
CLB,INB
JSB .DIO.
OCT 0
DEF END
JSB .IOR.
STA I
END (program continuation)
*SAMPLE OF A REAL NUMBER OUTPUT CALL
EXT .DIO.,.IOR.,.DTA.
UNIT OCT 2
X OCT 077777
OCT 177776
FMT ASC 4,(E13.6)
START LDA UNIT
CLB
JSB .DIO.
DEF FMT
DEF END
DLD X
JSB .IOR.
JSB .DTA.
END (program continuation)
SAMPLE OUTPUT
-1.781476-39
*For internal conversion only substitute the following –
INITIAL CALL
CLA (unit = 0)
CLB,INB (in) or CLB (out)
JSB.DIO.
DEF (buffer)
DEF (format)
DEF (end of list)
in - ASCII to binary
out - binary to ASCII
START
FORMATTED?
YES
NO
FREE FIELD?
INITIAL CALL
LDA (unit)
CLB,INB
JSB.BIO.
INITIAL CALL*
LDA (unit)
CLB,INB
JSB.DIO.
OCT 0
DEF (end of list)
INITIAL CALL*
LDA (unit)
CLB,INB
JSB.DIO.
DEF (format)
DEF (end of list)
ANY MORE DATA?
NO
END
YES
ARRAY?
TYPE
REAL
INTEGER
DATA CALL
LDA (= of elements)
LDB (array address)
JSB.RAR.
DATA CALL
JSB.IDI.
STA i
DATA CALL
JSB.IOR.
DST x
INITIAL CALL*
LDA (unit)
CLB
JSB.DIO.
DEF (format)
DEF (end of list)
INITIAL CALL
LDA (unit)
CLB
JSB.BIO.
TERMINATION CALL
JSB.DTA.
DATA CALL
LDA i
JSB.IOI.
DATA CALL
DLD x
JSB.IOR.

Homework Assignment

Lesson: ASSEMBLER PROGRAMMING TECHNIQUES

Assignment: 4-1

Reference: Course Manual, pages 4-19 through 4-23 and Students Manual page 4-4.

1. An entire array of "real number" data is to be output in binary form using Unit #4. The data block has 1ØØ elements and may be referenced by the label "BUFFR". In the space below, write the Assembly language calling sequence that will enable the FORMATTER to complete this operation. Declare all EXTernal symbols used and consider the instruction following the calling sequence to be a NOP for all questions in this assignment.

LABEL	OP CODE	OPERAND	REMARKS
	EXT		

2. An entire array of "real numbers" is to be output on Unit #4 as "formatted" data. The array name is SAM and it has 1Ø elements. The Format to be used should provide the following output form.

±XXX.XX

In the space below write the Assembly language calling sequence that will enable the FORMATTER to complete this operation.

LABEL	OP CODE	OPERAND	REMARKS

3. Assume that 2 integer values are to be output on Unit #4 as "formatted" data. The labels for the values are "J" and "K". The Format should provide 3 digits of output for "J" and 4 digits for "K".

   In the space below write the Assembly language calling sequence that will enable the FORMATTER to complete this operation.

   LABEL OP CODE OPERAND REMARKS

4. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #1. Use Unit #5 for input.

   LABEL OP CODE OPERAND REMARKS

5. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #2. Use Unit #5 for input.

LABEL	OP CODE	OPERAND	REMARKS

6. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #3. Use Unit #5 for this operation.

LABEL	OP CODE	OPERAND	REMARKS

Laboratory Exercise Guide

Lesson: **HP BASIC CONTROL SYSTEM, IOC SECTION**

Exercise: 5-1

A. OBJECTIVE

1. To provide a programming task that requires communicating directly with IOC.

B. PROBLEM

1. A paper tape is to be searched for error messages. Control characters within each record are examined to determine error records.

C. PROCEDURES

1. The specific problem requirements are outlined in the flowchart, however, the following information is also provided.

   a. Read a data message consisting of twenty characters.

   b. If characters 15 & 16 are equal to ** print ERROR MESSAGE on the teleprinter.

   c. If characters 1 & 2 are equal to @@ the end of the message tape has been reached – halt the computer.

   d. If neither b or c are met, read the next record.

   e. Use Unit #5 for input and Unit #2 for output.

   f. A system status check should be made to insure that all output operations are complete before the computer is brought to a halt.

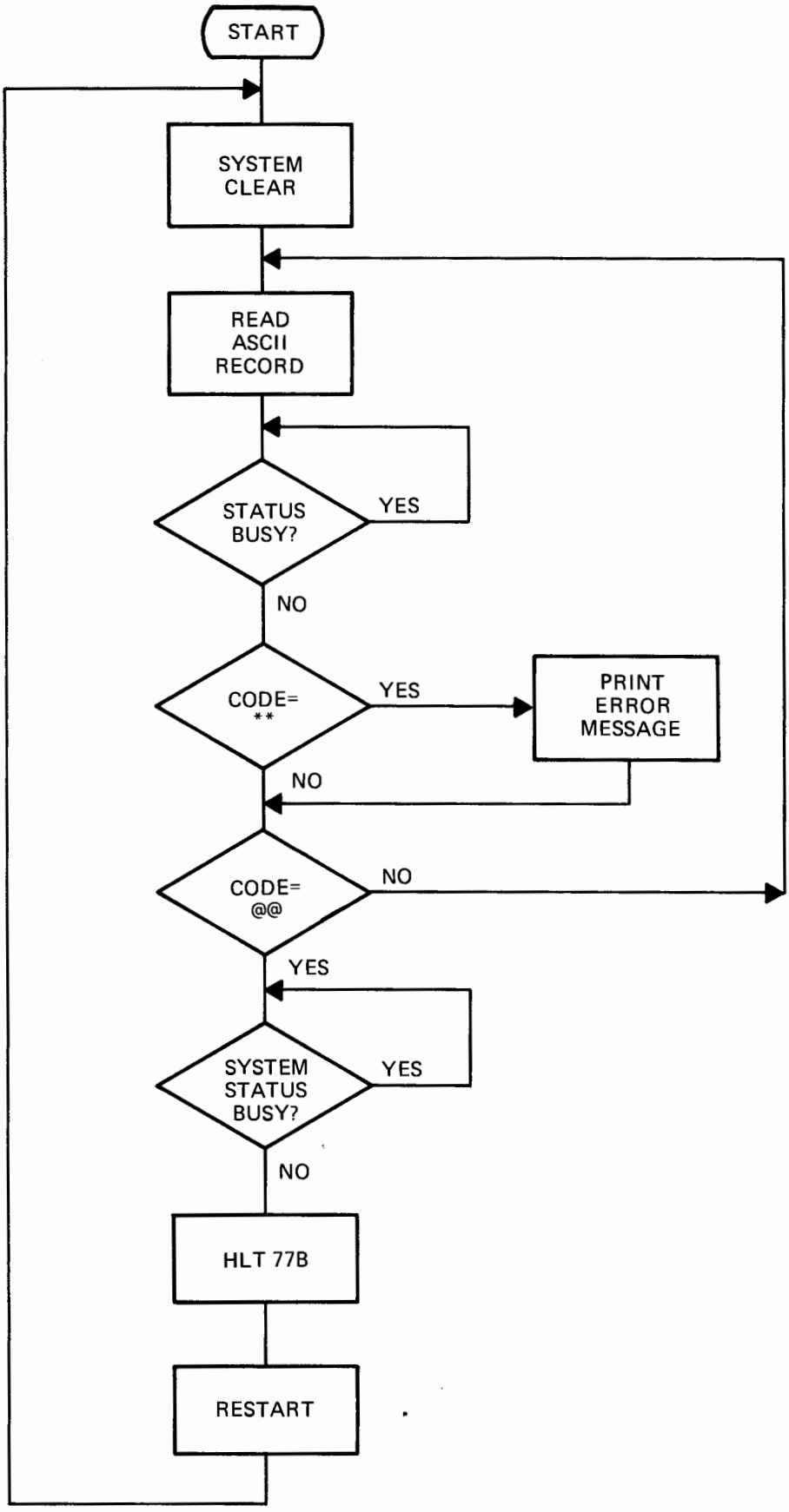
   OPTIONAL

   Each time an error message is encountered, print along with the words ERROR MESSAGE the record number.

D. RESULTS

1. ERROR MESSAGE should be printed 1Ø times.

2. A solution is shown on page A-5-1 & A-5-2.


START
SYSTEM CLEAR
READ ASCII RECORD
STATUS BUSY?
YES
NO
CODE= **
YES
PRINT ERROR MESSAGE
NO
CODE= @@
NO
YES
SYSTEM STATUS BUSY?
YES
NO
HLT 77B
RESTART

Homework Assignment Guide

Lesson: **HP BASIC CONTROL SYSTEM**

Assignment: 5-1

1. What are the three main elements of the Basic Control System?

   a.

   b.

   c.

2. The Basic Control System resides in what portion of memory?

3. The Basic Control System is loaded into the computer by means of what program?

4. The Basic Control System is:

   a) an absolute program

   b) a relocatable program

   c) a source program

   d) an object program

5. Why do we use unit reference numbers to specify an I/O device rather than the physical I/O device channel number?

6. What is the reason for specifying a standard unit table in addition to the equipment table when configuring a BCS system?

   a) FORTRAN I/O requests can use standard unit table numbers, but not equipment table entries.

   b) Since standard unit table numbers specify whether the operation is input or output, the programmer's task is made easier.

c) References to standard unit table numbers rather than equipment table numbers will allow us to run the program on any other HP computer, regardless of its BCS configuration.

d) Since standard unit table numbers (1-6) are less than the equivalent table numbers (7-77), our I/O operation will have higher priority if a standard unit table number is used.

7. Each entry in the equipment table consists of how many computer words?

8. The Input/Output driver routines consist of two sections. What are they called?

   a.

   b.

9. What information must be supplied by the programmer in the last 4 words of the calling sequence to IOC?

10. The basic control system transfers control to the reject point of the calling sequence under what condition?

    a) subfunction is indicated to be V = 1 and n = 1

    b) DMA channel not available

    c) if we try to read an ASCII tape

    d) the referenced unit reference number is not defined

11. What information is made available to the user when IOC rejects a call?

    Register A

    Register B

12. What information is made available to the user when a Status Request is executed?

    Register A

    Register B

13. Of what use is the systems status check?

14. The Clear Request command to IOC performs what type of function?

15. Code a calling sequence to IOC to perform the following operations:

    1. Read an ASCII record consisting of 20 characters.
    2. The input device is associated with Equipment Table entry 15.
    3. If the command cannot be executed a subroutine called XAMIN is to be entered.
    4. The data is to be read into a memory buffer called INBUF which you are to reserve.

Classroom Exercise Guide

Lesson: **CONFIGURATION ROUTINE**

Exercise: 6-1

A. OBJECTIVE

1. To provide the student with a practical exercise in planning a Basic Control System.

B. PROBLEM

1. Create a Basic Control System for a Computer with 8K of memory and the following:

	B.C.S. Driver ID. Code	Standard Unit Number	Equipment Table Number	I/O Channel Number
ASR-33 Teleprinter	D.ØØ	1,2,6	11	16
Photoelectric Tape Reader	D.Ø1	3,5	7	17
Paper Tape Punch	D.Ø2	4	1Ø	2Ø

C. PROCEDURE

1. In order to complete this exercise it is only necessary to follow the instructions included with the P.C.S. Planning Package provided for your use. This procedure is highly recommended for actual, as well as classroom, configurations. The planning form has 3 parts.

    1. Instructions
    2. Configuration Form
    3. Worksheet

D. RESULTS

1. The instructor will check the student worksheet portion of the planning form at the conclusion of the exercise.

PREPARE CONTROL SYSTEM (P.C.S.) PLANNING PACKAGE

1. Use the table on page 6-1 to determine select code assignments, system and equipment tables, interrupt linkages and software driver designations.

2. The format of this worksheet allows an exact procedure for configuring a basic control system. To use this form all blank spaces on the worksheet portion must be filled in. All the required information is available from the completed table on page 6-1.

3. All responses to statements ending with a question mark (?), are typed on the Teleprinter keyboard, followed by CR LF . All responses to the "*LOAD" statement will be the loading of a B.C.S. module using paper tape.

4. XXXXX – Are used to indicate variable values that will be listed on Teleprinter listing. Only the general form of these listings can be shown here.

CONFIGURATION DIAGRAM

SYSTEM SERIAL NO. _____ DATE _____

Higher ← PRIORITY → Lower

FIRST WORD OF AVAILABLE MEMORY (FWAM) _____

LAST WORD OF AVAILABLE MEMORY (LWAM) _____

																I/O INTERFACE CARD
DRIVER IDENTIFICATION																
INTERRUPT LOCATION	10	11	12	13	14	15	16	17	20	21	22	23	24	25	26	27
LINKAGE LOCATION																
INTERRUPT IDENT.																
EQUIPMENT TABLE →																

	STANDARD UNIT TABLE																
1.	KEYBOARD INPUT																
2.	TELEPRINTER OUTPUT																
3.	PROGRAM LIBRARY																
4.	PUNCH OUTPUT																
5	INPUT																
6.	LIST OUTPUT																

NOTE: ASSIGN STANDARD UNITS BY PLACING A CHECKMARK AT THE CORRECT X-Y INTERSECTION.

COMPUTER PROGRAMMING COURSE

P.C.S. WORKSHEET

THESE ENTRIES APPLY ONLY TO THE "CONFIGURING" SYSTEM.

HS INP ? _____ (ENTRY = DEVICE CHANNEL NUMBER)

HS PUN ? _____

FWA MEM ? _____

LWA MEM ? _____

*LOAD Load drivers in any order.

D.XX

XXXXX XXXXX This sequence continues until all drivers have been processed.

*LOAD

IOC Load the IOC Module.
XXXXX XXXXX

*TABLE ENTRY

UNIT REF #	EQT ?	(Push RUN and type;) (FORMAT = CHAN #, D.XX)
(7)	_____	
(1Ø)	_____	
(11)	_____	
(12)	_____	
:		
ETC	/E	Terminates equipment table entries.

SQT ? (ENTRY = EQUIP. TABLE ENTRY NUMBER)

-KYBD ?

_____

-TTY ?

_____

-LIB ?

_____

-PUNCH ?

_____

-INPUT ?

_____

-LIST ?

_____

DMA ?

_____

*LOAD Load the relocatable loader module.

LOADR

XXXXX XXXXX

INTERRUPT LINKAGE ? (PUSH RUN and type)
(FORMAT = INT. LOC., LINK LOC, I.XX)

_____

_____

_____

_____

_____

_____

_____

_____

/E List terminator

APPENDIX A

ANSWERS TO EXERCISES

Classroom Exercise Answer Sheet

Exercise: 2-1


```
ASMB,A,B,L,T
*
* THIS IS A SAMPLE BIT MANIPULATION PROBLEM
*
* CLASSROOM EXERCISE 2-1
*
       ORG 100B
START  LIA 1          READ SWITCH REGISTER
       STA SAVE       SAVE THE VALUE
       LDB CONST      SET -1 VALUE
       STB Z          IN Z
       SLA            SKIP IF BIT 0 = 0
       JMP END        BIT 0 = 1 TEST OVER
       ALF,RAR        PUT BIT 12 INTO POSITION 15
       SSA,RSS        SKIP IF BIT 15 = 1
       JMP B1         BIT 15 = 0
       CLB            SET B = 0
       STB Z          STORE IN Z
B1     LDA SAVE       RESTORE TEST VALUE
       AND M16        ISOLATE BITS 3, 2 & 1
       SZA            SKIP IF THEY ALL = 0
       JMP END        ONE OR MORE = 1
       CLB,INB        SET B = 1
       STB Z          STORE IN Z
END    HLT 33B
       JMP START      RESTART
M16    OCT  16        AND MASK
Z      NOP           RESERVE A PLACE FOR Z
SAVE   NOP           RESERVE A PLACE FOR SWITCH VALUE
CONST  OCT -1         CONSTANT
       END
```

Laboratory Exercise Answer Sheet

Exercise: 3-1

```
ASMB,A,B,L,T
*
* Laboratory Exercise 3-1
*
        ORG 100B
CNTR    BSS 1           CONSTANTS
M100    DEC -100        CONSTANTS
*
START   JSB READ        GET A CHARACTER
        SZA,RSS         IS CHARACTER A BLANK?
        JMP *-2         YES, IGNORE
AGAIN   LDB M100        NO, GET -100 AND
        STB CNTR        SET COUNTER
GO      JSB PUNCH       PUNCH CHARACTER
        JSB READ        GET ANOTHER CHARACTER
        SZA             IS CHARACTER A BLANK?
        JMP AGAIN       NO, RESET COUNTER
        ISZ CNTR        YES, ADD 1 TO COUNTER, ZERO?
        JMP GO          NO, PUNCH THE CHARACTER
        HLT 77B         YES, HALT THE COMPUTER
        JMP START       RESTART PROGRAM
*
READ    NOP             ENTER
        STC 13B,C       START READER
        SFS 13B         IS FLAG SET?
        JMP *-1         NO, WAIT
        LIA 13B         YES, GET CHARACTER
        JMP READ,I      RETURN
*
PUNCH   NOP             ENTER
        OTA 16B         OUTPUT CHARACTER
        STC 16B,C       START PUNCH
        SFS 16B         IS FLAG SET?
        JMP *-1         NO, WAIT
        JMP PUNCH,I     YES, RETURN
        END
```

Laboratory Exercise Answer Sheet

Exercise: 4-1

```
ASMB,R,B,L
*
* THIS IS A FUNCTION
*
* LABORATORY EXERCISE 4-1
*
       NAM ILIMT
       ENT ILIMT
       EXT .ENTR
UPPER  BSS 1          ADDRESS OF UPPER
LOWER  BSS 1          ADDRESS OF LOWER
TEST   BSS 1          ADDRESS OF TEST
ILIMT  NOP            ENTER HERE
       JSB .ENTR      TRANSFER THE
       DEF UPPER      PARAMETER ADDRESSES
       DLD UPPER,I    GET UPPER LIMIT
       FSB TEST,I     SUBTRACT TEST
       SSA,RSS        SKIP IF NEGATIVE (OVER)
       SZA,RSS        SKIP IF NOT ZERO
       JMP EQROV      EQUAL OR OVER UPPER LIMIT
       DLD LOWER,I    GET LOWER LIMIT
       FSB TEST,I     SUBTRACT TEST
       SSA            SKIP IF POSITIVE (UNDER)
       SZA,RSS        SKIP IF NOT ZERO
       JMP EQRUN      EQUAL OR UNDER LOWER
       CLA            SET A = Ø
       JMP ILIMT,I    RETURN TO CALLING PROGRAM
EQROV  CLA,INA        SET A = 1
       JMP *-2
EQRUN  CCA            SET A = -1
       JMP *-4
       END
```

Homework Assignment

Lesson: **ASSEMBLER PROGRAMMING TECHNIQUES**

Assignment: 4-1

1. An entire array of "real number" data is to be output in binary form using Unit #4. The data block has 100 elements and may be referenced by the label "BUFFR". In the space below, write the Assembly language calling sequence that will enable the FORMATTER to complete this operation. Declare all EXTernal symbols used and consider the instruction following the calling sequence to be a NOP for all questions in this assignment.

```
LABEL   OP CODE   OPERAND    REMARKS
        EXT       .BIO.,.RAR.,.DTA.
N       DEC       100
UNIT4   OCT       4
ADDRS   DEF       BUFFR
START   LDA       UNIT4      UNIT#4
        CLB                  Ø TO "B" FOR OUTPUT
        JSB       .BIO.      INITIAL CALL(BINARY)
        LDA       N          # OF ELEMENTS
        LDB       ADDRS      ARRAY ADDRESS
        JSB       .RAR.      DATA CALL
        JSB       .DTA.      TERMINATION CALL
        NOP                  (NEXT INSTRUCTION)
```

2. An entire array of "real numbers" is to be output on Unit #4 as "formatted" data. The array name is SAM and it has 10 elements. The Format to be used should provide the following output form.

±XXX.XX

In the space below write the Assembly language calling sequence that will enable the FORMATTER to complete this operation.

```
LABEL   OP CODE   OPERAND    REMARKS
        EXT       .DIO.,.RAR.,.DTA.
TEN     DEC       10
UNIT4   OCT       4
FMT     ASC       3,(F7.2)
ADDRS   DEF       SAM
          -
START   LDA       UNIT4      UNIT#4
        CLB                  Ø TO "B" FOR OUTPUT
        JSB       .DIO.      INITIAL CALL(FORMATTED)
        DEF       FMT        ASCII STRING ADDRESS
        DEF       EOL        END OF LIST ADDRESS
        LDA       TEN        # OF ELEMENTS
        LDB       ADDRS      ARRAY ADDRESS
        JSB       .RAR.      DATA CALL
        JSB       .DTA.      TERMINATION CALL
EOL     NOP                  (NEXT INSTRUCTION)
```

3. Assume that 2 integer values are to be output on Unit #4 as "formatted" data. The labels for the values are "J" and "K". The Format should provide 3 digits of output for "J" and 4 digits for "K".

   In the space below write the Assembly language calling sequence that will enable the FORMATTER to complete this operation.

```
LABEL   OP CODE   OPERAND     REMARKS
        EXT       .DIO.,.IOI.,.DTA.
UNIT4   OCT       4
FMT     ASC       4,(I3,I4)
        -
START   LDA       UNIT4       UNIT#4
        CLB                   Ø TO "B" FOR OUTPUT
        JSB       .DIO.       INITIAL CALL(FORMATTED)
        DEF       FMT         ASCII FORMAT ADDRESS
        DEF       EOL         END OF LIST ADDRESS
        LDA       J           FIRST
        JSB       .IOI.       DATA CALL
        LDA       K           SECOND
        JSB       .IOI.       DATA CALL
        JSB       .DTA.       TERMINATION CALL
EOL     NOP                   (NEXT INSTRUCTION)
```

4. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #1. Use Unit #5 for input.

```
LABEL   OP CODE   OPERAND     REMARKS
        EXT       .BIO.,.RAR.
N       DEC       100
UNIT5   OCT       5
ADDRS   DEF       BUFFR
        -
START   LDA       UNIT5       UNIT#5
        CLB,INB               1 TO "B" FOR INPUT
        JSB       .BIO.       INITIAL CALL(BINARY)
        LDA       N           # OF ELEMENTS
        LDB       ADDRS       ARRAY ADDRESS
        JSB       .RAR.       DATA CALL
        NOP                   (NEXT INSTRUCTION)
```

5. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #2. Use Unit #5 for input.

```
LABEL    OP CODE    OPERAND      REMARKS
         EXT        .DIO.,.RAR.
TEN      DEC        10
UNIT5    OCT        5
FMT      ASC        3,(F7.2)
ADDRS    DEF        SAM
         -
START    LDA        UNIT5        UNIT#5
         CLB,INB                 1 TO "B" FOR INPUT
         JSB        .DIO.        INITIAL CALL(FORMATTED)
         DEF        FMT          ASCII FORMAT ADDRESS
         DEF        EOL          END OF LIST ADDRESS
         LDA        TEN          # OF ELEMENTS
         LDB        ADDRS        ARRAY ADDRESS
         JSB        .RAR.        DATA CALL
EOL      NOP                     (NEXT INSTRUCTION)
```

6. In the space below write the Assembly language calling sequence that will allow the FORMATTER to read the tape produced as a result of Question #3. Use Unit #5 for this operation.

```
LABEL    OP CODE    OPERAND      REMARKS
         EXT        .DIO.,.IOI.
UNIT5    OCT        5
FMT      ASC        4,(I3,I4)
START    LDA        UNIT5        UNIT#5
         CLB,INB                 1 TO "B" FOR INPUT
         JSB        .DIO.        INITIAL CALL
         DEF        FMT          ASCII FORMAT ADDRESS
         DEF        EOL          END OF LIST ADDRESS
         JSB        .IOI.        DATA CALL FOR
         STA        J            FIRST ELEMENT
         JSB        .IOI.        DATA CALL FOR
         STA        K            SECOND ELEMENT
EOL      NOP                     (NEXT INSTRUCTION)
```

Laboratory Exercise Answer Sheet

Exercise: 5-1

```
ASMB,R,B,L
*
* LABORATORY EXERCISE 5-1
*
       NAM MESSG
       ENT START
       EXT .IOC.
START  NOP
       JSB .IOC.     SYSTEM CLEAR
       OCT 0
-------------------------------------------------
READ   JSB .IOC.     READ MESSAGE
       OCT 010005
       JMP *-2       REJECT
       DEF INBUF
       DEC -20
-------------------------------------------------
       JSB .IOC.     STATUS CHECK
       OCT 040005
       SSA           BUSY?
       JMP *-3       YES
-------------------------------------------------
       LDA INBUF+7   GET CHARACTER 15 & 16
       CPA ASTER     = TO ** ?
       JSB PRINT     YES - OUTPUT ERROR MESSAGE
       LDA INBUF     GET CHARACTERS 1 & 2
       CPA ATSGN     = TO @@ ?
       JMP DONE      YES
       JMP READ
-------------------------------------------------
*
PRINT  NOP           ENTER HERE
       JSB .IOC.     PRINT REQUEST
       OCT 020002
       JMP *-2       REJECT - PROBABLY BUSY
       DEF OUTBF
       DEC -13
       JMP PRINT,I
-------------------------------------------------
DONE   JSB .IOC.     SYSTEM STATUS CHECK
       OCT 040000
       SSA           ANYTHING BUSY?
       JMP *-3       YES - WAIT
       HLT 77B       NO - OK TO STOP
       JMP START+1
-------------------------------------------------
INBUF  BSS 10
OUTBF  ASC 7,ERROR MESSAGE
ASTER  ASC 1,**
ATSGN  ASC 1,@@
       END START
```

Laboratory Exercise Answer Sheet

```
ASMB,R,B,L
*
* LABORATORY EXERCISE 5-1 (OPTIONAL)
*
       NAM OPTIO
       ENT BEGIN
       EXT .IOC.,.DIO.,.IOI.,.DTA.
BEGIN  NOP
       JSB .IOC.      SYSTEM CLEAR
       OCT 0
       CLA
       STA COUNT      RECORD COUNTER
READ   JSB .IOC.      READ MESSAGE
       OCT 010005
       JMP *-2        REJECT
       DEF INBUF
       DEC -20
       JSB .IOC.      STATUS CHECK
       OCT 040005
       SSA            BUSY?
       JMP *-3        YES
       ISZ COUNT
       LDA INBUF+7    GET CHARACTER 15 & 16
       CPA ASTER      = TO ** ?
       JSB PRINT      YES
       LDA INBUF      GET CHARACTERS 1 & 2
       CPA ATSGN      = TO @@ ?
       JMP DONE       YES
       JMP READ
*
PRINT  NOP            ENTER HERE
       LDA UNIT       CALL TO FORMATTER
       CLB
       JSB .DIO.
       DEF FMT
       DEF EOL
       LDA COUNT
       JSB .IOI.
       JSB .DTA.
EOL    JMP PRINT,I    RETURN
*
UNIT   OCT 2
COUNT  BSS 1
FMT    ASC 11,("ERROR MESSAGE #"I6)
*
DONE   JSB .IOC.      SYSTEM STATUS CHECK
       OCT 040000
       SSA            ANYTHING BUSY?
       JMP *-3        YES - WAIT
       HLT 77B        NO - OK TO STOP
       JMP BEGIN+1
INBUF  BSS 10
OUTBF  ASC 7,ERROR MESSAGE
ASTER  ASC 1,**
ATSGN  ASC 1,@@
       END BEGIN
```

Homework Exercise Answer Sheet

1. a. Input Output Control (IOC)
   b. Input Output Driver Routines (I/O Drivers)
   c. The Relocatable Loader (Loader)
2. High memory
3. The basic binary loader
4. a)
5. So that if we decide to change or upgrade our system (with the addition of new peripheral devices, for instance) we do not have to re-write all existing programs. We simply reconfigure BCS.
6. c)
7. 4 words per entry
8. a. Initiator section
   b. Continuator section
9. 2nd word: read or write operation?
   on what device?
   ASCII or binary?
   3rd word: reject address
   4th word: where is the data?
   5th word: how many words or characters?
10. b)
11. Reg. A = Physical status of equipment
    Reg. B = Cause of the reject
12. Reg. A = Physical status of equipment
    Reg. B = Transmission Log
13. To make sure all devices have completed their I/O operations before halting the computer.
14. Causes the termination of a previously issued input or output operation before all data is transmitted.
15.
```
INBUF   BSS   10
        JSB   .IOC.
        OCT   010015
        JSB   XAMIN
        DEF   INBUF
        DEC   10 (WDS) or -20 (CHARS.)
```

CONFIGURATION DIAGRAM

SYSTEM SERIAL NO. _____ DATE 9/10/68

Higher ← PRIORITY → Lower

FIRST WORD OF AVAILABLE MEMORY (FWAM) 17677

LAST WORD OF AVAILABLE MEMORY (LWAM) 24

							TELEPRINTER	PHOTOREADER	PUNCH							I/O INTERFACE CARD
DRIVER IDENTIFICATION							D.00	D.01	D.02							
INTERRUPT LOCATION	10	11	12	13	14	15	16	17	20	21	22	23	24	25	26	27
LINKAGE LOCATION							21	22	23							
INTERRUPT IDENT.							I.00	I.01	I.02							
EQUIPMENT TABLE →							11	7	10							

	STANDARD UNIT TABLE																
1.	KEYBOARD INPUT							√									
2.	TELEPRINTER OUTPUT							√									
3.	PROGRAM LIBRARY								√								
4.	PUNCH OUTPUT									√							
5	INPUT								√								
6.	LIST OUTPUT							√									

NOTE: ASSIGN STANDARD UNITS BY PLACING A CHECKMARK AT THE CORRECT X-Y INTERSECTION.

COMPUTER PROGRAMMING COURSE

P.C.S. WORKSHEET

THESE ENTRIES APPLY ONLY TO THE "CONFIGURING" SYSTEM.		
HS	INP ? 17	(ENTRY = DEVICE CHANNEL NUMBER)
HS	PUN ? 20	

FWA MEM ? 24

LWA MEM ? 17677

*LOAD	Load drivers in any order.	
D.XX		
XXXXX	XXXXX	This sequence continues until all drivers have been processed.
*LOAD		
IOC	Load the IOC Module.	
XXXXX	XXXXX	

*TABLE ENTRY

UNIT REF #	EQT ?	(Push RUN and type;) (FORMAT = CHAN #, D.XX)
(7)	17, D.Ø1	
(10)	2Ø, D.Ø2	
(11)	16, D.ØØ	
(12)		
:		
ETC	/E	Terminates equipment table entries.

Classroom Exercise Guide

Lesson: INTRODUCTION TO HP ASSEMBLER LANGUAGE

Exercise: 7-1

A. OBJECTIVE

1. To provide a basic exercise using HP Assembly language that will enable the student to use the techniques presented in class. This checkpoint exercise will uncover problems in understanding that might not be apparent to the student at the outset.

B. PROBLEM

1. Using Assembly language programming techniques write a program that will:

   a. Add the contents of a location called IDATA to the contents of a location called ISUM.

   b. Add the contents of a memory location called JDATA to the contents of a location called JSUM.

   c. Store the result of step "a" to location JSUM.

   d. Store the result of step "b" to location ISUM.

   e. At the completion of steps "c" and "d" halt the computer. Provide a method to restart the program without manually selecting the program starting address.

   f. The following constants should be used:

      IDATA = -32767_{10}, JDATA = 9678_{10}, ISUM = -1,
      JSUM = 597_{10}

C. PROCEDURE

1. Using the techniques and the instructions presented in Lesson VII up to this point, create the Assembly language program to solve the problem outlined in step B. Remember all programs must begin with a control statement and each must have a name. Use the simulated coding form provided below for your program. If you can not proceed ask your instructor for assistance.

Classroom Exercise Answer Sheet

Exercise: 7-1

```
ASMB,R,B,L,T
*
* CLASS EXERCISE 7-1
*
       NAM EXER7      PROGRAM NAME
       ENT GO         DEFINE PROGRAM ENTRY POINT
GO     NOP            ENTRY POINT
       LDA IDATA      GET IDATA
       ADA ISUM       ADD ISUM
       LDB JDATA      GET JDATA
       ADB JSUM       ADD JSUM
       STA JSUM       STORE TO JSUM
       STB ISUM       STORE TO ISUM
       HLT 77B        HALT COMPUTER
       JMP GO+1       PROGRAM RESTART
IDATA  DEC -32767     DATA VALUE
ISUM   DEC -1         DATA VALUE
JDATA  DEC 9678       DATA VALUE
JSUM   DEC 597        DATA VALUE
       END GO         END,DEFINE STARTING ADDRESS
```

SQT ? (ENTRY = EQUIP. TABLE ENTRY NUMBER)

-KYBD ?

11

-TTY ?

11

-LIB ?

7

-PUNCH ?

10

-INPUT ?

7

-LIST ?

11

DMA ?

0

*LOAD Load the relocatable loader module.

LOADR

XXXXX XXXXX

INTERRUPT LINKAGE ? (PUSH RUN and type)
(FORMAT = INT. LOC., LINK LOC, I.XX)

16,21,I.00

17,22,I.01

20,23,I.02

/E List terminator

Classroom Exercise Guide

Lesson: INTRODUCTION TO THE HP ASSEMBLER PROGRAM

Exercise: 7-2

A. OBJECTIVE

1. To provide a checkpoint exercise to re-enforce the basic Assembly language programming techniques discussed so far in the course.

B. PROBLEM

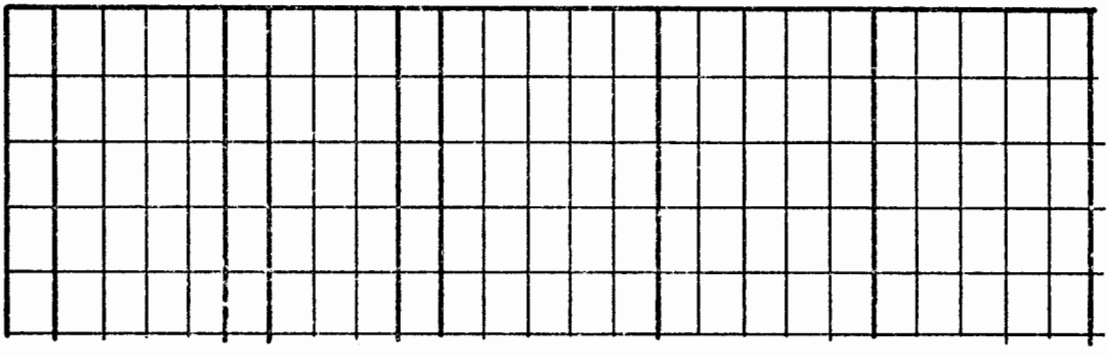
1. This problem has two parts. Assume each part to be a segment of a larger program so just supply the actual coding required without the control statements, entry points, etc.

   a. Clear bit 8 of register "A" and test bit 9. If bit 9 is 1, clear register "B". If bit 9 is a Ø, do not clear register "B". At the completion of the test restore all register "A" bits to their original positions.

   b. Clear register "B". Set register "E" to 1. Test register "A"; if positive, take the 2's complement of register "A". If negative multiply the contents of register "A" by 2 and provide an unconditional skip of the next instruction.

C. PROCEDURES

1. Using HP Assembler language prepare a program segment to implement the operations described in B. Use the Assembly language techniques outlined in Lesson VII and write the two program segments on the simulated coding forms provided below.



Classroom Exercise Answer Guide

Exercise: 7-2

```
*
* CLASS EXERCISE 7-2
*
*  FIRST PROBLEM
*
      ALF,ALF              POSITION BITS
      ERA,CLE,SLA,ELA      CLE,TEST BIT 9
      CLB                  BIT 9=1, CLEAR B
      ALF,ALF              RESTORE REGISTER A
*
*  SECOND PROBLEM
*
      CLB                  0 TO B
      CCE,SSA              1 TO E,TEST A
      ALS,CLE,SLA          A<0,0 TO E,SKIP
      CMA,INA              TWOS COMPLEMENT
*
*
```

11000 WOLFE ROAD, CUPERTINO, CALIFORNIA 95014, TELEPHONE 408 257-7000, TWX 910-338-0221
EUROPE: 1217 MEYRIN-GENEVA, SWITZERLAND CABLE "HEWPACKSA" GENEVA TEL. (022) 41.54.00

Printed in U.S.A. 6/72