8800 Program Library

PURPOSE: This Program Library will serve several purposes.

1) To store away programs used with the 8800 for use later.

2) A library of basic routines so that each one of us does not have to "reinvent the wheel". Subroutines are especially good for this purpose.

3) A source of sample programming so that we can share styles and techniques. This should be a storehouse of ideas when one of us is learning to use a new I/O device or instructions that have already been used by someone else in the group.

USING THE LIBRARY:

Programs will be classified into categories. Listings should not be removed from this binder except for making copies. The library will live near the computer.

ADDING TO THE LIBRARY:

Since it is better to have a bad copy of an idea than to have no record at all, there will be no strict rules about the format of entries to the library. However, nicely documented listings are generally more useful than illegible laconic scribblings. Imagination, Bruce?

If a program is untested, it should be marked as such. When someone later uses it he can cross out "untested" and leave his initials. if David. In the same spirit, known bugs should be listed with the programs, or added as they are discovered. If one finds a fix to these bugs, that should also be noted.

Comments and instructions with the programs are usually very worth while.

OWNERSHIP:

Most of the programs listed in here will be so basic as not to be copyrightable. Any electronic music programs will generally belong to ARP. Program listings copied into here from magazines, etc.. generally remain the possession of the original author. B.C. may occasionally add useful programs to here that belong to him but may be used but not sold by ARP for the time being(ownership remains his on these independently dereloped programs).

Bruce Gichowlas 10/7/75

HOW TO WRITE PROGRAMS THAT ARE ACCESSIBLE THROUGH THE MONITOR

It is often convenient to write programs that can be called and tested directly from the Monitor (also called Executive or Operating System in some contexts). This information will allow program writers to write programs that interface directly with the Processor Technology Monitor.

RUNNING YOUR PROGRAM FROM THE OPERATING SYSTEM

This is easy to do. Just use the EXEC command to get your program started, e.g.

```
EXEC D83E
```

if your program starts at D83E

RETURNING TO THE MONITOR

It is really a good idea that your program be able to return to the monitor when you are finished running it. Otherwise the monitor will have to be restarted, and the file directory will generally be lost. (In an emergency, one can get around this problem by restarting at the location of the monitor loop, i.e. F00C.)

The best way to return to the monitor is to use the statement

```
        JMP MONTR
```

where elsewhere in your program you have defined

```
MONTR EQU 0F00CH
```

Note that if your program is the type that ends up just repeating a loop ad infinitum, you can program in a provision to return if a certain sense switch is set. In fact, let's plan on using sense switch 0 on the main panel for this purpose.

(continued)

BC 9/9752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTIN: UART (PT)
PAGE 1 of 1

BC 9/22/75-1

ADDRESS	INSTR	MNEMONIC		COMMENTS
03F5	DB 00	OUT8	IN USTA	:OUTPUT A BYTE TO UART
03F7	E6 80		ANI H'80'	: DATA COMES FROM REG B (PT)
03F9	CA F5 03		JZ OUT8	
03FC	78		MOV A,B	LABELED OK IN PROC. TECH. COPY
03FD	D3 01		OUT UDAO	
03FF	C9		RET	
				B_{16} words
03E8	DB 00	IN8	IN USTA	*READ A BYTE FROM UART
03EA	E6 40		ANI H'40'	* DATA IS PUT IN REG B (PT)
03EC	CA E8 03		JZ IN8	
03EF	DB 01		IN UDAI	READ DATA
03F1	E6 7F		ANI H'7F'	STRIP OFF PARITY
03F3	47		MOV B,A	
03F4	C9		RET	
				D_{16} words
				* OUTPUT CR AND LF FOLLOWED BY
				*TWO DELETE CHARACTERS TO
				* GIVE PRINT HEAD TIME TO
				*RETURN (PT) DELETE CHARACTERS
				* NOT REALLY NEEDED IN OUR CASE
03D5	06 0D	CRLF	MVI B,13	CR
03D7	CD F5 03		CALL OUT8	
03DA	06 0A	LF	MVI B,10	LF
03DC	CD F5 03		CALL OUT8	
03DF	06 7F		MVI B,127	DEL
03E1	CD F5 03		CALL OUT8	
03E4	CD F5 03		CALL OUT8	
03E7	C9		RET	
				13_{16} words

BCO 9752

8080 MACHINE LANGUAGE/ ASSEMBLER
CODING SHEET

ROUTINE MONITOR (PT)
PAGE 3 of —

ADDRESS	INSTR	MNEMONIC		COMMENTS
02E5	31 00 01	START:	LXI SP,0100	;PROGRAM TO START WITH
02E8	C3 FD 02		JMP S458	

		Computer	Terminal
38	NDB1	S	S(H)
37	NDB2	S	1
	NSB (stop)	S	1 for 110 0 for other
35	NPB (parity)	S	S(F)
39	POE (odd/even)	S	S(J)
33	DB8	C	S(D)

IEK Gnd

Desired — 7 bits data
1 bit parity

NPB - Low (parity enable)
(35) - NSB - High (2 stop bits (at 1200))
36 - NDB2 - High] - (7 data bits)
37 - NDB1 - Low]
39 - POE - High - (Even Parity)

BC 9/22/75-2

ADDRESS	INSTR	MNEMONIC		COMMENTS
03CD	8D	CLER	CMP L	*BLANK OUT A PORTION OF MEMORY (PT)
03CE	C8		RZ	*LAST LOW ORDER TO CLEAR IS
03CF	36 20		MVI M,A' '	*PLACED IN ACCUMULATOR (R0)
03D1	23		INX H	
03D2	C3 CD 03		JMP CLER	
				8_{16} words
037A	-	IBUF	RES 83_{10}	*INPUT BUFFER (PT)
				53_{16} words
0378	-	ADDS (HCON)	RES 2	* ? (PT)
				2_{16} words
				* LINE READING ROUTINE
				* SEE PT DOCUMENTATION
0317	21 7A 03	READ	LXI H,IBUF	
031A	22 78 03		SHLD ADDS	
031D	1E 02		MVI E,2	
031F	CD E8 03	NEXT	CALL IN8	
0322	78		MOV A,B	
0323	FE 18		CPI 24	
0325	C2 2E 03		JNZ CR	
0328	CD D5 03		CALL CRLF	
032B	C3 17 03		JMP READ	
032E	FE 0D	CR	CPI ASCR	
0330	C2 49 03		JNZ DEL	
0333	7D		MOV A,L	
0334	FE 7A		CPI >IBUF	
0336	CA 17 03		JZ READ	
0339	36 0D		MVI M,ASCR	
033B	23		INX H	
033C	3601		MVI M,1	
033E	23		INX H	
033F	3F ED		MVI A,>IBUF+83	
0341	CD CD 03		CALL CLER	
0344	21 79 03		LXI H,IBUF-1	
0347	73		MOV M,E	
0348	C9		RET	
0349	FE 7A	DEL	CPI 127	(MAY WANT TO CHANGE THIS)

7A
83
ED

H'1A' (control Z - at least we can type it)

ADDRESS	INSTR	MNEMONIC	COMMENTS
034B	C2 5E03		JNZ CHAR
034E	3E 7A		MVI A,>IBUF
0350	BD		CMP L
0351	CA 1F03		JZ NEXT
0354	2B		DCX H
0355	1D		DCR E
0356	06 5F	BSPA	MVI B,H'5F
0358	CD F503		CALL OUT8
035B	C3 1F03		JMP NEXT
035E	FE 20	CHAR	CPI A' '
0360	DA 1F03		JC NEXT
0363	FE 5B		CPI A'Z'+1
0365	D2 1F03		JNC NEXT
0368	47		MOV B,A
0369	CD F503		CALL OUT8
036C	77		MOV M,A
036D	3E CB		MVI A,>IBUF+H'1
036F	BD		CMP L
0370	CA 5603		JZ BSPA
0373	23		INX H
0374	1C		INR E
0375	C3 1F03		JMP NEXT
			61_{16} words
0316	C9	NOOP	RET ROUTINE THAT DOES NOTHING
02FD	31 FD02	SYS8	LXI SP,AREA+18 SYS8: PT's main thing
0300	CD 1703		CALL READ
0303	23		INX H
0304	7E		MOV A,M
0305	FE 3A		CPI A'9'+1
0307	DA 1003		JC EOR Should be "LINE"
030A	CD 1603	LINE	CALL NOOP(VALC) TEMPORARY LABEL PLACEMENT OF LINE
030D	CD 1603		CALL NOOP(COMM)
0310	CD D503	EOR	CALL CRLF
0313	C3 FD02		JMP SYS8
			19_{16} words
02EB	—	AREA	RES 18_{10}
			12_{16} words

F086
F025
61

READ A CHARACTER (from Proc Tech)

```
①  IN    333
   206   206 /* status */
   ANI   346 /* data there? */
   002   002
   JZ    312 /* if not, jump */
   ①L → 364
   ①H → 003
   IN    333 /* get that character */
   207   207
   ANI   346 /* strip off parity */
   177   177
   RET   311 /* proc. Tech. moved the data to the */
             /* B register also */
```

14₈ words.

OUTPUT A CHARACTER (from Proc. Tech. with Mods)

```
   PUSH  SW  365 /* save the character */
①  IN    333
   206   206 /* status */
   ANI   346
   001   001 /* ready for data? */
   JZ    312 /* if not, jump back */
   ①L → 345
   ①H → 003
   POP   SW  361 /* get it back */
   PUSH  SW  365 /* resave it */
   JPE   352 /* if parity is already ok, jump */
   ②L → 374
   ②H → 003
   XRI   200 /* complement the highest bit */
   200
②  OUT   323 /* put it out */
   207   207
   POP   SW  361 /* put back original for caller */
   RET   311
```

23₈ words

NEW LINE (mostly original)

```
①  PUSH BC   305  /* save B&C */
   PUSH SWA  365  /* save SW & A */
   MVI A     076
   012       012  /* LF */
   CALL      315
   OAC_L →   341
   OAC_H →   003
   MVI A     076
   015       015  /* CR */
   CALL      315
   OAC_L →   341
   OAC_H →   003
   MVI B     006  /* 40₈ + 79₁₀ */
   157       157
   MVI A     076
   040       040  /* blank */
②  CALL      315
   OAC_L →   341
   OAC_H →   003
   DCR B     005
   CMP B     270
   JNZ       302
   ②L →      341
   ②H →      003
   MVI A     076
   015       015  /* CR */
   CALL      315
   OAC_L →   341
   OAC_H →   003
   POP SWA   361  /* restore SW & A */
   POP BC    301  /* restore B&C */
   RET       311
```

40₈ words

READ A LINE; (PUT IN BUFFER)

```
① PUSH DE       /* save DE */
  LXI
  IBUF_L
  IBUF_H
  ~~SHLD~~      /* ~~put address of IBUF in ADDS~~ */
  ~~ADDS_L~~
  ~~ADDS_H~~
  MVI E         /* E = [illegible]; 1 for the Carriage Return */
  1
② CALL
  RAC_L
  RAC_H
  CPI           /* look for cancel */
  030
  JNZ           /* if not, jump */
  ③_L
  ③_H
  CALL          /* give the user a new line */
  NL_L
  NL_H
  JMP           /* try again */
  ①_L
  ①_H
③ CPI           /* look for carriage return */
  015
  JNZ           /* if not, jump */
  ④_L
  ④_H
  MOV A,L
  CPI           /* see if we are still at first character */
  IBUF_L
  JZ            /* start over if we are */
  ①_L
  ①_H
  MVI M         /* put in the carriage return */
  015
  [illegible]
```

IBUF (80 $_{10}$ words) = 120 [illegible] $_8$ words

~~ADDS~~

~~2$_8$ words~~

```
④ CPI           /* is it control H? */
  010
  JNZ           /* if not, jump */
  ⑤_L
  ⑤_H
  MVI A
  IBUF_L
  CMP L         /* is it the first character */
  JZ            /* if so, ignore */
  ①_L
  ①_H
  MVI A
  134           /* backslash */
  CALL
  OAC_L
  OAC_H
  MVI A
  130           /* X */
  CALL
  OAC_L
  OAC_H
  DCX H         /* decrement things */
  DCR E
  MVI A
  134           /* backslash */
  CALL
  OAC_L
  OAC_H
  MVI A
  110           /* H */
  CALL
  OAC_L
  OAC_H
  JMP
  ②_L
  ②_H
⑤ CALL          /* echo it */
  OAC_L
  OAC_H
  MOV M,A       /* store */
  INX H         /* increment */
  INR E
  MOV A,E
  CPI           /* max length */
  120
  JZ
  ③_L
  ③_H
  MOV A,E       /* A = E */
  POP DE        /* restore DE */
  RET
```

JMP | BC 9/10/25-2

②$_L$

②$_H$

(end)

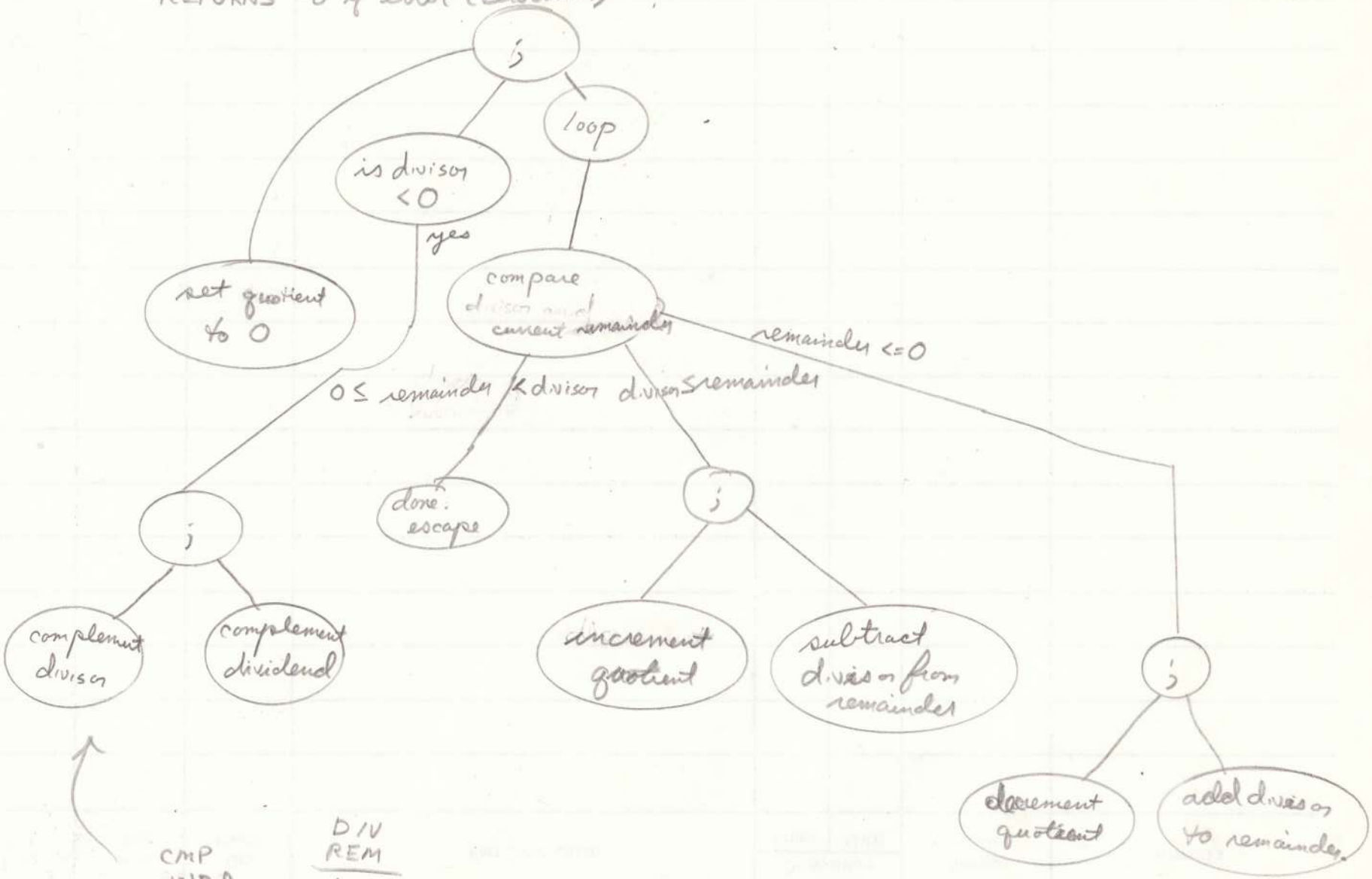
~~132~~ 127$_8$ words

FIXED POINT DIVIDE

BC-9/24/75

SWS / SWS

DIVISOR-R - A
DIVIDEND-R - B
QUOTIENT- B
RETURNS 0 if error (zero divide)



CMP
INRA

DIV
REM
+
0 or -

ADDRESS	INSTR	MNEMONIC		COMMENTS
	FE 7F	SWSABS	CPI H'7F'	ROUTINE TO CALCULATE THE ABSOLUTE VALUE
	D8		RC	OF A SIGNED SINGLE WORD. INPUT/OUTPUT IS IN A.
	2F		CMA	
	3C		INR A	
	C9		RET	
				6_8 words
	0E 00	SWSDIV	MVI C,0	ROUTINE TO DIVIDE ONE SWS INTO ANOTHER.
	FE 00		CPI 0	DIVISOR IN A DIVIDEND IN B
	C8		RZ	RETURNS - QUOTIENT IN C, REMAINDER IN B.
	FE 7F		CPI H'7F'	DIVISOR MAY BE DESTROYED.
	DA____		JC DIVL00	NOTE ZERO RETURNED ON ZERO DIVIDE.
	2F		CMA	HERE, IF A WAS NEGATIVE, WE WILL
	3C		INR A	COMPLEMENT (X-1)A, THEN B
	F5		PUSH SWA	PUT ASIDE FOR AWHILE
	78		MOV A,B	B = B* -1
	2F		CMA	
	3C		INR A	
	47		MOV B,A	
	F1		POP SWA	GET A BACK
	F5	DIVL00	PUSH SWA	DIVISION LOOP STARTS HERE; NOW WE
	3E 80	DIVL002	MVI A,'80'	KNOW A IS >0
	B8		CMP B	CHECK REMAINDER
	DA____		JC NEGREM	JUMP IF NEGATIVE
	F1		POP SWA	(WAS NOT NEGATIVE IF HERE)
	B8		CMP B	
	CA____		JZ DIVLSR	IF RESULT IS 0 or -,
	DA____		JC DIVLSR	JMP TO DIVLSR (DIV < REM)
	C9		RET	(WE ARE DONE)
	F1	NEGREM	POP SWA	GET BACK ORIGINAL A
	0D		DCR C	
	F5		PUSH SWA	SAVE
	80		ADD B	
	47		MOV B,A	
	C3____		JMP DIVL002	AVOID UNNECESSARY POP & PUSH
	F1	DIVLSR	POP SWA	
	0C		INR C	
	F5		PUSH SWA	SAVE

Memory Test Diagnostic Program

BC 9/14/75 -1

Parameters - start address
end address

pause or not - SS 15

continue sw - SS 14
continuous read mode SS 13
you supply the data SS 12

registers - HL - address memory
B - word being tested

write memory diags. check voltages on power supply.

what to do? maybe order some static 4K's for quick delivery. Demands MITS take their memory back.

IBUF = 037A
+81 = 51
03CB

have technician check dropouts for any patterns.

STACK 2E7=22 on return from IN8
2E8=03

02EB
+18
0303

02EB
+12
02FD

02EB				DF	
02EC	F7	EF	FD	EB	
02F0	00	00	00	00	
02F4	00	00	FB	FB	
02F8	18	22	03	03	RETURN TO READ FROM IN8
02FC	03	31	FD	02	RETURN TO SYS8 FROM READ
03	00	CD			

SYS8

IBUF 31 EB FA 18
1 ? ? ?

32
32
16000

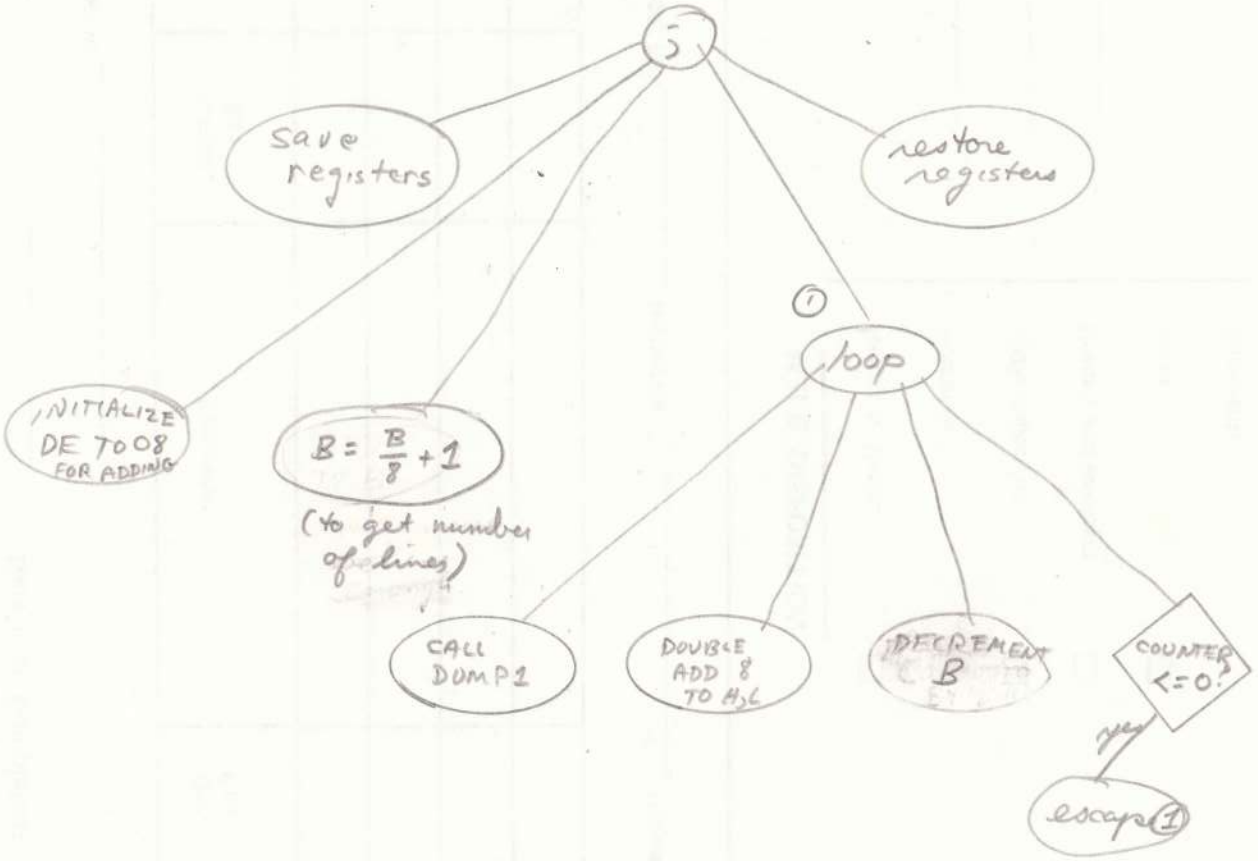
BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE MEM DIAG.
PAGE 1 of 1

ADDRESS	INSTR	MNEMONIC		COMMENTS
				FIND FF's in 4k of Memory
0100	3E 00		MVI A,0	
0102	D3 02		OUT 2	CLEAR BOXES
0104	D3 03		OUT 3	
0106	21	REDO	LXI HL	INITIALIZE HL
0107	FF		FF	
0108	FF		EF	
0109	23	AGAIN	INX HL	
010A	3E FF		MVI A,H'FF'	
010C	BE		CMP M	
010D	CA 26 01		JZ PROBLEM	
0110	3E BB	GOON	MVI A,H' '	LOW ORDER LAST LOCATION
0112	BD		CMP L	
0113	C2 09 01		JNZ AGAIN	
0116	3E EC		MVI A,H' '	HIGH ORDER LAST LOCATION
0118	BC		CMP H	
0119	C2 09 01		JNZ AGAIN	
011C	C9		RET	
				$1F_{16}$ words
011D	31 FF 03		LXI SP, 03FF	
0120	CD 00 01	DRIV	CALL 0100	DRIVER PROGRAM FOR ABOVE
0123	C3 23 01		JMP DRIV	
0126	7C	PROBLEM	MOV A,H	
0127	D3 02		OUT 2	
0129	7D		MOV A,L	
012A	D3 03		OUT 3	
012C	C3 10 01		JMP GOON	

BC 9/25/75-8

DUMP A COUPLE LINES OF HEX



STARTING ADDRESS COMES IN H,L

DESIRED COUNT COMES IN B

BC923754

DISPLAY IMAGE FOR SWTP CRT TERMINAL DIAGNOSTIC FORMAT
OR OTHER 16 x 32 DISPLAY

	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	1	A		9	9		S	9	9	9			B		9	9		S	9	9	9			C		9	9		S	9	9	9
2	2	D		9	9		S	9	9	9			E		9	9		S	9	9	9			M		9	9		S	9	9	9
3	3																															
4	4	H	L		9	9	9	9					S	P		9	9	9	9					C	,	A	C	,	S	,	Z	,	P
5	5																															
6	6	9	9	9	9		9	9		9	9		9	9		9	9		9	9		9	9		9	9		9	9			
7	7																															
8	8																															
9	9																															
A	10																															
B	11																															
C	12																															
D	13																															
E	14																															
F	15																															
10	16																															

will be 0 or 8

NOTE THAT IF THE LAST POSITION ON EACH LINE IS USED, THE CURSOR WILL BE AUTOMATICALLY POSITIONED AT THE START OF THE NEXT LINE.

~~Load Map~~ Memory Allocation

3/377 = 1777 High End of Memory
3/364 = 1764 Read a Character
3/341 = 1741 Output a Character
3/301 = 1701 New Line
3/152 = 1552 Read a Line
3/032 = 1432 Input Buffer

9/10 ~~3/363 - 3/377~~

3/354 - 3/377 - Output a Character

9/22 0400 Just Outside of Memory
- B
03F5 OUT8: P.T. Write UART - 1 char
- D
03E8 IN8: P.T. Read UART - 1 char
- 13
03D5 CRLF: Put out new line (uses OUT8)
- 8
03CD CLER: Clear a portion of memory with blanks
- 53
037A IBUF: Input Buffer
- 2
0378 ADDS: HCON: ? (PT)
- 61
0317 READ: a line. (uses IN8, OUT8 CRLF, CLER, IBUF, ADDS)
- 1
0316 NOOP: routine which does nothing

0316
- 19
02FD SYS8: P.T. main monitor routine (uses AREA, READ, NOOP)
- 12
02EB AREA: ? (PT)
- 6
02E5 STARTUP:

OUT

00	21	LXI H
01	00	00
02	F0	F0
03	DB	IN
04	06	06
05	E6	ANI
06	~~02~~80	~~02~~ 80
07	~~C2~~	~~JZ~~ JNZ (C2)
08	~~00~~ 03	~~00~~ 03
09	00	00
0A	7E	MOV A,M
0B	D3	OUT
0C	07	07
0D	7D	MOV A, L
0E	~~23~~	INX H
0F	FE	CPI
10	~~FF~~ 00	~~FF~~ 00
11	~~C3~~ C2	~~JMP~~ JNZ
12	03	~~00~~ 03
13	00	00
14	7C	MOV A, ~~L~~ H
15	FE	CPI
16	~~00~~ FF	00
17	~~C3~~ C2	JNZ
18	03	03
19	00	00
1A	FB	EI
1B	JMP	
1C	00	
1D	00	

OUT 2

D3
02
C2
03
00
FB
C3
00
00
(end)

IN

F126

0000 - ~~0037~~ 03FF

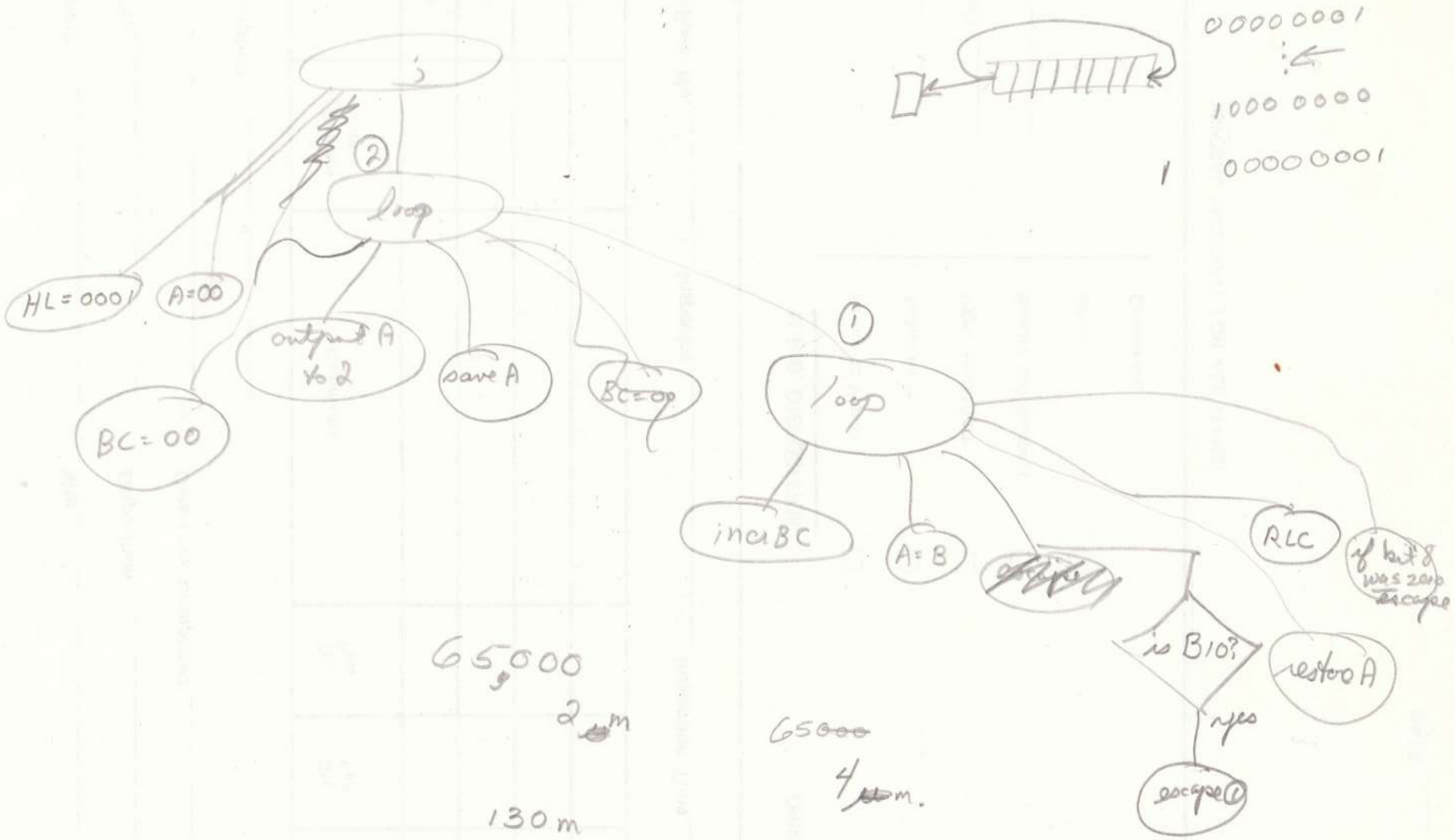
D000 - DFFF

F000 - FFFF

0000 ~~0038~~ is already claimed
D000 - D7FF is used.
F000 - FCBB is used.

~~0038~~
0039 - 03FF
D800 - DFFF
FCBC - FFFF - if protect is off

44 min.



65,000

2m

130m

65000

4m.

250 mill

```
:1CF65800CD93F23A5ED0B7C268F62A66D02268D03AB0D0FE45C271F6AF326AD0
:1CF67400AF3274D032F0D03A66D0226ED02A05D0224ED02A4ED0318ED07EFE01
:1CF69000CAE8F8EB13218ED03EA7CD86F00E0DCD34F571EB224ED03A70D0B7C2
:1CF6AC00B4F8CDD7F6C387F6CD9CF7218ED0CDC0F6C387F63A6AD0B7C2CDF63A
:1CF6C800A0D0FE20C8218ED0CDACF0CD13F2C9CDFEF03270D021ACD02272D07E
:1CF6E400FE20CA19F7FE2AC8CD07FBDAC9FACAAAFCCD30F7C2C9FA0E05215AD0
:1CF70000C7E1213230DC200F7EB2A6CD03A6FD077233A6ED077217 4D034CDFEF0
:1CF71C00CDF7F8DAF0FACD58FBF120DA4FFAC2F0FAC34FFA2A72D07EFE20C8FE
:1CF73800 3AC0232272D0C9CDF7F8AB7CA5BF7FA8DF7E270F7FE05DA85F7C2E8
:1CF75400F80E02AFC3DFFACD7AF13AA0D0FE20C0226ED03AACD0FE20C8C37BF7
:1CF77000CD7AFB3AACD0FE20CA8FFCEB2A6CD072237 3C3C7F6CD7AFB444DC3E7
:1CF78C00F7C36EFA218ED03A6FD CD22F2233A6ED0CD22F2227AD0CDFEF021AC
:1CF7A800D02272D07EFE20CA19F FE2AC3CD0AFBDAA FCCD30F7C2A5FCC319F7
:1CF7C400 1AB7CAF4F7FAEBF7E0FE05DADBF7C2E8F8CDC7F8C355F7CD77FB444D
:1CF7E000 2A68D0092268D0AFC3E FACD86F8AF0E01C3DFFACD77FB3AA0D0FE20
:1CF7FC00C0EB2A6ED0EB226ED071935F7C9A572A68D0192268D0C9CDD4F8C9CD
:1CF81800 77FBC464FC7DB7CA3BF FE02C464FCC33BF8CD77FBC464FC7D0FDC64
:1CF83400FC17FE08D464FC071717 471A80FE76CC64FCC313F8CD77FBC464FC7D
:1CF85000FE08D464FC1AFE40CA67F8FEC77DCA3BF8FA3EF8C33BF82929298512
:1CF86C00CDA5F8CD7AFBC464FC7 FE08D464FCC33EF8FE06CC93F8CDD4F8CD77
:1CF88800FB3CFE02D47DFC7DC31 F8CD77FBC464FC7DFE08D464FC2929291A85
:1CF8A400 5F2A72D07EFE2C23227 D0C26DFC7BC9FE01C2C4F8CD93F8E608C464
:1CF8C000FC7BE6F7CDD4F8CD77FB7D54CDD4F87AC313F8C92A68D07723268D0
:1CF8DC00 2A7AD02323CD22F2227A D0C93A70D0B7C21FF0CDA6F03E01C378F62A
:1CF8F800 72D07EFE20C0232272D0C3FAF8215BD02250D00602CD3AFAC94F5247
:1CF9140000004555550001444200 00FF4453000003445700 0005454E44000600
:1CF9300048 4C547652 4C43075252 430F52414C175241521F52455 4C9434D412F
:1CF94C0053544337444141274341 433F454900FB444900F34E4F500000584348
:1CF9680047EB585448 4CE3535048 4CF9504348 4CDF005354415802 4C4441580A
:1CF9840000505553 48C5504F500 1C1494E5800034443580 00B4441440009004 9
:1CF9A0004E5204444352054D4F5 4041444480414443885355429053424298 41
:1CF9BC004E41A0585241A84F524 1B0434D50B8525354C700414449C6414349CE
:1CF9D800535549D6534249DE414 49E6585249EE4F5249F643 049FE494E00DB
:1CF9F4004F5554D34D56490600 4A4D5000C343414C4CCD4C 4900 1 444100
:1CFA10003A5354410032534 84C44224C484C442A004E5A005 00084E43104300
:1CFA2C0018504F2050452850 00304D0038002A50D01AB7CA4CFA48CDEBF01AC8
:1CFA48001 3C33AFA3C13C9215AD0 2250D01111F90604CD3AFACAF8FA05CD3AFA
:1CFA6400CA6BFA04CD3AFA2113F8 E01CACBFACD3AFA2117F8CA6EFACD3AFA21
:1CFA80002AF8CA6EFA05CD3AFA21 9F8CA6EFACD3AFA217EF80E02CACBFA04CD
:1CFA9C003AFACAC6FACD05F9C2F0 AC6C05706033A5AD04FFB527ACA6BFA7914
:1CFAB8001 4FE4ACAC5FAFE43C2F0 A14147A21B4F80E033279D03E5A805F3ED0
:1CFAD400CE0057 1AB7C2F0FA3A70 D0060 0EB2A6ED009226ED0B7C83A79D0EBE9
:1CFAF0002190FC0E03C3DCFA215E 07EB7C2F0FA3A70D0B7CA3FF7C3C4F7FE41
:1CFB0C00D8FE5B3FD8CD58FB215A 0225 0DC05C22EFB041147FBCD3AFAC22EFB
:1CFB28006F2600C341FB3A74D047 1FFDCB7CA44FB3E053271D0CDD4F0373FC9
:1CFB44003CB7C941074200430144 2450348044C054D0600060012047 8FE0BD0
:1CFB600013232272D07EFE30D8FE 4DA5AFBFE41D8FE5BDA5AFBC9CDF7F82100
:1CFB7C0000276D0242277D02A72D 02BCDFEF03275D0237EFE21DA36FCFE2CCA
:1CFB980036FCFE2BCAA7FBFE2DC2 7FB3275D03A78D0FE02CA6DFC3E023278D0
:1CFBB400C38EFB4F3A78D0B7CA6D C79FE24C2CFFB232272D02A6ED0C30BFCFE
:1CFBD00027C2FBFB1100000E0323 272D07EFE0DCA8BFCFE27C2F2FB232272D0
:1CFBEC007EFE27C20CFC0DCA8BFC 35FC3D9FBFE30DA8BFCFE3AD22AFCCD46FC
:1CFC0800DA8BFCEB2A76D0AF3278D 3A75D0B7C221FC192276D0C384FB7D936F
:1CFC24007C9A67C31BFCCD0AFBCA 0FCDA8LFCC378FC3A78D0B7C26DFC2A76D0
:1CFC40007C1179D0B7C9CD58FB1B 1A015AD0FE48CA5EFCFE44C25AFCAF12CD9A
:1CFC5C00F1C9AF12CDB4F1C93E52 0000032A0D0C93E5332A0D0210000C340FC
:1CFC78003E55C36FFC3E56C366FC 4D32A0D0CDCDF6C93E41C36FFC3E4F32A0
:1CFC9400D03A70D0B7C80E03AFCD 4F80DC29CFCC93E4CC392FC3E4432A0D0CD
:0CFCB000CDF6C319F723C243F3C34 F3
:00
```

```
***SOFTWARE PACKAGE #1***

ASSME 0000

EXEC 0000

:1CF000002100D00E4EAF77230DC206F0318ED0CD25F0237EFE3ADA76F4CD0BF1
:1CF01C00CDC3F0CDA6F0C30CF021ACD02250D01E02CD8EF078FE18C23CF0CDA6
:1CF03800F0C325F0FE0DC257F07DFEACCA25F0360D233601233EFFCD86F021AB
:1CF05400D073C9FE7FC26CF03EACBDCA2DF02B1D065FCD9BF0C320F0FE20DA2D
:1CF07000F0FE5BD22DF047CD9BF0773EFDBDCA64F0231CC32DF0BDC8362023C3
:1CF08C0086F0DB00E640CA8EF0DB01E67F47C9DB00E680CA9BF073D3013C9060D
:1CF0A800CD9BF0060ACD9BF0067FCD9BF0CD9BF0C9CD93F2CDA6F02A66D0E911
:1CF0C40057F2060A3E043271D0CDD4F0C21AF4E92A50D03A71D04FCDEBF01A67
:1CF0E000131A6FC81305C2D4F004C91ABEC2F7F023130DC2EBF0C9130DC2F7F0
:1CF0FC000CC9AF1166D0060C1B1205C204F1C9CD12F1DA1AF4C9E10000268D0
:1CF118002252D0CDFEF021ABD0237EFE203FD0C221F12272D0CDF7F83FD0FE2F
:1CF13400C25DF11152D00E05237EFE2FCA4DF10DFA1AF4120013C33CF13E200D
:1CF15000FA58F11213C34FF1CDFEF83FD0115AD0CD58FB78FE053FD8015AD0CD
:1CF16C00B4F1D82266D0215AD0CD71F5CDF7F83FD0115ED0CD58FB78FE053FD8
:1CF18800015ED0CDB4F1D82268D0215ED0CD71F5B7C92100000AB7C8545D2929
:1CF1A4001929D630FE0A3FD85F16001903C39DF12100000AB7C829292929CDCB
:1CF1C000F1FE103FD8856F03C3B7F1D630FE0AD8D607C9CD1FF22150D046CD9B
:1CF1DC00F02346CD9BF0C9CDD3F1CDF6F1C9CD3CF2CDD6F12346CD9BF0C90620
:1CF1F800CD9BF0C92A66D03A69D0BCC20EF23A68D0BDC20EF2372232266D0C946
:1CF214003E0DB8C8CD9BF023C313F22150D0471F1F1F1FCD32F2772378CD32F2
:1CF2300077C9E60FC630FE3AD8C607C92150D00664CD4DF2060ACD4DF2C63077
:1CF24C00C9362F3490D24FF28023C944554D50F29B45584543F0B9454E5452F4
:1CF26800374649 4C45F2FF4C495354F58444454C54F5E14153534DF658504147
:1CF2840045F2D850524F4DF59D43555354E0003A5AD0B7CA1AF4C9CD93F23E10
:1CF2A000326AD0CDA6F03A67D0CDD3F13A66D0CDE3F13A6AD0326BD02A66D07D
:1CF2BC00D3077CFEEF7EC2C7F2DB06CDE3F1CDFCF1D83A6BD03DC2B5F2C3A3F2
:1CF2D800CD93F23A5ED0B7CA1AF42A66D0EB2A68D0D6007BD3077AFEFF1AC2F7
:1CF2F400F2DB0677231305C2EBF2C9CDA6F03A52D0B7CA79F3CDD8F3EBC224F3
:1CF3100C3A5AD0B7CA1DF43A59D0B7C239F3212CF4C320F43A5AD0B7CA43F32A
:1CF32C0066D07CB5CA4BF32131F4C320F42A57D0EB2152D0D50E057E1E130DC3
:1CF34800B5FCD12100D00E0D1A467778121323 0DC250F33A5AD0B7CA83F32A66
:1CF36400D02205D02207D07DB4CA72F33601AF3209D0C383F33AB0D0FE530E06
:1CF38000CA85F30E012100D0793259D0E5110500197EB7C2A3F3238623C2A3F3
:1CF39C0033332323C3B8F3E10E0546CD9BF00D23C2A6F3CDC4F3CDC4F3CDA6F0
:1CF3B800110400193A59D03DC289F3C9CDF6F1237E2BE5CDD3F1E17E2323E5CD
:1CF3D400E3F1E1C9AF3259D006061100D02152D00E05CDEBF0F5D51AB7C20EF4
:1CF3F000131AB7C20EF4EB11FAFF192257D07A3259D0E1F1110800 19EB05C8C3
:1CF40C00E1F3E1F1C204F411FBFF197AB7C9CDA6F02126F4CD13F2C31FF05748
:1CF4280041543F0D46554C4C0D4E4F204E4F0DCD93F2CD44F4DA1AF4C9A6F0C9
:1CF44400CDA6F0CD25F021ACD02272D0CDFEF0CDF7F8DA44F4FE2FC8CD58FB78
:1CF46000FE033FD8015AD0CDB4F1D87D2A66D077CD0EF2C350F40E0421AED023
:1CF47C007EFE30DA1AF4FE3AD21AF40DC27BF42250D0110CD0CD56F5D2B3F423
:1CF49800CD46F5210CD0CD4EF511ABD02A07D00E01CD34F5712207D0C31FF0CD
:1CF4B4000BF50E02CABCF40D462B3602224ED03AABD00DCAD1F490CAF4F4DAE4
:1CF4D000F42A07D0545DCD2FF52207D00E02CD3DF5C3F4F42F3C545DCD2FF5EB
:1CF4EC00CD34F536012207D02A4ED0360D2311ABD00E01CD34F5C31FF0215DD0
:1CF508002250D02A05D0CD28F5EB2A50D0EB3E04CD2FF5CD56F5D8C87ECD2FF5
:1CF52400C30EF5233E01BEC0C31FF0856FD024C91A13B9C8772BC334F51A1BB9
:1CF54000C8772BC33DF546234E2356235EC9732B722B712B70C906 01CE04B71A
:1CF55C009ECA61F5041B2B0DC25BF505C90E041AD601C35CF5CD46F5AFB8C8BB
:1CF57800C44EF5C05A5148 0630C377F5CDA6F0CD05F523CD13F2CDA6F0CD27F5
:1CF59400DBFFE680C023C38BF5CD93F2CDA6F03A66D0CDE3F116032A66D07DD3
:1CF5B000077ED3083E02D309CDD5F5AFD309DB06BECACEF5063FCD9BF015C2AE
:1CF5CC00F5C9CDFCF1D8C3A0F51E96AF3DC2D8F51DC2D7F5C9 D93F2CD05F522
:1CF5E8004ED02161D07EB7C2F5F5215DD02250D0EB210CD0CD56F52A4CD0DA43
:1CF60400F62207D03601EB2A05D0EB060D2B7D937C9A3E0DDAC4F6052BBEC212
:1CF62000F62B7D937C9ADA3BF6BE2323CA30F623CD46F5210CD0CD4EF5C9B8EB
:1CF63C00C22FF63209D0C9CD0EF5CC20F5EB2A4ED00E01CD34F52207D03601C9
```

```
:1CF65800CD93F23A5ED0B7C268F62A66D02268D03AB0D0FE45C271F6AF326AD0
:1CF67400AF3274D03270D02A66D0226ED02A05D0224ED02A4ED0318ED07EFE01
:1CF69000CAE8F8EB13218ED03EA7CD86F00E0DCD34F571EB224ED03A70D0B7C2
:1CF6AC00B4F6CDD7F6C387F6CD90F7218ED0CDC0F6C387F63A6AD0B7C2CDF63A
:1CF6C800A0D0FE20C8218ED0CDA6F0CD13F2C9CDFEF03270D021ACD02272D07E
:1CF6E400FE20CA19F7FE2AC8CD0AFBDAC9FACAAAFCCD30F7C2C9FA0E05215AD0
:1CF70000 7E1213230DC200F7EB226CD03A6FD077233A6ED077217 4D034CDFEF0
:1CF71C00CDF7F8DAF0FACD58FBFE20DA4FFAC2F0FAC34FFA2A72D07EFE20C8FE
:1CF73800 3AC0232272D0C9CDF7F81AB70A5BF7FA8DF7E270F7FE05DA85F7C2E8
:1CF75400F80E02AFC3DFFACD7AFB3AA0D0FE20C0226ED03AACD0FE20C8C37BF7
:1CF77000CD7AFB3AACD0FE20CA82FCEB2A6CD072237 3C3C7F6CD7AFB444DC3E7
:1CF78C00F7C36EFA218ED03A6FD0CD22F2233A6ED0CD22F2227AD0CDFEF021AC
:1CF7A800D02272D07EFE20CA19F7FE2AC8CD0AFBDAA5FCCD30F7C2A5FCC319F7
:1CF7C400 1AB7CAF4F7FAEBF7E0FE05DADBF7C2E8F8CDC7F8C355F7CD77FB444D
:1CF7E000 2A68D0092268D0AFC3E2FACD86F8AF0E01C3DFFACD77FB3AA0D0FE20
:1CF7FC00C0EB2A6ED0EB226ED07D935F7C9A572A68D019 2268D0C9CDD4F8C9CD
:1CF81800 77FBC464FC7DB7CA3BF8FE02C464FCC33BF8CD77FBC464FC7D0FDC64
:1CF83400FC17FE08D464FC07171747 1A80FE76CC64FCC313F8CD77FBC464FC7D
:1CF85000FE08D464FC1AFE40CA67F8FEC77DCA3BF8FA3EF8C33BF8292929851 2
:1CF86C00CDA5F8CD7AFBC464FC7DFE08D464FCC33EF8FE06CC93F8CDD4F8CD77
:1CF88800FB3CFE02D47DFC7DC313F8CD77FBC464FC7DFE08D464FC2929291A85
:1CF8A4005F2A72D07EFE2C232272D0C26DFC7BC9FE01C2C4F8CD93F8E608C464
:1CF8C000FC7BE6F7CDD4F8CD77FB7D54CDD4F87AC313F8C92A68D07723 2268D0
:1CF8DC002A7AD02323CD22F2227AD0C93A70D0B7C21FF0CDA6F03E01C378F62A
:1CF8F80072D07EFE20C0232272D0C3FAF8215BD02250D00602CD3AFAC94F5247
:1CF9140000004551550001444200 00FF44530000034457000005454E44000600
:1CF9300048 4C5476524C43075252430F52414C175241521F52455 4C9434D412F
:1CF94C00535443374441412743 4D433F454900FB444900F34E4F500000584348
:1CF9680047EB5854484CE3535048 4CF95043484CDF0053544158024C4441580A
:1CF9840000505553 48C5504F5000C1494E5800034443580000B444144000900 49
:1CF9A0004E52044443520 54D4F56404144448041444388535542905342429841
:1CF9BC004E41A0585241A84F5241B0434D50B8525354C700414449C6414349CE
:1CF9D800535549D6534249DE414E49E6585249EE4F5249F6435049FE494E00DB
:1CF9F4004F5554D34D5649060 04A4D5000C343414C4CCD4C5849000 14C444100
:1CFA1000 3A535441003253484C44224C484C442A004E5A005A00084E43104300
:1CFA2C0018504F2050452850 00304D0038002A50D01AB7CA4CFA48CDEBF01AC8
:1CFA480013C33AFA3C13C9215AD02250D01111F90604CD3AFACAF8FA05CD3AFA
:1CFA6400CA6BFA04CD3AFA2113F80E01CACBFACD3AFA2117F8CA6EFACD3AFA21
:1CFA800002AF8CA6EFA05CD3AFA2149F8CA6EFACD3AFA217EF80E02CACBFA04CD
:1CFA9C003AFACAC6FACD05F9C2F0FAC6C05706033A5AD04FFE527ACA6BFA7914
:1CFAB80014FE4ACAC5FAFE43C2F0FA141 47A21B4F80E033279D03E5A805F3ED0
:1CFAD400CE00571AB7C2F0FA3A70D00600EB2A6ED009226ED0B7C83A79D0EBE9
:1CFAF0002190FC0E03C3DCFA215ED07EB7C2F0FA3A70D0B7CA3FF7C3C4F7FE41
:1CFB0C00D8FE5B3FD8CD58FB215AD02250D005C22EFB041147FBCD3AFAC22EFB
:1CFB280006F2600C341FB3A74D04711FFD0B7CA44FB3E053271D0CDD4F0373FC9
:1CFB44003CB7C941074200430144024503480 44C054D06000600120478FE0BD0
:1CFB600013232272D07EFE30D8FE3ADA5AFBFE41D8FE5BDA5AFBC9CDF7F82100
:1CFB7C0002276D0242277D02A72D02BCDFEF03275D0237EFE21DA36FCFE2CCA
:1CFB980036FCFE2BCAA7FBFE2DC2B7FB3275D03A78D0FE02CA6DFC3E023278D0
:1CFBB400C38EFB4F3A78D0B7CA6DFC79FE24C2CFFB232272D02A6ED0C30BFCFE
:1CFBD00027C2FBFB1100000E03232272D07EFE0DCA8BFCFE27C2F2FB232272D0
:1CFBEC007EFE27C20CFC0DCA8BFC535FC3D9FBFE30DA8BFCFE3AD22AFCCD46FC
:1CFC0800DA8BFCEB2A76D0AF3278D03A75D0B7C221FC192276D0C384FB7D936F
:1CFC24007C9A67C31BFCCD0AFBCA0BFCDA8BFCC378FC3A78D0B7C26DFC2A76D0
:1CFC40007C1179D0B7C9CD58FB1B1A015AD0FE48CA5EFCFE44C25AFCAF12CD9A
:1CFC5C00F1C9AF12CDB4F1C93E52210000 32A0D0C93E5332A0D0210000C340FC
:1CFC78003E55C36FFC3E56C366FC3E4D32A0D0CDCDF6C93E41C36FFC3E4F32A0
:1CFC9400D03A70D0B7C80E03AFCDD4F80DC29CFCC93E4CC392FC3E4432A0D0CD
:0CFCB000CDF6C319F723C243F3C34AF3
:00
```

```
***SOFTWARE PACKAGE #1***

ASSME 0000

EXEC 0000

:1CF00000210DD00E4EAF77230DC206F0318ED0CD25F0237EFE3ADA76F4CD0BF1
:1CF01C00CDC3F0CDA6F0C30CF021ACD02250D01E02CD8EF078FE18C23CF0CDA6
:1CF03800F0C325F0FE0DC257F07DFEACCA25F0360D23601233EFFCD86F021AB
:1CF05400D073C9FE7FC26CF03EACBDCA2DF02B1D065FCD9BF0C32DF0FE20DA2D
:1CF07000F0FE5BD22DF047CD9BF077CEFDBDCA64F0231CC32DF0BDC8362023C3
:1CF08C0086F0DB00E640CA8EF0DB01E67F47C9DB00E680CA9BF078D301C9060D
:1CF0A800CD9BF0060ACD9BF0067FCD9BF0CD9BF0C9CD93F2CDA6F02A66D0E911
:1CF0C40057F2060A3E043271D0CDD4F0C21AF4E92A50D03A71D04FCDEBF01A67
:1CF0E000131A6FC81305C2D4F004C91ABEC2F7F023130DC2EBF0C9130DC2F7F0
:1CF0FC000CC9AF1166D0060C1B1205C204F1C9CD12F1DA1AF4C92100002268D0
:1CF11800225D0CDFEF021ABD0237EFE203FD0C221F12272D0CDF7F83FD0FE2F
:1CF13400C25DF11152D00E05237EFE2FCA4DF10DFA1AF4120013C33CF13E200D
:1CF15000FA58F11213C34FF1CDFEF83FD0115AD0CD58FB78FE053FD8015AD0CD
:1CF16C00B4F1D82266D0215AD0CD71F5CDF7F83FD0115ED0CD58FB78FE053FD8
:1CF18800015ED0CDB4F1D82268D0215ED0CD71F5B7C92100000AB7C8545D2929
:1CF1A4001929D630FE0A3FD85F16001903C39DF12100000AB7C829292929CDCB
:1CF1C000F1FE103FD8856F03C3B7F1D630FE0AD8D607C9CD1FF22150D046CD9B
:1CF1DC00F02346CD9BF0C9CDD3F1CDF6F1C9CD3CF2CDD6F12346CD9BF0C90620
:1CF1F800CD9BF0C92A66D03A69D0BCC20EF23A68D0BDC20EF237232266D0C946
:1CF21400E0DB8C8CD9BF023C313F22150D0471F1F1F1FCD32F2772378CD32F2
:1CF23000779E60FC630FE3AD8C607C92150D00664CD4DF2060ACD4DF2C63077
:1CF24C00C9362F3490D24FF28023C944554D50F29B45584543F0B9454E5452F4
:1CF26800374649 4C45F2FF4C495354F58444454C54F5E14153534DF658504147
:1CF28400 45F2D850524F4DF59D43555354E0003A5AD0B7CA1AF4C9CD93F23E10
:1CF2A000326AD0CDA6F03A67D0CDD3F13A66D0CDE3F13A6AD0326BD02A66D07D
:1CF2BC00D3077CFEEF7EC2C7F2DB06CDE3F1CDFCF1D83A6BD03DC2B5F2C3A3F2
:1CF2D800CD93F23A5ED0B7CA1AF42A66D0EB2A68D0D06007BD3077AFEFF1AC2F7
:1CF2F400F2DB0677231305C2EBF2C9CDA6F03A52D0B7CA79F3CDD8F3EBC224F3
:1CF31000 3A5AD0B7CA1DF43A59D0B7C239F3212CF4C320F43A5AD0B7CA4BF32A
:1CF32C0066D07CB5CA4BF32131F4C320F42A57D0EB2152D0D50E057E12130DC3
:1CF34800B5FCD12100D00E0D1A467778121323 0DC250F33A5AD0B7CA83F32A66
:1CF36400D02205D02207D07DB4CA72F33601AF3209D0C383F33AB0D0FE530E06
:1CF38000CA85F30E012100D0793259D0E5110500197EB7C2A3F3238623C2A3F3
:1CF39C0033332323C3B8F3E10E0546CD9BF00D23C2A6F3CDC4F3CDC4F3CDA6F0
:1CF3B800110400193A59D03DC289F3C9CDF6F1237E2BE5CDD3F1E17E2323E5CD
:1CF3D400E3F1E1C9AF3259D006061100D02152D00E05CDEBF0F5D51AB7C20EF4
:1CF3F000131AB7C20EF4EB11FAFF192257D07A3259D0E1F1110800 19EB05C8C3
:1CF40C00E1F3E1F1C204F411FBFF197AB7C9CDA6F02126F4CD13F2C31FF05748
:1CF42800415 43F0D46554C4C0D4E4F204E4F0DCD93F2CD44F4DA1AF4CDA6F0C9
:1CF44400CDA6F0CD25F021ACD0227 2D0CDFEF0CDF7F8DA44F4FE2FC8CD58FB78
:1CF46000FE033FD8015AD0CDB4F1D87D2A66D077CD0EF2C350F40E0421ABD023
:1CF47C007EFE30DA1AF4FE3AD21AF40DC27BF42250D0110CD0CD56F5D2B3F423
:1CF49800CD46F5210CD0CD4EF511ABD02A07D00E01CD34F5712207D0C31FF0CD
:1CF4B4000BF50E02CABCF40D462B3602224ED03AABD00DCAD1F490CAF4F4DAE4
:1CF4D000F42A07D0545DCD2FF52207D00E02CD3DF5C3F4F42F3C545DCD2FF5EB
:1CF4EC00CD34F536012207D02A4ED0360D2311ABD00E01CD34F5C31FF0215DD0
:1CF50800 2250D02A05D0CD28F5EB2A50D0EB3E04CD2FF5CD56F5D8C87ECD2FF5
:1CF52400C30EF5233E01BEC0C31FF0856FD024C91A13B9C87723C334F51A1BB9
:1CF54000C8772BC33DF546234E2356235EC9732B722B712B70C906010E04B71A
:1CF55C009ECA61F5041B2B0DC25BF505C90E041AD601C35CF5CD46F5AFB8C8BB
:1CF57800C44EF5C05A5148063 0C377F5CDA6F0CD05F523CD13F2CDA6F0CD27F5
:1CF59400DBFFE680C023C38BF5CD93F2CDA6F03A66D0CDE3F116032A66D07DD3
:1CF5B000077ED3083E02D309CDD5F5AFD309DB06BECACEF5063FCD9BF015C2AE
:1CF5CC00F5C9CDECF1D8C3A0F51E96AF3DC2D8F51DC2D7F5C9CD93F2CD05F522
:1CF5E8004ED02161D07EB7C2F5F5215DD02250D0EB210CD0CD56F52A4ED0DA43
:1CF60400F62207D03601EB2A05D0EB060D2B7D937C9A3E0DDA3AF6052BBEC212
:1CF62000F62B7D937C9ADA3BF6BE2323CA30F623CD46F5210CD0CD4EF5C9B8EB
:1CF63C00C22FF63209D0C9CD0EF5CC20F5EB2A4ED00E01CD34F52207D03601C9
```

11001
16 8 1 = 25

× 21
5
105

108 = 6C
9C+12

5)108

21 = 15₁₆ = 10101

To do: 101)0110 1100
u v

After normalization:

1010)110 11000 2⁴
u v

Calculate: m: number of leading zeroes in u.

multiply u':=u by 2^m

Compare u' & v

u' ≤ v u' > v

q := q*2 q := q*2

q := q+1

m := m-1

if m<0 quit, v is remainder when reshifted by v shift (d)

0000 101)0110 1100
u v

↑ determine # left zeroes in u
- # left zeroes in v + 1
= number of digits in answer - 1
= m

u' = u shifted (m+1) times to left.

Compare u' & v

q := q*2

u' ≤ v u' > v

q := q+1
v := v - u'

m := m-1

m = -1 ?

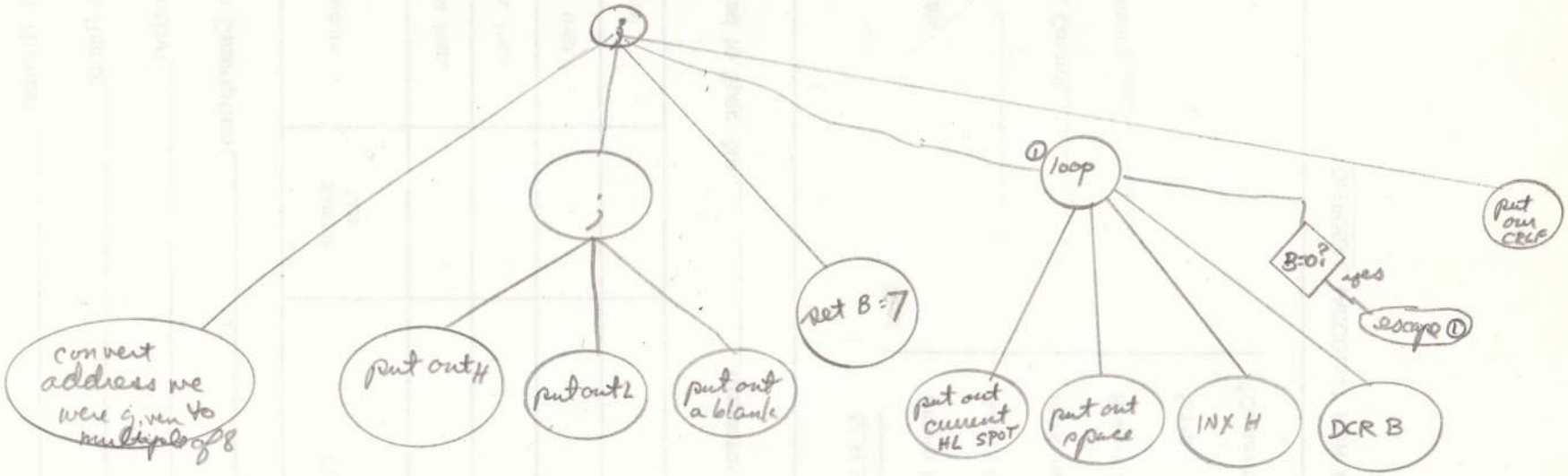
yes

done
(v is remainder)

u' := u'/2 (shift u' right)

DUMP ONE LINE OF HEX ROUTINE

BC 9/25( )-5



BC91952

8080 MACHINE LANGUAGE/ASSEMBLER CODING SHEET

ROUTINE DUMPN

PAGE 1 of —

ADDRESS	INSTR	MNEMONIC		COMMENTS
	E5	DUMPN	PUSH HL	DUMP N WORDS OF HEX (MORE OR LESS)
	C5		PUSH BC	ON THE TERMINAL. H and L are
	D5		PUSH DE	where TO BEGIN AND B IS THE
	11 08 00		LXI D,#08,H00#	DESIRED NUMBER OF WORDS.
	78		MOV A,B	B = $\frac{B}{8}+1$
	0F		RRC	
	0F		RRC	
	0F		RRC	
	3C		INR A	
	47		MOV B,A	
	CD ____	DNLOOP	CALL DUMP1	
	19		DAD D	
	05		DCR B	
	3E 00		MVI A,0	
	B8		CMP B;	JUMP
	DA ____		JC DNLOOP	IF B>0
	D1		POP DE	RESTORE REG's
	C1		POP BC	
	E1		POP HL	
	C9		RET	
				$1B_{16}$ words

8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE CONVERSIONS

BC 9/23/75-1

ADDRESS	INSTR	MNEMONIC	COMMENTS
	E6 0F	BIN1 ANI H'0F'	* CONVERT LOW ORDER FOUR BITS OF A
	C6 30	ADI 48	* WORD TO ASCII HEX REPRESENTATION
	FE 3A	CPI 58	* OF SAME, I.E.
	D8	RC	* 00 → '0'
	C6 07	ADI 7	* 1B → 'B' (PT)
	C9	RET	
			A, 6 words
			* PRINT OUT A WORD IN HEX
	F5	PHEX PUSH SWA	SAVE FOR LATER
	1F	PHEX RAR	
	1F	RAR	
	1F	RAR	
	1F	RAR	
	CD ____	CALL BIN1	
	CD ____	CALL OUT8	
	F1	POP SWA	RESTORE A,
	F5	PUSH SWA	BUT KEEP IT ON STACK
	CD ____	CALL BIN1	
	CD ____	CALL OUT8	
	F1	POP SW	RESTORE A FOR CALLER
	C9	RET	
			13/16 words
			*PRINT OUT A WORD AS A SIGNED
			*DECIMAL NUMBER (FOUR CHARACTERS)
		SDEC PUSH BC	SAVE
		PUSH SWA	
		MOV B,A	/* second copy */
		CPI H'77'	/* see if it's negative */
		JNC SDECNEG	/* jump if it is */
		MVI C,A' '	/* put sign in C temporarily
		JMP SDECSGN	
		SDECNEG CMA	/* complement the version in B */
		INR A	/* and add one to get the positive */
		MOV B,A	/* equivalent of the number */
		MVI C,A'-'	/* put sign in C temporarily */
		SDECSGN	/* check out order of magnitude

BC919752

8080 MACHINE LANGUAGE/ASSEMBLER CODING SHEET

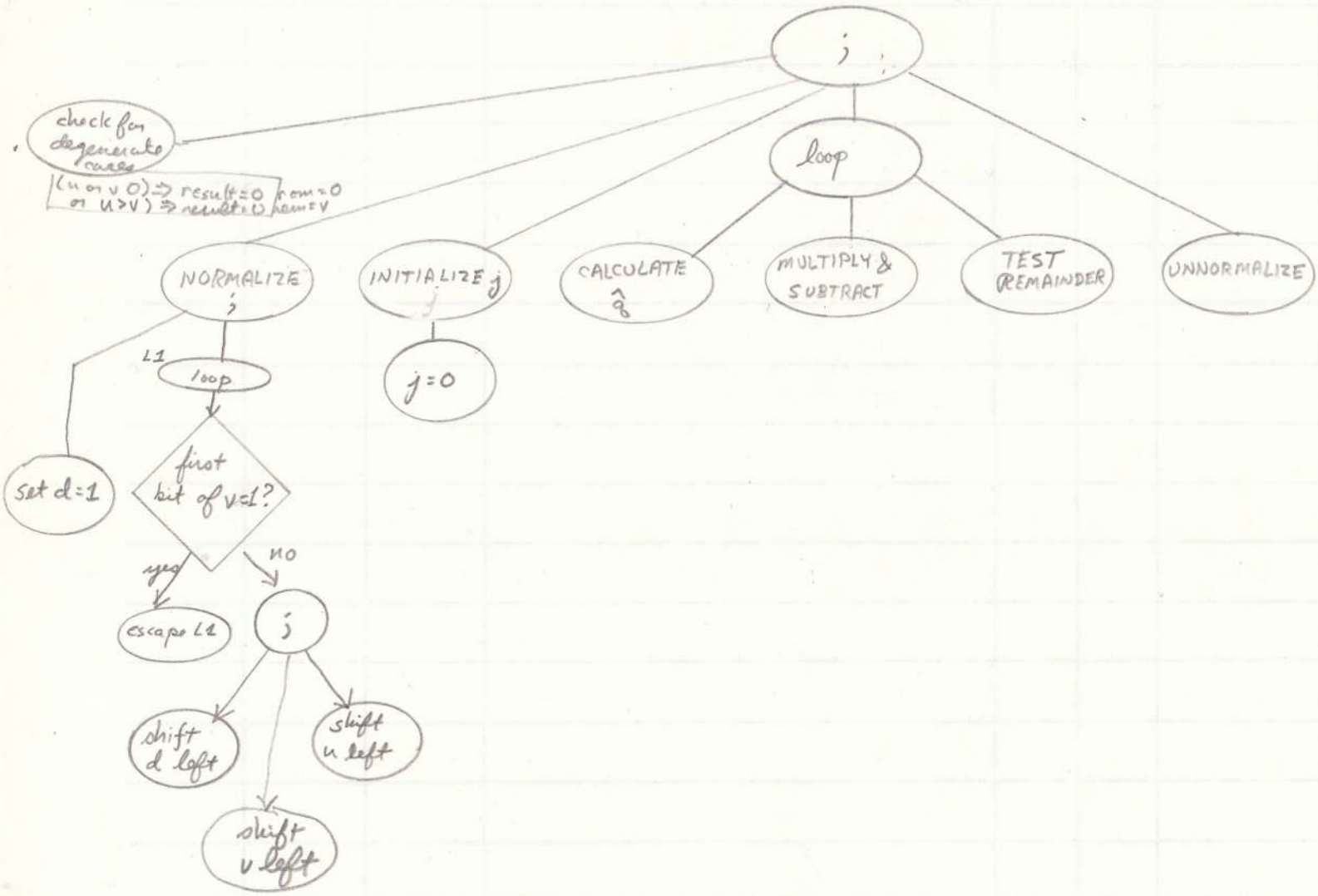
ROUTINE ARITHMETIC

PAGE 1 of ___

BC 9/24/75 4

ADDRESS	INSTR	MNEMONIC		COMMENTS
	C5	SUBCMP	PUSH BC	COMPARE TWO SW'S (IN A AND B) SET
				REGULAR BITS.
	4F		MOV C,A	
	3E 80		MVI A,'80'	
	A8		XRA B	
	47		MOV B,A	
	79		MOV A,C	
	EE 80		XRI '80'	
	B8		CMP B	PRODUCES THE COMPARISON RESULTS
	EE 80		XRI '80'	
	C1		POP BC	GET BACK ORIGINAL B & C.
	C9		RET	
				E_{16} words.

FIXED POINT UNSIGNED BINARY DIVISION (attempted adaptation from Knuth) BC 9/25/75 -1



SUBROUTINE REGISTER CONVENTIONS

It is a subroutine's responsibility to ensure that the contents of the B, C, D, E, H, L, and SP registers are as they were when control is returned (unless it returns arguments there). In general, subroutines will trample on the A register and the switches.

Most Processor Technology Routines do not respect these conventions so caveat emptor. If you call them, don't let your caller get screwed, though.

Exceptions may be made to the above where imperative, but such exceptions should be well documented.

ADDRESS	INSTR	MNEMONIC	COMMENTS
			*PRINT OUT A WORD AS A SIGNED
			*DECIMAL NUMBER (FOUR CHARACTERS)
	SDEC	PUSH BC	SAVE.
		~~PUSH SWA~~	
		MVI C,0	/* C will be number of leading blanks */
		MVI B,A' '	/* B will be sign. */
		CPI H'77'	
		JC ISPOS	/*jump if positive*/
		MVI B,A'-'	
		~~DCR B~~	~~/* has leading blank due to sign */~~
		CMA	/*multiply by -1 ... */
		INR	
	ISPOS	CPI H'63'	/*H'63' = 99 <=99? */
		JNC SDECSGN	/* if not we now know leading blanks, so jump */
		INR C	/* 1 for <99 */
		CPI H'09'	/* <= 9 */
		JNC SDECSGN	/* if not, we're set */
		INR C	
	SDECSGN	PUSH SWA	/*save absolute number */
		MVI A,0	/* put zero in A for comparing */
	RECMP	CMP C	/* compare C & A */
		JZ PRTSGN	/* ready to print sign, if they're the same*/
		DCR C	/* otherwise DCR C ... */
		CALL BLK1	/* put out a blank... */
		JMP RECMP	/* and try again */
	PRTSGN	MOV A,B	/* put out the sign */
		CALL NICEOUT	
		POP SWA	/*get back absolute number */
		CMP 'C7'	/* <= 199? */
		JNC	

BC 9/23/75-5

ADDRESS	INSTR	MNEMONIC		COMMENTS
				* PRINT OUT CURRENT SWITCH
				* SETTINGS (ALWAYS PRINTS 10_{10}
				* SPACES
	E5	PRSW	PUSH HL	SAVE...
	C5		PUSH BC	"
	F5		PUSH SWA	" (THIS ROUTINE WILL TRY NOT TO TOUCH A THING)
	F5		PUSH SWA	PUSH SWA AND BRING IT BACK
	C1		POP BC	IN B&C (SW WILL COME BACK IN B)
	21 ____		LXI H, PRSWBUF	
	3E __		MVI A, >PRSWBUF+11	/I believe the 11 is correct*/
	CD ____		CALL CLER	BUFFER IS NOW CLEARED
	21 ____		LXI H, PRSWBUF	RESET H&L TO START OF BUFFER
	78		MOV A, B	CARRY?
	E6 01		ANI H'01'	
	FE 00		CPI 0	
	CA ____		JZ NOCAR	
	36 43		MVI M, A'C'	
	23		INX H	
	36 2C		MVI M, A','	
	23		INX H	
	78	NOCAR	MOV A, B	AUXILIARY CARRY?
	E6 10		ANI H'10'	
	FE 00		CPI 0	
	CA ____		JZ NO AUX	
	36 41		MVI M, A'A'	
	23		INX H	
	36 43		MVI M, A'C'	
	23		INX H	
	36 2C		MVI M, A','	
	23		INX H	
	78	NOAUX	MOV A, B	SIGN?
	E6 80		ANI H'80'	
	FE 00		CPI 0	
	CA ____		JZ NOSIGN	
	36 53		MVI M, 'S',	
	23		INX H	
	36 2C		MVI M, ','	
	23		INX H	
	78	NOSIGN	MOV A, B	ZERO?

BC 9/23/75-6

ADDRESS	INSTR	MNEMONIC	COMMENTS
	E6 40		ANI H'40'
	FE 00		CPI 0
	CA ____		JZ NOZERO
	36 5A		MVI M,'Z'
	23		INX H
	36 2C		MVI M,','
	23		INX H
	78	NOZERO	MOV A,B PARITY?
	E6 04		ANI H'04'
	FE 00		CPI 0
	CA ____		JZ NOPAR
	36 50		MVI M,'P'
	23		INX H
	36 2C		MVI M,','
	23		INX H
	3E __	NOPAR	MVI A, >PRSWBUF
	BD		CMP L /* anything written? */
	CA ____		JZ PRSWPR /* If not, get ready to print */
	3E 2C		MVI A,',' /* otherwise we have an extra comma */
	2B	COMRETRY	DCX H /* to find. */
	BE		CMP M /* Is that it? */
	C2 ____		JNZ COMRETRY /* Try again if it isn't */
	36 20		MVI M,' ' /* Change the comma to a blank */
	21 ____		LXI H, >PRSWBUF+10 /* Point to the end */
	36 03		MVI M,'ETX' /* Put an ETX there */
	21 ____	PRSWPR	LXI H, >PRSWBUF /* Point to the start */
	CD ____		CALL STROUT /* Call string printer */
	F1		POP SWA /* restore */
	C1		POP BC
	E1		POP HL
	C9		RET /* all done */

78_{16} words

BC 9/23/75-3

ADDRESS	INSTR	MNEMONIC		COMMENTS
~~F5~~		NICEOUT	~~PUSH SWA~~	* THIS ROUTINE USES OUT 8, AND TAKES CHAR FROM A ^ BUT
C5		NICEOUT	PUSH BC	* DOESN'T DESTROY REGISTERS
CD	____		MOV A,B CALL OUT8	
C1			POP BC	
~~F1~~			~~POP SWA~~	
C9			RET	

6 7/16 words.

BC 9/25/75 - 3

ADDRESS	INSTR	MNEMONIC		COMMENTS
		~~DVIU1~~	~~PUSH SWA~~	B CONTAINS DIVISOR (W), C DIVIDEND (V);
	D5	DVIV1	PUSH DE	AFTERWARD B CONTAINS QUOTIENT; C REMAINDER
	3E 00		MVI A,0	(SEE BC 9/25/75-2) D in Q; E in M
	B8		CMP B	
	CA____		JZ DZRSLT	DOUBLE ZERO RESULT
	B9		CMP C	
	CA____		JZ D2RSLT	" " "
	79		MOV A,C	PUT DIVIDEND IN A
	B8		CMP B	IS IT LESS THAN B?
	DA____		JC LS1RSLT	LESS THAN ONE RESULT
	16 00		MVI D,0	Q=0
	5A		MOV E,D	M=0
	FE 80	TESTV	CPI 80	C IS STILL IN A; IS THE HIGH ORDER BIT ONE?
	D2____		JNC TESTU	IF SO JUMP
	07		RLC	OTHERWISE ROTATE ONE LEFT
	1D		DCR E	M=M-1
	C3____		JMP TESTV	TRY AGAIN
	78	TESTU	MOV A,B	SO MUCH FOR V; GET U
	FE 80	TESTUC	CPI 80	IS THE HIGH ORDER BIT ONE?
	D2____		JNC SAVU	IF SO, GO TO THE MAIN LOOP
	07		RLC	OTHERWISE ROTATE ONE LEFT
	3C		INR A	M=M-1
	C3____		JMP TESTUC	TRY AGAIN
	47	SAVU	MOV B,A	
	7A	DVLOOP	MOV A,D	MAIN LOOP STARTS HERE; SHIFT Q ONE LEFT
	07		RLC	
	57		MOV D,A	
	79		MOV A,C	PUT DIVIDEND (U) IN ACCUMULATOR
	B8		CMP B	COMPARE U AND V
	DA____		JC UBIG	JUMP IF U WAS BIGGER
	14		INR D	INCREMENT THE QUOTIENT
	D6		SUI B	SUBTRACT THE DIVISOR FROM THE DIVIDEND
	3E 00	UBIG	MVI A,0	
	BB		CMP E	IS M ZERO?
	CA____		JZ REGRSLT	IF SO, GO TO "REGULAR RESULT"
	1D		DCR E	M=M-1
	78		MOV A,B	SHIFT THE DIVISOR RIGHT
	0F		RRC	
	47		MOV B,A	
	C3____		JMP DVLOOP	

(more)

BC 9/25/75-4

ADDRESS	INSTR	MNEMONIC			COMMENTS
0E	00	D2RSLT	MVI	C,0	RETURN TWO ZERO'S
06	00	LS1RSLT	MVI	B,0	
C3	____		JMP	DIVRET	READY TO RETURN
42		REGRSLT	MOV	B,D	PUT QUOTIENT IN B
D1		DIVRET	POP	DE	RESTORE REGISTERS WE USED
C9			RET		

$4A_{16}$ words

BC 9/23/75-2

ADDRESS	INSTR	MNEMONIC	COMMENTS
	F5	BLK1 PUSH SWA	* PUT OUT A BLANK
	C5	BLK1 PUSH BC	(DOESN'T DESTROY REGISTERS)
	06 20	MVI B,A' '	
	CD ____	CALL OUTS	
	C1	POP BC	
	F1	POP SWA	
	C9	RET	
			8_{16} words
	F5	BLKN PUSH SWA	* PUT OUT 'N' BLANKS
	FE 00	BLKN CPI 0	* NUMBER COMES FROM ACCUMULATOR
	C8	RZ	
	3D	DCR A	
	CD ____	CALL BLK1	
	C3 ____	JMP BLKN	
	F1	BDONE POP SWA	
	C9	RET	
			A_{16} words
			* PUT OUT STRING OF CHARACTERS
			* TILL ONE ENCOUNTERS 'ETX'
			* STARTING ADDRESS IS IN H, L
	E5	STROUT PUSH HL	SAVE
	F5	PUSH SWA	SAVE
	7E	STRNXT MOV A,M	GET NEXT CHARACTER
	FE 03	CPI H'03'	IS IT ETX?
	CA ____	JZ STRRET	IF SO GET READY TO RETURN
	CD ____	CALL NICEOUT	OTHERWISE, PUT IT OUT
	23	INX H	GO TO THE NEXT ONE
	C3 ____	JMP STRNXT	
	F1	STRRET POP SWA	
	E1	STRRET POP HL	
	C9	RET	
			10_{16} words

(more)

ADDRESS	INSTR	MNEMONIC		COMMENTS
	90		SUB B	ACTUALLY THIS GIVES US -1 TIMES WHAT
	2F		CMA	WE WANT SO... WE MULTIPLY BY -1
	3C		INR A	
	47		MOV B, A	
	C3 ___		JMP DIVL002	AVOID UNNECESSARY POP & PUSH

34/16 words

LOADER

200	240	300	340 332
201	241	301	341 316 321
202	242	302	342 000
203	243	303	343 376
204	244	304	344 070 067 070
205	245	305	345 322 322
206	246	306	346 316 324
207	247	307	347 000
210	250	310 041	350 326 326
211	251	311 000	351 060
212	252	312 000	352 127
213	253	313 006	353 014
214	254	314 000	354 170
215	255	315 110	355 007
216	256	316 333	356 007
217	257	317 206	357 007
220	260	320 346	360 202
221	261	321 002	361 107
222	262	322 312	362 171
223	263	323 316 321	363 376
224	264	324 000	364 003
225	265	325 333	365 302 324
226	266	326 207	366 316
227	267	327 346	367 000
230	270	330 177	370 160
231	271	331 376	371 043
232	272	332 107	372 303
233	273	333 312	373 313 346
234	274	334 000	374 000
235	275	335 000	375 X
236	276	336 376	376 X
237	277	337 057 060	377 X

HEX BOOT STRAPPER

BC 9/11/75 -1

```
00   LXI H   21
     00      00   /*address to start */
     00      00
03 ② MVI B   06
04   00      00
08 ① IN      DB   /* is symbol there to be read? */
     00      00
     ANI     E6
     40      40
0C   JZ      CA
     ①L      08
     ①H      XX
     IN      DB
10   01      01
     CPI     FE   /* >= 0? */
     30      30
     JC      DA   /* NO, TRY AGAIN */
14   ①L      08
     ①H      XX
     CPI     FE   /* >= F+1? */
     47      47
18   JNC     D2   /* YES, TRY AGAIN */
     ①L      08
     ①H      XX
     CPI     FE   /* >= '9'+1 ? */
1C   3A      3A
     JC      DA   /* if NO, jump ahead */
     ③L      27
     ③H      XX
20   CPI     FE   /* >= 'A' ? */
     41      41
     JC      DA   /* if not, try again */
     ①L      08
24   ①H      XX
     SUI     D6   /* ASCII BIAS */
     07      07
27 ③ SUI     D6   /* get down to real number */
28   30      30
     MOV C,A  4F   /* put in C temporarily */
     MOV A,B  78   /* look at B */
     CPI     FE   /* is it zero? */
2C   0       00
     JNZ     C2   /* if not, jump again */
     ④L      3C
     ④H      XX
30   INRB    04   /* B was zero if we are here, make it one */
     MOV A,C  79   /* get back what was in C */
     RLC     07   /* shift it four times */
     RLC     07
34   RLC     07
     RLC     07
```

```
05 ⑤ MOV A,L  7D
     OUT     D3   /* show us where we are */
     03      03
```

```
     MOV M,A  77   /* put it out to memory */
     OUT     D3   /* also to channel two */
38   02      02
     JMP     C3   /* move on to next symbol */
     ⑤L      05
     ⑤H      XX
3C ④ DCR B   05   /* B was one, make it zero */
     MOV A,M  7E   /* get our partial result from memory */
     ADD C    81   /* add C to it */
     MOV M,A  77   /* store it */
40   OUT     D3   /* also to channel two */
     02      02
     INX H   23
     JMP     C3
44   ⑤L      05
45   ⑤H      XX
```

46_{16} words = 70 words = 106_8 words

XX = 01

start at 0100 Hex

0 0

BC 9/16/75 -1

LXI H		21
00		00
FO or DO		FO or DO
① MOV M,L		75
INR L		2C
JMP		C3
①L		03
①H		00

② LXI H	400	21	0	
00	⌒	00	1	] 14
~~FO~~ DO	—	DO	2	] 00
① MOV M,L	~~FO~~3	75	3	
INX H	—	23	4	
MOV A,L	—	7D	5	
CPI	—	FE	6	
FF	—	FF	7	
JNZ	—	C2!	8	
①L	—	03.	9	
①H	—	00	A	
MOV A,H	—	7C	B	
CPI	—	FE	C	
FF	—	FF	D	
~~JZ~~ JNZ	—	~~CA~~ C2	E	
②L	—	~~00~~03	F	
②H	—	00	10	
JMP	—	C3	11	
①L	—	~~03~~00.	12	
①H	—	00	13	

- starting address for getting in memory

DIFF

23
~~FB~~
7C
D3
02
C3
00
01

F

7 6 5 4 3 2 1 0

0-FF

D

7 6 5 4 3 2 1 0

0-FF

② 7.2 the worst

EC919752

8080 MACHINE LANGUAGE/ASSEMBLER CODING SHEET

ROUTINE _____

PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
00	21 00 F0		LXI H00F0
03	DB 06	L1	IN 06
05	E6 01		ANI 01
07	C2 0300		JNZ L1
0A	DB 07		IN 07
0C	77		MOVM,A
0D	7C		MOVA,H
0E	D3 02		OUT 02
10	23		INX HL
11	C3 0300		JMP L1
			LXI H00F0
			LXI B0021
		L1	IN 06
			ANI 01
			JNZ L1
			IN 07
			XOR B
			JNZ L1
			MOV M,A
			MOV A,H
			OUT 02
			INX HL
			JMP L1

0 if same

```
ASSM FE14

FE14 E5                 0010 HXANS PUSH H
FE15 CD E2 FF           0020  CALL QANDA
FE18 21 AC D0           0030  LXI H,IBUF
FE1B CD B8 FF           0040  CALL RHEX
FE1E E1                 0050  POP H
FE1F C9                 0060  RET
FE20                    0070 IBUF EQU 0D0ACH
FE20                    0080 QANDA EQU 0FFE2H
FE20                    0090 RHEX EQU 0FFB8H
ENTR DFFF
0/
```

MEMORY ALLOC BC-10/16/75-1

V13

		CATCH	FCBC - FCC2 - tentative DBL
FCBC - FCC5	A	~~free~~	A
FCC6 - FCCE	9	INTP	1F
FCCF - FCED	~~1F~~ 2 1D	OUTNC ~~free~~	1
FCEE - FD22	35	MCHRC	E
FD23 - FD35	13	WSYNC	9
FD36 - FD4E	19	RSYNC	41 - remaining unused space.
FD4F - FD5A	C	INNC	
FD5B - FD62 3	~~8~~ 9	~~CATCH~~ free	
~~FD63 - FD63~~	~~D~~	~~free~~ free (oom)	
FD64 - FD6D	A	RDNC	
FD6E - FD7D	10	QANDA	
FD7E - FD88	B	OUTTP	
FD89 - FD93	B	OUTNC	
FD94 - FD9E	B	TPDMP	
FD9F - FDB4	16	MVCHR	
FDB5 - FDC1	D	TELL	
FDC2 - FDED	2C	RHEX	
FDEE - FE0D	20	NEWOS	
FE0E - FE13	6	NLAB	
FE14 - FE ~~21~~ 1F	E	~~free~~ HXANS	
FE ~~22~~ 20 - FEA1	~~80~~ 82	WRFIL	
FEA2 - FED3	32	FNXT	
FED4 - FF0E	3B	CHGF	
FF0F - FF ~~32~~ 37	~~24~~ 29	DUMPR	
FF ~~38~~ 3F - FF3E	~~E~~ 7	~~HXANS~~ free	
FF3F - FFA3	65	RDFIL	B2 free
FFA4 - FFAF	C	HOTNC	FFB0 - FFB3 ~~EI~~ 4
~~FFB0 - FFB8~~	~~9~~	~~free~~	FFB4 - FFB8 free 5
FFB9 - FFFF	OSTV		

EC919752

8080 MACHINE LANGUAGE / ASSEMBLER CODING SHEET

ROUTINE _____

PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		TPDMP CALL INTP	
		OUT 03	
		CALL OUTNC	
		JMP TPDMP	
		INTP EQU 0FCC6H	
		OUTNC EQU 0FD89H	

BC919752 80 0 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE _____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		QANDA MOV A, M	
		CALL OUTNC	
		CPI 03	
		JNZ QANDA.	
		QANDA PUSH B H	
		WR MOV A, M	
		CALL OUTNC	
		INX H	
		CPI 03	
		JNZ WR	
		CALL RDNC	
		POP H	
		RET	
		RDNC EQU 0FD64H	
		OUTNC EQU 0FD5BH 89	
		RDNC PUSH H	
		PUSH D	
		PUSH B	
		CALL READ	
		POP B	
		POP D	
		POP H	
		RET	
		READ EQU 0F025H	

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE _____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		TQ	LXI H, DATA
			CALL QANDA
			JMP MONTR
		DATA	DW 'AT'
			DW 'KL'
			DW 'T '
			DW ' O'
			DW 'EM'
			DW '..'
			DW 032EH
		QANDA	EQU 0FD6EH
		MONTR	EQU 0F00CH

BC919752

8080 MACHINE LANGUAGE/ ASSEMBLER
CODING SHEET

ROUTINE _____
PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS	
		RHEX	~~PUSH D~~	HL POINTS TO FOUR ASCII CHARACTERS
			PUSH H	
			PUSH B	
			LXI D, 0H	
			MVI B, 4	
		LOOP4	MVI C, 4	
		SHFT4	STC	
			MOV A, E	
			RAL	NOW HAS 1 IN RIGHT MOST POSITION
			MOV E, A	
			MOV A, D	
			RAL	CARRY IS FROM E
			MOV D, A	
			JC CON	
			DCR E	SET RIGHT MOST BACK TO ZERO
		CON	DCR C	
			JNZ SHFT4	
			MOV A, M	CONVERT CHAR TO BIN (CRUDELY)
			SUI 37H	
			CPI 8	
			~~JNC~~ JP CON2	
			ADI 7	
		CON2	ADD E	ADD A to what's there
			MOV E, A	
			JNC CON3	
			INR D	~~E OVERFLOWED~~
		~~CON3~~	INX H	
			DCR B	
			JNZ LOOP4	
			POP B	
			POP H	
			~~POP D~~	
			RET	

37 1-31 57/a 37

a-61 57/a

✓ ed

BC919752

8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE _____

PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		THEX LXI H,DATA	
		CALL RHEX	
		LXI H,OUTP	
		MOV M,D	
		INX H	
		MOV M,E	
		JMP MONTR	
		DATA DW '21'	
		DW '43'	
		OUTP DW 0	
		MONTR EQU 0F00CH	
		RHEX EQU 0FDC2H	

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE OUTTP
PAGE — of —

ADDRESS	INSTR	MNEMONIC		COMMENTS
				Outputs word to tape
			PUSH SW	
FD7E	F5 ~~FD7E~~	OUTTP	~~PUSH PSW~~	Save A
FD7F	DB 06	TRY	IN 06	
FE01	17		~~ANI H'80~~ RAL	
FE02	DA 7FFD		~~JC~~ ~~JN~~ TRY	
FE05	F1		POP SW	
	9 —		~~POP PSW~~	
FE06	D3 07		Out 07	
FE08	C9 10 —		RET	
			WAIT 6,128,255	

~30

BC9[illegible]752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE _____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		QANDA2 MOV A,M	NICE OUT OUTNC
		CPI 3	NICE IN INNC
		JZ ANSW	
		MOV B,A	
		OUTNC PUSH B	
		MOV B,A	
		CALL OUT8	
		POP B	
		RET	
		INNC PUSH B	
		CALL IN8	
		MOV	
		OUTNC PUSH SW	
		L1 NICE IN 0	
		ANI RAL	
		JNC OUTNC	L1
		JZ POP SW	
		OUT 1	
		RET	
		INNC IN 0	
		ANI 404	
		JZ INNC	
		IN 1	
		ANI 127	
		RET	

NAME COMPARISON ALGORITHM

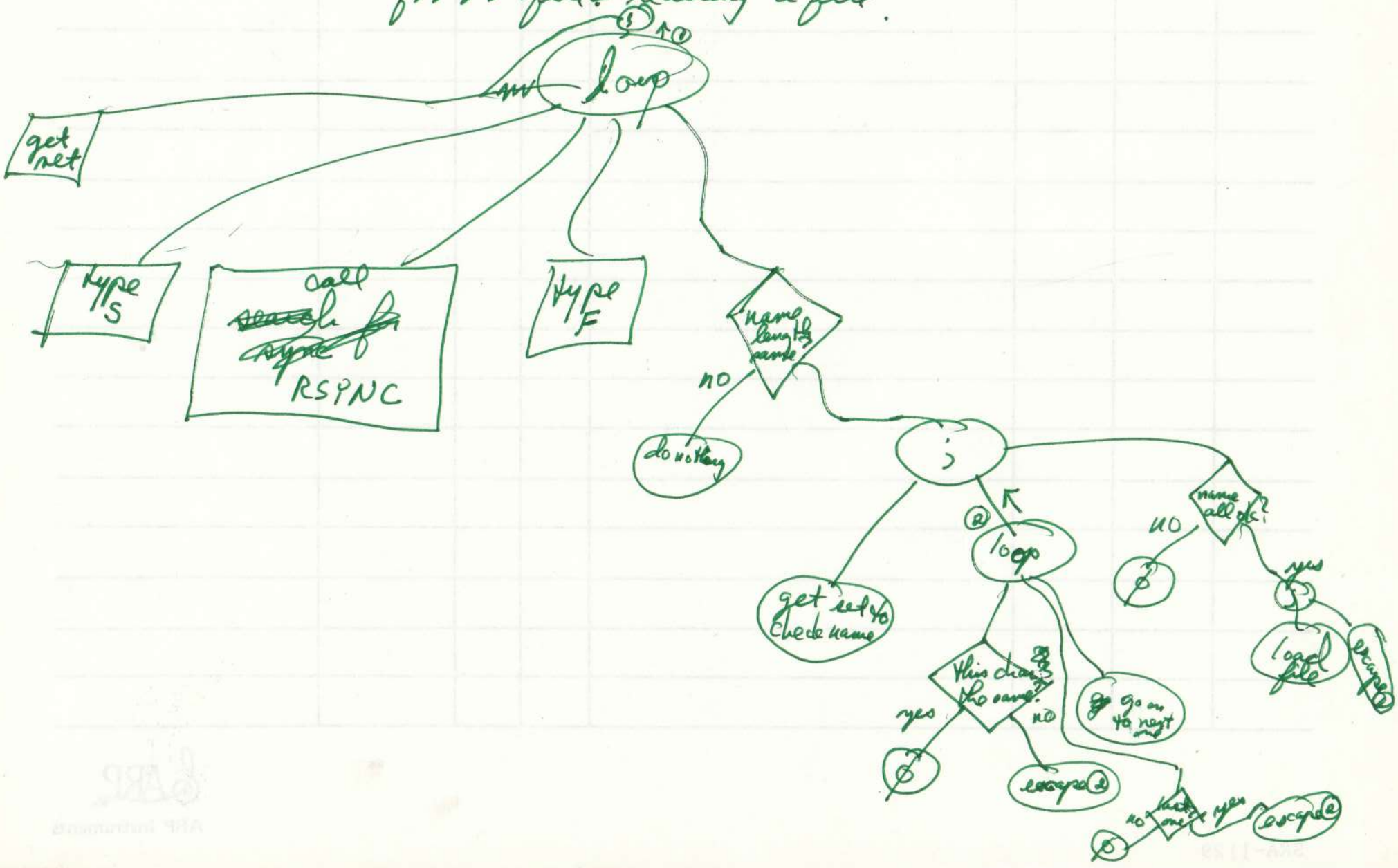
IBUF is OD
and TAPE is 03] successful comparison.

TAPE AND IBUF MATCH] so far so good; continue

TAPE AND IBUF DO NOT MATCH] no good; back to sync routine

TAPE SEARCH PROGRAM

PHASES: S looking for sync
F ~~found a file~~ reading a file.



BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE _____ PAGE 1 of —

ADDRESS	INSTR	MNEMONIC	COMMENTS
		~~PUSH B~~	NOT REALLY NECESSARY WHEN USED AS A
		~~PUSH D~~	COMMAND.
		~~PUSH H~~	
10	NRFIL	LXI H, ~~START~~	
20		CALL QANDA	ASK HIM FOR THE START OF THE FILE
30		LXI H, IBUF	
40		CALL RHEX	
50		PUSH D	GET IT OUT OF THE WAY FOR NOW
60		LXI H, ENDQ	ASK HIM FOR THE END OF THE FILE
70		CALL QANDA	
80		LXI H, IBUF	
90		CALL RHEX	
100		POP H	STARTING ADDRESS IS IN H
110		PUSH H	SAVE
120		MVI B, FFH	FOR COMPLEMENTING
130		MOV A, H	MULTIPLY HL by -1
140		XRA B	
150		MOV H, A	
160		MOV A, L	
170		XRA B	
		~~INR A~~	
180		MOV L, A	
190			
200		DAD D	LENGTH IS NOW IN H, L
210		~~MOV D, H~~ POP D	START IS IN D
220		XCHG	BUT EXCHANGE THESE
230		~~CALL QANDA~~ PUSH H	SAVE H
240		LXI H, NAME	GET NAME IN IBUF
250		CALL QANDA	
260		CALL WSYNC	NOW WERE READY TO SPIN THE TAPE
270		~~MVI A, FFH~~ MOV A, B	PUT OUT FF
280		CALL OUTTP	
290		LXI H, IBUF	
300	AGN	MOV A, M	
310		CPI 0DH	(CR)
320		JZ CON	
330		CALL OUTTP	
340		INX H	
350		JMP AGN	
360	CON	MVI A, 03	
370		CALL OUTTP	PUT OUT 03 (END OF NAME)

INX H
INX H

DD41

STACK

D0 80 - 12
1 - AA
2 - 30
3 - FD
4 - 56
5 - 03

D0 86 - 12

D0 87 - 43

D0 88 - [redacted] 66

D0 89 - [redacted] FE

D0 8A - 07

D0 8B - FE

D0 8C - 1F

D0 8D - F0

D0 8E - 46

D0 8F - 45

D0 90 - 39

ADDRESS	INSTR	MNEMONIC	COMMENTS
380		POP H	H IS NOW START OF FILE
390	AGN2	MOV A,M	
400		CPI 03	
410		JNZ CON2	
420		CALL OUTTP	GOES OUT TWICE IF IT'S AN 03.
430	CON2	CALL OUTTP	
440		DCX D	
450		~~MVI A,0~~ XRA O	
460		INX H	
470		CMP E	
480		JNZ AGN2	
490		CMP D	
500		JNZ AGN2	
510		MVI A,3	MUST BE DONE, PUT OUT TRAILER
520		CALL OUTTP	
530		~~MVI A,0FFH~~ MOV A,B	GET FF
540		CALL OUTTP	
550		MVI A,'!'	SIGNAL THAT WE'RE DONE
560		CALL OUTNC	
570		~~POP H~~ ~~POP D~~ ~~POP B~~ JMP ~~RST~~ MONTR	
580	START	DW "TS"	"START? ↓03"
590		DW "RA"	
600		DW "?T"	
610		DW 0320H	
620	ENDQ	DW "NE"	"END? ↓03"
630		DW "?D"	
640		DW 0320H	
650	NAME	DW ~~4E0DH~~ 0D0AH	cr"NAME? ↓03"
660		DW "MA"	DW 4E7FH
670		DW "?E"	
680		DW 0320H	
690	IBUF	EQU 0D0ACH	
700	QANDA	EQU 0FD6EH	
710	OUTTP	EQU 0FD1EH	
720	RHEX	EQU 0FDC2H	
730	OUTNC	EQU 0FD89H	
740	MONTR	EQU 0F00CH	
750	WSYNC	EQU 0FD23H	

BC919752 8080 MACHINE LANGUAGE/ASSEMBLER CODING SHEET ROUTINE _____ PAGE — of —

ADDRESS	INSTR	MNEMONIC	COMMENTS
0010	FNXT	LXI H,QADDR.	
20		CALL HXANS	not in yet on v9
30		~~XCHG~~ PUSH D	
40		LXI H,QPTN	
50		CALL HXANS	
60		MOV ~~E,A~~ A,E	
70		POP H	
80		~~CMP A~~ DCX H	
90	LOOP	INX H	
100		CMP M	
110		JNZ LOOP	
120		PUSH H	
130		MOV A,H	
140		CALL HOUT	
150		POP H	
160		MOV A,L	
170		CALL HOTB	
180		~~JMP MONTR~~ NLAB	(new line and back
190	QADDR	DW 'DA'	
200		DW 'RD'	
210		DW '?'	
220	PTN	DW 0303H	
230	Q	DW 'P'	
240		DW 'RT'	
250		DW '?N'	
260		DW 0320H	
270	HXANS	EQU 0FF33H	
280	HOUT	EQU 0F1D3H	
290	HOTB	EQU 0F1E3H	
300	~~MONTR~~ NLAB	EQU 0F00C4H 0FE0EH	

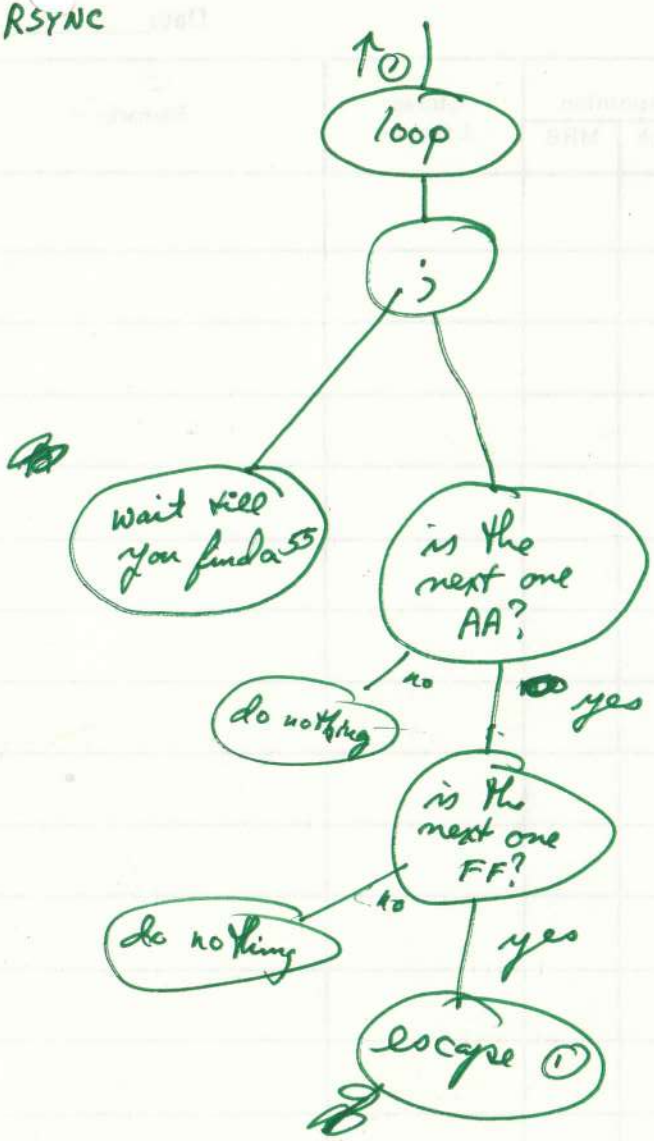
ADDRESS	INSTR	MNEMONIC	COMMENTS
		QADDR DW 'DA'	
		DW 'RD'	
		DW '?'	
		DW 0303H	
		QNAME DW 4E0DH	
		DW 'MA'	
		DW '?E'	
		DW 0320H	
		QANDA EQU OFD6EH	
		~~NEOUT~~ OUTNC EQU OFD89H	
		INTP EQU OFCC6H	
		IBUF EQU 0D0ACH	
		MONTR EQU OF00CH	
		HXANS EQU OFF33H	
		RSYNC EQU OFD36H	

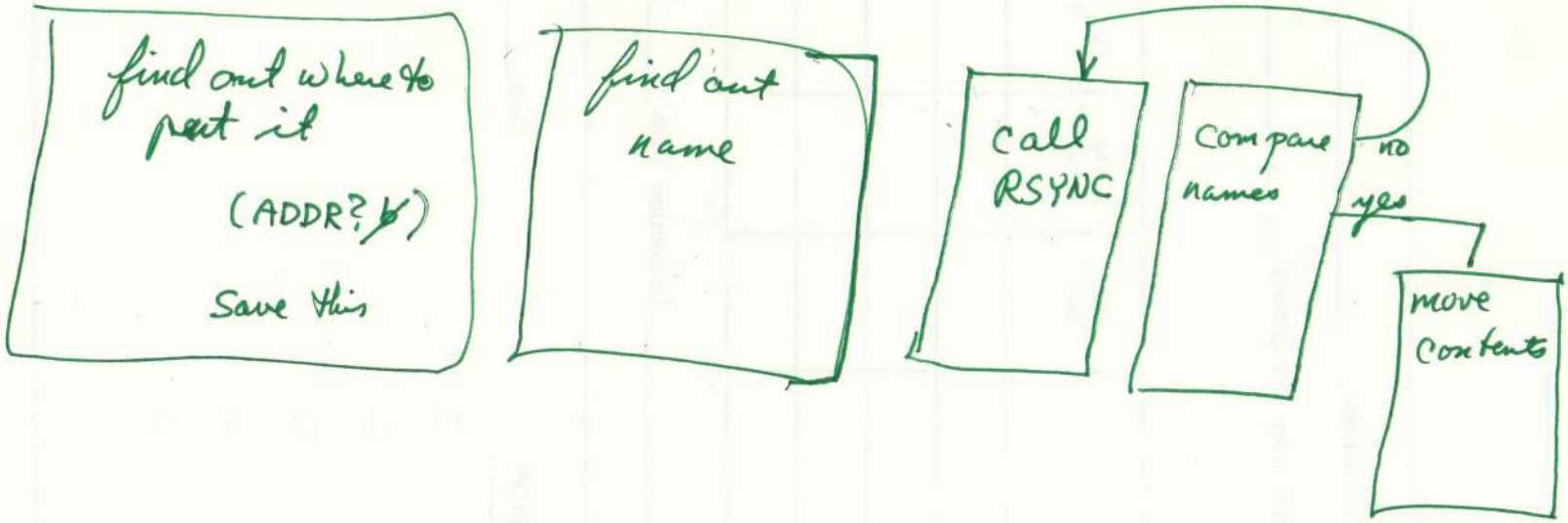
ADDRESS	INSTR	MNEMONIC	COMMENTS
		RDFIL LXI H, QADDR	FIND OUT WHERE TO PUT IT
		CALL HXANS	
		~~DCRD D~~	
		LXI H, QNAME	FIND OUT NAME
		CALL QANDA	(IBUF HAS NAME - D,E, HAS ADDR)
		~~LOOK CAL~~ CALL RSYNC	SPIN THE TAPE.
		LXI H, IBUF	COMPARE NAMES
		MVI A, 'F'	FOR ~~SCAN~~ FILE
		CALL OUTNC	
		LOOK MVI A, 's'	FOR SCAN
		CALL ~~R~~OUTNC	
		CALL RSYNC	
		MVI A, 'F'	FOR FILE
		CALL ~~R~~OUTNC	
		LXI H, IBUF	
		CMPR ~~COMPARE~~ CALL INTP	
		CPI 3	
		JZ BUFOD	see if ibuf is OD
		CMP M	are they the same
		JNZ LOOK	NO, NO GOOD
		INX H	O.K., TRY THE NEXT ONE
		JMP CMPR	
		BUFOD MVI A, ODH	
		CMP M	
		JNZ LOOK	TAPE NAME TOO SHORT
		MVI A, 'L'	FOUND IT (LOAD)
		CALL ~~R~~OUTNC	
		XCHG	GET FIRST ADDRESS IN H FROM D
		LDTP CALL INTP	
		CPI 03	
		JZ LGC3	IF ITS A THREE, TRY THREE LOGIC
		MOVM MOV M, A	OTHERWISE MOVE IT IN
		INX H	AND
		JMP LDTP	GO GET ANOTHER
		LGC3 CALL INTP	LOOK AT THE NEXT ONE.
		CPI 03	ALSO A THREE?
		~~JMP~~ JZ MOVM	IF SO, MOVE THIS ONE IN.
		MVI A, '!'	OTHERWISE, WE'RE DONE.
		CALL ~~R~~OUTNC	
		JMP MONTR	

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE _____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		HEXANS PUSH H	
		CALL QANDA	
		LXI H,IBUF	
		CALL RHEX	
		POP H	
		RET	
		QANDA EQU 0FD6EH	
		IBUF EQU 0D0ACH	
		RHEX EQU 0FDC2H	

RSYNC


loop
;
wait till you find a 55
is the next one AA?
no
yes
do nothing
is the next one FF?
no
yes
do nothing
escape


find out where to put it
(ADDR?)
Save this
find out name
call RSYNC
compare names
no
yes
move contents

BC919752

8080 MACHINE LANGUAGE/ ASSEMBLER
CODING SHEET

ROUTINE _____
PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS	
		MCHRC	LXI H,QFROM	MOVE CHARACTERS COMMANDS
			CALL QANDA	
			LXI H,IBUF	
			CALL RHEX	
			PUSH D	
			LXI H,QTO	
			CALL QANDA	
			LXI H,IBUF	
			CALL RHEX	
			PUSH D	
			LXI H,QCNT	
			CALL QANDA	
			LXI H,IBUF	
			CALL RHEX	
			MOV A,E	THE COUNT
			POP D	THE "TO"
			POP H	THE "FROM"
			CALL MVCHR	
			CALL CRLF	
			JMP MONTR	
		QFROM	DW 'RF'	'FROM?b'
			DW 'MO'	
			DW ' ?'	DB 03H
		? QTO	DW 'T'	'bb TO?b'
			DW '?O'	
			DW '03AOH'	
		? QCNT	DW 'C'	'b COUNT?b'
			DW 'UO'	
			DW 'TN'	
			DW ' ?'	DB 03H
		QANDA	EQU OFD6EH	
		MVCHR	EQU OFD9FH	
		RHEX	EQU OFDC2H	
		CRLF	EQU OFOA6H	
		MONTR	EQU OFOOCH	
		IBUF	EQU ODOACH	

Why were these necessary? Assembler problem? over thing to do with db.

BC91 752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE ____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		CHGF CALL ~~RHEX~~ READ	LXI H, IBUF — MAKE THE OPERATING SYSTEM
		CALL RHEX	CHANGE THE LOCATION OF A
		XCHG	FILE.
		SHLD BOFP	
		CALL READ	LXI H, IBUF — NO CONVERSATION TO SAVE ROOM
		CALL RHEX	
		XCHG	
		SHLD EOFP	
		CALL CRLF	
		JMP MONTR	
		READ EQU 0F025H	
		RHEX EQU 0FDC2H	
		BOFP EQU 0D005H	
		EOFP EQU 0D007H	
		CRLF EQU 0F0A6H	
		MONTR EQU 0F00CH	
		IBUF EQU 0D0ACH	

BC919752

8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE _____

PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC	COMMENTS
		DUMPR PUSH SW	DOESN'T DESTROY ANYTHING
		MOV B, A	
		CALL HOTNC	
		MOV C, A	
		CALL HOTNC	
		MOV D, A	
		CALL HOTNC	
		MOV E, A	
		CALL HOTNC	
		MOV H, A	
		CALL HOTNC	
		MOV L, A	
		CALL HOTNC	
		MOV M, A	
		CALL HOTNC	
		POP SW	
		PUSH SW	
		CALL HOTNC	
		POP SW	
		RET	
		HOTNC PUSH SW	
		PUSH D	
		PUSH H	
		PUSH B	
		CALL HOT8	
		POP B	
		POP H	
		POP D	
		POP SW	
		RET	

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

ROUTINE INTP
PAGE 1 of 1

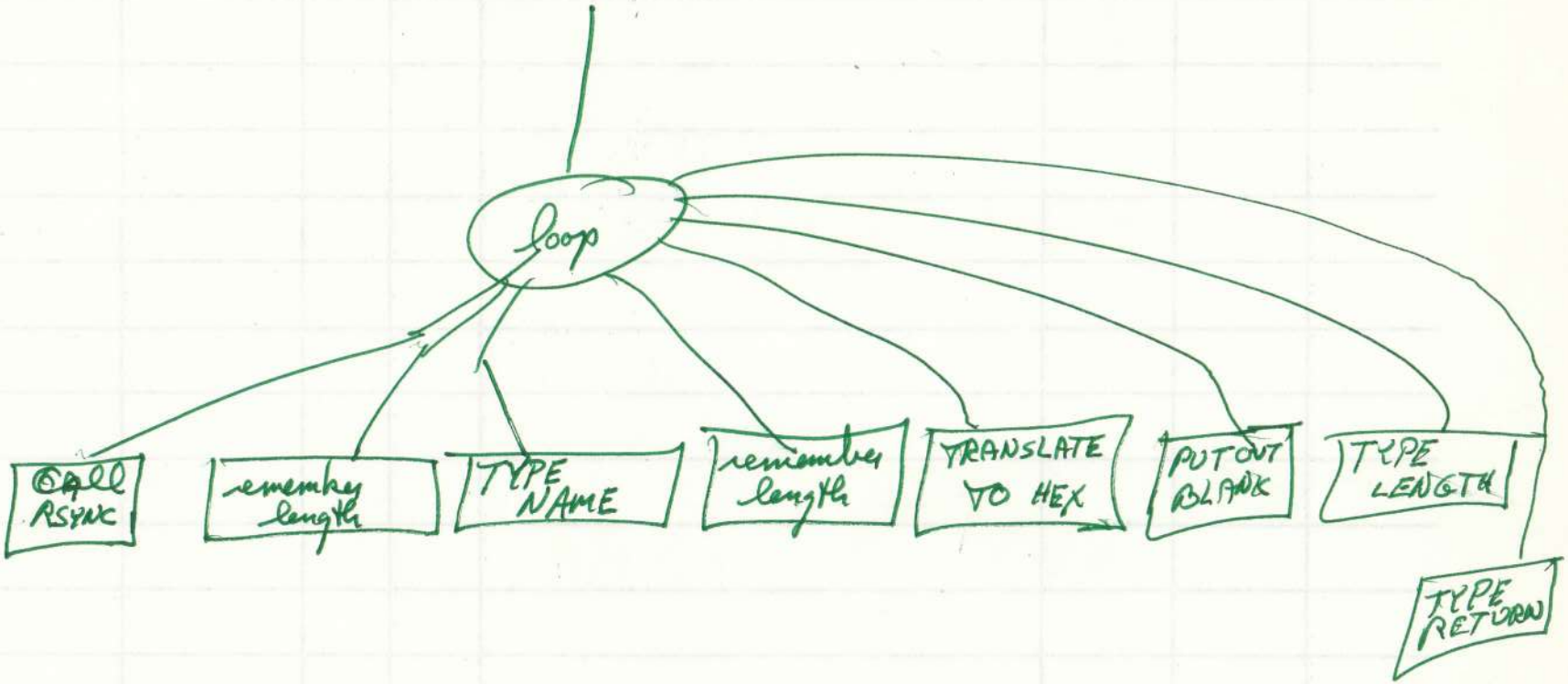
ADDRESS	INSTR	MNEMONIC	COMMENTS
			Inputs word from Tape
		~~Push H~~	
		~~Push Psw~~	
		TRY IN 06	
		RAR	Check status
		~~JNC~~ JC TRY	
		~~Pop H~~	
		~~Pop Psw~~	If OK, Read
		IN 07	Store
		~~MOV M,A~~	
		RET	

QUESTION ASKER

params are ~~B — length of?~~
HL — addr of? (ends in 03)
~~B — length of answer~~

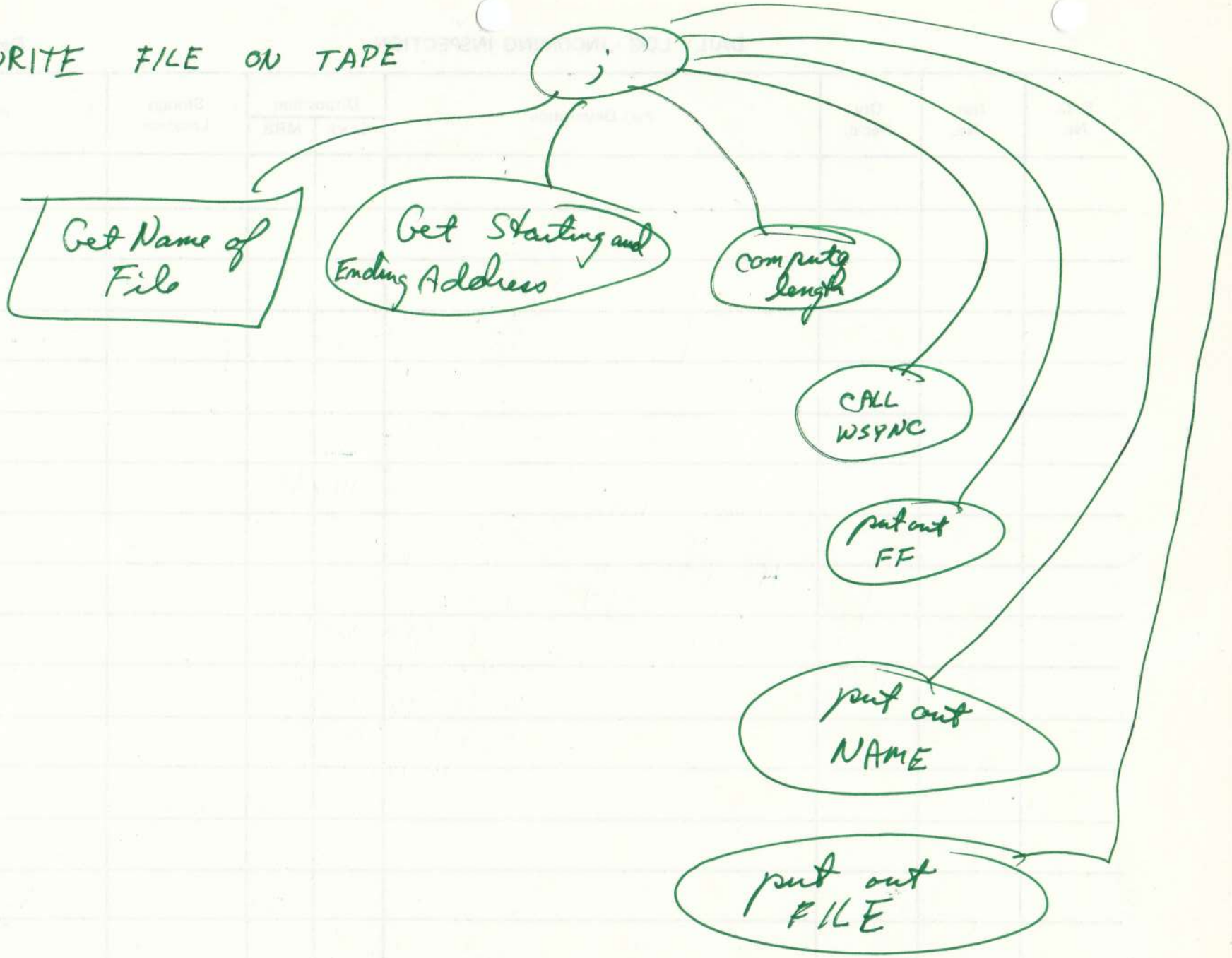
returns A — length of answer
(ANSWER IS IN IBUF.)

PRINT CATALOG

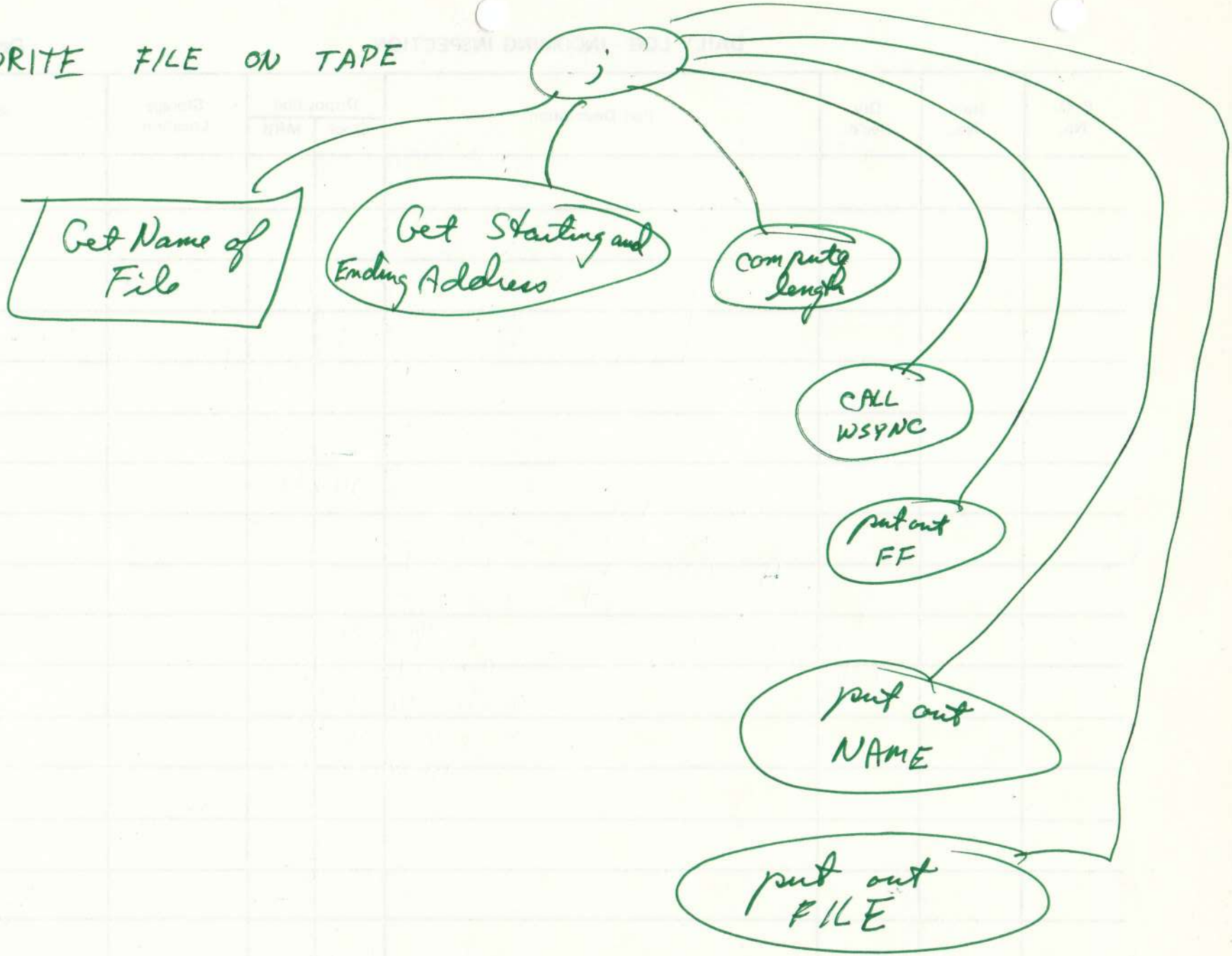


User must interrupt to end this program.

WRITE FILE ON TAPE



WRITE FILE ON TAPE



3C919 52 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE ____ PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC		COMMENTS
		[redacted]		MOVE CHARACTERS
		MVCHR	PUSH H	H,L - FROM
			PUSH D	D, E - TO
			PUSH B	A - LENGTH
			MOV B, A	
			INR B	NOW LENGTH +1
		LOOP	DCR B	
			JZ DONE	
			MOV A, M	
			XCHG	
			MOV M, A	
			XCHG	
			INX D	
			INX H	
			JMP LOOP	
		DONE	POP B	
			POP D	
			POP H	
			RET	

BC9197-2

8080 MACHINE LANGUAGE/ ASSEMBLER
CODING SHEET

ROUTINE _____
PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC		COMMENTS
		TELL	PUSH H	HL POINTS TO STRING TO PRINT
		AGAIN	MOV A,M	
			CALL OUTNC	
			INX H	
			CPI 03	
			JNZ AGAIN	
			POP H	
			RET	
		OUTNC	EQU 0FD89H	

READ IN A FILE TO ...

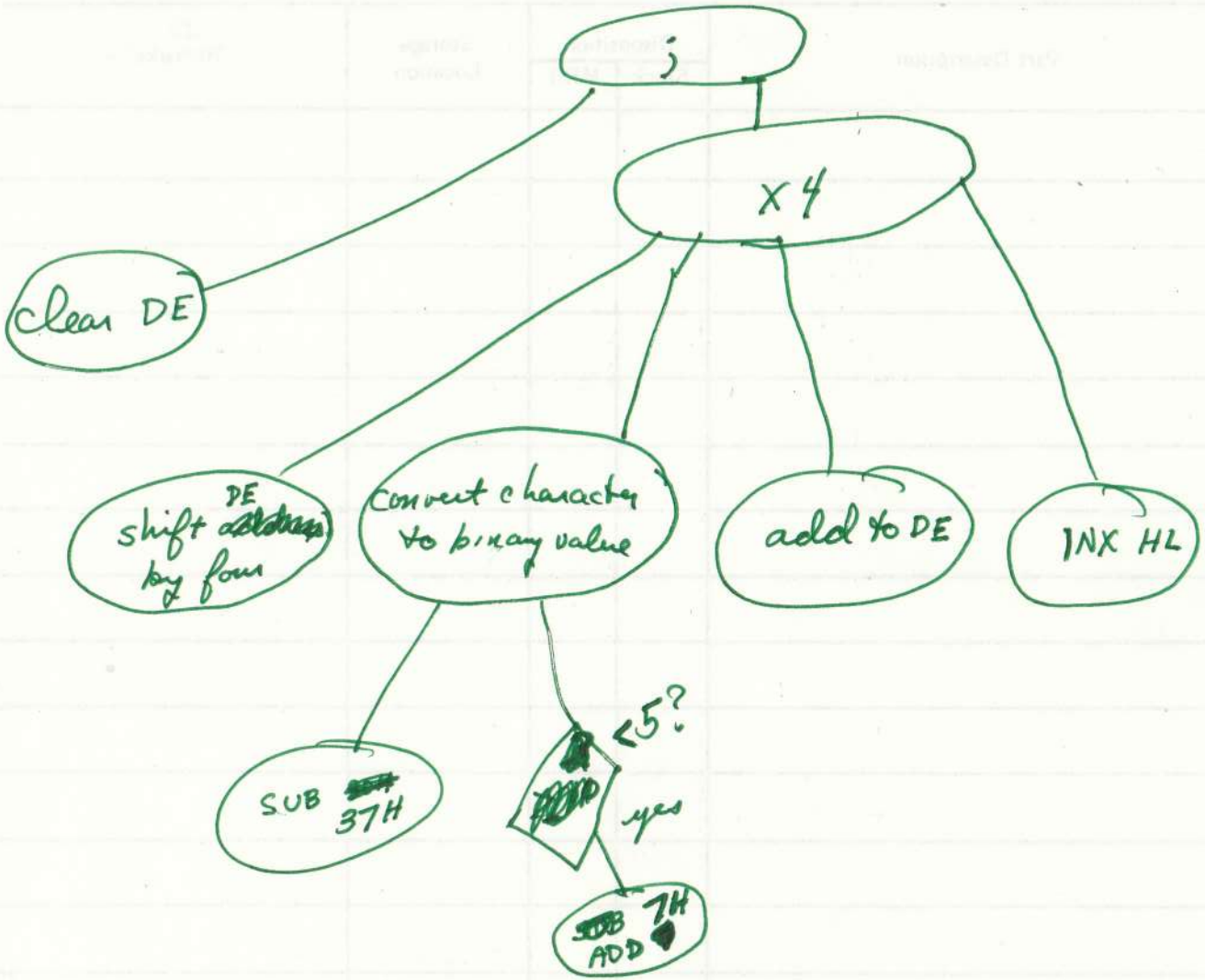
8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

PAGE ___ of ___

ADDRESS	INSTR	MNEMONIC		COMMENTS
		WSYNC	PUSH B	WRITE SYNCHRONIZING SEQUENCE ON TAPE
			MVI B,30	(decimal) 30 times (2 seconds)
		AGAIN	MVI A,55H	
			CALL OUTTP	
			MVI A,0AAH	
			CALL OUTTP	
			DCR B	
			JNZ AGAIN	
			POP B	
			RET	
		OUTTP	EQU 0FCBCH	

BC91752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE TPWRET PAGE — of —

ADDRESS	INSTR	MNEMONIC		COMMENTS
				Writes a tape from incremental series of memory HL ⇒ first ADDR ; DE last
		TPWRT	Push HL	
			Push BC	
			INX DE	Adjust to make sure we get last one
		BEE	MOV A, M	Put out current character
			Call outtp	
			INX HL	Go to next character
			MOV A, L	
			XRA E	H'00' if same
			MOV B, A	save
			MOV A, H	
			XRA D.	H'00' if same
			ORA B	0 if same
			CPI H'00'	will be 0 if both are same
			JNZ BEE	
			DCX D,E	Readjust DE
			Pop BC	
			Pop HL	
			RET	



regardless of input

F2 62

1 2 3 4

00100010 00110100

11110010 01100010

1000100010011000

41
-37
A

37

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET ROUTINE _____ PAGE ___ of ___

ADDRESS	INSTR	MNEM NIC		COMMENTS
		RSYNC:	CALL INTP	lets tape go by till it finds a "good sync"
			CPI 55H	
			JNZ RSYNC	
		FINDA:	CALL INTP	
			CPI 0AAH	
			JNZ RSYNC	
			CALL INTP	
			CPI 0FFH	
			JNZ ~~RSYNC~~ FINDA	
			RET	WE FOUND A SYNC.
		INTP:	EQU 0FCC6H	

BC919752 8080 MACHINE LANGUAGE/ ASSEMBLER CODING SHEET

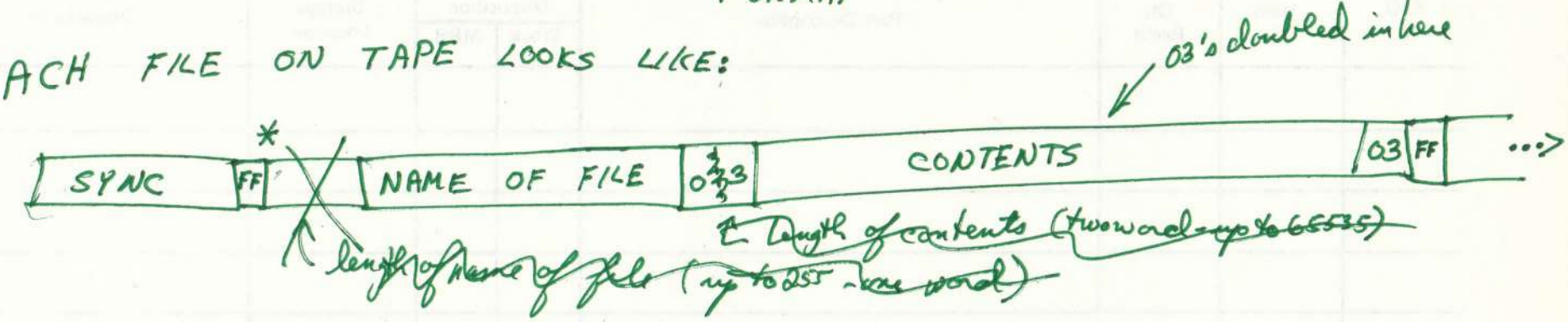
ROUTINE TPRD
PAGE 1 of 1

ADDRESS	INSTR	MNEMONIC	COMMENTS
			loads tape into mem. H,L contains first ~~ADDR~~; D,E last.
		TPRD Push B,C	
		Push H,L	
		INX D,E	take care of final word
		~~BEE MOV A,~~	
		BEE Call INTP	
		MOV M,A	
		INX H,L	
		MOV A,L	
		XRA E	
		MOV B,A	
		MOV A,H	
		XRA D	
		ORA B	
		CPI H'00'	
		JNZ BEE	
		DCX D,E	
		Pop H,L	
		Pop B,C	
		Ret	

TEST
↓
H'00' if same
↓
'0' if same

NAMED FILE TAPE FORMAT

EACH FILE ON TAPE LOOKS LIKE:



.....
.....SRC

NAME LENGTH ≤ 10 chars.

* FF is written by your program, but is readby RSYNC.

ARP INTERNAL MEMO

DATE: 10/9/75

TO: Ron Ross, Software Files

FROM: Bruce Cichowlas

SUBJECT: Proposed format for load tapes for the Altiar 8800.

LOAD TAPES

A load tape is a tape containing one or more object modules in ready to execute format. It differs from "memory image" tapes like the ones we are storing the operating system on, because it may contain several program modules that will be loaded into distinct sections of memory.

The purpose of a load tape is to set up a system that may need most of the memory including possibly that used by the operating system, for execution of a series of programs. For instance, a load tape might contain all of the programs needed for a proposed ARP application.

If our load tapes have a consistent format then a simple loader program will be able to load any one of them in an identical manner.

PROPOSED FORMAT

A magnetic tape as used by the Altair 8800 can be viewed as a stream of sequential 8-bit words. Each program on the load tape will have the following format:

Words 0-1: Load address of this program. (Two words; word 0 MSB's)
Words 2-3: Length of this program in words. (Two words; word 2 is MSB's)
Words 4-??: The program. The length of this field is specified by words 2-3 above.

This pattern would be repeated for each program on the tape. At the end of the tape there is some trailer information so that the loader will stop loading and start execution if this is desired. The format of this data would be as follows:

Words 0-3: FF FF FF FF (16) (Four words; indicates end of programs.)
Word 4: 00 or FF. (one word:
if this word is 00, this is the end of the tape and the Altair will pause. If this word is FF, and sense switch 1 is NOT up, then the computer will start execution at the address specified in words 5-6).
Words 5-6: Address to start execution. Two words. Word 5 is MSB's. This field is only used if word 4 is FF (16), and sense switch 1 is up.

N.B. This scheme does not allow for parity checking or dynamic loading or relocatable object code, and may be altered later to provide for these.

ARP Instruments

INCOMING INSPECTION REPORT

No._____

P. O. Number: _____ Mfg: _____

Date Rec'd: _____ Disposition: _____

Quantity: _____ Date Insp. Completed: _____

Item Description: _____

Procedure	Sample Size	Comments	Qty. Pass	Qty. Fail	Pass/Fail
Visual Insp.					
Mech. Insp.					
Elec. Insp.					
Special					

Tested to Spec. No._____ Inspector_____ Inspection Time_____Hours

M.R.B. DISPOSITION

Date_____

Initials

Q. A._____

Prod. Control_____

Purchasing_____

Return to Vendor ☐

Accept As Is ☐

100% inspect/sort ☐

Rework and reinspect ☐

Scrap ☐

Comments:

DISPOSAL (ACCOUNT FOR ALL ITEMS)

DATE: _____

SKA-1128

DAILY LOG - INCOMING INSPECTION

Date_____

P. O. No.	Item No.	Qty. Rec'd.	Part Description	Disposition		Storage Location	Remarks
				Stock	MRB		

ARP Instruments

DAILY LOG - INCOMING INSPECTION

Date_____

P. O. No.	Item No.	Qty. Rec'd.	Part Description	Disposition		Storage Location	Remarks
				Stock	MRB		

ARP Instruments