OCTAL UTILITY PACKAGE

The purpose of this program is to read in 8 bit codes from the paper tape and arrange them into 24 bit command or data words for the PB 250. These words are then stored in memory line 02, beginning with sector 000.

Two talleys are kept; the C register keeps tract of the number of words stored in line 02 and the B register keeps track of the number of bits assembled into words.

When the program is first started both the B and C registers are loaded with their respective constants (locations 377, 000, 001 and 002). During each iteration a "left shift and decrement" command (in sector 016) causes the (C) to be decremented by 8 and (B) to be shifted left 8 places. After 3 of these shifts the marker bit in B is in the sign bit position of B so that a "transfer if B-negative" command (cell 013) determines whether a word is complete.

A "transfer if C negative" command (cell 012) determines whether the required number of words have been assembled and stored in memory. The original value of C (which is stored in 377) must be calculated to go negative when the correct number of words is stored.

After the B and C registers are set, a "read paper tape" (003) and "transfer on external signal" (004) are used to make sure that the tape reader has passed the character read on the previous cycle. As long as an external signal is present the program stays in a small loop of reading and testing, etc. When no external signal is present (or when the reader has passed the character from the previous cycle a "clear input buffer" command (005) is given to clear out the old information.

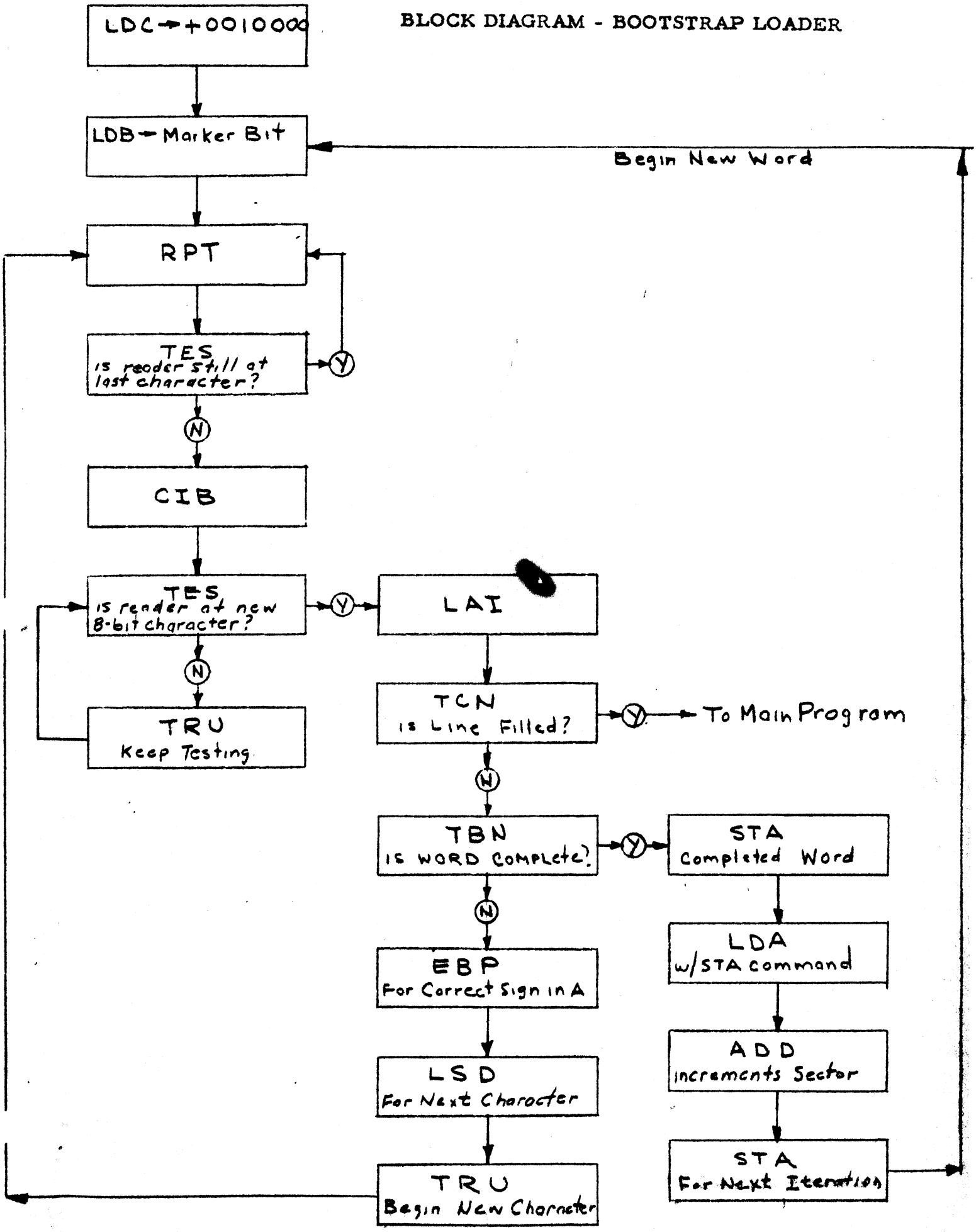
The next two commands, "transfer on external signal" (006) and "transfer unconditionally" (007) cause another loop to be set up to wait for the new characters to be read from the tape. When this new information is sensed, a "load A from input buffer" command (010) brings the 8 bits into the A register. The mask for the "load A from input buffer" instruction is in 011.

At this point, the B and C registers are tested as described above. If C is negative (line 02 filled) the program branches to the first instruction of the main program. If not, it tests B (word complete).

If the word is not complete an "extend bit pattern" instruction (014, with mask in location 015) is given so that the sign bit may be moved into the A register. The unconditional transfer (location 017) now returns the program to location 003, or the read paper tape command so that a new character can be brought in and the process repeated until a word is formed.

The "transfer if B negative" command (013) branches to location 020 if the word is complete. In this cell is a "store A" command. The first iteration through (A) which are the assembled word, are stored in line 02, sector 000. The next instruction (021) is to "load A" with the command word which said "store A in line 02 sector 000." A constant is added to this command word (instructions 022 and 023) which advances the sector number by one. This is now stored back in location 020 ready for the next word (instruction 024) and an unconditional transfer (instruction 025) loops the program back to location 001 to begin assembling a new word.

BLOCK DIAGRAM - BOOTSTRAP LOADER



BOOTSTRAP INPUT

PROGRAM 8 Bit Loader

	B	SELECTOR			S	OPL		INP			B	REMARKS
377	0	00	000	000	0	100	000	00000	0	0	CR	$+0010000_8$
00	0	11	111	111	0	000	100	00001	0	0	CR	LDC → (377)
01	1	00	000	010	1	000	110	00001	0	0	CR	LDB → (002)
02	1	00	000	000	0	000	000	01000	0	0	CR	Marker Bit
03	0	00	000	100	0	101	010	00000	0	0	CR	RPT
04	0	00	000	011	0	111	111	11110	0	0	CR	TES → 003
05	1	00	000	101	0	101	111	00000	0	0	CR	CIB
06	1	00	001	000	0	111	111	11110	0	0	CR	TES → 010
07	1	00	000	110	1	011	111	00001	0	0	CR	TRU → 006
10	0	00	001	001	1	101	101	00001	0	0	CR	LAI
11	0	00	000	000	0	000	001	11111	1	1	CR	Mask for LAI
12	0	01	100	110	0	011	[illegible]	00010	0	0	CR	TCN → (02)146 (3)
13	0	00	010	000	0	011	110	00001	0	0	CR	TBN → 020
14	0	00	001	101	1	100	000	00001	0	0	CR	EBP
15	0	11	111	111	0	000	000	00000	0	0	CR	Mask for EBP
16	0	00	010	111	0	010	001	00000	0	0	CR	LSD (8 Places to 027)
17	1	00	000	011	1	011	111	00001	0	0	CR	TRU → 003
20	1	00	000	000	0	001	001	00010	0	0	CR	STA → (02)000
21	0	00	010	000	0	000	101	00001	0	0	CR	LDA → (020)
22	1	00	010	011	1	001	100	00001	0	0	CR	Add (023)
23	1	00	000	001	0	000	000	00000	0	0	CR	constant
24	0	00	010	000	0	001	001	00001	0	0	CR	STA → 020
25	0	00	000	001	1	011	111	00001	0	0	CR	TRU → 001
26												
27												
30												
31												
3												

USE OF THE LOADER - UNLOADER ROUTINE

This program is written to perform the following functions:

1. Accept and load information from the typewriter in the prescribed octal format.
2. Accept and load information from the paper tape reader in the prescribed octal format.
3. Type out in instruction format the contents of any line or sector.
4. Punch out in instruction format the contents of any line.
5. Punch out in eight-bit codes the contents of any line using only three characters per word.
6. Accept and load information prepared in the eight-bit code.

The program uses all locations of line one, and various sectors of line 00 for temporary storage. In addition, the last two sectors of lines 05 and 06 are used for typeout and punchout instructions. The sectors used are:

05-376
05-377
06-376
06-377
00-012
00-013
00-014
00-015
00-000

Loading the Program:

A bootstrap program is provided which will load the main program (256 words) in about two and one-half minutes including the time to load the bootstrap.

The tape is started in the reader in the normal bootstrap loading manner. The starting point is not critical as long as it is before the first set of holes. The

FILL switch is then turned to the ON position (up) and the ENABLE and BREAKPOINT switches are depressed simultaneously but momentarily. The tape will start to read and will stop about halfway through its length. When it stops, turn the "FILL" switch off; push the ENABLE switch down, hit the I key on the Flexowriter, push the BREAKPOINT switch down, and then lift both the ENABLE and the BREAKPOINT switches and the tape will start to load.

The information on this portion of the tape is loaded into line 02. When the line has been filled, the information is copied into line 01 and

the Flexowriter light will be turned on indicating that the program is loaded and is now waiting for instructions from the operator.

The procedures for utilizing the various portions of the program are as follows:

To load a program manually punched in the octal format:

A. To set up a starting sector and line no. for loading information via the typewriter.
    1. Type the desired first sector to be loaded. This should be three octal digits numbered from 000 to 377.
    2. Type the number of the desired line (in octal).
    3. Type a $.

    This will destroy the contents of the location just specified and will set up the program to load the following information into the specified location.

B. To set up the starting sector and line no. for loading octal information via the paper tape reader. The same procedure is followed with the exception that an R (for Reader) will be typed instead of the $.

C. The tape must be prepunched in the proper format. Nine characters per word are used. For Example:
    1. 034S3702;
    2. 000 0600I

These two instructions may be shown as:

	Sector	Sequence Tag	Operation	Line No.	Index tag
1.	034	S	37	02	;
2.	000		06	00	I

In example No. 1 the instruction is an unconditional transfer (TRU) to sector 034 (octal) with a sequence tag indicated by the S, and no indextag indicated by the ;. Example No. 2 indicates a load B (LDB) instruction from sector 000 (octal) of line 00 (octal) with no sequence tag indicated by the space, and an index tag indicated by the I.

Octal numbers may also be loaded by punching (or typing) as follows:

+ 1234567 or,

- 2634721

where the sign may be either + or - followed by any seven octal digits. These will be entered exactly as presented. Note that only eight characters of this type are required per word.

Program control is normally given to the typewriter and the preceding information could be entered via the keyboard.

A carriage return must be typed (or punched) after every word of information. No information is loaded until the carriage return is found. This means that if an error is made (and discovered) before the carriage return, the correction may be made simply by continuing to type until the last nine characters prepared are those desired. The carriage return may then be typed and the correct data will be loaded.

As each carriage return code is found, the data is loaded and the sector number of the instruction which does the loading (XXXSTBLL) is incremented.

The sequential loading may be overidden by typing a new starting sector, line and $.

Once control has been given to the reader, there are only two ways to return to

keyboard control. The first is the ENABLE and I keys depressed simultaneously. The second is to have a transfer indicator punched on the tape.

Transfer of Control

The transfer of control may be effected by the use of the period (.). This is performed after the usual preliminary typing (or reading from tape) of the three octal digits denoting the sector and the two denoting the line number. For example, the transfer of control from the tape reader back to the keyboard can be effected by a transfer to sector 000 of line 01. This information would appear as:

00001.

Similarly a transfer to any desired location can be effected. For example:

22305.

This will transfer to the command stored in sector $223_{(8)}$ of line 05.

Instruction Typeout.

Instructions may be typed out in the same octal format as previously described. However no provision has been made for the typing of data words. These are converted to the instruction format and typed as instructions. It should be noted that the Lo bit will not show in this format.

The procedure for typeout is similar to the foregoing. The five octal digits denoting sector and line number must be typed, followed by a T. The Breakpoint switch should be up. Sequential sectors of the denoted line will be typed starting at the prescribed sector. After typing a complete line, the control will revert to the keyboard. Typeout can be interrupted at any time however by pushing the Breakpoint switch down. This will cause the program to complete the word being typed and then transfer control to the typewriter keyboard.

Punchout.

Punchout of information may be performed in either of two modes. When the Breakpoint switch is up, data will be punched exactly as it is typed. That is, in the instruction format only. If such a tape is reread back into the computer bits of

data words may be lost from the Lo position. This type of punchout may be interrupted after the initial start by pushing the Breakpoint switch down.

The second type of punchout prepares tapes for high speed loading of programs or data into the computer. Information is punched out in eight-bit codes, three characters per word. The information is in straight binary with the first two bits of each word always presented as zeros. These two bits are discarded when the tapes are read back into the computer. No information is lost in this type of output tape.

Since the loading of this type of tape always starts at sector 000, the keyboard instruction should always show 000 as the starting sector. To punch out the 8 bit code tape the following procedure should be followed.

1. Punch the Breakpoint switch down.
2. Type 000LLP where LL represents the two octal digits of the desired line no.

The tape will then be punched out until the entire contents of the line have been punched after which control will be returned to the operator via the Flexowriter keyboard.

NOTE:

This type of tape must be started at the exact first character on reading. It is therefore wise to mark a line along the punch block PRIOR to punching the tape. Since zeros will appear to be the same as leader, the mark will indicate the correct starting point.

Loading eight-bit coded tapes into the computer.

The loading procedure for this type of tape is as follows:

1. Insert the tape in the reader correctly aligned at the first character. (See the paragraph describing the punching of these tapes.)
2. Type 000LLR with the Breakpoint switch down. (Where LL is the desired line no.)

Only a complete line (256 words) may be loaded in this fashion, however the loading time should only be approximately seventy-five seconds. After the line has been loaded control will return to the typewriter keyboard.

LOCATION	INSTRUCTION	CODE	
000	034S3701;	TRU	Transfer to typewriter control from "Enable" & "I"
001	001 4500;	CLA	to zeros and start of binary loader.
002	140 1100;	STA	Zeros in loc 140, line 00 for start of check sum.
003	377 0401;	LDC	LDC → total shifts for 1 line.
004	005S0601;	LDB	with marker bit
005	000 0010;		marker bit to indicate completed word.
006	007 5200;	RPT	Energizes reader for each character.
007	006 7736;	TES	Has reader passed old character.
010	010 5700;	CIB	To remove old information
011	013 7736;	TES	has new character arrived
012	011S3701;	TRU	If no character transfer back to TES
013	014S5501;	LAI	New information into A
014	000 01371		Mask for LAI
015	202 3401;	TCN	C will be negative when line is full
016	022 3601;	TBN	B will be negative when word complete
017	032 4001;	EBP	To get sign bit correct in A register
020	031 2100;	LSD	8 places to make space for new character
021	006S3701;	TRU	For new character from reader
022	000 1100;	STA	→ N (This will have index tag, see
023	140 1400;	ADD	contents of A to running check sum
024	140 1100;	STA	back to loc 140, line 00 with new check sum
025	022 0501;	LDA	with [STA→ N]
026	027S1401;	ADD	to increment sector
027	001 0000;		increment
030	022 1101;	STA	back into 022 with new sector no.
031	004S3701;	TRU	to reset marker in B for new word
032	377 0000;		mask for EBP in 017 & end of binary loader
033	000 6000;	WOC	To be added to character during octal & 8 bit output
034	035S0501;	LDA	with [RTK] for replacement after octal tape read
035	043 5100;	RTK	
036	042 1101;	STA	to replace [RTK] for typewriter control
037	113 0501;	LDA	[TOF] for restoration of exit from subroutines

040	127 1101;	STA	at end of start sector subroutine
041	143 1101;	STA	at end of incrementer sub routine
042	043 5100;	RTK	(or RPT) as determined from input
043	042 7736;	TES	has old character passed
044	046 7736;	TES	has new character arrived
045	046$5700;	CIB	
046	047$4500;	CLA	
047	044$3701;	TRU	if no new character
050	014 5501;	LAI	8 bits loaded into A
051	052$5601;	CAM	for T
052	000 00101		T (actually + 0000043)
053	205 7501;	TOF	to type subroutine
054	055$5601;	CAM	for P
055	000 00051		P (actually + 0000027)
056	210 7501;	TOF	to punch subroutine
057	060$5601;	CAM	for R
060	000 00021		R (actually + 0000011)
061	146 7501;	TOF	to read subroutine
062	063$5601;	CAM	for period
063	000 00121		. (actually + 0000053)
064	171 7501;	TOF	to transfer subroutine
065	066$5601;	CAM	for $
066	000 00141		$ (actually + 0000061)
067	151 7501;	TOF	to start sector subroutine
070	071$5601;	CAM	for S
071	000 0014;		S (actually + 0000062)
072	166 7501;	TOF	to subroutine for inserting sequence tag
073	074$5601;	CAM	for space
074	000 0004;		space (actually + 0000020)
075	167 7501;	TOF	to subroutine for inserting no sequence tag
076	077$5601;	CAM	for I
077	000 00161		I (actually + 0000071)

LOCATION	INSTRUCTION	CODE	REMARKS
100	154 7501;	TOF	to subroutine for inserting index tag
101	102S5601;	CAM	for ;
102	000 0014;		; (actually + 0000060)
103	154 7501;	TOF	to subroutine for inserting no index tag
104	105S5601;	CAM	for carriage return
105	000 0013;		carriage return (actually + 0000056)
106	132 7501;	TOF	to load and increment sector subroutine
107	115 1100;	STA	to temporary word forming location
110	114 2100;	LSD	3 spaces for new octal character
111	115 0400;	LDC	with word now being formed
112	114 4601;	AOC	combines word being formed with new character
113	042S7501;	TOF	TRU for new character
114	000 00011		mask for AOC of octal character
115	116 1100;	STA	Temporary storage of selected operation [LDB or STB or TRU]
116	120 2100;	LSD	1 bit for alignment with index register for line no.
117	120 1237;	STB	into index register
120	130 2100;	LSD	to line up sector with instruction
121	116 0400;	LDC	with instruction [LDB or STB or TRU] from temp. store
122	123S4601;	AOC	combine instruction and sector
123	000S77371		mask for AOC
124	126 1201;	STB	location for starting sector or TRU
125	132 1201;	STB	into start of incrementer subroutine
126	000 0000;		new command [TRU or LDB or STB]
127	042S7501;	TOF	TRU for new data (This is replaced for typeout or punchout)
130	130 4500;	CLA	
131	340S1100;	STA	to clear check sum and transfer to binary punchout
132	000S3701;	LDB STB TRU	start of sector incrementes subroutine
133	132 0501;	LDA	with [command LDB or STB]
134	027 1401;	ADD	increment sector
135	126 5601;	CAM	compare with starting sector for complete line
136	337 7501;	TOF	if complete line
137	132 1101;	STA	with sector incremented, store back into 132

LOCATION	INSTRUCTION	CODE	REMARKS
140	154 1200;	STB	Temporary location for typeout or punchout
141	140 1600;	DPA	to compute check sum
142	140 1200;	STB	in loc for check sum
143	144 7101;	MCL	used after bootstrap only while in line 02
144	034S3701;	TRU	from line 02 to 01 & restore above to 042S7501
145	000 0000;		used to compare 0 in octal output
146	174 7735;	TES	test for BP-start of reader subroutine
147	006 0501;	LDA	with [RPT]
150	042 1101;	STA	to replace RTK with RPT
151	152S0501;	LDA	with [STB→N]
152	000 12001	STB	[STB with index for loading]
153	115S3701;	TRU	to start sector subroutine
154	153 1300;	STD	stores B & A into temporary locations for I subroutine
155	157 2100;	LSD	1 bit for alignment of line no.
156	153 0400;	LDC	for sector sequence tag and op code
157	160S4601;	AOC	combines complete instruction less index tag
160	377S7720;		mask for AOC
161	163 2100;	LSD	1 bit to allow for index
162	154 0400;	LDC	with I. or ;
163	164S4601;	AOC	combines index tag with instruction
164	000 00001		mask for AOC (+ 0000001)
165	042S3701;	TRU	back for next character end of subroutine
166	164 1401;	ADD	start of S.T. subroutine. puts 1 in L.S.B. (entry for S)
167	154 1100;	STA	temporary location (entry for space)
170	161S3701;	TRU	for combination of sequence tag with sector
171	172S0501;	LDA	with [TRU] for start of • subroutine
172	000S37001	TRU	[TRU - I]
173	115S3701;	TRU	transfer to start sector subroutine
174	176 2100;	LSD	1 bit for index register alignment-start 8 bit loader
175	176 1237;	STB	into index register for line no.
176	177S0501;	LDA	with [STA - I] for 8 bit loader
177	000 11001	STA	[STA - I]

```
200  022 1101;   STA  into 8 bit loader subroutine for start at 000
201  001S3701;   TRU  to 8 bit loader start
202  140 5600;   CAM  compare check sum at end of binary loader
203  034 7501;   TOF  transfer to beginning if check sum matches
204  033 0601;   LDB  [WOC] five least significant digits are 14000
205  206S0501;   LDA  with [TRU] entry for typeout
206  376S3705;   TRU  transfer to line 05 for type
207  213S3701;   TRU  to common subroutine for octal output
210  211S0501;   LDA  with [TRU] entry for punchout
211  376S3706;   TRU  transfer to line 06 for punch
212  130 7735;   TES  test for BP switch for 8 bit punch
213  265 1101;   STA  stores transfer to 05 or 06
214  215S0501;   LDA  with [TRU]
215  222S3701;   TRU  new return from start sector subroutine
216  127 1101;   STA  at end of start sector subroutine
217  220S0501;   LDA  with [LDB-I]
220  000 0600I   LDB  [LDB - I]
221  115S3701;   TRU  to start sector subroutine
222  223S0501;   LDA  with [TRU] (return from start sector subroutine)
223  266S3701;   TRU  return from 05 or 06 octal output
224  377 1105;   STA  in line 05
225  377 1106;   STA  in line 06
226  227S0501;   LDA  with [TRU] - (TOF - S to reset overflow)
227  232S7501;   TOF  TRU - return from incrementer
230  143 1101;   STA  to end of incrementer
231  132S3701;   TRU  to start of incrementer
232  114 0401;   LDC  (start of typeout) - no. of shifts for sector
233  234 0600;   LDB  word to be typed - was stored in incrementer
234  234 4500;   CLA
235  237 2200;   RSI  shift 1 bit for 0 in MSB of sector
236  236 4500;   CLA  entry point from sub loops
237  270 3401;   TCN  determines breakpoints of sector, ST, OP, line & I
```

2	244 2100;	LSD	shifts octal digit into A
2	232 1100;	STA	into temporary location (output character)
2	242 4100;	GTB	determines if character parity is correct
2	253 3501;	TAN	A will be negative if parity is correct
2	232 0500;	LDA	from temporary storage
2	145 5601;	CAM	test for 0
2	251 7501;	TOF	if 0
2	074 1401;	ADD	bit in parity position for character
2	24153701;	TRU	to typeout loop
2	005 1401;	ADD	bit in position for 0 code on flex
2	24153701;	TRU	to typeout loop
2	235 1000;	STC	to temporary storage to keep character count
2	253 1200;	STB	to temporary storage (remaining characters)
2	232 0500;	LDA	with output character (parity corrected)
2	257 4300;	CLB	to shift 0's into L.S. bits of A
2	262 2100;	LSD	shift for alignment with WOC
2	033 1401;	ADD	adds character WOC to command
2	376 1105;	STA	stores command into line 05
2	376 1106;	STA	stores command into line 06
2	220 0401;	LDC	with delay no.
2	264 7737;	TES	tests for free Flexowriter
2	37653705;	TRU	to line 05 or 06 (see 213)
2	253 0600;	LDB	with remaining characters -return from output
2	23550400;	LDC	with no. of shifts remaining & transfer to 236
2	274 0501;	LDA	with TRU-N start of loop selector
2	027 1501;	SUB	001 from sector
2	274 1101;	STA	to TRU-N, location 274
2	274 4500;	CLA	
2	30353701;	TRU	varying transfer for subloops
2	30353701;	TRU	for replacement of original command
2	33053701;	TRU	last output loop
2	32653701;	TRU	fifth loop

300	317S3701;	TRU	fourth loop
301	315S3701;	TRU	third loop
302	304 2100;	LSD	second output loop start for sequence tag
303	253 1200;	STB	to temporary storage for remaining characters
304	164 1501;	SUB	1 from LSB
305	310 3501;	TAN	negative for no sequence tag-positive for sequence tag
306	071 0501;	LDA	with "S"
307	311S3701;	TRU	to common subloop
310	074 0501;	LDA	with space
311	312S0401;	LDC	with shift count for op code characters
312	000 0001;		actually (+ 0000004)
313	315 1000;	STC	to temporary location
314	256S3701;	TRU	to typeout loop
315	312 0401;	LDC	with shift count for line number
316	234S3701;	TRU	to CLA, RSI for MSB of line number
317	334 0500;	LDA	with original word for (index tag)
320	322 2200;	RSI	index tag into B
321	324 3601;	TBN	negative for index tag positive for ;
322	102 0501;	LDA	[;]
323	256S3701;	TRU	to typeout loop
324	077 0501;	LDA	[I]
325	256S3701;	TRU	to typeout loop
326	105 0501;	LDA	with carriage return
327	256S3701;	TRU	to typeout loop
330	275 0501;	LDA	with [TRU] to restore loop selector
331	274 1101;	STA	restores loop selector
332	034 7735;	TES	for BP to interrupt typeout
333	132S3701;	TRU	to incrementer for next word
334	000 0501;	LDA	with [TRU] at end of complete line
335	140 0600;	LDB	with check sum
336	137S3701;	TRU	to middle of incrementer to type check sum
337	361 0501;	LDA	with (TRU ⟶ 334) entry from incrementer test

3	137S3701;	TRU	back to incrementer
3	342S0501;	LDA	[TRU] this is start of 8 bit punch
3	362S3701;	TRU	return from line 06 after punch
3	377 1106;	STA	to line 06
3	345S0501;	LDA	with [TRU]
3	351S3701;	TRU	return from start sector subroutine
3	127 1101;	STA	at end of start sector subroutine
3	220 0501;	LDA	with [LDB-I]
3	115S3701;	TRU	to start sector subroutine
3	352S0501;	LDA	[TRU] return from incrementer
3	355S3701;	TRU	return to this subroutine
3	143 1101;	STA	at end of incrementer
3	132S3701;	TRU	to start of incrementer
3	355 4500;	CLA	start of actual punch out subroutine
3	074 0401;	LDC	shift count of one word
3	374 0600;	LDB	with word to be punched see sector 140
3	367S2100;	LSD	6 bits only to keep sign located
3	334S3701;	TRU	used by sector 337 not part of punchout
3	376 0400;	LDC	with running shift count-return from 06
3	132 3401;	TCN	back to incrementer after complete word punched
3	375 0600;	LDB	with shifted word
3	365 4500;	CLA	
3	377 2100;	LSD	shift 8 bits into A for second & third characters
3	375 1200;	STB	temporary storage of shifted word
3	376 1000;	STC	temporary storage of running shift count
3	371 4300;	CLB	to avoid an unintentional index tag
3	375 2100;	LSD	to line up for WOC addition
3	033 1401;	ADD	WOC to character
3	376 1106;	STA	to line 06 for punchout
3	220 0401;	LDC	with delay number
3	376S3706;	TRU	to line 06 for punch out
3	000 4002;		total shifts for 8 bit loader + check sum

pb 250 CODING SHEET

LOC.	SECT.	OP.	S.
000	01/034	37	S
1	00/001	45	
2	00/140	11	
3	01/377	04	
4	01/005	06	S
5	+0000040		
6	00/007	52	
7	36/006	77	
010	00/010	57	
1	36/013	77	
2	01/011	37	S
3	01/014	55	S
4	+0000377		
5	01/202	34	
6	01/022	36	
7	01/032	40	
020	00/031	21	
1	01/006	37	S
2	00/000	11	I
3	00/140	14	
4	00/140	11	
5	01/022	05	
6	01/027	14	S
7	+0040000		
030	01/022	11	
1	01/004	37	S
2	-7740000		
3	00/000	60	
4	01/035	05	S
5	00/043	51	
6	01/042	11	
7	01/113	05	

LOC.	SECT.	OP.	S.
040	01/127	11	
1	01/143	11	
2	00/043	51	
3	36/042	77	
4	36/046	77	
5	00/045	57	S
6	00/047	45	S
7	01/011	37	S
050	01/014	55	
1	01/052	56	S
T 2	+0000043		
3	01/205	75	
4	01/055	56	S
P 5	0000027		
6	01/210	75	
7	01/060	56	S
R060	+0000011		
1	01/146	75	
2	01/063	56	S
• 3	0000053		
4	01/171	75	
5	01/066	56	S
# 6	0000061		
7	01/151	75	
070	01/071	56	S
1	+0000062		
2	01/166	75	
3	01/074	56	S
4	+0000020		
5	01/167	75	
6	01/077	56	S
7	+0000071		

LOC.	SECT.	OP.	S.
100	01/154	75	
1	01/102	56	S
2	+0000060		
3	01/154	75	
4	01/105	56	S
5	+0000056		
6	01/132	75	
7	00/115	11	
110	00/114	21	
1	00/115	04	
2	01/114	46	
3	01/142	75	S
4	+0000007		
5	00/116	11	
6	00/120	21	
7	37/120	12	
120	00/130	21	
1	00/116	04	
2	01/123	46	S
3	+0037777		
4	01/126	12	
5	01/132	12	
6	+0000006		
7	01/042	75	S
130	00/130	45	
1	00/340	11	S
2	01/000	37	S
3	01/132	05	
4	01/036	14	
5	01/126	56	
6	01/337	75	
7	01/132	11	

LOC.	SECT.	OP.	S.
140	00/154	12	
1	00/140	16	
2	00/140	12	
3	01/144	71	
4	01/034	37	S
5	+0000000		
6	35/174	77	
7	01/[illegible]	05	
150	01/042	11	
1	01/152	05	S
2	00/[illegible]	12	I
3	01/116	37	S
4	00/153	13	
5	00/157	21	
6	00/153	04	
7	01/[illegible]	46	S
160	-7777700		
1	00/163	21	
2	00/154	04	
3	01/164	46	S
4	+0000001		
5	01/042	37	S
6	01/164	14	
7	00/154	11	
170	01/161	37	S
1	01/172	05	S
2	00/000	37	S
3	01/115	37	
4	00/176	21	
5	37/176	12	
6	01/177	05	S
7	00/000	11	I

LOC.	SECT.	OP.	S.
200	01/022	11	
1	01/201	37	S
2	00/140	56	
3	01/224	75	
4	01/[illegible]	00	
5	01/200	05	S
6	05/376	37	S
7	01/213	37	S
210	01/211	05	S
1	06/376	37	S
2	35/130	77	
3	01/205	11	
4	01/212	05	S
5	01/222	37	S
6	01/141	11	
7	01/220	05	S
220	00/[illegible]	06	I
1	01/115	37	S
2	01/223	05	S
3	01/266	37	S
4	05/377	11	
5	06/377	11	
6	01/227	05	S
7	01/232	75	S
230	01/143	11	
1	01/132	37	S
2	01/114	04	
3	00/234	06	
4	00/234	45	
5	00/237	22	
6	00/236	45	
7	01/270	34	

LOC.	SECT.	OP.	S.
240	00/[illegible]	21	
1	00/232	11	
2	00/242	41	
3	01/253	35	
4	00/232	05	
5	01/145	56	
6	01/251	75	
7	01/074	14	
250	01/241	37	S
1	01/005	14	
2	01/241	37	S
3	00/235	10	
4	00/253	12	
5	00/232	05	
6	00/257	43	
7	00/262	21	
260	01/033	14	
1	05/376	11	
2	06/376	11	
3	01/220	04	
4	37/264	77	
5	05/376	37	S
6	00/253	06	
7	00/235	04	S
270	01/274	05	
1	01/027	15	
2	01/274	11	
3	00/274	45	
4	01/303	37	S
5	01/303	37	S
6	01/330	37	S
7	01/326	37	S

LOC.	SECT.	OP.	S.
300	01/317	37	S
1	01/315	37	S
2	00/304	21	
3	00/253	12	
4	01/104	15	
5	01/310	35	
6	01/071	05	
7	01/311	37	S
310	01/074	05	
1	01/312	04	S
2	+0000004		
3	00/315	10	
4	01/256	37	S
5	01/312	04	
6	01/234	37	S
7	00/334	05	
320	00/322	22	
1	01/324	36	
2	01/102	05	
3	01/256	37	S
4	01/077	05	
5	01/256	37	S
6	01/105	05	
7	01/256	37	S
330	01/275	05	
1	01/274	11	
2	35/034	77	
3	01/132	37	S
4	01/000	05	
5	00/140	06	
6	01/137	37	S
7	01/[illegible]	05	

LOC.	SECT.	OP.	S.
340	01/137	37	S
1	01/342	05	S
2	01/362	37	S
3	06/377	11	
4	01/345	05.	S
5	01/351	37	S
6	01/121	11	
7	01/220	05	
350	01/115	37	S
1	01/352	05	S
2	01/355	37	S
3	01/143	11	
4	01/132	37	S
5	00/355	45	
6	01/074	04	
7	00/374	06	
360	00/367	21	S
1	01/334	37	S
2	00/376	04	
3	01/132	34	
4	00/376	06	
5	00/365	45	
6	00/377	21	
7	00/375	12	
370	00/376	10	
1	00/371	43	
2	00/375	21	
3	01/[illegible]	14	
4	06/376	11	
5	01/220	04	
6	06/376	37	S
7	+0010010		