

PCL11-A,B

PCL11 EXERCISER V020
CZPLAB0

AH-E260B-MC
COPYRIGHT © 1978
FICHE 1 OF 1

DEC 1978
digital
MADE IN USA

IDENTIFICATION

B 1

SEQ 0001

PROGRAM CODE: AC-E259B-MC
 PROGRAM NAME: CZPLAB0 PCL11 EXERCISER V-02A
 VERSION: V-02
 DATE CREATED: 21-SEP-76
 UPDATED: 13-MAR-78
 MODIFIED: 11-SEP-78
 MAINTAINER: SPECIAL SYSTEMS, KANATA
 AUTHOR: DAVID G. WIENS

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1978, BY DIGITAL EQUIPMENT OF CANADA, LIMITED.

	TABLE OF CONTENTS	PAGE
	-----	----
1	GENERAL	1-1
1.1	GENERAL DESCRIPTION	1-1
1.2	DIFFERENCES IN EXERCISER V-02	1-2
2	EXERCISER TABLES	2-1
2.1	STATUS TABLE	2-1
2.2	SUMMARY TABLE	2-2
2.3	ERRORS TABLES	2-3
2.4	CLEARING OF THE TABLES	2-4
3	EXERCISER COMMANDS	3-1
3.1	CONTROL CHARACTERS	3-1
3.1.1	CONTROL-C	3-1
3.1.2	CONTROL-O	3-1
3.1.3	CONTROL-U	3-1
3.1.4	CONTROL-S	3-2
3.1.5	CONTROL-Q	3-2
3.1.6	CARRIAGE RETURN	3-2
3.1.7	LINE FEED	3-2
3.1.8	RUBOUT	3-2
3.2	COMMANDS	3-3
3.2.1	SILO	3-3
3.2.2	MASTER	3-4
3.2.3	SECONDARY	3-4
3.2.4	RIB	3-5
3.2.5	RANGE	3-5
3.2.6	ADD	3-6
3.2.7	DELETE	3-6
3.2.8	CLEAR	3-7
3.2.9	INITIALIZE	3-7
3.2.10	STATUS	3-8
3.2.11	SUMMARY	3-8
3.2.12	ERRORS	3-8
3.2.13	ASSIGN	3-9
3.2.14	GO	3-10
3.2.15	CONTINUE	3-10
3.2.16	**SYNTAX ERROR**	3-11
4	GETTING THE EXERCISER STARTED	4-1
4.1	PREPARATION	4-1
4.2	LOADING	4-1
4.3	DEVICE ADDRESSES	4-1
4.4	STARTING ADDRESSES	4-2
4.5	OPERATING PROCEDURES	4-2
4.6	ERRORS	4-3

TABLE OF CONTENT'S (CONTD)		PAGE
-----		----
5	COMMAND INPUT MODE DESCRIPTION	5-1
5.1	SHORTENED COMMANDS	5-1
5.2	RUBOUT FEATURES	5-1
5.3	ENTERING COMMAND MODE	5-1
5.4	UPPER OR LOWER CASE	5-2
6	EXERCISE MODE DESCRIPTION	6-1
6.1	TRANSMIT EVENT	6-1
6.2	RECEIVE EVENT	6-1
6.3	DATA GENERATION EVENT	6-1
6.4	ADDRESS QUEUE EVENT	6-2
6.5	ERRORS UPDATE EVENT	6-2
6.6	SPECIAL EVENTS	6-2
APPENDIX A		
	EXERCISER OVERALL FLOW	A-1
APPENDIX B		
	EXERCISER COMMANDS	B-1
APPENDIX C		
	ERROR DESCRIPTIONS	C-1
APPENDIX D		
	STATISTICAL INFORMATION	D-1
LISTING...		

PARALLEL COMMUNICATIONS LINK
EXERCISER PROGRAM

1 GENERAL

1.1 GENERAL DESCRIPTION

THE PDP-11 PARALLEL COMMUNICATIONS LINK (PCL11) EXERCISER IS WRITTEN TO EXERCISE A FULL SET OR PARTIAL SET OF PCL11 UNITS, EACH ON ITS OWN PDP-11. THE MAXIMUM NUMBER OF UNITS (FULL SET) WHICH MAY BE EXERCISED IS 31, ALTHOUGH THE NORMAL MAXIMUM CONFIGURATION INCLUDES ONLY 16 PCL11'S. THE MINIMUM NUMBER OF UNITS IS 1.

THE EXERCISER IS OPERATED BY MEANS OF THE OPERATOR ENTERING COMMANDS AT EACH PDP-11 CONSOLE TERMINAL WHICH WILL DESCRIBE TO THE EXERCISER THE NUMBER OF TARGET RECEIVERS TO BE COMMUNICATED WITH AND THEIR T.D.M. BUS ADDRESSES. STATUS AND ERROR REPORTS ARE ACHIEVED ALSO IN RESPONSE TO OPERATOR COMMANDS.

EACH PCL11 TRANSMITTER IN THE 'CHAIN' OF PDP-11'S MUST BE TOLD HOW MANY OF THE RECEIVERS IN THE 'CHAIN' IT SHOULD SEND DATA TO. THEN, UPON THE OPERATOR'S COMMAND, EACH PCL11 TRANSMITTER INVOLVED WILL BEGIN SENDING RANDOM DATA PATTERNS TO EACH RECEIVER IT HAS BEEN TOLD TO COMMUNICATE WITH.

RECEIVERS IN THE CHAIN ARE ALWAYS RECEPTIVE TO DATA FROM ANY ONE TRANSMITTER AT A TIME AND WILL CHECK THAT THE DATA RECEIVED IS CORRECT. THE RECEIVER HANDLER PORTION OF THE EXERCISER WILL ALSO GENERATE A TABLE OF ERRORS THAT MAY BE EXAMINED PON ISSUANCE OF THE 'ERRORS' COMMAND.

TRANSMITTERS IN THE CHAIN ARE ACTIVATED OR DE-ACTIVATED BY THE OPERATOR AND ARE DE-ACTIVATED BY A 'MASTER-DOWN' ERROR OR A UNIBUS TIMEOUT (TRAP TO 4). THE TRANSMITTER HANDLER PORTION OF THE EXERCISER WILL LOOK AFTER THE TRANSMISSION OF RANDOM DATA TO EACH RECEIVER ON ITS 'LIST' AND GENERATE A STATUS TABLE AND AN ERROR TABLE. THE STATUS TABLE WILL INFORM THE OPERATOR OF THE SUCCESS OR FAILURE OF THAT TRANSMITTER TO COMMUNICATE WITH EACH OF THE RECEIVERS ON ITS LIST. THE ERROR TABLE WILL SHOW THE OPERATOR ANY HARDWARE ERRORS ENCOUNTERED DURING COMMUNICATION.

A SPECIAL 'SUMMARY' COMMAND WILL GIVE A CONDENSED ERROR TABLE INDICATING ONLY THE ERROR NUMBER, THE ADDRESS IN THE LISTING OF THE ERROR, AND THE TOTAL NUMBER OF OCCURRENCES OF THAT ERROR WITHOUT REGARD TO WHICH RECEIVER AND TRANSMITTER WERE CONNECTED.

1.2 DIFFERENCES IN EXERCISER V-02

- 1.2.1 STARTING AND RESTARTING (SEE SEC. 4.4)
V-02 OF THE EXERCISER STARTS AT LOCATION 200. WHEN RESTARTED AT LOCATION 204, THE STATUS AND ERRORS ARE PRESERVED.
- 1.2.2 UPPER/LOWER CASE INPUT (SEE SEC. 5.4)
EITHER UPPER OR LOWER CASE ALPHA CHARACTERS ARE ACCEPTED BY V-02 OF THE EXERCISER. WHICHEVER IS TYPED IN, THE UPPER CASE VERSION IS ECHOED.
- 1.2.3 ERRORS COMMAND (SEE SEC. 3.2.12)
THE 'ERRORS' COMMAND HAS BEEN ADDED TO THE EXERCISER TO GIVE MORE DETAIL ABOUT THE ERROR CONDITIONS.
- 1.2.4 RIB COMMAND (SEE SEC. 3.2.4)
THE 'RIB' COMMAND HAS BEEN ADDED TO THE EXERCISER TO ALLOW RUNNING IN A RE-TRY - IF - BUSY MODE, WHICH BRINGS THE ATTEMPTS AND SUCCESSES COUNTS MORE IN LINE.
- 1.2.5 ASSIGN COMMAND (SEE SEC. 3.2.13)
THE 'ASSIGN' COMMAND WAS ADDED TO ALLOW CONSOLE CHANGING OF THE PCL11 UNIBUS ADDRESSES AND VECTORS WITHIN THE EXERCISER.
- 1.2.6 GO COMMAND (SEE SEC. 3.2.14)
THE 'GO' COMMAND HAS BEEN ALTERED SO THAT THE RECEIVER ADDRESS QUEUE, STATUS OF ATTEMPTS AND SUCCESSES, AND THE ERROR TABLE ARE ALL CLEARED.
- 1.2.7 ADDRESS QUEUE EVENT (SEE SEC. 6.4)
THIS EVENT WAS CHANGED SO THAT IF THE EXERCISER WAS TOLD TO 'GO' OR 'CONTINUE', AND NOTHING WAS LOADED INTO THE 'RECEIVER ADDRESS QUEUE', THE EXERCISER WOULD NOT GO.
- 1.2.8 LOADING OF THE SILO (SEE SEC. 3.2.1)
IN THE PREVIOUS EXERCISER, THE SILO COULD BE LOADED IN SUCH A WAY AS TO CAUSE 'HARD TO TRACE' HARDWARE ERROR INDICATIONS. NOW A 'PAD' VALUE HAS BEEN INSERTED BETWEEN SIMILAR ENTRIES SO THAT THIS CANNOT HAPPEN.
- 1.2.9 NEW CONTROL CHARACTERS (SEE SEC. 3.1)
V-02 OF THE EXERCISER HAS THROWN OUT THE 'ESCAPE/ALTMODE' CHARACTER IN FAVOUR OF 'CNTRL-C'. ALSO, CNTRL-S AND CNTRL-Q WERE ADDED TO CONTROL THE PRINTOUT ON VIDEO TERMINALS.
- 1.2.10 CARRIAGE RETURN (SEE SEC. 3.1.6)
THIS VERSION OF THE EXERCISER WILL NOT GO INTO 'LIMBO' IF A CARRIAGE RETURN IS ENTERED BY ITSELF. IT SIMPLY ECHOES ANOTHER \$.
- 1.2.11 RUBOUT (SEE SEC. 3.1.8)
THE EXERCISER WILL NOW INDICATE WHEN ALL HAS BEEN RUBBED OUT BY RETURNING A <CR-LF> AND '\$' WHEN THE COMMAND INPUT BUFFER IS EMPTY.

1.2.12 STATUS TABLE (SEE SEC. 2.1)

THE 'ATTEMPTS' AND 'SUCCESSSES' COUNTS IN THE STATUS TABLE HAVE BEEN BEEFED UP TO DOUBLE PRECISION DECIMAL NUMBERS. ALSO, AN INDICATION HAS BEEN ADDED TO THE TABLE OF WHETHER THE UNIT IS MASTER OR SECONDARY, AND WHETHER 'RIB' IS SET OR CLEAR. ALSO, THE ELAPSED TIME OF THE RUN SO-FAR IS PRINTED PROVIDING THAT THE PDP-11 HAS A LINE CLOCK.

1.2.13 SUMMARY TABLE (SEE SEC. 2.2)

THE 'NO. OF OCCURRENCES' COUNT IN THE SUMMARY TABLE HAS BEEN GIVEN A CEILING OF 65,528 (DECIMAL) AND WILL PRINT 'MXCNT' AFTER THAT. THE COUNT IS PRINTED IN DECIMAL.

1.2.14 DIFFERENCES IN V02A

THIS REVISION WAS BROUGHT ABOUT BY THE CHANGE OF THE PROMPT PRINTED WHILE IN COMMAND MODE. THE PROMPT WAS SIMPLY '\$'. IT HAS BEEN CHANGED TO 'PCL>' TO REDUCE THE CONFUSION ON SYSTEMS WHERE PDP-11'S ARE ATTACHED TO VAX11780'S. (IT IS ONLY THE DIAGNOSTIC SUPERVISOR ON THE VAX11780 THAT IS ALLOWED TO USE '\$'.)

2 EXERCISER TABLES

THE PCL11 EXERCISER MAINTAINS THREE TABLES: A 'STATUS' TABLE AN 'ERRORS' TABLE, AND A 'SUMMARY' TABLE.

2.1 STATUS TABLE

THE STATUS TABLE IS DESIGNED TO SHOW THE OPERATOR THE NUMBER OF SUCCESSFUL TRANSMISSIONS TO EACH RECEIVER RELATIVE TO THE NUMBER OF ATTEMPTS TO TRANSFER DATA TO EACH RECEIVER. IF THE TRANSMITTER HAS BEEN TOLD TO 'RE-TRY - IF - BUSY', THE NUMBER OF SUCCESSFUL TRANSFERS WILL BE VERY MUCH CLOSER TO THE NUMBER OF ATTEMPTS THAN WOULD BE THE CASE IF 'RE-TRY - IF - BUSY' WERE NOT THE ORDER. HOWEVER, THE TOTAL NUMBER OF ATTEMPTS WOULD BE GREATER WITH 'RIB' CLEAR.

THE INFORMATION GIVEN IN THE TABLE INCLUDES:

RUN STATE - STATE OF MASTER, SECONDARY, AND 'RIB'

RECEIVER ADDRESS - THAT THIS TRANSMITTER HAS BEEN INSTRUCTED TO COMMUNICATE WITH.

CONNECTION ATTEMPTS - NUMBER OF TIMES THIS XMTR TRIED CONNECTING TO THE RECEIVER OF THE ABOVE ADDRESS.

SUCCESSFUL CONNECTIONS- NUMBER OF TIMES THAT THE ABOVE ATTEMPTS WERE SUCCESSFUL.

RIB IS -SET- (OR CLEAR)

THIS UNIT IS -MASTER- (OR SECONDARY)

ELAPSED TIME (HRS,MIN,SEC,TK)...0:0:4:35

RCVR ADDRESS CONNECTION ATTEMPTS SUCCESSFUL CONNECTIONS

1	X	Y
2	X	Y
3	X	Y
.	.	.
.	.	.
37	X	Y

THE ENTRIES UNDER 'RCVR ADDRESS' INCLUDE ONLY THOSE ADDRESSES ENTERED BY THE OPERATOR USING THE 'RANGE' OR 'ADD' COMMANDS.

THE ENTRIES UNDER 'CONNECTION ATTEMPTS' AND 'SUCCESSFUL CONNECTIONS' ARE DOUBLE PRECISION DECIMAL NUMBERS. THEY ARE CAPABLE OF INCREASING TO A VALUE OF 655,359,999. IF THERE ARE A LARGER NUMBER OF ATTEMPTS OR SUCCESSES FOR A PARTICULAR RECEIVER THAN THAT AMOUNT, THE ENTRY IN THE TABLE WILL APPEAR AS : '*****'.

SINCE ANY DATA RECEIVED BY A PCL11 RECEIVER MUST BE CHECKED, THE DATA PATTERN, WHICH IS RANDOM, MUST BE RECREATED BY THE RECEIVING EXERCISER BASED ON THE FIRST WORD RECEIVED. THIS MUST THEN BE COMPARED WORD-FOR-WORD WITH THE RECEIVED DATA. IT IS READILY APPARENT THAT THIS TAKES TIME AND WILL MAKE THAT PARTICULAR RECEIVER UNAVAILABLE DURING THAT TIME. ON A LARGE SYSTEM, WITH MANY PCL11'S IT MIGHT BE QUITE A WHILE BETWEEN SUCCESSIVE SUCCESSFUL CONNECTIONS OF A GIVEN TRANSMITTER TO THE SAME RECEIVER, SINCE ALL TRANSMITTERS MAY VERY WELL BE TRYING FOR THE SAME RECEIVERS AT THE SAME TIME. IT IS FOR THIS REASON, THEN, THAT IT NEED NOT BE ALARMING THAT

THERE IS A DIFFERENCE BETWEEN THE NUMBER OF SUCCESSES AND THE NUMBER^{I 1}
OF ATTEMPTS.

SEQ 0008

3.

THE PRINTING OF THE STATUS TABLE MUST BE INITIATED BY THE OPERATOR ON EACH PDP-11 INVOLVED DURING THE COURSE OF AN EXERCISE RUN. THE STATUS TABLE OUTPUT IS NOT CAUSED BY ANY OTHER MEANS THAN THE OPERATOR ENTERING COMMAND MODE AND TYPING 'STATUS'. ALSO IT MUST BE NOTED THAT ALL TRANSMITTER ACTIVITY IS SUSPENDED ON THE UNIT WHILE THE STATUS IS BEING PRINTED OUT.

2.2 SUMMARY TABLE

THE SUMMARY TABLE IS DESIGNED TO SHOW THE OPERATOR A SUMMARY OF ERRORS WHICH HAVE OCCURRED IN EITHER THE RECEIVER OR TRANSMITTER HARDWARE ON EACH PDP-11.

INCLUDED IN THIS TABLE ARE:

ERROR NUMBER - FOR REFERENCE LATER IN THIS DOCUMENT TO INDICATE THE TYPE OF FAILURE.

ERROR ADDRESS - FOR REFERENCE IN THE PCL11 EXERCISER LISTING.

NO. OF OCCURRENCES - TO INDICATE HOW OFTEN THIS ERROR HAS OCCURRED DURING RUN TIME.

THE SUMMARY TABLE IS PRINTED IN THE FORM:

ERROR NUMBER	ERROR ADDRESS	NO. OF OCCURRENCES
1	xxx	yyyy
2	xxx	yyyy
3	xxx	yyyy
.	.	.
30	xxx	yyyy

THE ENTRIES UNDER 'ERROR NUMBER' INCLUDE ONLY THOSE ERRORS WHICH HAVE A 'NO. OF OCCURRENCES' GREATER THAN 0.

THE ENTRIES UNDER 'ERROR ADDRESS' REPRESENT THE OCTAL MEMORY ADDRESS OF THE OCCURRENCE OF THE ERROR WHICH OCCURRED. THE OPERATOR MAY FIND THIS USEFUL FOR LOCATING, IN THE LISTING, THE PORTION OF THE PROGRAM WHERE THE ERROR OCCURRED.

THE ENTRIES UNDER 'NO. OF OCCURRENCES' IS THE TOTAL (OR SUMMARY VALUE) OF ALL THE ERRORS OF THAT ERROR NUMBER WHICH OCCURRED AT THIS TERMINAL REGARDLESS OF WHAT TRANSMITTER WAS CONNECTED TO WHAT RECEIVER.

THE PRINTING OF THE SUMMARY TABLE MUST BE INITIATED BY THE OPERATOR ON EACH PDP-11 INVOLVED DURING THE COURSE OF AN EXERCISE RUN. THE SUMMARY TABLE OUTPUT IS NOT CAUSED BY ANY OTHER MEANS THAN THE OPERATOR ENTERING COMMAND MODE AND TYPING 'SUMMARY'. ALSO IT MUST BE NOTED THAT ALL TRANSMITTER ACTIVITY IS SUSPENDED ON THE UNIT WHILE THE SUMMARY TABLE IS BEING PRINTED OUT.

2.3 ERROR TABLES

THE ERROR TABLES ARE DESIGNED TO SHOW THE OPERATOR THE ERRORS WHICH OCCURRED DURING AN EXERCISE RUN IN ENOUGH DETAIL SO THAT HE CAN DETERMINE AS CLOSELY AS POSSIBLE WHICH TRANSMITTER AND/OR WHICH RECEIVER WERE CONNECTED AT THE TIME OF THE ERROR. THERE ARE 15 OCTAL TRANSMITTER TYPE ERRORS AND 13 OCTAL RECEIVER TYPE ERRORS. FOR EACH OF THE TRANSMITTER ERRORS, ANY ONE OF 31 POSSIBLE RECEIVERS MAY HAVE BEEN CONNECTED TO THAT TRANSMITTER AT THE TIME OF THE ERROR. THE ONE THAT ACTUALLY WAS CONNECTED IS LISTED IN THIS ERROR TABLE ALONGSIDE THE ERROR NUMBER AND THE COUNT OF THAT OCCURRENCE. FOR EACH OF THE RECEIVER-TYPE ERRORS, ANY ONE OF 31 TRANSMITTERS MAY HAVE BEEN TALKING TO THIS RECEIVER AT THE TIME OF THE ERROR. AGAIN, THE ACTUAL ONE IS LISTED IN THE RECEIVER ERRORS TABLE.

THE ENTRIES IN THESE TABLES THEN, INCLUDE:

ERROR NUMBER - FOR REFERENCE LATER IN THIS DOCUMENT TO INDICATE THE TYPE OF FAILURE.

CONNECTED XMTR (RCVR) - TO INDICATE THE ACTUAL ERRONEOUS LINK.

ERROR COUNT - TO SHOW THE NUMBER OF OCCURRENCES OF THIS ERROR WITH THIS CONNECTION.

THE TABLES ARE ALWAYS PRINTED TOGETHER AND ARE PRINTED IN THE FOLLOWING FORMAT:

TRANSMITTER ERRORS:

ERROR NO.	CONCTD RCVR	ERROR COUNT
1	1	63
1	2	7
1	6	48
15	37	2

RECEIVER ERRORS:

ERROR NO.	CONCTD XMTR	ERROR COUNT
16	1	22
16	2	22
27	37	12
30	6	2

IF THE COMMAND IS GIVEN TO DUMP THESE ERROR TABLES AND IT IS FOUND THAT NO ERRORS EXIST, THE EXERCISER RESPONDS IN THE FOLLOWING WAY:

** NO ERRORS TO REPORT YET **

IF THERE ARE, FOR EXAMPLE, TRANSMITTER ERRORS BUT NO RECEIVER ERRORS, THEN UNDER THE HEADING 'ERROR NO.' OF THE RECEIVER ERRORS WILL BE PRINTED: (NONE).

2.4 CLEARING OF THE TABLES

THE 'STATUS TABLE' IS NORMALLY CREATED BY MEANS OF THE OPERATOR ENTERING, VIA THE 'RANGE, OR ADD' COMMANDS, SOME RECEIVER ADDRESSES.

FOR EXAMPLE, IF THE 'ADD' COMMAND WERE USED TO ENTER RECEIVER ADDRESSES 1, 2, 3, 4, & 5, AND IMMEDIATELY THEREAFTER THE STATUS WAS REQUESTED, THE STATUS TABLE WOULD INDICATE ALL OF THE SELECTED RECEIVERS BUT THE ATTEMPTED AND SUCCESSFUL CONNECTION COUNTS FOR EACH OF THOSE RECEIVERS WOULD BE 0.

THE STATUS TABLE IS COMPLETELY CLEARED BY USING THE 'CLEAR' COMMAND, OR BY USING THE 'DELETE' COMMAND AND DELETING ALL THOSE ADDRESSES INDICATED IN THE STATUS TABLE. THE 'INITIALIZE' COMMAND ALSO CLEARS THE STATUS TABLE COMPLETELY BY INTERNALLY CALLING THE 'CLEAR' COMMAND.

THE 'SUMMARY TABLE' IS NOT AFFECTED BY THE 'CLEAR' COMMAND BUT IS ENTIRELY CLEARED BY THE 'INITIALIZE' COMMAND.

LIKewise, THE 'ERRORS TABLE' IS COMPLETELY CLEARED BY THE 'INITIALIZE' COMMAND BUT IS NOT AFFECTED BY THE 'CLEAR' COMMAND.

WHEN THE OPERATOR 'STARTS' THE EXERCISER BY USING THE 'GO' COMMAND, THE FOLLOWING RESULTS CAN BE EXPECTED ON THE TABLES:

THE ADDRESS ENTRIES OF THE 'STATUS' TABLE ARE UNAFFECTED.

THE ATTEMPTS AND SUCCESSES ENTRIES OF THE STATUS TABLE ARE CLR'D

THE ENTIRE SUMMARY TABLE IS CLEARED.

THE ENTIRE ERRORS TABLE IS CLEARED.

!-----CAUTION-----!

IT IS IMPORTANT TO NOTE THAT THE RECEIVER ERRORS ARE CLEARED ALONG WITH THE TRANSMITTER ERRORS UPON THE ISSUANCE OF THE 'GO' COMMAND. THEREFORE, THE OPERATOR MUST NOTE THAT THE NUMBER OF RECEIVER ERRORS INDICATED IN EITHER THE SUMMARY TABLE OR THE RECEIVER ERRORS TABLE IS ONLY THE NUMBER ACCUMULATED SINCE 'GO' WAS TYPED.

3 EXERCISE COMMANDS

3.1 CONTROL CHARACTERS

3.1.1 CONTROL-C

THIS CHARACTER IS USED TO GET THE EXERCISER INTO 'COMMAND' MODE SO THAT ANY OF THE CONTROLLING COMMANDS MAY BE ENTERED.

TYPING CONTROL-C (^C) ECHOS '^C' AND TERMINATES ANY OTHER FUNCTIONS WHICH MAY BE TAKING PLACE AT THE TIME. IF ANY OF THE TABLES ARE BEING PRINTED, THE CURRENT LINE WILL BE COMPLETED BEFORE '^C' IS ECHOED. THEN THE TABLE WILL BE TRUNCATED. THE EXERCISER WILL THEN ENTER 'COMMAND' MODE AND INDICATE THIS BY PRINTING 'PCL>'.

'PCL>' IS THE PROMPT WHICH INDICATES THAT A COMMAND MAY BE ENTERED BY THE OPERATOR.

TYPING 'CNTRL-C' WHILE THE EXERCISER IS 'EXERCISING' WILL CAUSE TERMINATION OF TRANSMITTER ACTIVITY ON THAT PDP-11, ECHO '^C', AND PRINT THE COMMAND MODE PROMPT 'PCL>'.

3.1.2 CONTROL-O

THIS CHARACTER IS USED TO TERMINATE THE CURRENT PRINTOUT. IT CAUSES THE ENTIRE CONTENTS OF THE OUTPUT QUEUE TO BE THROWN AWAY. WHEN TABLES ARE BEING PRINTED, ONLY A SINGLE LINE IS IN THE OUTPUT QUEUE AT ANY GIVEN TIME. THIS ALLOWS 'SKIPPING' OF LINES DURING TABLE OUTPUT TO SAVE TIME.

WHEN CONTROL-O IS TYPED, THE EXERCISER WILL ECHO '^O', CLEAR THE OUTPUT QUEUE, AND THEN RETURN TO WHICHEVER MODE IT WAS IN WHEN 'CNTRL-O' WAS TYPED IN. THEREFORE, IF ^O IS TYPED IN DURING EXERCISE TIME, THE '^O' WILL STILL BE ECHOED. IF 'CNTRL-O' IS TYPED ARBITRARILY WHILE IN COMMAND MODE, THE '^O' WILL BE ECHOED BUT THERE WILL NOT BE ANOTHER 'PCL>' GIVEN (YET THE EXERCISER WILL ACCEPT COMMANDS).

3.1.3 CONTROL-U

THIS CHARACTER IS USED (AS IT IS IN MOST D.E.C. SOFTWARE) TO THROW AWAY THE INPUT TYPED THUS FAR. THIS CONTROL CHARACTER SHOULD BE USED IF A MISTAKE IS NOTICED IN THE COMMAND STRING BEFORE CARRIAGE RETURN IS HIT. IF CARRIAGE RETURN IS HIT, THE ERRONEOUS COMMAND WILL BE EXECUTED UP TO THE POINT OF THE ERROR, THEN A SYNTAX ERROR MESSAGE WILL BE PRINTED.

WHEN CONTROL-U IS TYPED, THE EXERCISER WILL ECHO '^U', CLEAR THE INPUT QUEUE, THEN ISSUE A NEW 'PCL>' AND AWAIT A FURTHER COMMAND.

3.1.4 CONTROL-S

THIS CHARACTER IS USED TO SUSPEND PRINTER OUTPUT. NOTHING IS LOST WHEN IT IS USED, BUT THE PRINTOUT IS 'HELD' UNTIL THE ISSUANCE OF CONTROL-Q OR SIMPLY TYPING ANY OTHER CHARACTER. THIS FEATURE IS USEFUL WHEN GETTING TABLE PRINTOUTS ON A VIDEO TERMINAL AND IT IS DESIREOUS TO STOP THE DISPLAY. IT IS ALSO USEFUL TO STOP THE PRINTOUT IN ORDER TO FIX THE PAPER OR ADD MORE PAPER ON ANY HARD COPY PRINTER.

NOTHING IS ECHOED UPON , OR AFTER, THE ISSUANCE OF A CONTROL-S.

3.1.5 CONTROL-Q

THIS CHARACTER IS USED TO RESUME PRINTOUT AFTER IT WAS STOPPED BY USE OF CONTROL-S. TYPING ANY OTHER CHARACTER WILL ALSO RESUME OUTPUT BUT THE CONTROL-Q CHARACTER IS NOT ENTERED INTO THE INPUT QUEUE, AND SIMPLY RESUMES PRINTOUT.

NOTHING IS ECHOED AS A RESULT OF TYPING CONTROL-Q EXCEPT WHATEVER PRINTOUT HAD BEEN SUSPENDED BY THE CONTROL-S CHARACTER.

3.1.6 CARRIAGE RETURN

THE CARRIAGE RETURN CHARACTER IS USED TO TERMINATE COMMAND STRINGS AND SIGNIFIES ENTRANCE OF A COMMAND. ALL COMMANDS MUST BE TERMINATED WITH EITHER A CARRIAGE RETURN OR A LINE FEED. IF ONLY CARRIAGE RETURN IS TYPED, (BLANK COMMAND) THE EXERCISER SIMPLY ISSUES A NEW 'PCL>'.

WHEN CARRIAGE RETURN 'CR>' IS TYPED, BOTH CARRIAGE RETURN AND LINE FEED ARE ECHOED.

3.1.7 LINE FEED

THE LINE FEED CHARACTER IS TREATED EXACTLY AS THE CARRIAGE RETURN CHARACTER. IF ONLY A LINE FEED IS TYPED, (BLANK COMMAND) THE EXERCISER ISSUES A NEW 'PCL>'.

3.1.8 RUBOUT

THIS CHARACTER IS USED TO 'EDIT' THE COMMAND STRING WHILE IT IS BEING TYPED. EACH RUBOUT TYPED WILL REMOVE A CHARACTER FROM THE COMMAND BUFFER. WHEN ALL THE CHARACTERS HAVE BEEN REMOVED, THE EXERCISER WILL ISSUE A NEW 'PCL>' TO INDICATE THE ENTIRE COMMAND HAS BEEN ERASED.

EACH TIME A RUBOUT IS TYPED, A '^' IS ECHOED AND THE LAST CHARACTER WHICH WAS INPUT IS REMOVED.

3.2 COMMANDS

THIS SECTION DEALS WITH THE FULL COMMANDS OF THE EXERCISER WHICH ALLOW THE SETTING UP OF CONDITIONS TO BEST SUIT THE SYSTEM BEING TESTED. IN THE COMMANDS SHOWN, THE MINIMUM AMMOUNT REQUIRED TO BE TYPED IS SET IN SQUARE BRACKETS []. FOR EXAMPLE, IN THE COMMAND:

[I]INITIALIZE

ALL OF THE FOLLOWING ARE ACCEPTABLE:

I, IN, INI, INIT, INITI, INITIA, INITIAL, INITIALI, INITIALIZ
AND INITIALIZE

WHEREAS THE FOLLOWING ARE NOT ACCEPTABLE:

INT, INITL, INITIALIZES

THERE ARE SOME COMMANDS WHICH MAY BE EXECUTED WITH OR WITHOUT ARGUMENTS. IN THESE CASES, THE <ARGUMENTS> WILL BE SET IN ANGLE BRACKETS. ARGUMENTS FOR COMMANDS MUST BE NUMERIC, EXCEPT FOR THE COMMANDS 'MASTER', 'SECONDARY', AND 'RIB', AND IF THEY ARE TYPED IN AS DECIMAL NUMBERS, THEY MUST BE IMMEDIATELY FOLLOWED BY A DECIMAL POINT (PERIOD). FOR THE 'RANGE' AND 'ASSIGN' COMMANDS, THE ORDER OF THE INPUT OF ARGUMENTS MUST ADHERE TO THE DESCRIPTIONS (SECT 3.2.5, & 3.2.13)

3.2.1 SILO

[SI]LO <A B CN>

SILO -- CLEAR THE MASTER ADDRESS SILO AND RETURN TO 'AUTO ADDRESS MODE'.

SILO A B C . . . N

LOAD THE MASTER ADDRESS SILO WITH THE SEQUENCE 'ABC..N' AS MANY TIMES AS THE WHOLE SEQUENCE WILL FIT INTO THE SILO'S 50 LOCATIONS.

THIS COMMAND MONITORS THE ARGUMENTS CAREFULLY CHECKING FOR SEQUENTIAL ARGUMENTS WHICH ARE THE SAME, ARGUMENTS WHICH ARE 0, OR ARE GREATER THAN 37 OCTAL, AND CHECKING THAT THE LAST ARGUMENT IS NOT THE SAME AS THE FIRST.

IF ANY TWO SEQUENTIAL ARGUMENTS ARE FOUND TO BE THE SAME, INCLUDING THE FIRST AND THE LAST, A 'PAD' VALUE IS INSERTED BETWEEN THEM. THE 'PAD' VALUE IS 0. THIS ALSO INCREASES THE NUMBER OF ARGUMENTS IN THE SEQUENCE.

IF ANY OF THE ARGUMENTS ARE '0', OR GREATER THAN '37' OR NON-NUMERIC, OR IF THERE ARE MORE THAN 50. ARGUMENTS, THE COMMAND WILL BE ABORTED AND THE '**SYNTAX ERROR**' MESSAGE WILL BE PRINTED.

WHEN THE SILO HAS BEEN SUCCESSFULLY LOADED, THERE IS THE POSSIBILITY THAT ONE OR TWO MESSAGES WILL BE PRINTED, OR NONE MAY BE PRINTED.

IF THIS UNIT IS NOT MASTER, THE FOLLOWING WILL BE PRINTED:

'THIS UNIT IS NOT MASTER BUT HAS BEEN MADE SECONDARY
THE SILO YOU HAVE JUST LOADED WILL BE USED IF YOU CLEAR
THE CURRENT MASTER.'

IF IT WAS REQUIRED FOR THE EXERCISER TO 'PAD' THE SILO WITH 0'S

THE FOLLOWING WILL BE PRINTED:

SEQ 0015

'THE SILO HAS BEEN PADDED WITH THE ADDRESS '0'.'

S
S
S
T
T
H
M
S
S
C
T
S

S

C
W
C
T

S

C
M

E
1
A
1
C

3.2.2 MASTER

[M]ASTER [S]ET OR [C]LEAR

MASTER SET - WRITE A (1) INTO THE TMMR REGISTER BIT 08 IN THIS TRANSMITTER TO ATTEMPT TO SET MASTER. ANOTHER UNIT HAVING 'MASTER' SET, WILL DISALLOW THIS BIT TO BE SET.

MASTER CLEAR - WRITE A (0) INTO THE TMMR REGISTER BIT 08 IN THIS TRANSMITTER TO CLEAR 'MASTER'. IF NO OTHER UNIT HAS BEEN SET TO BE 'SECONDARY', THIS ACTION WILL CAUSE 'MASTER DOWN' ERRORS ON ALL UNITS ATTEMPTING TO TRANSMIT.

THIS COMMAND IS USED TO 'SOFTWARE SET' ONE OF THE PCL11 UNITS TO BE MASTER OF THE T.D.M. BUS. THE ONLY ARGUMENTS ALLOWED ARE '[S]ET' AND '[C]LEAR'. NO ARGUMENTS, OR ANY OTHER ARGUMENTS THAN THESE, WILL RESULT IN THE MESSAGE '**SYNTAX ERROR**'.

SINCE THE HARDWARE DESIGN WILL NOT ALLOW MASTER TO BE SET ON MORE THAN ONE UNIT CONNECTED TO THE SAME T.D.M. BUS, THIS COMMAND MAY NOT WORK IN SETTING MASTER. THERE IS NOT, HOWEVER, ANY IMMEDIATE INDICATION THAT THE 'MASTER SET' COMMAND WAS OR WASN'T SUCCESSFUL. THE OPERATOR MUST ISSUE THE 'STATUS' COMMAND TO DETERMINE IF A PARTICULAR UNIT IS MASTER.

3.2.3 SECONDARY

[SE]CONDARY [S]ET OR [C]LEAR

SECONDARY SET - WRITE A (1) INTO THE TMMR REGISTER BIT 09 IN THIS TRANSMITTER TO ATTEMPT TO SET 'SECONDARY'. THIS UNIT BEING 'MASTER' WILL DISALLOW THIS BIT TO BE SET.

SECONDARY CLEAR - WRITE A (0) INTO THE TMMR REGISTER BIT 09 IN THIS TRANSMITTER TO CLEAR 'SECONDARY'.

THIS COMMAND IS USED TO 'SOFTWARE SET' ONE OF THE PCL11 UNITS TO BE SECONDARY MASTER OF THE T.D.M. BUS. IF THE PCL11 UNIT WHICH CURRENTLY HAS MASTER SET CLEARS IT, AND THIS UNIT HAS SECONDARY SET, THEN THIS UNIT WILL BECOME THE NEW MASTER. IF THIS UNIT IS IN COMMUNICATION WITH A RECEIVER AT THE TIME IT BECOMES NEW MASTER, THE MESSAGE :

** THIS UNIT HAS BECOME 'NEW MASTER' **

IS PRINTED ON THE CONSOLE AND THE EXERCISER AUTOMATICALLY CONTINUES EXERCISING.

3.2.4 RIB

[RIB] [SET] OR [CLEAR]

RIB SET - WRITE A (1) INTO THE MEMORY LOCATION (BIT 15) WHICH IS USED AS THE TRANSMITTER COMMAND WORD. THE LOCATION IS TAGGED: 'TXMST'. THIS WILL CAUSE THE NEXT AND SUBSEQUENT TRANSMITTER EVENTS TO OCCUR IN THE 'RE-TRY - IF - BUSY' MODE.

RIB CLEAR - WRITE A (0) INTO THE MEMORY LOCATION (BIT 15) WHICH IS USED AS THE TRANSMITTER COMMAND WORD TAGGED 'TXMST'. THIS WILL CAUSE 'BUSY' INTERRUPTS TO OCCUR WHENEVER THE RECEIVER BEING ADDRESSED IS NOT READY OR NOT THERE.

THIS COMMAND IS USED TO 'SOFTWARE SET' THE 'RIB' OR RE-TRY IF BUSY FEATURE IN THE PCL11 TRANSMITTER. WITH 'RIB' SET, THE TRANSMITTER HARDWARE WILL CONTINUOUSLY RE-TRY TO CONNECT TO THE SELECTED RECEIVER UNTIL EITHER A 'TIME-OUT' OCCURS, OR THE CONNECTION IS MADE AND A 'SUCCESSFUL TRANSFER' OCCURS.

WITH 'RIB' CLEAR, THE HARDWARE PRODUCES AN INTERRUPT IMMEDIATELY UPON FINDING THE ADDRESSED RECEIVER BUSY OR NOT THERE. THE USE OF THIS COMMAND WILL CAUSE A DIFFERENCE IN THE 'SUCCESSFUL CONNECTION' COUNT RELATIVE TO THE 'ATTEMPTED CONNECTION' COUNT IN THE STATUS PRINT-OUT.

3.2.4 RANGE

[RANGE] [LOW HIGH]

RANGE A B PRODUCE A LIST, IN THE STATUS TABLE, OF ADDRESSES FROM ADDRESS 'A' TO 'B' AND MAKE THEM 'ACTIVE' RECEIVER ADDRESSES. 'B' MUST BE HIGHER THAN 'A'. BOTH 'A' AND 'B' ARE REPRESENTATIVE OF NUMERICAL VALUES WITHIN THE RANGE OF 1-37 OCTAL.

EXAMPLE:

THE COMMAND 'RANGE 12. 16.' WILL ACTIVATE RECEIVER ADDRESSES 12 UP TO AND INCLUDING 16 (DECIMAL), OR 14 UP TO AND INCLUDING 20 (OCTAL). THESE ADDRESSES WILL BE USED SEQUENTIALLY BY THE TRANSMITTER MODULE IN THE EXERCISER AS TARGET RECEIVER ADDRESSES.

3.2.6 ADD

[AD]D [1] 2 3 . . 37

ADD A B C

SELECT TARGET RECEIVER ADDRESSES. ADD THE NUMERIC VALUES OF A, B, C TO THE STATUS TABLE AND MAKE THEM 'ACTIVE RECEIVER ADDRESSES'. THIS COMMAND MAY HAVE ANY NUMBER OF ARGUMENTS PROVIDING THAT THERE IS AT LEAST ONE. THERE IS NO RESTRICTION ON THE ORDER OF THE ARGUMENTS AT ALL. THE SAME NUMBER MAY BE ADDED AS MANY TIMES AS YOU LIKE. THE ARGUMENTS MUST BE NUMERICAL AND BETWEEN THE VALUES 1 AND 31. (DECIMAL) OR 1 AND 37 (OCTAL).

EXAMPLE:

THE COMMAND 'ADD 1 12 3 3 22 8. ' WILL ACTIVATE RECEIVER ADDRESSES 1 3 10 12 AND 22 (OCTAL) OR 1 3 8. 10. AND 18. (DECIMAL) THIS COMMAND AFFECTS ONLY THOSE ADDRESSES INCLUDED IN THE ARGUMENTS OF THE COMMAND. IT MAY BE USED EFFECTIVELY IN CONJUNCTION WITH THE 'RANGE' COMMAND TO PRODUCE A AN EFFICIENT LIST OF THOSE RECEIVER ADDRESSES WHICH ARE TO BE COMMUNICATED WITH BY THIS TRANSMITTER.

3.2.7 DELETE

[D]ELETE [1] 2 3 . . 37

DELETE A B C

DELETE THE NUMERIC VALUES OF A, B, C FROM THE STATUS TABLE AND MAKE THEM 'INACTIVE RECEIVER ADDRESSES'. THIS COMMAND MAY HAVE ANY NUMBER OF ARGUMENTS PROVIDING THERE IS AT LEAST ONE. THERE IS NO RESTRICTION ON THE ORDER OF THE ARGUMENTS AT ALL. AND REDUNDANCIES ARE ACCEPTABLE. THE ARGUMENTS MUST BE NUMERIC AND BETWEEN THE VALUES 1 AND 31. (DECIMAL) OR 1 AND 37 (OCTAL).

EXAMPLE:

THE COMMAND 'DELETE 12 22 ' WILL MAKE RECEIVER ADDRESSES 12 AND 22 'INACTIVE' SO THAT THIS TRANSMITTER WILL NOT ATTEMPT TO COMMUNICATE WITH THEM. THIS COMMAND MAY BE USED EFFECTIVELY IN CONJUNCTION WITH THE 'RANGE' COMMAND AS FOLLOWS:

RANGE 1 7
DELETE 2 5

WILL PRODUCE A LIST OF 'ACTIVE' RECEIVERS WITH THE ADDRESSES:

1, 3, 4, 6, AND 7

3.2.8 CLEAR

[CL]EAR

CLEAR - CLEAR THE ENTIRE STATUS TABLE. REMOVE ALL RECEIVER ACTIVE FLAGS AND CLEAR ALL CONNECTION ATTEMPTS AND SUCCESSES FROM EVERY TABLE LOCATION. THIS COMMAND IS MOSTLY USEFUL WHEN IT IS DESIRED TO 'RE-START' THE EXERCISER WITH A FRESH SLATE BUT NOT DISTURB THE ERROR SUMMARY TABLE NOR THE ERRORS TABLES.

THE 'CLEAR' COMMAND HAS NO AFFECT ON ANY OTHER TABLES IN THE EXERCISER BUT IT DOES ALSO CLEAR THE INTERNAL (SOFTWARE) QUEUE OF RECEIVER ADDRESSES.

EXAMPLE:

CONSIDER THE FOLLOWING STRING OF COMMANDS:

RANGE 1 21
ADD 27 30 31 32 33 34
DELETE 17 20
CLEAR
RANGE 1 16
ADD 21 37

THE RESULT WOULD BE THE 'ACTIVATION' OF RECEIVER ADDRESSES:

1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 21, 37

3.2.9 INITIALIZE

[I]INITIALIZE

INITIALIZE - INITIALIZE EVERYTHING ABOUT THIS TRANSMITTER AND CLEAR ALL TABLES ASSOCIATED WITH THIS PDP-11. INITIALIZE PERFORMS A HARDWARE CLEAR OF THE TRANSMITTER REGISTERS, RESETS ALL OF THE EVENT FLAGS ASSOCIATED WITH THE SENDING OF DATA, CLEARS THE SOFTWARE QUEUE OF RECEIVER ADDRESSES, INITIALIZES THE SEEDS USED FOR GENERATION OF RANDOM DATA, DOES A 'CLEAR' COMMAND AS ABOVE AND CLEARS BOTH ERROR TABLES.

3.2.10 STATUS

[ST]ATUS

STATUS -

SET THE EVENT FLAG WHICH CAUSES THE CURRENT STATUS TABLE TO BE PRINTED. EVERY TIME THE STATUS COMMAND IS ISSUED, A STATUS HEADER IS PRINTED (SEE 2.1). THEN, IN NUMERICAL ORDER THE 'ACTIVE' RECEIVER ADDRESSES, NUMBER OF ATTEMPTED CONNECTIONS, AND NUMBER OF SUCCESSFUL CONNECTIONS IS PRINTED ON A LINE-BY-LINE BASIS. THE PRINTING OF THE STATUS TABLE DOES (LIKE ALL COMMANDS) INHIBIT ANY ACTION BY THE TRANSMITTER IN THAT PDP-11 BUT THE RECEIVER IS ALWAYS ACTIVE.

3.2.11 SUMMARY

[SL]MMARY

SUMMARY -

SET THE EVENT FLAG WHICH CAUSES THE CURRENT ERROR SUMMARY TABLE TO BE PRINTED. EVERY TIME THE SUMMARY COMMAND IS ISSUED, A SUMMARY HEADER IS PRINTED (SEE 2.2). THEN, IN NUMERICAL ORDER, THE ERROR NUMBERS, ERROR ADDRESSES, AND NO. OF OCCURRENCES ARE PRINTED (IF ANY) ON A LINE-BY-LINE BASIS. AGAIN, TRANSMITTER ACTIVITY IS SUSPENDED UNTIL THE EXERCISER IS 'CONTINUED'.

3.2.12 ERRORS

[E]RRORS

ERRORS -

SET THE EVENT FLAG WHICH CAUSES THE CURRENT TRANSMITTER ERROR TABLE TO BE PRINTED. WHEN THE TRANSMITTER ERROR EVENT IS FINISHED, IT WILL AUTOMATICALLY SET THE RECEIVER ERROR EVENT FLAG WHEREBY THE RECEIVER ERROR TABLE WILL BE PRINTED (SEE 2.3).

WHEN THE 'ERROR' COMMAND IS ISSUED, A CHECK IS MADE OF THE ENTIRE TRANSMITTER AND RECEIVER TABLES TO DETERMINE IF THERE HAD BEEN ANY ERROR OCCURRENCES TO DATE. IF THERE WERE NO ERRORS IN EITHER TABLE, THEN THE MESSAGE:

'NO ERRORS TO REPORT YET'

IS PRINTED. OTHERWISE, THE ERROR NUMBER (IN NUMERICAL ORDER), THE CONNECTED RCVR/XMTR, AND THE ERROR COUNT ARE PRINTED ON A LINE-BY-LINE BASIS. TRANSMITTER ACTIVITY IS AGAIN SUSPENDED UNTIL THE EXERCISER IS 'CONTINUED'.

3.2.13 ASSIGN

[AS]SIGN <XM ADDR XM VCT RC ADDR RC VCTR>

ASSIGN -

GIVE TO THE EXERCISER, THE UNIBUS ADDRESSES AND VECTORS OF THE PCL11 UNIT WHICH THE OPERATOR DESIRES TO EXERCISE.

AS IS INDICATED BY THE ANGLE BRACKETS, THE ASSIGN COMMANDS' ARGUMENTS ARE OPTIONAL. IF THE ASSIGN COMMAND IS ISSUED WITH NO ARGUMENTS, THEN THE 'DEFAULT' (NORMAL) ADDRESSES AND VECTORS ARE ASSIGNED. THESE ARE:

XMTR ADDR	164200
XMTR VECTOR	170
RCVR ADDR	164220
RCVR VECTOR	174

THE 'ASSIGN' COMMAND MAY ALSO BE USED WITH ANY, OR ALL OF FOUR ARGUMENTS. HOWEVER THE PROPER FIELD MUST BE USED TO ENTER THE DESIRED ADDRESS:

ASSIGN AAAAA BBB CCCCC DDD

TO ENTER ONLY THE TRANSMITTER ADDRESS, ONLY THE FIELD AAAAA NEED BE USED.

ASSIGN 166200

TO ENTER THE TRANSMITTER VECTOR (BBB), FIRST THE TRANSMITTER ADDRESS, THEN THE VECTOR IS TYPED:

ASSIGN 166200 700

TO ENTER THE RECEIVER ADDRESS (FIELD CCCCC), FIRST THE TRANSMITTER ADDRESS, THEN THE TRANSMITTER VECTOR, THEN THE RECEIVER ADDRESS IS TYPED:

ASSIGN 166200 700 166220

FINALLY, TO ASSIGN THE RECEIVER VECTOR, THE XMTR ADDRESS, XMTR VECTOR, RCVR ADDRESS AND THEN THE RECEIVER VECTOR IS TYPED:

ASSIGN 166200 700 166220 704

NOTE THAT EACH ARGUMENT MUST BE SEPARATED BY A SPACE (NOT A COMMA).

AT THE SUCCESSFUL COMPLETION OF THE 'ASSIGN' COMMAND, THE EXERCISER WILL BE STARTED OVER JUST AS THOUGH THE OPERATOR HAD STARTED AT 200.

A '**SYNTAX ERROR**' WILL OCCUR WITH ANY OF THE FOLLOWING CONDITIONS:

TOO MANY ARGUMENTS

ARGUMENT IS NOT NUMERIC

ADDRESS ARGUMENT NOT IN I/O ADDRESS FIELD
(I.E. ABOVE 163776)

VECTOR ARGUMENT IS NOT IN VECTOR FIELD
(I.E. FROM 0 TO 776)

ADDRESS ARGUMENT HAS WRONG OFFSET
(I.E. LAST 4 BITS MUST BE 0)

VECTOR ARGUMENT HAS WRONG OFFSET
(I.E. LAST 2 BITS MUST BE 0)

IMPROPER SPELLING OF 'ASSIGN' OR IMPROPER USE OF DECIMAL NUMBERS.

3.2.14 GO

[G]O

GC -

START THE EXERCISER. ENTER 'EXERCISE MODE'. GO IS ISSUED TO INITIALLY START THE EXERCISER TRANSMITTING TO OTHER RECEIVERS ON THE T.D.M. BUS. ALL TARGET RECEIVER ADDRESS SHOULD HAVE ALREADY BEEN ENTERED VIA THE 'RANGE' OR 'ADD' COMMANDS, AND ONE OF THE PCL11 UNITS SHOULD HAVE BEEN SET TO BE T.D.M. BUS MASTER. IF THE 'SILO' IS BEING USED TO GENERATE TRANSMITTER ADDRESSES, IT SHOULD HAVE BEEN LOADED PRIOR TO 'GO'.

THE 'GO' COMMAND WILL CAUSE THE CLEARING OF THE ERRORS TABLES AND THE SUMMARY TABLE AND THE 'ATTEMPTS' AND 'SUCCESSSES' PORTIONS OF THE STATUS TABLE. IT ALSO CAUSES THE CLEARING OF THE RECEIVER ADDRESS QUEUE (SOFTWARE). NOTE THAT THE RECEIVER ERRORS ACCUMULATED UP TO THE POINT OF TYPING 'GO' ARE LOST. ONLY THOSE ACCUMULATED AFTER 'GO' IS TYPED CAN BE DISPLAYED IN THE ERRORS TABLE. IT WILL NOT, HOWEVER, CAUSE THE CLEARING OF THE RECEIVER ADDRESSES IN THE STATUS TABLE; SO THAT ALL RECEIVERS SELECTED BY THE RANGE OR ADD COMMANDS WILL STILL BE ACTIVE.

THE TRANSMITTER 'EVENT' FLAG IS SET BY THE GO COMMAND WHICH BEGINS THE TRANSMISSION OF DATA TO THE DE-QUEUED TARGET RECEIVER ADDRESSES. WHEN 'GO' IS ISSUED, THE IMMEDIATE RESPONSE SHOULD BE THAT THE EXERCISER PRINTS:

EXERCISER STARTED

HOWEVER, IF THE OPERATOR FORGOT TO ENTER THE TARGET RECEIVER ADDRESSES, AND NONE WERE PREVIOUSLY ENTERED, THE EXERCISER WILL NOT BE ABLY TO RUN AND WILL APOLOGIZE THUS:

'* SORRY, I HAVE NO RECEIVER ADDRESSES *'

3.2.15 CONTINUE

[CO]NTINUE

CONTINUE -

CONTINUE EXERCISING. RE-ENTER EXERCISE MODE. 'CONTINUE' IS ISSUED TO CAUSE THE EXERCISER TO CONTINUE AFTER A TABLE HAS FINISHED BEING PRINTED, OR AFTER SOME OTHER COMMAND HAS BEEN EXECUTED TO POSSIBLY CHANGE THE RUNNING OF THE EXERCISER.

THE 'CONTINUE' COMMAND DOES NOTHING TO ANYTHING EXCEPT THAT IT RE-STARTS EXERCISE MODE. ALL TABLES ARE LEFT INTACT AND SOFTWARE QUEUES ARE UNTOUCHED. WHEN 'CONTINUE' IS ISSUED, THE IMMEDIATE RESPONSE SHOULD BE THAT THE EXERCISER PRINTS:

'EXERCISER CONTINUING'

IF, HOWEVER, WHILE THE EXERCISER WAS STOPPED, THE OPERATOR USED THE INITIALIZE OR CLEAR COMMANDS AND DID NOT RE-ENTER ANY TARGET RECEIVER ADDRESSES (VIA RANGE OR ADD), THE RECEIVER ADDRESS QUEUE WOULD EVENTUALLY BECOME EMPTY AND WOULD NEVER BE RE-FILLED. THIS WOULD AGAIN CAUSE THE PRINTOUT:
** SORRY, I HAVE NO RECEIVER ADDRESSES **

3.2.16 **SYNTAX ERROR**

SYNTAX ERROR IS NOT A COMMAND BUT IS RELATED TO ALL OF THE COMMANDS OF SECTION 3.2. IN GENERAL, A SYNTAX ERROR WILL RESULT FOR THE FOLLOWING REASONS:

- .1 COMMAND DOES NOT EXIST IN 'KEYWORD' TABLE
- .2 COMMAND WORD IS MISSPELLED
- .3 FEWER THAN THE MINIMUM CHARACTERS WERE USED
(I.E. LESS THAN THAT ENCLOSED IN SQUARE BRACKETS [CL]EAR)
- .4 NO ARGUMENTS GIVEN WHERE REQUIRED
- .5 ARGUMENTS GIVEN WHERE NONE REQUIRED
- .6 ARGUMENTS IN WRONG ORDER (WHERE ORDER IS IMPORTANT)
- .7 NOT ENOUGH ARGUMENTS
- .8 TOO MANY ARGUMENTS
- .9 ARGUMENTS ARE WRONG CLASS (USUALLY SHOULD BE NUMBERS)
- .10 ARGUMENTS ARE OUTSIDE SPECIFIC BOUNDARIES (SEE 3.2.13)
- .11 ARGUMENTS SEPARATED BY OTHER THAN A SPACE
- .12 COMMAND SEPARATED FROM ARGUMENTS BY OTHER THAN A SPACE
- .13 DECIMAL NUMERIC ARGUMENTS USED WITHOUT THE POINT (.)
- .14 ARGUMENTS ARE THE SAME (ONLY IN 'RANGE' COMMAND)

4 GETTING THE EXERCISER STARTED

4.1 PREPARATION

BEFORE RUNNING THE PCL11 EXERCISER, THE FOLLOWING MUST HAVE BEEN PREVIOUSLY PERFORMED:

- INSURE ALL PCL11 UNITS ARE CORRECTLY INSTALLED IN EACH PDP-11 PROCESSOR.
- DETERMINE ALL OF THE T.D.M. BUS ADDRESSES OF THE TRANSMITTERS AND THE RECEIVERS OF THE UNITS WHICH IT IS DESIRED TO BE TESTED. INSURE THAT NO TWO TRANSMITTERS AND NO TWO RECEIVERS HAVE BEEN ASSIGNED THE SAME T.D.M. BUS ADDRESSES.
- RUN THE PCL11 'STANDALONE' TEST (CZPLBA0) WITHOUT ERRORS BEFORE CONNECTING ALL UNITS TOGETHER VIA THE T.D.M. BUS.

4.2 LOADING

THE PCL11 EXERCISER IS SUPPLIED IN ABSOLUTE BINARY FORMAT AND IS LOADED BY MEANS OF THE STANDARD PDP-11 ABSOLUTE LOADER OR THE 'XXDP' LOAD COMMAND.

THE EXERCISER IS APPROXIMATELY 7-K LONG. THEREFORE, WHEN USING 'XXDP', THE PROCESSOR MUST HAVE 16-K OF MEMORY.

THE EXERCISER MUST BE LOADED INTO EACH PDP-11 HOSTING A PCL11 WHICH IS TO BE TESTED.

4.3 DEVICE ADDRESSES

IT MAY BE FOUND THAT THE UNIBUS ADDRESSES OF THE PCL11 UNIT ARE DIFFERENT THAN THE DEFAULT ADDRESSES IN THE EXERCISER (SEE 3.2.13). IF THIS IS THE CASE, AND THERE ARE NO OTHER DEVICES ON THE UNIBUS WITH THE ADDRESSES 164200-164226, THE FOLLOWING PRINTOUT WILL OCCUR:
'DEVICE ADDRESS ERROR. USE 'ASSIGN' COMMAND'

ALSO, IT MAY BE THE CASE THAT THERE IS MORE THAN ONE PCL11 UNIT HOSTED BY ONE PDP-11 AND EACH ONE MUST BE TESTED 'ON-LINE' USING THE EXERCISER.

IN EITHER CASE, THE OPERATOR MUST USE THE 'ASSIGN' COMMAND AS SHOWN IN SECTION 3.2.13 TO ASSIGN THE CORRECT UNIBUS ADDRESSES TO THE EXERCISER IN ORDER TO EXERCISE THE RIGHT PCL11.

NOTE THAT THE EXERCISER WILL RUN WITH ONLY ONE PCL11 PER PDP-11 AT A TIME.

EX
--

CH
--
CN
CN
CN
CN
CN
CN
RU
CA
LI

4.4 STARTING ADDRESSES

THE EXERCISER STARTING ADDRESS IS 200
THIS WILL INITIALIZE EVERYTHING AND START IN COMMAND INPUT MODE.

THE EXERCISER IS RESTARTED AT LOCATION 204
THIS WILL PRESERVE THE ERROR TABLES, STATUS TABLE, AND THE
RECEIVER ADDRESS QUEUE. IT WILL NOT, HOWEVER, PRESERVE THE
STATE OF THE TRANSMITTER ADDRESS SILO (HARDWARE CLEARED BY 'RESET')
NOR THE STATE OF 'MASTER' OR 'SECONDARY'. THESE UNPRESERVED
STATES MUST BE RE-ESTABLISHED PRIOR TO STARTING THE EXERCISE MODE
WITH THE 'GO' COMMAND.

START = 200
RESTART = 204

4.5 OPERATING PROCEDURES

-
- A) LOAD THE PROGRAM USING THE PDP-11 ABS LOADER OR THE LOAD
COMMAND OF 'XXDP'. (SEE 4.2)
 - B) LOAD ADDRESS 200; PRESS START. THE TEST WILL IDENTIFY
ITSELF AND TEST THE DEVICE ADDRESSES.
 - C) IF THE PCL11 UNIBUS ADDRESSES OF THE UNIT TO BE TESTED
ARE NON-STANDARD, USE THE 'ASSIGN' COMMAND AS IN 3.2.13
 - D) DO A) TO C) (ABOVE) ON ALL PDP-11S BEFORE CONTINUING.
 - E) ASSIGN ONE OF THE UNITS AS MASTER EITHER BY USING THE
'MASTER SET' COMMAND, (SEC. 3.2.2) OR BY LOADING THE
XMTR ADDRESS SILO (SEC. 3.2.1) ON THE SELECTED UNIT.
 - F) AT EACH UNIT, DECIDE WHETHER IT IS DESIRED TO RUN THE
TRANSMITTER IN THE 'RE-TRY - IF - BUSY' MODE AND SET OR
CLEAR 'RIB' ACCORDINGLY (SEC. 3.2.4).
 - G) AT EACH UNIT, ENTER THE RECEIVER ADDRESSES OF ALL THE
RECEIVERS THAT THIS TRANSMITTER IS TO COMMUNICATE TO,
INCLUDING THE ADDRESS OF ITS OWN RECEIVER. (3.2.6 OR 3.2.5)
 - H) ON EACH UNIT, TYPE 'GO' TO START THE EXERCISER(S).
 - I) PERIODICALLY, ON EACH UNIT, TYPE CNTRL-C, THEN ISSUE THE
'STATUS' COMMAND TO INSURE THAT ALL RECEIVERS ARE BEING
TALKED TO AND THAT THE CORRECT UNIT IS MASTER.
 - J) ALSO PERIODICALLY, ON EACH UNIT, ISSUE THE 'SUMMARY'
COMMAND TO DISCOVER IF THERE HAVE BEEN ANY ERRORS.
 - K) AT ANY TIME, THE 'ERRORS' COMMAND MAY BE ISSUED TO
DETERMINE WHICH ERRORS HAVE OCCURRED BETWEEN WHICH
RECEIVER AND TRANSMITTER CONNECTION.
 - L) TO RESUME EXERCISING AS BEFORE ON ANY UNIT, TYPE
'CONTINUE' ON THE UNITS WHICH HAD BEEN STOPPED BY

CNTRL-C. (OR MASTER DOWN).

N 2

SEQ 0026

ER
--

4.6 ERRORS

A LIST OF ERROR NUMBERS MAY BE FOUND IN APPENDIX C OF THIS DOCUMENT. THESE 'ERROR NUMBERS' ARE THOSE REFERRED TO IN THE SUMMARY TABLE AND IN THE ERRORS TABLES. A LITTLE MORE DETAIL MAY BE DETERMINED ABOUT THE ERROR BY REFERRING TO THE PROGRAM LISTING AROUND THE ADDRESS SHOWN IN THE SUMMARY TABLE. THE LISTING WILL HAVE, IN THE COMMENT FIELD, THE ERROR IDENTIFIER:

**** XMTR ERROR N **** OR:

**** RCVR ERROR N ****

WHERE 'N' IS THE ERROR NUMBER. ABOVE THIS IDENTIFIER, WILL BE THE DESCRIPTION OR CAUSE OF THE ERROR WITH THAT NUMBER.

IT MAY BE NOTED THAT THE TRANSMITTER ERRORS ARE NUMBERED FROM 1 TO 15 (OCTAL), AND THE RECEIVER ERRORS ARE NUMBERED FROM 16 TO 30 (OCTAL). THIS IS DONE SO THAT ONLY ONE TABLE IS REQUIRED TO SUMMARIZE ALL THE ERRORS (SUMMARY TABLE).

IF THERE IS AN ALARMING NUMBER OF OCCURRENCES OF ERRORS, THE FOLLOWING STEPS SHOULD BE TAKEN:

- A) DETERMINE THE ERROR CAUSE (FROM APPENDIX C)
- B) DETERMINE WHICH TRANSMITTER AND/OR RECEIVER ARE SUSPECT (FROM THE ERRORS TABLES)
- C) IF THE ERROR WAS NOT 'MASTER DOWN' (ERROR 10), RUN THE PCL11 'STANDALONE TEST' (YC-2017D-08) ON THE UNITS WITH THE SUSPECTED RECEIVER OR TRANSMITTER. SEE IF THE SAME ERROR TYPE CAN BE ACHIEVED WITH THE 'STANDALONE TEST'. IF NOT, THEN THE T.D.M. DRIVERS, OR CABLES, OR TERMINATORS ETC. ARE SUSPECT.
- D) IF THE ERROR WAS 'ERROR 10' BUT THIS ERROR HAD NOT OCCURRED ON OTHER UNITS, THE CABLE, OR RECEIVER CHIPS ETC. ARE AGAIN SUSPECT.
- E) USING EITHER THE EXERCISER, OR THE STANDALONE TEST, A DEFECTIVE SINGLE MODULE SHOULD BE RELATIVELY SIMPLE TO LOCALIZE AND REPLACE, CORRECTING THE PROBLEM.
- F) ONCE A MODULE HAS BEEN REPLACED, ALWAYS RUN THE PCL11 STANDALONE TEST (EVEN IF IT WAS A LINE DRIVER MODULE) BEFORE RUNNING THE EXERCISER.

A SMALL NUMBER OF CERTAIN ERRORS IS ACCEPTABLE DURING A LONG EXERCISE RUN. THESE ERRORS WOULD BE ATTRIBUTED TO LINE NOISE, GENERAL SYSTEM NOISE ETC. THESE ERRORS ARE:

ERROR 6 TRANSMITTER CRC ERROR
ERROR 7 TRANSMITTER MISCELLANEOUS TXM ERROR

ERROR 22 RECEIVER CRC ERROR
ERROR 24 RECEIVER PARITY ERROR

5 COMMAND MODE DESCRIPTION

5.1 SHORTENED COMMANDS

ANY OF THE COMMANDS MAY BE TYPED IN AS SHORT A FORM AS WOULD SEEM REASONABLE. THAT IS, ONLY ENOUGH LETTERS NEED BE TYPED SO AS TO DISTINGUISH ONE COMMAND FROM ANOTHER WITH THE SAME FIRST LETTER.

FOR EXAMPLE, SINCE THERE IS ONLY ONE COMMAND BEGINNING WITH THE LETTER 'E' (ERRORS), ONLY THE 'E' NEED BE TYPED FOR THAT COMMAND. HOWEVER, THERE ARE FOUR COMMANDS BEGINNING WITH THE LETTER 'S' (SUMMARY, STATUS, SILO, AND SECONDARY). IN EACH OF THESE COMMANDS, THE SECOND LETTER IS DIFFERENT, SO JUST TWO LETTERS NEED BE TYPED: (SU, ST, SI, AND SE).

ON THE OTHER HAND, THE COMMAND DECODER WILL NOT ACCEPT ANY COMMAND WORDS WITH ANY OF THE LETTERS WRONG. THAT IS, EVERY LETTER TYPED IN FOR A PARTICULAR COMMAND MUST BE AT LEAST ON THE WAY TO SPELLING THE WORD CORRECTLY.

FOR EXAMPLE: FOR THE 'INITIALIZE' COMMAND:

INITIAL IS ACCEPTABLE, WHEREAS:
INITL IS UNACCEPTABLE.

5.2 RUBOUT FEATURES

THERE ARE TWO EDITING FEATURES EMPLOYED IN THE COMMAND DECODER. 'RUBOUT' (DELETE) CHARACTER WILL DELETE THE LAST CHARACTER WHICH WAS TYPED IN AS PART OF A COMMAND WORD OR ARGUMENT. CONTROL-U CHARACTER WILL REMOVE ALL THAT HAS BEEN TYPED IN SO FAR ON THIS LINE.

ONE OTHER METHOD OF HAVING THE EXERCISER IGNORE EVERYTHING TYPED IN SO FAR IS TO TYPE 'CONTROL-C'.

- A) RUBOUT DELETE LAST CHARACTER
- B) CNTRL-U DELETE THIS LINE
- C) CNTRL-C DELETE THIS LINE

5.3 ENTERING COMMAND MODE

COMMAND MODE IS AUTOMATICALLY ENTERED AT STARTUP OR RESTART OF THE EXERCISER. THERE ARE TIMES, HOWEVER, WHEN IT IS NOT IN 'COMMAND' MODE. THEY ARE:

- A) WHEN IN EXERCISE MODE (RUNNING)
- B) WHILE PRINTING THE STATUS, SUMMARY, OR ERRORS TABLES

AT ANY TIME THAT IT IS DESIRED TO ENTER COMMAND MODE, THE OPERATOR NEED ONLY TYPE 'CONTROL-C' (^C). THIS WILL TERMINATE ALL TRANSMITTER ACTIVITY ON THE UNIT AND ENTER COMMAND MODE. IT WILL ALSO, (AT COMPLETION OF THE CURRENT LINE), TERMINATE ALL TABLE PRINTING AND RETURN TO COMMAND MODE.

THERE IS ANOTHER CHARACTER WHICH WILL PERFORM THE SAME AS CONTROL-C DUE TO ITS FUNCTION; THAT IS CONTROL-U.

5.4 UPPER OR LOWER CASE

WHEN IN COMMAND MODE, THE OPERATOR MAY FIND HIMSELF USING A KEYBOARD WHICH DOES NOT HAVE A 'CAPS LOCK' KEY. SINCE THE COMMAND DECODER REQUIRES THAT ALL INPUT BE IN CAPITAL LETTERS, THE KEYBOARD INPUT ROUTINE WILL AUTOMATICALLY CONVERT ALL LOWER CASE ALPHA CHARACTERS INTO UPPER CASE ALPHA CHARACTERS BY CLEARING BIT05 IN THE ASCII CODE OF THE INPUT CHARACTER.

6 EXERCISER MODE DESCRIPTION

6.1 TRANSMIT EVENT

AN 'EVENT FLAG' IS CHECKED IN THE MAIN LOOP OF THE PROGRAM TO DETERMINE WHETHER THE EXERCISER HAS BEEN TOLD TO 'GO'. THIS FLAG IS THE TRANSMIT EVENT FLAG. IT IS SET WHENEVER THE OPERATOR ISSUES THE 'GO' COMMAND, OR THE 'CONTINUE' COMMAND TO THE EXERCISER. THIS FLAG IS CLEARED WHENEVER THE OPERATOR TYPES CONTROL-C OR IF A MASTER DOWN ERROR OCCURS.

WHEN THE FLAG IS DETECTED AS BEING SET, THE EXERCISER CALLS THE TRANSMITTER MODULE WHICH TRANSMITS A BLOCK OF DATA THAT HAD BEEN PREVIOUSLY GENERATED BY THE DATA GENERATION MODULE. IF THIS DATA HAS ALREADY BEEN USED FOR THE FIFTH TIME, IT SETS THE DATA GENERATION EVENT FLAG AND NEW RANDOM DATA WILL BE GENERATED. WHEN THE TRANSMIT MODULE IS CALLED, THE TRANSMIT EVENT FLAG IS CLEARED AND IS NOT SET AGAIN UNTIL SOME TYPE OF 'COMPLETION' INTERRUPT HAS OCCURRED SUCH AS 'SUCCESSFUL TRANSFER' OR 'ERROR'.

THE TRANSMIT EVENT IS ALSO RESPONSIBLE FOR UPDATING THE ERROR TABLES FOR TRANSMITTER ERRORS, AND ALSO THE STATUS TABLE FOR ATTEMPTS AND SUCCESSES TO EACH RECEIVER IN THE RECEIVER ADDRESS QUEUE.

6.2 RECEIVE EVENT

WHEN THE EXERCISER IS STARTED BY THE OPERATOR STARTING AT LOCATION 200, A SOFTWARE FLAG CALLED 'RECEIVER EVENT FLAG' IS SET. WHEN THIS IS DETECTED IN THE MAIN LOOP, THE RECEIVER MODULE IS CALLED TO SET UP THE RECEIVER TO RECEIVE UP TO 600 (OCTAL) WORDS FROM A TRANSMITTER THAT TRIES. WHEN THE MODULE IS CALLED, THE FLAG (RCVR EVENT) IS CLEARED AND NOT SET AGAIN UNTIL SOME TYPE OF COMPLETION INTERRUPT IS RECEIVED SUCH AS 'SUCCESSFUL TRANSFER', 'ERROR', OR 'REJECT COMPLETED'.

THE RECEIVE EVENT IS ALSO RESPONSIBLE FOR UPDATING THE ERROR TABLES FOR RECEIVER ERRORS, AND FOR CHECKING THE DATA RECEIVED TO DETERMINE ITS CORRECTNESS AND THAT THE RIGHT NUMBER OF WORDS WERE RECEIVED.

UNLIKE THE TRANSMIT EVENT, THE RECEIVE EVENT CANNOT BE SUSPENDED BY THE OPERATOR ISSUING ANY COMMANDS OR CONTROL CHARACTERS. HALTING THE EXERCISER, OR A HARDWARE FAILURE TO INTERRUPT ARE THE ONLY WAYS TO PREVENT THE RECEIVE EVENT FROM OCCURRING.

6.3 DATA GENERATION EVENT

ANOTHER EVENT WHICH OCCURS IN EXERCISE MODE IS 'DATA GENERATION'. A NEW BUFFER FULL OF RANDOM DATA IS GENERATED AFTER 5 PASSES WITH THE OLD DATA ARE COMPLETED. THE LENGTH OF THE DATA BUFFER ALSO RANDOMLY VARIES FROM 1 TO 1000 (OCTAL) WORDS. IF THE BUFFER IS LONGER THAN 600 WORDS, THE RECEIVER WILL BE EXPECTED TO TRUNCATE THE MESSAGE AFTER RECEIVING THE 600TH WORD.

ALSO, IF THE FIRST WORD OF THE BUFFER ('FLAGS' WORD) HAS THE FOUR MOST SIGNIFICANT BITS SET, THE RECEIVER WILL REJECT THE MESSAGE ENTIRELY. ALL OF THESE 'REJECT' AND 'TRUNCATE' OCCURRENCES ARE EXPECTED BY THE TRANSMITTER EVENT AND CHECKS ARE MADE THAT THEY OCCUR AS THEY SHOULD.

6.4 ADDRESS QUEUE EVENT

WHEN THE USER HAS COMPLETED GENERATING THE LIST OF RECEIVER ADDRESSES HE WISHES A PARTICULAR TRANSMITTER TO COMMUNICATE WITH, THE STATUS TABLE HAS THOSE ENTRIES 'ACTIVATED'. DURING EXERCISE MODE ACTIVE ADDRESSES IN THE STATUS TABLE ARE LOADED INTO A SOFTWARE QUEUE TO AWAIT THEIR TURN WITH THE TRANSMIT MODULE. ADDRESSES ARE DEQUEUED ON EVERY ENTRANCE TO THE TRANSMIT MODULE AND THE 'ATTEMPTS' ENTRY OF THE STATUS TABLE IS UPDATED. WHEN THE QUEUE IS EMPTY, THE ADDRESS QUEUE EVENT IS CALLED TO RE-FILL IT FROM THE STATUS TABLE.

NOTE THAT ADDRESSES ARE ALWAYS QUEUED AND DEQUEUED IN NUMERICAL ORDER REGARDLESS OF THE ORDER IN WHICH THEY WERE ENTERED.

6.5 ERRORS UPDATE EVENT

AT ANY TIME, WHETHER THE EXERCISER IS IN 'EXERCISE' MODE OR COMMAND MODE, THE RECEIVE EVENT IS ACTIVE. THEREFORE, RECEIVER ERRORS CAN OCCUR AT ANY TIME. HOWEVER, TRANSMITTER ERRORS CAN ONLY OCCUR WHEN THE EXERCISER IS 'GOING'. WHEN AN ERROR OCCURS, OF THE TRANSMITTER TYPE, THE TRANSMITTER COMMAND REGISTER (TCR) IS READ TO GET THE ADDRESS OF THE CONNECTED RECEIVER. THIS IS USED TO DECIDE WHICH TRANSMITTER ERROR TABLE LOCATION TO INCREMENT ALONG WITH THE ERROR NUMBER. THIS UPDATE IS ACCOMPLISHED AT THE ACTUAL TIME THAT THE ERROR IS DISCOVERED. WHEN AN ERROR OF THE RECEIVE TYPE OCCURS THE RECEIVER COMMAND REGISTER (RCR) IS READ TO GET THE ADDRESS OF THE CONNECTED TRANSMITTER. AGAIN, THIS IS USED TO DETERMINE THE CORRECT TABLE LOCATION TO INCREMENT.

AGAIN, PLEASE NOTE THAT THE 'GO' COMMAND CLEARS BOTH THE TRANSMITTER ERROR TABLES AND THE RECEIVER ERROR TABLES.

ANOTHER FUNCTION OF THE ERRORS EVENT IS TO UPDATE THE 'SUMMARY' TABLE ACCORDING ONLY TO THE ERROR NUMBER.

THERE ARE, THEN, ACTUALLY TWO ERROR EVENTS: ONE FOR TRANSMITTER ERRORS, AND ONE FOR RECEIVER ERRORS.

6.6 SPECIAL EVENTS

DURING AN EXERCISE RUN, THERE ARE TWO INTERRUPTS THAT CAN CAUSE PRINTOUTS ON THE CONSOLE. ONE OF THEM IS MASTER GOING DOWN. THIS WILL CAUSE THE CALLING OF A 'SPECIAL' EVENT TO PRINT THE MESSAGE:

*** MASTER DOWN ***

AND RETURN THE EXERCISER TO COMMAND MODE. THE OTHER INTERRUPT IS CAUSED BY THIS UNIT BECOMING 'MASTER' AFTER HAVING BEEN 'SECONDARY'. THIS WILL CAUSE THE CALLING OF ANOTHER 'SPECIAL' EVENT TO PRINT THE MESSAGE:

** THIS UNIT HAS BECOME 'NEW MASTER' **

AND LEAVE THE EXERCISER IN 'EXERCISE' MODE.

APPENDIX A, B, C, D

OVERALL FLOW

[S]---START

I
LOAD & TRY
DEVICE ADDRESSES

I
INITIALIZE

I
RESTART

[A]---MAIN LOOP

I
TIQ NOT
EMPTY

<PROCHR>
PROCESS
CHARACTER

PRCTLC
PRDEL
PRCTLS
PRCTLQ
PRCR-LF
PRCTLU
PRCTLO

[A]

I
TOQ NOT
EMPTY

DEQUEUE TOO
TO TTY IF
READY

[A]

I
OLDEV
EVENT SET

LOAD ADDRESS
QUEUE. IF NO-
THING LOADED,
FLAG ERROR, &
DON'T ALLOW
STARTUP

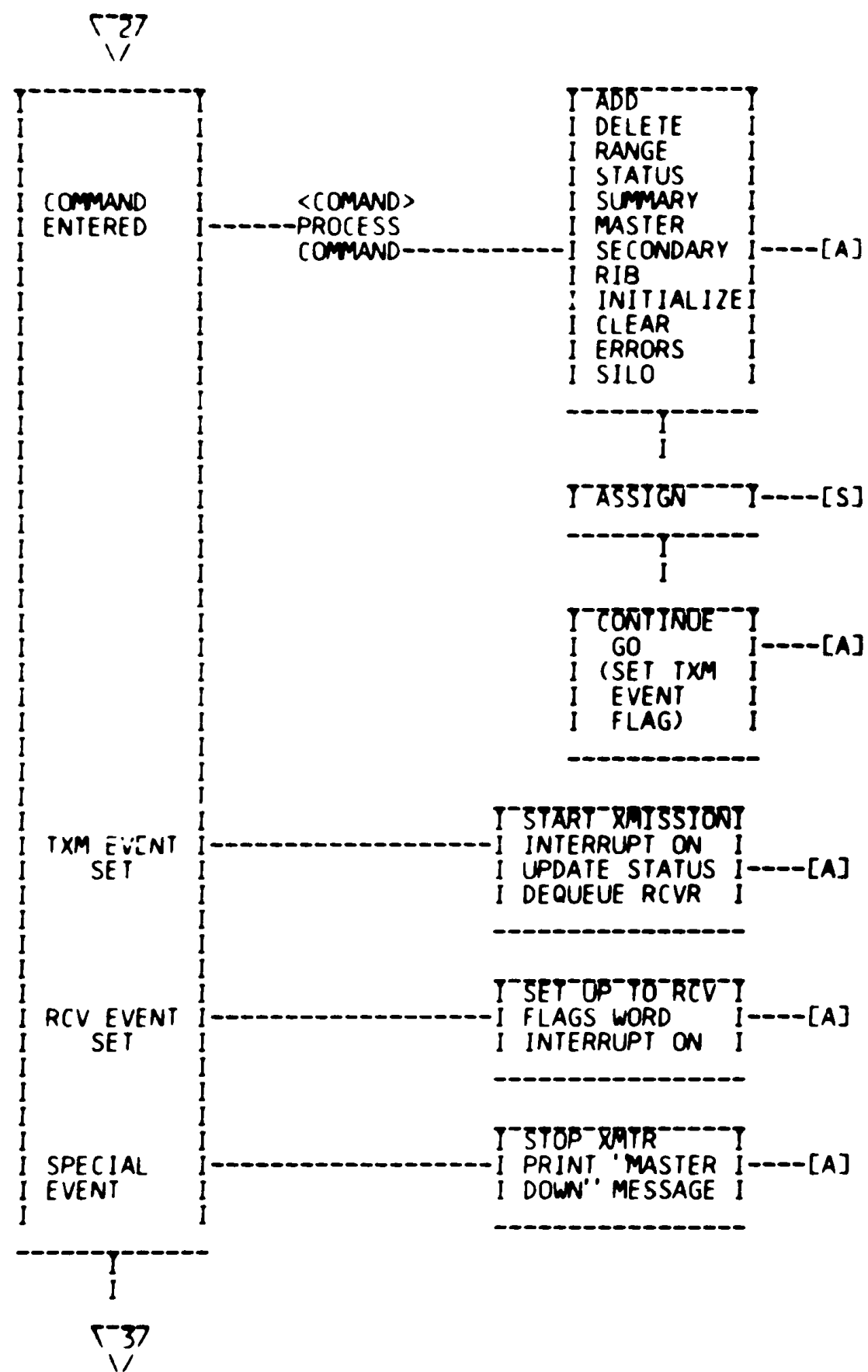
[A]

I
DATGEN
EVENT SET

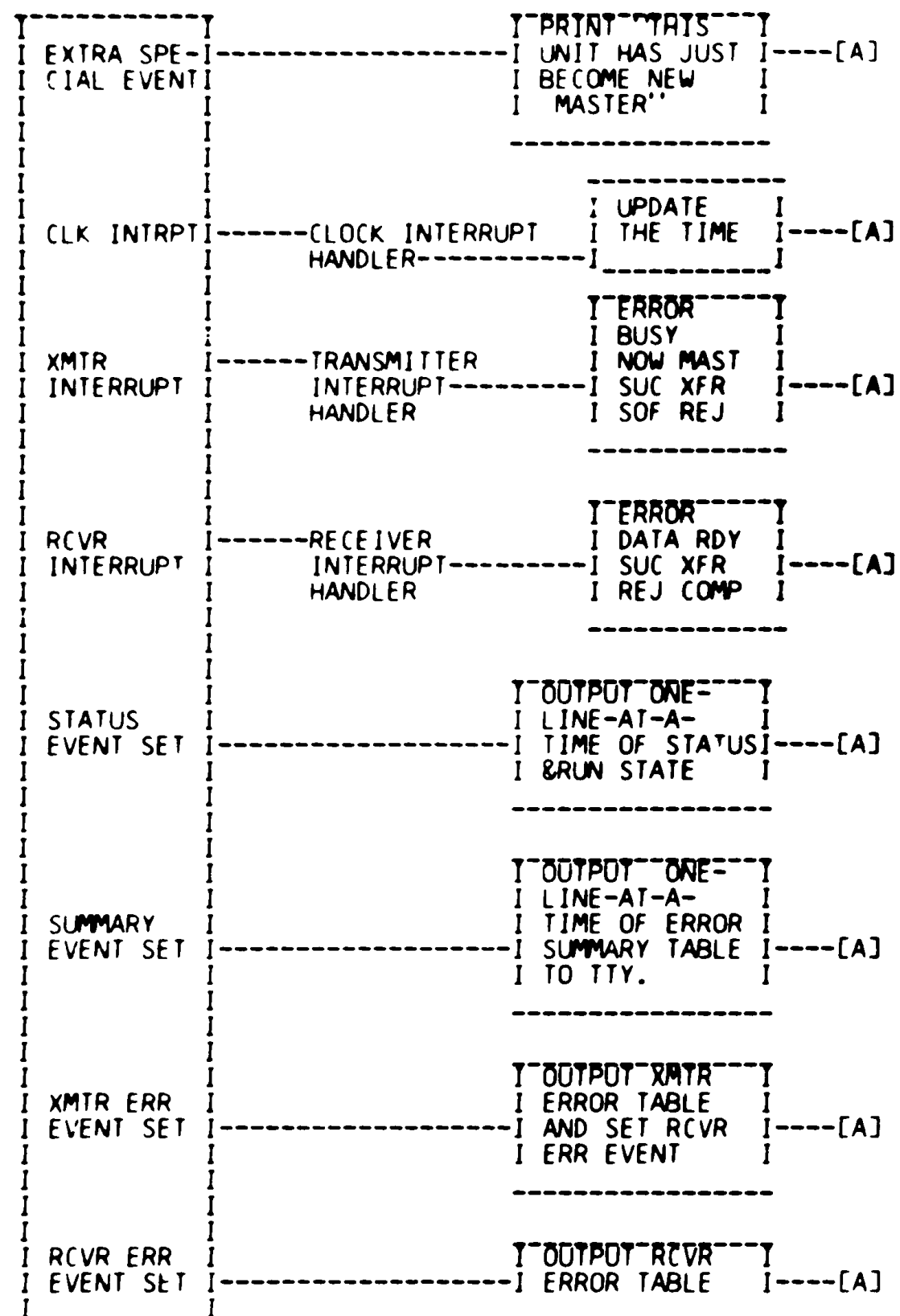
GENERATE NEW
RANDOM DATA
TABLE FOR NEXT
5 CYCLES.

[A]

27
V



37
V



I
[A]

K 3

SEQ 0036

CZ
PC

EXERCISER COMMANDS

CONTROL CHARACTERS

CHARACTER	ECHOES	EFFECT
-----	-----	-----
CNTRL-C	^C	ENTER COMMAND INPUT MODE
CNTRL-O	^O	THROW AWAY TTY OUTPUT
CNTRL-U	^U	DISCARD CURRENT INPUT LINE
CNTRL-S		SUSPEND TTY OUTPUT
CNTRL-Q		RESUME TTY OUTPUT
RUBOUT	\	DELETE LAST INPUT CHARACTER
CAR RET	<CR, LF>	PERFORM COMMAND JUST ENTERED
LINE FEED	<CR, LF>	SAME AS CAR RET

COMMANDS

COMMAND -----	ARGUMENTS -----	MINIMUM -----	EFFECT -----
[AD]D	A B C - N	AD A	ADD ADDRESSES A B C - N
[AS]SIGN	ADR VCT ADR VCT	AS	ASSIGN UNIBUS ADDRESSES AND VECTORS FOR TRANSMITTER AND RECEIVER.
[CL]EAR	-	CL	CLEAR THE STATUS TABLE
[CO]NTINUE	-	CO	CONTINUE EXERCISING
[D]ELETE	A B C - N	D A	DELETE ADDRS A B C - N
[E]RRORS	-	E	PRINT ERRORS TABLES
[G]O	-	G	START THE EXERCISER
[I]NITIALIZE	-	I	INIT THE EXERCISER
[M]ASTER	SET	M S	SET 'MASTER'
[M]ASTER	CLEAR	M C	CLEAR 'MASTER'
[RA]NGE	LOW HI	RA L H	ADD RANGE OF ADDRESSES FROM LOW TO HIGH INCLUSIVE.
[RI]B	SET	RI S	SET 'RIB'
[RI]B	CLEAR	RI C	CLEAR 'RIB'
[SE]CONDARY	SET	SE S	SET 'SECONDARY'
[SE]CONDARY	CLEAR	SE C	CLEAR 'SECONDARY'
[SI]LO	-	SI	CLEAR SILO; SET AUTO ADDR
[SI]LO	A B C - N	SI A	LOAD SILO WITH A B C - N
[ST]ATUS	-	ST	PRINT STATUS TABLE
[SU]MMARY	-	SU	PRINT SUMMARY TABLE

ERROR DESCRIPTIONS

ERROR NUMBER	DESCRIPTION
-----	-----
1	ERRONEOUS INTERRUPT FROM TRANSMITTER
2	NON EXISTANT LOC. ERROR IN XMTR
3	MEM OVERFLOW ERROR IN TRANSMITTER
4	XMTR TXM ERROR: RCVR ACCEPTED A NULL
5	XMTR TXM ERROR: RCVR HAS GONE OFF-LINE
6	XMTR TXM ERROR: WORD OR C.R.C. REJECTED
7	XMTR TXM ERROR: MISCELLANEOUS TXM ERROR
10	MASTER DOWN
11	TRANSMITTER TIMED OUT
12	SILO OVERRUN ERROR IN TRANSMITTER
13	MESSAGE TRUNCATED UNEXPECTEDLY
14	MESSAGE FAILED TO BE TRUNCATED
15	ERRONEOUS REJECT BY RECEIVER
16	UNKNOWN RECEIVER INTERRUPT OCCURRED
17	NON-EXISTANT LOC. ERROR IN XMTR
20	MEM OVERFLOW ERROR IN RECEIVER
21	RCVR TXM ERROR: XMTR HAS GONE OFF-LINE
22	RCVR TXM ERROR: RCVR C.R.C. ERROR
23	RCVR TXM ERROR: FIRST WORD INVALID
24	RECEIVER DETECTED INVALID PARITY.
25	RECEIVER TIMEOUT ERROR OCCURRED
26	RECEIVER GOT TOO MANY WORDS
27	DATA WORD RECEIVED WAS BAD
30	RECEIVER GOT TOO FEW WORDS

D-1

STATISTICAL INFORMATION

STARTING ADDRESS	200
RESTARTING ADDRESS	204
SWITCH OPTIONS	NONE
PROGRAM SIZE	APPROX 7K
MEMORY OCCUPIED	
LOW BOUNDARY	00000
HIGH BOUNDARY	34474

LOCATIONS TO CHANGE FOR
DIFFERENT DEVICE ADDRESSES

DEVICE	CHANGE LOCATION
-----	-----
KEYBOARD STATUS	16136
KEYBOARD DATA	16140
TTY STATUS	16142
TTY DATA	16144
KEYBOARD VECTOR	16146
LINE CLK STATUS	16150
XMTR PRIORITY	2334
RCVR PRIORITY	2346
KBD PRIORITY	2360

USEFUL LOCATIONS	ADDRESS
-----	-----
XMTR DATA BUFFER	20666
RCVR DATA BUFFER	22726
DATA SEED IN TRANSMITTER	17726
RANDOM MULTIPLIER	17746
RANDOM INCREMENT	17750

284	DEFINITIONS AND DEVICE INFO
367	PCL11 EXERCISER MAIN PROCEDURE
489	MAIN LOOP
546	COMMAND PROCESSORS:
547	COMMAND PROC. FOR SILO (LOAD)
644	COMMAND PROC. FOR MASTER, SECONDARY AND R.I.B.
713	COMMAND PROC. FOR RANGE
758	COMMAND PROC. FOR ADD AND DELETE
825	COMMAND PROC. FOR CLEAR, STATUS, AND CONTINUE
894	COMMAND PROC. FOR INIT, SUMMARY, AND GO
1008	COMMAND PROC. FOR 'ASSIGN'
1103	COMMAND DECODER AND PROCESSOR
1199	RECEIVER ADDRESS QUEUE LOADER ROUTINE
1243	DATA GENERATION (RANDOM) ROUTINE
1288	MULTIPLY ROUTINE FOR DATA GENERATION
1322	TRANSMIT MODULE
1534	RECEIVER MODULE
1716	STATUS MODULE
1809	TRANSMITTER ERRORS MODULE
1864	RECEIVER ERRORS MODULE
1915	SUMMARY MODULE
1971	ERROR UPDATE ROUTINES
1972	TRANSMITTER ERRORS
2010	RECEIVER ERRORS
2049	UTILITY ROUTINES
2050	PROCESS AN INPUT CHARACTER FROM THE TTY
2072	TTY INPUT CHARACTER PROCESSING ROUTINES
2207	TTY OUTPUT HANDLERS
2225	TTY INPUT INTERRUPT PROCESSORS
2236	MESSAGE PRINT ROUTINE
2265	DATA AREAS
2472	KEYWORD TABLE
2533	SOME MORE ASCII STORAGE:
2590	AUXILIARY ROUTINES
2591	CHARACTER PROCESSOR
2634	BINARY TO ASCII CONVERSION
2737	GENERAL BINARY TO ASCII CONVERSION
2786	DOUBLE PRECISION BINARY TO ASCII
2873	DOUBLE PRECISION DIVIDE ROUTINE
2909	INTEGER DIVIDE MAGNITUDE NUMBERS
2954	QUEUE HANDLING ROUTINES
3077	COMMAND PROCESSOR INITIATING ROUTINE
3122	KEYWORD PROCESSING ROUTINE
3211	REGISTER SAVE & RESTORE ROUTINES
3238	LEXICAL SCAN ROUTINE

.TITLE CZPLABO PCL11 EXERCISER V02
.IDENT /0003/ ;DRCMAC.MAC 6-JAN-76

; KEYWD MACRO

; THIS MACRO DETERMINES THE ROUTINE ADDRESS
; ASSOCIATED WITH THE SYNTACTIC OBJECT OBJ, ACCORDING
; TO A KEYWORD TABLE POINTED TO BY KWTABL.

; ON RETURN, IF OBJ WAS IN THE TABLE, THEN THE PS
; C BIT = 0 & THE ROUTINE ADDRESS IS AT THE TOP OF THE
; STACK. IF NOT, THEN C=1 & @SP = 0.

; EACH KEYWORD IN THE TABLE HAS ASSOCIATED WITH IT A
; MINIMUM LENGTH SPECIFYING THE MINIMUM NUMBER
; OF CHARACTERS IN THE KEYWORD THAT MUST MATCH
; THOSE IN OBJ FOR A MATCH TO HAVE BEEN DEEMED
; FOUND. IF OBJ CONTAINS MORE THAN THIS MINIMUM
; NO. OF CHARACTERS, THOUGH, ALL CHARACTERS IN OBJ MUST
; CORRESPOND TO THE TABLE KEYWORD FOR A MATCH TO
; HAVE BEEN DEEMED FOUND. THUS, FOR EXAMPLE, IF
; 'REPEAT' APPEARS IN THE KEYWORD TABLE WITH A
; MINIMUM LENGTH OF 3 ASSOCIATED WITH IT, THEN
; 'REP', 'REPE', 'REPEA', & 'REPEAT' WILL ALL MATCH IT, BUT
; 'R', 'RPT', 'REPEET', & 'REPEATER' WILL NOT.

; THE KEYWORD TABLE CONSISTS OF A SET OF ENTRIES AS FOLLOWS:

OFFSET	TYPE	ROUTINE ADDRESS
0	WORD	ROUTINE ADDRESS
2	BYTE	MINIMUM LENGTH OF KEYWORD
3	BYTE	FULL LENGTH OF KEYWORD
4	STRING	KEYWORD

; ENTRIES ARE STORED CONSECUTIVELY IN THE TABLE. EACH ENTRY
; MUST BEGIN ON A WORD BOUNDARY. THE KEYWORD OF
; THE PREVIOUS ENTRY MAY HAVE TO HAVE A BYTE (CONTAINING
; ANYTHING) APPENDED TO IT TO ACCOMPLISH THIS.
; ENTRIES MUST BE ARRANGED IN ALPHABETICAL
; ORDER (MORE SPECIFICALLY, IN ASCII COLLATING SEQUENCE).
; THE TABLE IS ENDED BY AN ENTRY WITH A FULL LENGTH
; OF 0.

```

; .MACRO KEYWD OBJ,KWTABL
; .IF NB OBJ
;   MOV OBJ,-(SP)
;   .IF B KWTABL
;     .ERROR ;CANNOT SPECIFY OBJ & NOT TABLE ADDRESS.
;     MOV #1,-(SP) ;IF RUN, THIS WILL CAUSE TRAP.
;   .ENDC
; .ENDC
; .IF NB KWTABL
;   MOV KWTABL,-(SP)
; .ENDC
; JSR PC,KEYWD

```

```
57          .ENDM ;KEYWD
```

```
58          ; -----
```

```
59          ; LXSCAN MACRO
```

```
60          ; THIS MACRO CALLS THE ROUTINE LXSCAN TO CONVERT THE CHARACTER  
61          ; STRING SPECIFIED BY STR INTO ITS CONSTITUENT SYNTACTIC OBJECTS.  
62          ; STR IS AN ASSEMBLER EXPRESSION SPECIFYING THE ADDRESS OF THE  
63          ; BEGINNING OF THE ASCII STRING TO BE CONVERTED. THE STRING  
64          ; MUST BE ENDED BY A CARRIAGE RETURN CODE.
```

```
65          ;  
66          ; .MACRO LXSCAN STR  
67          ; MOV STR,R1  
68          ; JSR R1,LXSCAN
```

```
69          ; .ENDM ;LXSCAN
```

```
70          ; -----
```

```
71          ; PROC MACRO
```

```
72          ; THIS MACRO DEFINES THE ENTRY POINT FOR A PROCEDURE. THE PARAMS  
73          ; SPECIFIED WILL BE GIVEN THEIR CORRESPONDING OFFSETS RELATIVE TO R5.  
74          ; THUS, A PARAMETER PP CAN BE REFERENCED IN THE PROCEDURE BY THE  
75          ; ASSEMBLER EXPRESSION '@PP(R5)'. THE PROCEDURE CAN BE CALLED  
76          ; USING THE CALL MACRO.
```

```
77          ; .MACRO PROC PNAME,PARAMS  
78          ; Zx1 = 0
```

```
79          ; .IRP Zx2 <PARAMS>  
80          ; Zx1 Zx1+2  
81          ; .IRP Zx3 \Zx1
```

```
82          ; .LIST  
83          ; Zx2 Zx3  
84          ; .NLIST
```

```
85          ; .ENDM  
86          ; .ENDM ;Zx2
```

```
87          ; .LIST
```

```
88          PNAME: ;**ENTRY POINT**
```

```
89          ; .NLIST  
90          ; .ENDM ;PROC
```

```
91          ; -----
```

```
92          ; RETURN MACRO
```

```
93          ; THIS MACRO RETURNS FROM A PROCEDURE. IF ANSWR IS SPECIFIED IT  
94          ; WILL BE LOADED INTO R0 BEFORE RETURNING.
```

```
95          ; .MACRO RETURN ANSWR  
96          ; .IF NB ANSWR  
97          ; MOV ANSWR,R0
```

```
98
```

```
99
```

```
100
```

```
101
```

```
102
```

```
103
```

```
104
```

```
105
```

```
106
```

```
107
```

```
108
```

```
109
```

```
110
```

```
111
```

```
112
```



```
113      .ENDC
114      RTS      FC
115      .ENDM      ;RETURN
116
117      ; -----
118
119      :      CALL MACRO
120
121      :      THIS MACRO CALLS A PROCEDURE, SUBR, WITH AN ARGUMENT LIST SPECIFIED
122      :      BY ARGS.  ARGS IS A LIST OF ADDRESSES WHICH WILL BE INCLUDED
123      :      IN THE ASSEMBLED EXPANSION OF THIS MACRO.
124
125      :      THE CALLING SEQUENCE GENERATED IS FORTRAN COMPATIBLE.  R5 IS LEFT
126      :      INTACT THROUGH THE EXECUTION OF THIS MACRO.  OTHER REGISTERS
127      :      ARE DESTROYED.
128      :
129      :      .MACRO  CALL      SUBR,ARGS,?PLIST,?ZXCALL
130      :      JSR      R5,ZXCALL
131      :      PLIST:  BR      ZXCALL
132      :      .IF      NB      <ARGS>
133      :      .WORD      ARGS
134      :      .ENDC
135      :      ZXCALL:  JSR      PC,SUBR
136      :      MOV      (SP)+,R5
137
138      :      .ENDM      ;CALL
139
140      ; -----
141
142      :      MACRO TO MULTIPLY A NUMBER BY A CONSTANT.
143      :      THE NUMBER IN DST IS MULTIPLIED BY THE VALUE OF THE EXPRESSION
144      :      CONST; THE RESULT IS LEFT IN DST.  A TEMPORARY LOCATION MAY
145      :      BE SPECIFIED AT WORK WHICH WILL BE USED IN THE MACRO EXPANSION
146      :      IF NECESSARY.  IF WORK IS NOT SPECIFIED & A TEMPORARY LOCATION
147      :      IS NEEDED, A STACK ELEMENT WILL BE ALLOCATED (& SUBSEQUENTLY
148      :      DEALLOCATED) FOR THE PURPOSE.  THE MACRO GENERATES A SERIES OF
149      :      SHIFT & ADD INSTRUCTIONS IN-LINE TO ACCOMPLISH THE MULTIPLICATION.
150      :
151      :      .MACRO  MULT      CONST,DST,WORK
152      :      ZX1      =      0      ;FLAG: 0==>LEAST SIG 1-BIT NOT TESTED
153      :      ;      YET; 1==>OPPOSITE.
154      :      ZX2      CONST      ;COPY CONSTANT FOR SHIFTING.
155
156      :      .IF      Z      ZX2
157      :      CLR      DST
158      :      .MEXIT
159
160      :      .ENDC
161
162      :      .REPT      16.
163      :      .IF      NZ      ZX2&1      ;IF BIT 0 - 1
164      :      ZX2      ZX2/2      ;SHIFT CONSTANT RIGHT 1 POSITION.
165      :      .IF      Z      ZX1      ;IF LEAST SIG 1-BIT
166      :      .IF      NZ      ZX2      ;IF NOT MOST SIG 1-BIT
167      :      .IF      NB      WORK      ;IF WORK SPECIFIED
168      :      MOV      DST,WORK
169      :      .IFF      ;ELSE IF WORK BLANK
```

```
169      MOV      DST,-(SP)
170      .ENDC      ;END [.IF NB WORK]
171      .IFF      ;ELSE IF MOST SIG 1-BIT
172      .MEXIT
173      .ENDC      ;END [.IF NZ ZX2]
174      ZX1      -      1      ;INDICATE NO LONGER LEAST SIG 1-BIT.
175      .IFF      ;ELSE IF NOT LEAST SIG 1-BIT
176      .IF      NZ      ZX2      ;IF NOT MOST SIG 1-BIT
177      .IF      NB      WORK      ;IF WORK SPECIFIED
178      ADD      DST,WORK
179      .IFF      ;ELSE IF WORK BLANK
180      ADD      DST,@SP
181      .ENDC      ;END [.IF NB WORK]
182      .IFF      ;ELSE IF MOST SIG 1-BIT
183      .IF      NB      WORK      ;IF WORK SPECIFIED
184      ADD      WORK,DST
185      .IFF      ;ELSE IF WORK BLANK
186      ADD      (SP)+,DST
187      .ENDC      ;END [.IF NB WORK]
188      .MEXIT
189      .ENDC      ;END [.IF NZ ZX2]
190      .ENDC      ;END [.IF Z ZX1]
191      .IFF      ;ELSE IF BIT 0 - 0
192      ZX2      =      ZX2/2      ;SHIFT CONSTANT RIGHT 1 POSITION.
193      .ENDC      ;END [.IF NZ ZX2&1]
194      ASL      DST
195      .ENDM      ;END [.REPT 16.] LOOP.
196      .ENDM      ;MULT
197
198      ; -----
199
200      .MACRO      HEDING      NAM,VER,EDIT,PATCH
201      .TITLE      HEDING
202      .IDENT      /VER'EDIT'PATCH/
203      .CSECT      HEDING
204      .GLOBL      HEDING,HEDLEN
205
206      HEDING:      .ASCII      /NAM'VER'-'EDIT'/
207      .IF      B      PATCH
208      .BYTE      40
209      .IFF
210      .ASCII      /PATCH/
211      .ENDC
212      HEDLEN      =      .-HEDING
213      .ENDM      ;HEDING
214
215      ;MULTIPLY MACRO FOR UNSIGNED MULTIPLY ROUTINE
216
217      .MACRO      MULD      A,B
218      MOV      A,-(SP)      ;SAVE A ON STACK
219      MOV      B,R4      ;SAVE B IN R4
220      JSR      R4,MLI      ;PERFORM MULTIPLICATION
221      .WORD      +2
222      MOV      (SP)+,B      ;PUT PRODUCT INTO B
223      .ENDM      ;MULD
224
```

```

225
226 ;BOARD INIT MACRO FOR CLEARING PCL HARDWARE
227 ;BOARD INIT RECEIVER OR TRANSMITTER.
228
229 .MACRO BDINIT DEV
230 .NLIST
231 .IF IDN <DEV>,<XMTR>
232 BIS #B01,@TCR
233 .IFF
234 .IF IDN <DEV>,<RCVR>
235 BIS #B01,@RCR
236 .IFF
237 .ERROR ;BAD ARGUMENT FOR BDINIT
238 .ENDC
239 .ENDC
240 .LIST
241 .ENDM
242
243 000001 N = 1 ;INITIAL ERROR NUMBER
244
245 ;ERROR MACROS
246
247 .MACRO ERROT P
248 ERADR =
249 CALL ERRMOD,<P,ERADR> ;UPDATE ENTRIES FOR ERROR P
250 N - N+1
251 .LIST
252
253 ;***** XMTR ERROR P *****
254
255 .NLIST
256 .ENDM
257
258 .MACRO ERROR P
259 ERADR =
260 CALL ERRMOD,<P,ERADR>
261 N - N+1
262 .LIST
263
264 ;***** RCVR ERROR P *****
265
266 .NLIST
267 .ENDM
268
269
270
271 ;REGISTER SAVE MACRO
272
273 .MACRO REGSAV
274 JSR R5,REGSAV
275 .ENDM
276
277 ;REGISTER RESTORE MACRO
278
279 .MACRO REGRES
280

```

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

MACY11 30A(1052) 20-OCT-78 09:48 PAGE 1-5

I 4

SEQ 0047

281
282

JSR R5,REGRES
.ENDM

CZ
PC

.SBTTL DEFINITIONS AND DEVICE INFO

.IDENT '02'

; COPYRIGHT AUGUST, 1975
; COMPUTER SPECIAL SYSTEMS,
; DIGITAL EQUIPMENT OF CANADA LTD.

; VARIABLE SYMBOL DEFINITIONS.

; DEVICE DEFAULT INFORMATION.

TTDEV	=	177560	; ADDR OF RCSR FOR TTY.
TTVCTR	=	60	; INPUT VECTOR ADDR FOR TTY.
TTPRIO	=	4	; PRIORITY LEVEL FOR TTY.
PCLTXM	=	164200	
PCLRCV	=	164220	
RCVECT	=	174	
TXVECT	=	170	
TXPRIO	=	5	
RCPRIO	=	5	

; QUEUE SIZES.

TISIZE	=	20.	; TIQ SIZE.
TOSIZE	=	256.	; TOQ SIZE.
AOSIZE	=	32.	; ADR QUEUE SIZE

QLEMS	=	0	; #ELEMENTS PRESENTLY IN QUEUE.
QSIZE	=	2	; #ELEMENTS IN QUEUE SPACE.
QTOP	=	4	; ADDR OF 1ST WORD OF QUEUE SPACE
QBOT	=	6	; =QTOP+(QSIZE*2)
QFRONT	=	10	; ADDR OF FRONT ELEMENT OF QUEUE.
QBACK	=	12	; ADDR OF BACK ELEMENT OF QUEUE.

; REGISTER DEFINITIONS.

R0	=	%0	
R1	=	%1	
R2	=	%2	
R3	=	%3	
R4	=	%4	
R5	=	%5	
SP	=	%6	
PC	=	%7	
PS	=	177776	
SR	=	177570	; SWITCH REGISTER.

; SPECIAL CHARACTER DEFINITIONS:

LF.	=	12	; CR OR LF TO END LINE.
CR.	=	15	
CTL.O	=	17	; ^O TO THROW AWAY TTY OUTPUT.
CTL.U	=	25	; ^U TO DELETE LINE.
CTL.C	=	3	; CNTRL-C TO START INPUT. (^C)
CTL.Q	=	21	; CNTRL-Q TO RESUME PRINTOUT
CTL.S	=	23	; CNTRL-S TO SUSPEND PRINTOUT
RUBOUT	=	177	; RUB OUT TO DELETE CHARACTER.

340			
341			
342	100000	:	DEVICE BIT DEFINITIONS.
343	004000	ERR	= 100000
344	000200	BUSY	= 4000
345	000100	DONE	= 200
		INTENB	= 100

```
347 ;BIT DEFINITIONS:
348
349 100000 B15 = 100000
350 040000 B14 = 40000
351 020000 B13 = 20000
352 010000 B12 = 10000
353 004000 B11 = 4000
354 002000 B10 = 2000
355 001000 B09 = 1000
356 000400 B08 = 400
357 000200 B07 = 200
358 000100 B06 = 100
359 000040 B05 = 40
360 000020 B04 = 20
361 000010 B03 = 10
362 000004 B02 = 4
363 000002 B01 = 2
364 000001 B00 = 1
365
```

```
367 .SBTTL PCL11 EXERCISER MAIN PROCEDURE
368
369 .ENABL ABS
370 = 0
371 .REPT 128.
372 .WORD .+2,0 ;TRAP CATCHERS
373
374 .ENDR
375
376 = 200
377 000200 000167 001574 . JMP PCLEX ;PROGRAM STARTS AT 200
378 000204 012706 002000 MOV #STKTOP,SP ;AND RESTARTS AT 204
379 000210 000167 002354 JMP PCREST
380
381 = 2000
382 002000 RELZRO =
383 002000 STKTOP =
384
385 002000 PCLEX: ;*****START HERE*****
386 002000 012706 002000 MOV #STKTOP,SP ;SET STACK POINTER TO TOP
387 002004 016700 014156 MOV TXDEV,R0 ;PREPARE TO GENERATE TXM ADDRS
388 002010 010067 014062 MOV R0,TCR ;GENERATE TCR ADDRESS
389 002014 062700 000002 ADD #2,R0
390 002020 010067 014054 MOV R0,TSR ;GENERATE TSR ADDRESS
391 002024 062700 000002 ADD #2,R0
392 002030 010067 014046 MOV R0,TSDB ;GENERATE TSDB ADDRESS
393 002034 062700 000002 ADD #2,R0
394 002040 010067 014040 MOV R0,TSBC ;GENERATE TSBC ADDRESS
395 002044 062700 000002 ADD #2,R0
396 002050 010067 014032 MOV R0,TSBA ;GENERATE TSBA ADDRESS
397 002054 062700 000002 ADD #2,R0
398 002060 010067 014024 MOV R0,TMMR ;GENERATE TMMR ADDRESS
399 002064 005200 INC R0
400 002066 010067 014020 MOV R0,TMMRH ;AND TMMR HIGH BYTE
401 002072 005200 INC R0
402 002074 010067 014014 MOV R0,TSCRC ;GENERATE TSCRC ADDRESS
403 002100 016767 014064 014024 MOV TXVEC,TXMVEC ;GENERATE TXMVEC ADDRESS
404
405 002106 016700 014060 MOV RCDEV,R0 ;PREPARE TO GENERATE RCVR ADDRS
406 002112 010067 014000 MOV R0,RCR ;GENERATE RCR ADDRESS
407 002116 062700 000002 ADD #2,R0
408 002122 010067 013772 MOV R0,RSR ;GENERATE RSR ADDRESS
409 002126 062700 000002 ADD #2,R0
410 002132 010067 013764 MOV R0,RDDB ;GENERATE RDDB ADDRESS
411 002136 062700 000002 ADD #2,R0
412 002142 010067 013756 MOV R0,RDBC ;GENERATE RDBC ADDRESS
413 002146 062700 000002 ADD #2,R0
414 002152 010067 013750 MOV R0,RDBA ;GENERATE RDBA ADDRESS
415 002156 062700 000004 ADD #4,R0
416 002162 010067 013742 MOV R0,RDCRC ;GENERATE RDCRC ADDRESS
417 002166 016767 014002 013740 MOV RCVEC,RCVVEC ;GENERATE RCVVEC ADDR
```



```
419 002174          PCRST:          ;*****RESTART HERE *****
420 002174 012706 002000          ;RESET STACK POINTER
421 002200 012737 003340 000004          ;SET UP VECTOR FOR ADDRESS ERROR
422 002206 012737 000340 000006          ;*****
423 002214 105067 015420          ;CLR INPUT REQUEST
424 002220 105067 015415          ;CLR COMMAND ENTERED FLAG
425 002224 005067 015450          ;CLR XMTR GO FLAG
426 002230 012767 000204 015524          ;SAVE RESTART ADDRESS
427          .IRP                  ;INITIALIZE IO QUEUES TO EMPT
428          .LIST                  ;*****
429          CLR LC'Q
430          MOV LC'Q+QTOP,LC'Q+QFRONT
431          MOV LC'Q+QTOP,LC'Q+QBACK
432          .NLIST
433          .ENDM
(1) 002236 005067 014276          CLR TIQ
(1) 002242 016767 014276 014300          MOV TIQ+QTOP,TIQ+QFRONT
(1) 002250 016767 014270 014274          MOV TIQ+QTOP,TIQ+QBACK
(1) 002256 005067 014342          CLR TOQ
(1) 002262 016767 014342 014344          MOV TOQ+QTOP,TOQ+QFRONT
(1) 002270 016767 014334 014340          MOV TOQ+QTOP,TOQ+QBACK
434 002276 012737 000340 177776          MOV #340,@#PS
435 002304 012777 007600 013620          MOV #XMTINT,@TXMVEC
436 002312 012777 011124 013614          MOV #RCVINT,@RCVVEC
437 002320 012777 015642 013620          MOV #TTIINT,@TTVECT
438 002326 016700 013600          MOV TXMVEC,R0
439 002332 012760 000240 000002          MOV #TXPRI0*32.,2(R0)
440 002340 016700 013570          MOV RCVVEC,R0
441 002344 012760 000240 000002          MOV #RCPRI0*32.,2(R0)
442 002352 016700 013570          MOV TTVECT,R0
443 002356 012760 000200 000002          MOV #TTPRI0*32.,2(R0)
444 002364          CALL PNCRLF
445 002400          CALL PNCRLF
446 002414          CALL PNTLIN,<PCLEXM>
447 002432          CALL PRINIT
448 002446 012700 031023          PRST: MOV #RSADD,R0
449 002452 016701 015304          MOV RSHOLD,R1
450 002456 012702 177777          MOV #-1,R2
451 002462 004767 030214          JSR PC,OCTJSP
452 002466          CALL PNTLIN,<RSTMSG>
453 002504 012737 002550 000004          MOV #STR,@#4
454 002512 005067 015132          CLR KWFLG
455 002516 005777 013426          TST @LCS
456 002522 012767 177777 015120          MOV #-1,KWFLG
457 002530 012737 012230 000100          MOV #CLKINT,@#100
458 002536 012737 000340 000102          MOV #340,@#102
459 002544 000167 000020          JMP PCREST
460 002550 022626          STR:  CMP (SP)+,(SP)+
461 002552 005067 015072          CLR KWFLG
462 002556 012737 000102 000100          MOV #102,@#100
463 002564 005037 000102          CLR @#102
464 002570 005037 177776          PCREST: CLR @#PS
465 002574 012737 002702 000004          MOV #TRAP4,@#4
466 002602 052767 100000 015072          BIS #B15,RCVEV
467 002610 052777 000100 013320          BIS #B06,@TTRCSR
468 002616 005067 015134          CLR ESCFLG
```

469	002622	005067	015030		CLR	QLDEV	;CLR QUEUE LOAD EVENT FLAG
470	002626	005067	015106		CLR	RJCTF	;CLEAR REJECT FLAG
471	002632	005067	015104		CLR	TRNKF	;CLEAR TRUNCATE FLAG
472	002636	005067	015042		CLR	SPCEV	;CLEAR MST DWN EVENT
473	002642	005067	015040		CLR	XSPCEV	;CLR NOW MST EVENT
474	002646			RTRYA:	CALL	PNCRLF	;ENQUEUE CR, & LF
475	002662				CALL	PRESC	;ENTER COMMAND MODE!
476	002676	000167	000060		JMP	PCLOOP	;GO TO MAIN LOOP
477							
478							
479	002702	011667	014756		TRAP4:	MOV	(SP),SUMSV
480	002706	162767	000002	014750		SUB	#2,SUMSV
481	002714	012700	031140			MOV	#TRP4AD,R0
482	002720	016701	014740			MOV	SUMSV,R1
483	002724	004767	027752			JSR	PC,OCTJSP
484	002730					CALL	PNTLIN,<TRPDMG>
485	002746	012706	002000			MOV	#STKTOP,SP
486	002752	005037	177776			CLR	#PS
487	002756	000167	177664			JMP	RTRYA

```

489 .SBTTL MAIN LOOP
490
491
492 002762 005767 013552 PCLOOP: TST TIQ ;IS TTY INPUT QUEUE EMPTY?
493 002766 001406 BEQ NXT0 ;YES, TEST ANOTHER FLAG
494 002770 CALL TTINP ;NO, PROCESS A CHARACTER
495 003004 105767 014631 NXT0: TSTB CMDENT ;HAS A COMMAND BEEN ENTERED?
496 003010 100006 BPL NXT1 ;NO, TEST ANOTHER FLAG
497 003012 CALL COMENT ;YES, PROCESS COMMAND
498 003026 005767 013572 NXT1: TST TOQ ;IS TTY OUTPUT QUEUE EMPTY?
499 003032 001406 BEQ NXT2 ;YES, TEST ANOTHER FLAG
500 003034 CALL TTOUT ;NO, OUTPUT A CHAR IF DEV RDY
501 003050 005767 014602 NXT2: TST QLDEV ;IS ADDR QUEUE EMPTY?
502 003054 100006 BPL NXT3 ;NO, TEST ANOTHER FLAG
503 003056 CALL ADQLD ;YES, LOAD ADDR QUEUE
504 003072 005767 014562 NXT3: TST DATGEV ;IS DATA GEN FLAG SET?
505 003076 100006 BPL NXT4 ;NO, TEST ANOTHER FLAG
506 003100 CALL DATGEN ;YES, GENERATE NEW RANDOM DATA
507 003114 005767 014554 NXT4: TST TXMEV ;IS TXTR EVENT FLAG SET?
508 003120 100006 BPL NXT5 ;NO, TEST ANOTHER FLAG
509 003122 CALL TXMIT ;YES, ENTER XMIT MODULE
510 003136 005767 014540 NXT5: TST RCVEV ;IS RCVR EVENT FLAG SET?
511 003142 100006 BPL NXT6 ;NO, TEST ANOTHER FLAG
512 003144 CALL RECV ;YES, ENTER RCVR MODULE
513 003160 005767 014520 NXT6: TST SPCEV ;IS SPECIAL EVENT FLAG SET?
514 003164 100006 BPL NXT7 ;NO, TEST ANOTHER FLAG
515 003166 CALL SPEC ;YES, HANDLE SPECIAL EVENT
516 003202 005767 014460 NXT7: TST STSEV ;IS STATUS EVENT FLAG SET?
517 003206 100006 BPL NXT8 ;NO, TEST ANOTHER FLAG
518 003210 CALL STATUS ;YES, OUTPUT STATUS
519 003224 005767 014432 NXT8: TST SUMEV ;IS SUMMARY EVENT FLAG SET?
520 003230 100006 BPL NXT9 ;NO, TEST ANOTHER FLAG
521 003232 CALL SUMRY ;YES, OUTPUT ERROR SUMMARY
522 003246 005767 014434 NXT9: TST XSPCEV ;IS EXTRA-SPECIAL EVENT SET?
523 003252 100006 BPL NXT10
524 003254 CALL XSPEC ;YES, HANDLE NOW MASTER.
525 003270 005767 014356 NXT10: TST TEREV ;IS XMTR ERROR EVENT SET?
526 003274 100006 BPL NXT11 ;NO, TEST ANOTHER FLAG
527 003276 CALL TEROS ;YES, OUTPUT XMTR ERROR TABLE
528 003312 005767 014336 NXT11: TST REREV ;IS RCVR ERROR EVENT SET?
529 003316 100006 BPL NXT12 ;NO,
530 003320 CALL REROS ;YES, OUTPUT RCVR ERROR TABLE
531 003334 000167 177422 NXT12: JMP PCLOOP ;STAY IN MAIN LOOP
532
533 ;TRAP TO 4 HANDLER
534
535
536 003340 ERTRAP: REGSAV
537 003344 CALL PNTLIN,<ERTMSG>
538 003362 005067 014314 CLR RCVEV
539 003366 005067 014306 CLR PCLGO
540 003372 052777 000100 012536 BIS #B06,@TTRCSR
541 003400 005037 177776 CLR @#PS
542 003404 REGRES
543 003410 012706 002000 MOV #STKTOP,SP
544 003414 000167 177226 JMP RTRYA

```

```

;;;SAVE R0...R5
;;;PRINT TRAP MESSAGE
;;;CLEAR RCVR EVENT FLAG
;;;CLR PCL GO FLAG
;;;SET TTY KBD INTR ENAB
;;;DROP CP PRIORITY

;FIX STACK
;ENTER COMMAND MODE

```

```

546 .SBTTL COMMAND PROCESSORS:
547 .SBTTL COMMAND PROC. FOR SILO (LOAD)
548
549
550 : LOAD OR CLEAR THE TRANSMITTER ADDRESS (HARDWARE) SILO.
551 : IF THERE ARE NO ARGUMENTS, CLEAR THE ADDRESS SILO AND SET
552 : AUTO ADDRESS. IF THERE ARE ARGUMENTS, LOAD THE TOTAL NUMBER
553 : OF ARGUMENTS (INCLUDING PAD VALUES) INTO THE SILO AS MANY TIMES
554 : AS THE TOTAL WILL GO INTO 50. LOCATIONS.
555
556 : IF ANY 2 SEQUENTIAL ARGUMENTS ARE THE SAME, SEPARATE THEM WITH
557 : A PAD VALUE OF '0'. IF THE FIRST ARGUMENT IS THE SAME AS THE LAST
558 : ARGUMENT, INSERT A PAD VALUE OF '0' AFTER THE LAST ARGUMENT.
559
560 : ARGUMENTS HIGHER THAN 37 (OCTAL) WILL NOT BE ACCEPTED
561 : ARGUMENTS LOWER THAN 1 WILL NOT BE ACCEPTED.
562
563 : ARGUMENTS MUST BE NUMERIC. DECIMAL ARGUMENTS MUST BE FOLLOWED BY
564 : A DECIMAL POINT. ('8.)
565 : PROC CPSILO
566 (1) 003420
567 (1) 003420 012701 017776
568 003420 016602 000002
569 003430 005302
570 003432 001002
571 003434 000167 000446
572 003440 010267 014254
573 003444 020227 000062
574 003450 101402
575 003452 000167 000422
576 003456 010267 014276
577 003462 010200
578 003464 005300
579 003466
580 003476 060600
581 003500 062700 000004
582 003504 016067 000004 014176
583 003512 012767 000000 014172
584 003520 021027 000002
585 003524 001402
586 003526 000167 000346
587 003532 026067 000004 014152
588 003540 001004
589 003542 112721 000000
590 003546 005267 014206
591 003552 116011 000004
592 003556 001002
593 003560 000167 00034
594 003564 122711 000037
595 003570 103002
596 003572 000167 000302
597 003576 112167 014110
598 003602 062700 177772
599 003606 005302
599 003610 001343

CPSILO:
MOV #ADSILO,R1
MOV 2(SP),R2
DEC R2
BNE 11$
JMP CLEAV
11$: MOV R2,PADFLG
CMP R2,#50.
BLOS 1$
JMP SYNRTM
1$: MOV R2,OBJCNT
MOV R2,R0
DEC R0
MULT 6,R0,R3
ADD SP,R0
ADD #4,R0
MOV 4(R0),FIRST
MOV #0,NEXT
CPMLP: CMP (R0),#2
BEQ 2$
JMP SYNRTM
2$: CMP 4(R0),NEXT
BNE 3$
MOVB #0,(R1)+
INC OBJCNT
3$: MOVB 4(R0),(R1)
BNE 4$
JMP SYNRTM
4$: CMPB #37,(R1)
BHS 5$
JMP SYNRTM
5$: MOVB (R1)+,NEXT
ADD #-6,R0
DEC R2
BNE CPMLP

:**ENTRY POINT**
:SET R1 TO POINT TO SILO BUFFER
:GET # OF ARGS. INTO R2
:O.K IF THERE ARE SOME
:OTHERWISE, EXIT
:SAVE COUNT FOR LATER USE.
:ARE THERE MORE THAN 50 OBJECTS?
:NO, CONTINUE
:YES, ERROR
:SAVE OBJECT COUNT
:TURN LOOK-UP AROUND

:SAVE FIRST ITEM
:SAVE CURRENT ITEM
:IS IT CLASS 2?
:YES, O.K.
:NO, ERROR
:IS THIS ITEM SAME AS LAST?
:NO, O.K.
:YES, INSERT A PAD VALUE
:KEEP OBJECT COUNT UP TO DATE
:PUT OBJECT INTO BUFFER
:ERROR IF OBJECT IS 0

:ERROR IF IT WAS > 37

:SAVE REAL ITEM
:SET UP TO GET NEXT OBJECT
:ARE WE DONE LOADING BUFF?
:NO, KEEP GOING

```

600	003612	026767	014074	014070		CMP	NEXT, FIRST	; IS LAST OBJECT = FIRST?
601	003620	001004				BNE	6\$; NO, O.K.
602	003622	112721	000000			MOVB	#0, (R1)+	; YES, INSERT A PAD VALUE
603	003626	005267	014126			INC	OBJCNT	; AND KEEP OBJECT COUNT UP TO DATE
604	003632	026727	014122	000062	6\$:	CMP	OBJCNT, #50.	; OBJECT COUNT GOTTEN TOO BIG?
605	003640	101402				B' OS	7\$; NO, O.K.
606	003642	000167	000232			JMP	SYNRTM	; YES, ERROR
607	003646	152777	000060	012236	7\$:	BISB	#B05+B04, @TMMRH	; PREPARE TO LOAD ADDR SILO
608	003654	012700	000062			MOV	#50., R0	; HOLD SILO SIZE
609	003660	026767	014074	014032		CMP	OBJCNT, PADFLG	; IS OBJECT COUNT DIFFERENT THAN START?
610	003666	001002				BNE	SILLD	; YES, LEAVE SOMETHING IN PADFLG (SET)
611	003670	005067	014024		8\$:	CLR	PADFLG	; NO, CLEAR PAD FLAG
612	003674	016702	014060		SILLD:	MOV	OBJCNT, R2'	; GET NO. OF OBJECTS
613	003700	012701	017776			MOV	#ADSILO, R1	; GET OBJECT BUFFER
614	003704	112177	012200		SLOLP:	MOVB	(R1)+, @TMMRH	; GET AN OBJECT INTO SILO
615	003710	005302				DEC	R2	; LOADED ALL OBJECTS?
616	003712	001374				B' IE	SLOLP	; NO, CONTINUE
617	003714	166700	014040			SUB	OBJCNT, R0	; YES, IS THAT ALL THAT'LL FIT?
618	003720	020067	014034			CMP	R0, OBJCNT	
619	003724	002363				BGE	SILLD	; IF NOT LOAD THEM AGAIN
620	003726	132777	000001	012156		BITB	#1, @TMMRH	; SEE IF I AM MASTER
621	003734	001037				BNE	CPSLV	; IF SO TURN ON SILO AND EXIT
622	003736					CALL	PNTLIN, <MSTMG1>	; PRINT 'THIS UNIT IS NOT MASTER
623	003754					CALL	PNTLIN, <MSTMG2>	; BUT HAS BEEN MADE SECONDARY
624	003772					CALL	PNTLIN, <MSTMG3>	; THE SILO YOU HAVE JUST FILLED
625	004010					CALL	PNTLIN, <MSTMG4>	; WILL BE USED IF YOU CLEAR THE
626	004026	152777	000002	012056		BISB	#B01, @TMMRH	; THE CURRENT MASTER'.
627	004034	005767	013660		CPSLV:	TST	PADFLG	; HAS SILO BEEN PADDED?
628	004040	001407				BEQ	9\$; NO, CARRY ON.
629	004042					CALL	PNTLIN, <MSTMG5>	; YES, TELL OPERATOR.
630	004060	012767	177777	013670	9\$:	MOV	#-1, ESCFLG	; FLAG COMENT TO ISSUE NEW 'PCL>'.
631	004066	142777	000020	012016		BICB	#B04, @TMMRH	; CLR AUTO ADDR
632	004074	000241				CLC		; CLEAR 'C' FOR NON ERROR RETURN
633	004076				CPSEX:	RETURN		
634								
635	004100	000261			SYNRTM:	SEC		; SET 'C' BIT FOR SYNTAX ERROR
636	004102	000167	177770			JMP	CPSEX	; AND RETURN
637								
638								
639	004106	152777	000060	011776	CLEAV:	BISB	#B05+B04, @TMMRH	; RESET AUTO ADDR & CLR SILO
640	004114	012767	177777	013634		MOV	#-1, ESCFLG	; FLAG COMENT TO ISSUE NEW 'PCL>'.
641	004122	000241				CLC		; CLEAR 'C' IN CASE IT WAS SET
642	004124	000167	177746			JMP	CPSEX	; RETURN

```
644 .SBTTL COMMAND PROC. FOR MASTER, SECONDARY AND R.I.B.
645
646 ;PROCESSOR FOR 'MASTER SET (OR CLEAR)' COMMAND
647 ; SET MASTER, OR CLEAR MASTER
648
649 004130 PROC CPMAS
(1)
(1) 004130 CPMAS: ;**ENTRY POINT**
650 004130 022766 000002 000002 CMP #2,2(SP) ;GET # OF ARGUMENTS
651 004136 001402 BEQ CPMOK ;OKAY IF 2
652 004140 000261 SEC
653 004142 000423 BR CPMRET ;OTHERWISE, SYNTAX ERROR
654 004144 010603 CPMOK: MOV SP,R3 ;GET 2ND WORD OF COMMAND
655 004146 062703 000004 ADD #4,R3
656 004152 KEYWD R3,#SCTBL ;PROCESS IT
657 004164 103446 BCS KWDERT ;SYNTAX ERROR IF 'C' SET
658 004166 117702 011720 MOVB @TMMRH,R2 ;GET OLD TMMR
659 004172 142702 000001 BICB #1,R2 ;REMOVE OLD STATE OF MASTER
660 004176 052602 BIS (SP)+,R2 ;SET NEW STATE OF MASTER
661 004200 110277 011706 MOVB R2,@TMMRH ;LOAD NEW TMMR
662 004204 012767 177777 013544 MOV #-1,ESCFLG ;FLAG COMENT TO ISSUE NEW 'PCL>'.
663 004212 CPMRET: RETURN ;EXIT
664
665
666 ;PROCESSOR FOR 'SECONDARY SET (OR CLEAR)' COMMAND
667 ; SET SECONDARY, OR CLEAR SECONDARY
668
669 004214 PROC CPSEC
(1)
(1) 004214 CPSEC: ;**ENTRY POINT**
670 004214 022766 000002 000002 CMP #2,2(SP) ;ARE THERE 2 ARGUMENTS?
671 004222 001402 BEQ CPSOK ;IF YES, PROCEED
672 004224 000261 SEC
673 004226 000424 BR CPSRET ;OTHERWISE, SYNTAX ERROR
674 004230 010603 CPSOK: MOV SP,R3 ;GET 2ND WORD OF COMMAND
675 004232 062703 000004 ADD #4,R3
676 004236 KEYWD R3,#SCTBL ;PROCESS IT
677 004250 103414 BCS KWDERT ;SYNTAX ERROR IF 'C' SET
678 004252 117702 011634 MOVB @TMMRH,R2 ;GET OLD TMMR
679 004256 142702 000002 BICB #2,R2 ;REMOVE OLD STATE OF SECONDARY
680 004262 006316 ASL (SP)
681 004264 052602 BIS (SP)+,R2 ;SET NEW STATE OF SECONDARY
682 004266 110277 011620 MOVB R2,@TMMRH ;LOAD NEW TMMR
683 004272 012767 177777 013456 MOV #-1,ESCFLG ;FLAG COMENT TO ISSUE NEW 'PCL>'.
684 004300 CPSRET: RETURN ;EXIT
685
686
687 004302 032600 KWDERT: BIT (SP)+,R0 ;POP BAD WORD OFF STACK
688 004304 000775 BR CPSRET ;EXIT
689
690
691 ;PROCESSOR FOR 'RIB SET' OR 'RIB CLEAR'
692 ; SET 'RIB' BIT IN XMTR COMMAND WORD LOCATION 'TXMST', OR CLEAR IT
693
694 004306 PROC CPRIB
(1)
```

ADDRESS	HEX	ASCII	DISPATCH	KEYWORD	OPERATION	COMMENT
(1)	004306					
695	004306	022766	000002	000002	CPRIB:	
696	004314	001402			CMP	#2,2(SP)
697	004316	000261			BEQ	CPRBK
698	004320	000426			SEC	
699	004322	010603			BR	CPRRET
700	004324	062703	000004		CPRBK:	MOV SP,R3
701	004330				ADD	#4,R3
702	004342	103757			KEYWD	R3,#SCTBL
703	004344	016702	003030		BCS	KWDERT
704	004350	042702	100000		MOV	TXMST,R2
705	004354	000241			BIC	#B15,R2
706	004356	006016			CLC	
707	004360	006016			FOR	(SP)
708	004362	052602			ROR	(SP)
709	004364	010267	003010		BIS	(SP)+,R2
710	004370	012767	177777	013360	MOV	R2,TXMST
711	004376				MOV	#-1,ESCFLG
					CPRRET:	RETURN

```

713 .SBTTL COMMAND PROC. FOR RANGE
714
715
716 ; SET RANGE OF RECEIVER ADDRESSES THAT THIS TRANSMITTER SHOULD TALK TO.
717 ;
718 ; ACCEPT A LOWER AND UPPER RECEIVER ADDRESS (IN THAT ORDER).
719 ; GENERATE A LIST (SET ACTIVE FLAGS) OF RCVR ADDRESSES FROM LOW ADDRESS
720 ; TO HIGH ADDRESS INCLUSIVE.
721
722
723 004400 PROC CPRANG
(1)
(1) 004400 (CPRANG: ;**ENTRY POINT**
724 004400 016600 000002 MOV 2(SP),R0 ;GET # OF OBJECTS
725 004404 022700 000003 CMP #3,R0 ;3 OBJECTS?
726 004410 001047 BNE SYNRTN ;IF NOT, INDICATE SYNTAX ERROR
727 004412 016600 000004 MOV 4(SP),R0 ;GET CLASS OF FIRST OBJECT
728 004416 022700 000002 CMP #2,R0 ;IS IT CLASS 2?
729 004422 001042 BNE SYNRTN ;IF NOT, INDICATE SYNTAX ERROR
730 004424 016601 000010 MOV 10(SP),R1 ;GET 'TO' ARGUMENT INTO R1
731 004430 016600 000012 MOV 12(SP),R0 ;GET CLASS OF SECOND OBJECT
732 004434 022700 000002 CMP #2,R0 ;IS IT CLASS 2?
733 004440 001033 BNE SYNRTN ;IF NOT, INDICATE SYNTAX ERROR
734 004442 016602 000016 MOV 16(SP),R2 ;GET 'FROM' ARGUMENT INTO R2
735 004446 020102 CMP R1,R2 ;IS 'FROM' LOWER THAN 'TO'?
736 004450 101427 BLOS SYNRTN ;IF NOT, INDICATE SYNTAX ERROR
737 004452 022701 000037 CMP #37,R1 ;IS 'TO' > 37?
738 004456 103424 BLO SYNRTN ;IF SO, INDICATE SYNTAX ERROR
739 004460 010200 MOV R2,R0 ;FIND 'FROM' ENTRY
740 004462 001422 BEQ SYNRTN ;IF 'FROM'=0, SYNTAX ERROR
741 004464 006300 ASL R0
742 004466 006300 ASL R0
743 004470 010003 MOV R0,R3
744 004472 006300 ASL R0
745 004474 060300 ADD R3,R0
746 004476 062700 020064 ADD #RAD80,R0 ;R0 CONTAINS 'FROM' ADDRESS
747 004502 012710 177777 (CPRFIL: MOV #-1,(R0) ;SET RCV ADDR ACTIVE FLAG.
748 004506 062700 000014 ADD #14,R0 ;UPDATE TABLE POINTER
749 004512 005202 INC R2 ;INCREMENT 'FROM' ADDRESS
750 004514 020201 CMP R2,R1 ; UNTIL EQUAL TO 'TO' ADDRESS
751 004516 003771 BLE CPRFIL ;EXIT WHEN COMPLETE.
752 004520 012767 177777 013230 MOV #-1,ESCFLG ;FLAG COMENT TO ISSUE NEW 'PCL>'
753 004526 (CPRRTN: RETURN
754
755 004530 000261 SYNRTN: SEC ;SET 'C' BIT TO INDICATE SYNTAX ERROR
756 004532 000167 177770 JMP (CPRRTN

```



```
758 .SBTTL COMMAND PROC. FOR ADD AND DELETE
759
760 ; ADD A RECEIVER ADDRESS TO THE LIST (SET ACTIVE FLAG) OF RCVR ADDRESSES
761 ;
762 ; ACCEPT ARGUMENTS (AT LEAST 1) CHECK THAT THEY FALL IN A RANGE FROM
763 ; 1 TO 37; THEN SET THE CORRESPONDING ACTIVE FLAGS IN THE STAT
764 ; TABLE.
765
766 004536 PROC CPADD
(1)
(1) 004536 CPADD: ;**ENTRY POINT**
767 004536 016602 000002 MOV 2(SP),R2 ;GET # OF ARGUMENTS INTO R2
768 004542 005302 DEC R2 ;IGNORE KEYWORD OBJECT
769 004544 010600 MOV SP,R0 ;GENERATE ADDRESS OF ARGUMENTS
770 004546 062700 000004 ADD #4,R0 ;POINT R0 AT 1ST OBJECT CLASS
771 004552 021027 000002 CPALP: CMP (R0),#2 ;IS CLASS = 2?
772 004556 001027 BNE SYNTRA ;NO, INDICATE SYNTAX ERROR
773 004560 016001 000004 MOV 4(R0),R1 ;GET OBJECT INTO R1
774 004564 001424 BEQ SYNTRA ;IF IT'S 0, SYNTAX ERROR
775 004566 022701 000037 CMP #37,R1
776 004572 103421 BLO SYNTRA ;IF IT'S >37, SYNTAX ERROR
777 004574 006301 ASL R1 ;FIND THE TABLE ELEMENT
778 004576 006301 ASL R1
779 004600 010103 MOV R1,R3
780 004602 006301 ASL R1
781 004604 060301 ADD R3,R1
782 004606 062701 020064 ADD #RADB0,R1
783 004612 012711 177777 MOV #-1,(R1) ;SET ACTIVE FLAG FOR THIS TABLE ENTRY
784 004616 062700 000006 ADD #6,R0 ;SET UP FOR NEXT ARGUMENT
785 004622 005302 DEC R2 ;ARE WE DONE?
786 004624 001352 BNE CPALP ;NO, CONTINUE.
787 004626 012767 177777 013122 MOV #-1,ESCFLG ;FLAG COMENT TO ISSUE NEW 'PCL>'
788 004634 CPARTN: RETURN
789 004636 000261 SYNTRA: SEC ;SET 'C' BIT FOR SYNTAX ERROR
790 004640 000167 177770 JMP CPARTN
791
792 ;DELETE RECEIVER ADDRESSES AND ASSOCIATED 'ATTEMPTS' AND 'SUCCESSSES'.
793
794 004644 PROC CPDEL
(1)
(1) 004644 CPDEL: ;**ENTRY POINT**
795 004644 016602 000002 MOV 2(SP),R2 ;GET # OF ARGUMENTS INTO R2
796 004650 005302 DEC R2 ;IGNORE KEYWORD OBJECT
797 004652 010600 MOV SP,R0 ;GENERATE ADDRESS OF ARGUMENTS
798 004654 062700 000004 ADD #4,R0 ;POINT R0 AT 1ST OBJECTS CLASS
799 004660 021027 000002 CPDLP: CMP (R0),#2 ;IS CLASS =2
800 004664 001034 BNE SYNTRD ;NO,INDICATE SYNTAX ERROR
801 004666 016001 000004 MOV 4(R0),R1 ;GET OBJECT INTO R1
802 004672 001431 BEQ SYNTRD ;IF IT'S 0, SYNTAX ERROR
803 004674 022701 000037 CMP #37,R1
804 004700 103426 BLO SYNTRD ;IF IT'S >37, SYNTAX ERROR
805 004702 006301 ASL R1 ;FIND TABLE ELEMENT
806 004704 006301 ASL R1
807 004706 010103 MOV R1,R3
808 004710 006301 ASL R1
809 004712 060301 ADD R3,R1
```

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

J 5
MACY11 30A(1052) 20-OCT-78 09:48 PAGE 10-1
COMMAND PROC. FOR ADD AND DELETE

SEQ 0061

810	004714	062701	020064		ADD	#RADB0,R1	
811	004720	005011			CLR	(R1)	;CLEAR ACTIVE FLAG AT TABLE LOCATION
812	004722	062701	000004		ADD	#4,R1	;AND OTHER ENTRIES...
813	004726	005021			CLR	(R1)+	
814	004730	005021			CLR	(R1)+	
815	004732	005021			CLR	(R1)+	
816	004734	005011			CLR	(R1)	
817	004736	062700	000006		ADD	#6,R0	;SET UP FOR NEXT ARGUMENT
818	004742	005302			DEC	R2	;ARE WE DONE?
819	004744	001345			BNE	CPDLP	;NO, CONTINUE
820	004746	012767	177777	013002	MOV	#-1,ESCFLG	;FLAG COMENT TO ISSUE NEW 'PCL>'
821	004754				CPDRTN: RETURN		
822	004756	000261			SYNTRD: SEC		;SET 'C' BIT FOR SYNTAX ERROR
823	004760	000167	177770		JMP	CPDRTN	

.SBTTL COMMAND PROC. FOR CLEAR, STATUS, AND CONTINUE

; CLEAR THE ENTIRE LIST OF RECEIVER ADDRESSES (CLEAR ACTIVE FLAGS)
; TO USE THIS ROUTINE BY A CALL MACRO, ENTER AT PRCLR AS FOLLOWS :; CALL PRCLR
; CLEAR STATUS TABLE

CPCLR: ;**ENTRY POINT**

MOV 2(SP),R2 ;GET NUMBER OF OBJECTS INTO R2
CMP #1,R2 ;ARE THERE ANY ARGUMENTS?
BNE SYNTRC ;IF SO, INDICATE SYNTAX ERROR
MOV #-1,ESCFLG ;FLAG COMMENT TO ISSUE NEW 'PCL>'
PROC PRCLRPRCLR: ;**ENTRY POINT**
MOV #37,R1 ;SAVE COUNT OF TABLE ELEMENTS
MOV #RADB,R0 ;SET UP TO CLR ACTIVE FLAGS
CPCLRC: CLR (R0) ;CLEAR ALL ELEMENTS OF ENTRY
ADD #4,R0
CLR (R0)+
CLR (R0)+
CLR (R0)+
CLR (R0)+
DEC R1
BNE CPCLRC ;DONE?
CLR AQ ;IF NOT, CONTINUE
MOV AQ+QTOP,AQ+QFRONT ;INITIALIZE ADDR QUEUE TO EMPTY
MOV AQ+QTOP,AQ+QBACKCPCLRT: RETURN ;AND LEAVE
SYNTRC: SEC ;SET 'C' BIT FOR SYNTAX ERROR
JMP CPCLRT

; SET THE STATUS EVENT FLAG SO THAT THE STATUS TABLE WILL BE PRINTED.

PROC CPSTAT

CPSTAT: ;**ENTRY POINT**
CMP #1,2(SP) ;ARE THERE ANY ARGUMENTS?
BNE SYNTRS ;IF SO, INDICATE SYNTAX ERROR
BIS #B15,STSEV ;SET STATUS EVENT FLAG
CLR STPNTR ;0 STPNTR INDICATES HEADER FIRST
BIC #B06,ATCR ;CLR TXM INTERRUPT ENABLECPSRTN: RETURN
SYNTRS: SEC ;SET 'C' BIT FOR SYNTAX ERROR
JMP CPSRTN

; CONTINUE EXERCISING. DO NOT AFFECT THE STATUS TABLE

```
875      : DO NOT AFFECT THE ERROR TABLE
876      : DO NOT AFFECT THE RCVR ADDRESS QUEUE
877      : SIMPLY SET PCLGO
878      :
879
880
881 005126      PROC      CPCNT
(1)
(1) 005126      CPCNT:      ;**ENTRY POINT**
882 005126 022766 000001 000002      CMP      #1,2(SP)      ;ARE THERE ANY ARGUMENTS?
883 005134 001027      BNE      SYNTCN      ;IF SO, INDICATE SYNTAX ERROR
884 005136 012767 100000 012534      MOV      #B15,PCLGO      ;SET PCL GO FLAG
885 005144      CALL      PNCRLF      ;PRINT CR & LF
886 005160      CALL      PNTLIN,<EXCNT>      ;PRINT 'EXERCISER CONTINUING'
887 005176 005767 012446      TST      KWFLG      ;GOT A CLOCK?
888 005202 001403      BEQ      CPCNRT      ;NO
889 005204 012777 000100 010736      MOV      #B06,@LCS      ;RE-ENABLE CLOCK.
890 005212      CPCNRT: RETURN
891 005214 000261      SYNTCN: SEC
892 005216 000167 177770      JMP      CPCNRT      ;SET 'C' BIT FOR SYNTAX ERROR
```

```
894 .S6TTL COMMAND PROC. FOR INIT, SUMMARY, AND GO
895
896 ;TO USE CALL MACRO, ENTER AT PRINT AND PARCLR
897 ; CLEAR STATUS TABLE, SUMMARY TABLE, ERROR TABLE
898 ; CLEAR 'RIB' IN XMTR COMMAND WORD 'TXMST'
899 ; CLEAR CLOCK DATA
900 ; CLEAR TRANSMITTER HARDWARE
901 ; INITIALIZE DATA PATTERN
902
903
904
905 005222 PROC CPINIT
(1)
(1) 005222 CPINIT: ;**ENTRY POINT**
906 005222 022766 000001 000002 CMP #1,2(SP) ;ARE THERE ANY ARGUMENTS?
907 005230 001114 BNE SYNTIN ;IF SO, INDICATE SYNTAX ERROR
908 005232 012767 177777 012516 MOV #-1,ESCFLG ;FLAG COMENT TO ISSUE NEW 'PCL>'
909 005240 PROC PRINIT
(1)
(1) 005240 ;**ENTRY POINT**
910 005240 BDINIT XMTR ;CLEAR XMITTER.
911 005246 105067 012371 CLRB SECNDS ;CLEAR SECONDS (TIME)
912 005252 105067 012366 CLRB MINUTS ;CLEAR MINUTES
913 005256 105067 012360 CLRB TICKS ;CLEAR TICKS
914 005262 005067 012360 CLR HOURS ;CLEAR HOURS
915 005266 042767 100000 002104 BIC #B15,TXMST ;CLEAR RIB IN COMMAND WORD FOR XMTR
916 005274 CALL PRCLR ;CLR STATUS TABLE & ADDR QUEUE
917 005310 PROC PARCLR
(1)
(1) 005310 PARCLR: ;**ENTRY POINT**
918 005310 012700 024726 MOV #ERTBL,R0 ;CLEAR ERROR SUMMARY TABLE
919 005314 022710 177777 ERTBCL: CMP #-1,(R0) ;IS ERR NUM = -1?
920 005320 001406 BEQ ERTCD ;IF SO, DONE CLEARING
921 005322 005060 000004 CLR 4(R0) ;OTHERWISE, CLR OCCURENCES
922 005326 062700 000006 ADD #6,R0 ;STEP TO NEXT ENTRY
923 005332 000167 177756 JMP ERTBCL ;AND CONTINUE
924 005336 012700 001350 ERTCD: MOV #37*30,R0 ;ALSO CLEAR ERROR DATA
925 005342 012701 025150 MOV #TERTBL,R1 ;FROM DETAILED ERR TABLE
926 005346 012721 000000 1$: MOV #0,(R1)+
927 005352 005300 DEC R0
928 005354 001374 BNE 1$
929 005356 012700 000037 MOV #37,R0 ;CLEAR ATTEMPS & SUCCESSES
930 005352 012701 020100 MOV #RADB,R1 ;FROM STATUS TABLE
931 005366 062701 000004 2$: ADD #4,R1
932 005372 005021 CLR (R1)+
933 005374 005021 CLR (R1)+
934 005376 005021 CLR (R1)+
935 005400 005021 CLR (R1)+
936 005402 005300 DEC R0
937 005404 001370 BNE 2$
938 005406 016767 012312 012312 MOV ORIGSD,DTSEED ;RESTORE ORIGINAL DATA SEED
939 005414 016767 012310 012310 MOV MLSD,MSGLSD ;AND MSG LENGTH SEED
940 005422 005067 012246 CLR TXMEV ;CLR XMIT EVENT FLAG
941 005426 005067 012230 CLR SUMEV ;CLR SUMMARY EVENT FLAG
942 005432 005067 012230 CLR STSEV ;CLR STATUS EVENT FLAG
943 005436 005067 012210 CLR TEREV
```

```

944 005442 005067 012206          CLR      REREV          ;CLR XMTR & RCVR ERROR EVENTS
945 005446 005067 012264          CLR      RCTXPS         ;CLEAR DATA PASS NO.
946 005452 052767 100000 012200  BIS      #B15,DATGEV      ;SET DATA GEN FLAG
947 005460          CPINRT: RETURN
948 005462 000261          SYNTIN: SEC          ;SET 'C' BIT FOR SYNTAX ERROR
949 005464 000167 177770          JMP      CPINRT
950
951
952
953          ; SET THE SUMMARY EVENT FLAG SO THAT THE ERROR SUMMARY TABLE WILL BE PRINTED
954
955
956 005470          PROC      CPSUM
(1)
(1) 005470          CPSUM:          ;**ENTRY POINT**
957 005470 022766 000001 000002  CMP      #1,2(SP)      ;ARE THERE ANY ARGUMENTS?
958 005476 001006          BNE      SYNTSM        ;IF SO, INDICATE SYNTAX ERROR
959 005500 012767 100000 012154  MOV      #B15,SUMEV      ;SET SUMMARY EVENT FLAG
960 005506 005067 012160          CLR      SUMPNT        ;0 SUMPNT INDICATES HEADER FIRST
961 005512          CPSURT: RETURN
962 005514 000261          SYNTSM: SEC          ;SET 'C' BIT FOR SYNTAX ERROR
963 005516 000167 177770          JMP      CPSURT
964
965
966
967
968          ; START THE EXERCISER
969          ; CLEAR 'ATTEMPTS' AND 'SUCCESSES' IN STATUS TABLE
970          ; CLEAR ERROR TABLES
971          ; CAUSE RCVR ADDRESS QUEUE TO BE LOADED
972          ; SET PCLGO
973          ; IF THE RCVR ADDRESS QUEUE IS EMPTY AFTER BEING LOADED,
974          ; INDICATE THIS BY PRINTING THE FOLLOWING MESSAGE:
975          ;
976          ;     '***SORRY, I HAVE NO RECEIVER ADDRESSES.***'
977          ;
978          ; AND RETURN TO COMMAND INPUT MODE.
979
980
981
982 005522          PROC      CPGO
(1)
(1) 005522          CPGO:          ;**ENTRY POINT**
983 005522 022766 000001 000002  CMP      #1,2(SP)      ;ARE THERE ANY ARGUMENTS?
984 005530 001062          BNE      SYNTAX        ;IF SO, INDICATE SYNTAX ERROR
985 005532 005067 012200          CLR      RCTXPS         ;CLEAR PASS NO.
986 005536 005067 010662          CLR      AOQ          ;INIT ADDR QUEUE TO EMPTY
987 005542 105067 012075          CLRB     SECNDS        ;CLEAR SECONDS REG.
988 005546 105067 012072          CLRB     MINUTS        ;CLEAR MINUTES REG.
989 005552 105067 012064          CLRB     TICKS         ;CLEAR TICKS REG.
990 005556 005067 012064          CLR      HOURS         ;CLEAR HOURS REG.
991 005562 016767 010642 010644  MOV      AOQ+QTOP,AOQ+QFRONT
992 005570 016767 010634 010640  MOV      AOQ+QTOP,AOQ+QBACK
993 005576 052767 100000 012052  BIS      #B15,QLDEV      ;SET ADDR QUEUE LOAD EVENT
994 005604 012767 177777 012066  MOV      #-1,PCLGO      ;SET PCL TO GO
995 005612          CALL     PARCLR          ;CLEAR STATUS & ERRORS (NOT RCVR)

```

CZPLAB0 PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

MACY11 30A(1052) 20-OCT-78 09:48 PAGE 12-2
COMMAND PROC. FOR INIT, SUMMARY, AND GO

SEQ 0066

996	005626				CALL	PNCRLF		:PRINT CR, LF
997	005642				CALL	PNTLIN,<EXRST>		:PRINT 'EXERCISER STARTED'.
998	005660	005767	011764		TST	KWFLG		:GOT A CLOCK?
999	005664	001403			BEQ	STRTRT		:NO.
1000	005666	012777	000100	010254	MOV	#B06,@LCS		:ENABLE CLOCK INTR.
1001	005674				STRTRT:	RETURN		:EXIT
1002								
1003	005676	000261			SYNTAX:	SEC		:SET 'C' TO INDICATE SYNTAX ERROR
1004	005700	000167	177770		JMP	STRTRT		:AND EXIT
1005								
1006								

```
1008 .SBTTL COMMAND PROC. FOR 'ASSIGN'
1009
1010 ; ASSIGN XMTR ADDR & VECTOR AND RCVR ADDR & VECTOR.
1011 ; THE ASSIGN COMMAND MAY HAVE 0, 1, 2, 3 OR 4 ARGUMENTS:
1012
1013 :
1014 : 0 = ASSIGN STANDARD ADDRESSES & VECTORS TO RCVR & XMTR
1015 : 1 = ARGUMENT IS ASSUMED TO BE XMTR ADDRESS
1016 : 2 = ARGUMENTS ARE ASSUMED TO BE : XMTR ADDR, XMTR VECTOR
1017 : 3 = ARGUMENTS ARE ASSUMED TO BE: XMTR ADDR, XMTR VECT, RCVR ADDR
1018 : 4 = ARGUMENTS IN THE FOLLOWING ORDER:
1019 :
1020 : XMTR ADDRESS, XMTR VECTOR, RCVR ADDRESS, AND RCVR VECTOR.
1021 : CAUTION MUST BE EXERCISED WHEN ASSIGNING ADDRESSES TO GET THEM
1022 : IN THE CORRECT SEQUENCE AND ORDER.
1023
1023 005704 PROC CPASS
1024 (1)
1024 (1) 005704 CPASS: ;**ENTRY POINT**
1025 005704 010600 MOV SP,R0 ;COPY LIST POINTER
1026 005712 016001 000002 MOV 2(R0),R1 ;GET NO. OF ARGUMENTS
1027 005716 020127 000005 CMP R1,#5 ;4 ARGUMENTS?
1028 005720 001402 BLOS 1$ ;ERROR TOO MANY ARGUMENTS
1029 005724 000167 000372 JMP SYNXR ;LESS THAN 4, PARTIAL ASSIGNMENT
1030 005726 001030 1$: BNE PART1 ;GET CLASS OF OBJECT
1031 005732 016002 000004 MOV 4(R0),R2 ;IF IT'S A NUMBER, O.K.
1032 005736 022702 000002 CMP #2,R2
1033 005740 001402 BEQ 2$ ;ERROR WRONG CLASS
1034 005744 000167 000352 JMP SYNXR ;NOW GET ACTUAL NUMBER
1035 005748 062700 000006 2$: ADD #6,R0
1036 005752 016002 000002 MOV 2(R0),R2
1037 005756 032702 000003 BIT #3,R2 ;IS IT ON VECTOR BOUNDARY?
1038 005760 001402 BEQ 3$ ;YES
1039 005764 000167 000330 JMP SYNXR ;ERROR NOT ON VECTOR BOUNDARY
1040 005768 020227 000774 3$: CMP R2,#774 ;IS IT WITHIN VECTOR AREA?
1041 005772 001402 BLOS 4$ ;YES
1042 005776 000167 000316 JMP SYNXR ;ERROR, OUTSIDE VECTOR AREA
1043 006000 010267 010170 4$: MOV R2,RCVEC ;SAVE AS RCVR VECTOR
1044 006004 005301 DEC R1
1045 006008 020127 000004 PART1: CMP R1,#4 ;ARE THERE ONLY 3 ARGS?
1046 006012 001030 BNE PART2 ;NO, MUST BE FEWER
1047 006016 016002 000004 MOV 4(R0),R2 ;GET CLASS OF OBJECT
1048 006020 022702 000002 CMP #2,R2 ;IF IT'S A NUMBER, O.K.
1049 006024 001402 BEQ 1$
1050 006028 000167 000264 JMP SYNXR ;ERROR, OBJECT NOT A NUMBER
1051 006032 062700 000006 1$: ADD #6,R0 ;NOW, GET THE NUMBER
1052 006036 016002 000002 MOV 2(R0),R2
1053 006040 032702 000017 BIT #17,R2 ;CHECK IF ITS A VALID ADDR
1054 006044 001402 BEQ 2$ ;IT IS. O.K.
1055 006048 000167 000242 JMP SYNXR ;ERROR INVALID ADDRESS
1056 006052 020227 164000 2$: CMP R2,#164000 ;IS IT IN THE ADDRESS FIELD?
1057 006056 010302 BHIS 3$ ;YES, O.K.
1058 006060 000167 000230 JMP SYNXR ;ERROR. ADDR = OUTSIDE DEVICE AREA
1059 006064 010267 010100 3$: MOV R2,RCDEV ;SAVE AS RCVR ADDRESS
1060 006068 005301 DEC R1
1061 006072 020127 000003 PART2: CMP R1,#3 ;ARE THERE ONLY 2 ARGS?
1061 006100 001030 BNE PART3 ;NO, MUST BE ONLY 1.
```


1062	006102	016002	000004		MOV	4(R0),R2	;GET CLASS OF OBJECT
1063	006106	022702	000002		CMP	#2,R2	
1064	006112	001402			BEQ	1\$;IF IT'S A NUMBER, O.K.
1065	006114	000167	000176		JMP	SYNXR	;ERROR, WRONG CLASS
1066	006120	062700	000006	1\$:	ADD	#6,R0	;NOW GET THE NUMBER
1067	006124	016002	000002		MOV	2(R0),R2	
1068	006130	032702	000003		BIT	#3,R2	;IS IT ON VECTOR BOUNDARY?
1069	006134	001402			BEQ	2\$;YES
1070	006136	000167	000154		JMP	SYNXR	;ERROR, NOT ON VECTOR BOUNDARY
1071	006142	020227	000774	2\$:	CMP	R2,#774	;IS IT WITHIN VECTOR AREA?
1072	006146	101402			BLOS	3\$;YES
1073	006150	000167	000142		JMP	SYNXR	;ERROR, OUTSIDE VECTOR AREA
1074	006154	010267	010010	3\$:	MOV	R2, TXVEC	;SAVE AS XMTR VECTOR
1075	006160	005301			DEC	R1	
1076	006162	020127	000002	PART3:	CMP	R1,#2	;ONLY 1 ARGUMENT?
1077	006166	001027			BNE	PART4	;NO, MUST BE NONE.
1078	006170	016002	000004		MOV	4(R0),R2	;GET CLASS OF OBJECT
1079	006174	022702	000002		CMP	#2,R2	
1080	006200	001402			BEQ	1\$;IF IT'S A NUMBER, O.K.
1081	006202	000167	000110		JMP	SYNXR	;ERROR, WRONG CLASS
1082	006206	062700	000006	1\$:	ADD	#6,R0	;NOW GET THE NUMBER
1083	006212	016002	000002		MOV	2(R0),R2	
1084	006216	032702	000017		BIT	#17,R2	;CHECK FOR VALID ADDRESS
1085	006222	001402			BEQ	2\$;O.K.
1086	006224	000167	000066		JMP	SYNXR	
1087	006230	020227	164000	2\$:	CMP	R2,#164000	;IS IT IN ADDRESS FIELD?
1088	006234	103002			BHIS	3\$;YES
1089	006236	000167	000054		JMP	SYNXR	;ERROR, OUTSIDE ADDRESS FIELD
1090	006242	010267	007720	3\$:	MOV	R2, TXDEV	;SAVE AS XMTR ADDRESS
1091	006246	020127	000001	PART4:	CMP	R1,#1	;NO ARGUMENTS?
1092	006252	001014			BNE	ASEXT	;WE'RE DONE!
1093	006254	012767	164200	007704	MOV	#164200, TXDEV	;NO ARGUMENTS, LOAD DEFAULTS.
1094	006262	012767	000170	007700	MOV	#170, TXVEC	
1095	006270	012767	164220	007674	MOV	#164220, RCDEV	
1096	006276	012767	000174	007670	MOV	#174, RCVEC	
1097	006304	012737	000340	177776	ASEXT:	MOV	#340, @#PS
1098	006312	000167	173462		JMP	PCLEX	
1099							
1100	006316	000261			SYNXR:	SEC	;SET 'C' TO FLAG ERROR
1101	006320					RETURN	;RETURN

```
1103 .SBTTL COMMAND DECODER AND PROCESSOR
1104
1105 ;CONVERT COMMAND TO IT'S PROCESSING ROUTINE ADDRESS
1106 ;IF C BIT IS SET, INDICATE SYNTAX ERROR
1107 ;RETURN TO COMMAND INPUT MODE
1108 ;IF ESCFLG <> 0, CALL PRESC (CNTRL-C)
1109
1110 006322 PROC COMENT
1111 (1)
1112 (1) 006322 COMENT: ;**ENTRY POINT**
1113 006322 105067 011313 CLR CMDENT ;CLEAR COMMAND ENTERED FLAG
1114 006326 005067 011424 CLR ESCFLG ;CLR COMMAND INPUT MODE FLAG
1115 006332 CALL COMAND,<CMDBUF,CMDTBL> ;CONVERT COMMAND
1116 006352 103015 BCC CMDRTN ;IF C = 0, EXIT
1117 006372 CALL PNTLIN,<SYNTAX> ;PRINT 'SYNTAX ERROR' IF C SET
1118 006406 005767 011344 CALL PRESC ;GIVE ANOTHER 'PCL>'
1119 006412 001406 CMDRTN: TST ESCFLG ;SHOULD WE FAKE CNTRL-C?
1120 006430 BEQ CMDRET ;NO, EXIT
1121 CALL PRESC ;YES, GIVE A 'PCL>'
1122 CMDRET: RETURN ;RETURN TO MAIN LOOP
1123
1124 ;PROCESSOR FOR SPECIAL EVENT
1125 ;INDICATE 'NO MASTER' AND CLEAR EVENTS
1126 ;THIS EVENT IS CALLED BY THE MAIN LOOP AFTER
1127 ; THE OCCURRENCE OF A MASTER DOWN INTERRUPT
1128 ;THE RESULT OF THIS EVENT IS THAT THE PROGRAM WILL
1129 ; PRINT MASTER DOWN IN THE FOLLOWING FASION:
1130 ;
1131 ; * * * * * MASTER DOWN * * * * *
1132 ;
1133 ;THEN CLEAR ACTIVE EVENT FLAGS BY ENTERING COMMAND MODE.
1134 ;TO RECOVER FROM THIS STATE AND RESUME EXERCISING:
1135 ; (A) SET MASTER ON ANY ONE PCL11
1136 ; (B) TYPE THE COMMAND 'CONTINUE'
1137 006432 PROC SPEC
1138 (1)
1139 (1) 006432 SPEC: ;**ENTRY POINT**
1140 006446 CALL PNCRLF ;ENQUEUE CR & LF
1141 006500 005067 011200 CALL PNTLIN,<MDNER> ;ENQUEUE MASTER DOWN MESSAGE
1142 006504 005067 011170 CALL PNCRLF ;ANOTHER CR & LF
1143 006510 005067 011152 CLR SPCEV ;CLR SPECIAL EVENT FLAG
1144 006514 005067 011142 SPERTN: CLR PCLGO ;DON'T ALLOW TXM EVENT
1145 006520 CLR STSEV ;DISCONTINUE STATUS EVENT
1146 006534 CALL SUMEV ;DISCONTINUE SUM EVENT
1147 RETURN ;FAKE CNTRL-C KEY
```

```

1148 ;PROCESSOR FOR 'EXTRA' SPECIAL EVENT
1149 ;TO INDICATE WHEN THIS UNIT HAS JUST BECOME
1150 ;MASTER AS A RESULT OF PREVIOUSLY HAVING
1151 ;SECONDARY SET AND THE CURRENT BUS MASTER
1152 ;HAVING JUST DROPPED MASTER.
1153 ;WHEN 'NOW MASTER' INTERRUPT OCCURS, THE
1154 ;CONSOLE DEVICE WILL PRINT
1155 ;   '** THIS UNIT HAS BECOME NEW MASTER **'
1156
1157 006536      PROC      XSPEC
(1)
(1) 006536      XSPEC:      ;**ENTRY POINT**
1158 006536      CALL      PNCRLF      ;ENQUEUE CR & LF
1159 006552      CALL      PNTLIN,<MSCHNG> ;ENQUEUE NEW MASTER MESSAGE
1160 006570      CALL      PNCRLF      ;ANOTHER CR & LF
1161 006604      CLR       XSPECV      ;CLR XSPEC EVENT FLAG
1162 006610      RETURN     ;RETURN TO MAIN _LOOP
005067 011076

1163
1164
1165 ;PROCESSOR FOR 'DETAILED' ERROR TABLE PRINTOUT COMMAND 'ERRORS'
1166
1167 ;THIS ROUTINE WILL INITIALIZE THE TEMPORARY LOCATIONS REQUIRED
1168 ;BY THE XMTR AND RCVR ERROR EVENTS. IT WILL THEN QUICKLY CHECK THE
1169 ;RCVR AND XMTR ERROR TABLES FOR ANY ENTRIES. IF NONE EXIST, THE MESSAGE
1170 ;   '**NO ERRORS TO REPORT YET **'
1171 ;WILL BE PRINTED ON THE CONSOLE AND THAT IS ALL.
1172 ; IF ANY ENTRIES WERE FOUND, THE XMTR ERROR EVENT FLAG 'TEREV' WILL BE SET
1173 ; WHICH WILL CAUSE THE XMTR ERRORS AND THE RCVR ERRORS (IF ANY) TO
1174 ; BE PRINTED.
1175
1176
1177 006612      PROC      CPERR
(1)
(1) 006612      CPERR:      ;**ENTRY POINT**
1178 006612      022766      000001      000002      CMP      #1,2(SP) ;ANY ARGUMENTS?
1179 006620      001044      BNE      SYNRR      ;ERROR IF THERE ARE.
1180 006622      012700      025150      MOV      #TERTBL,R0 ;POINT AT XMTR TABLE
1181 006626      012701      001350      MOV      #37*30,R1 ;SET TABLE LENGTH COUNTER
1182 006632      005720      1$:      TST      (R0)+ ;ANY ENTRIES?
1183 006634      001016      BNE      SERFL      ;YES, CALL ERROR MODULE
1184 006636      005301      DEC      R1 ;NO, CHECK WHOLE TABLE
1185 006640      001374      BNE      1$
1186 006642      CALL      PNTLIN,<NOERMG> ;PRINT 'NO ERRORS'
1187 006660      012767      177777      011070      MOV      #-1,ESCFLG ;FLAG COMENT TO ISSUE 'PCL>'
1188 006666      000167      000036      JMP      CPERTN ;AND RETURN
1189 006672      012767      100000      010752      SERFL: MOV      #B15,TEREV ;SET XMTR EVENT FLAG
1190 006700      012767      000001      011062      MOV      #1,ERR0 ;INITIATE ERROR #
1191 006706      012767      000001      011056      MOV      #1,ERR1 ;INITIATE RCVR ADDR
1192 006714      012767      025150      011052      MOV      #TERTBL,ERR2 ;INITIATE TABLE POINTER
1193 006722      052767      100000      010766      BIS      #B15,HDRFLG ;SET HEADER FLAG
1194 006730      CPERTN: RETURN
1195
1196 006732      000261      SYNRR: SEC ;SET 'C' BIT TO INDICATE SYNTAX ERR
1197 006734      000167      177770      JMP      CPERTN

```

```

1199          .SBTTL RECEIVER ADDRESS QUEUE LOADER ROUTINE
1200
1201          : LOAD THE RECEIVER ADDRESS SOFTWARE QUEUE
1202          :
1203          : THE TRANSMITTER EVENT WILL PICK RCVR ADDRESSES FROM THE QUEUE
1204          : UNTIL IT IS EMPT.
1205          :
1206          : IF THE QUEUE IS LOADED BUT STILL REMAINS EMPTY, AND PCLGO IS SET,
1207          : THIS MEANS THE OPERATOR IS TRYING TO RUN BUT HASN'T ENTERED
1208          : ANY RECEIVER ADDRESSES (USING 'RANGE' OR 'ADD' ETC.)
1209          :
1210
1211
1212 006740          PROC      ADQLD
1213          (1)          ADQLD:          ;**ENTRY POINT**
1214          006740          REGSAV          ;SAVE R0...R5
1215          006744 005067 010706          CLR          QLDEV          ;CLR QUEUE LOAD EVENT FLAG
1216          006750 012702 000037          MOV          #37,R2          ;POSITION COUNTER
1217          006754 012700 020100          MOV          #RADB,R0          ;R0 POINTS AT TABLE
1218          006760 022710 177777          QCHACT: CMP          #-1,(R0)          ;IS ACTIVE FLAG SET IN TABLE?
1219          006766 004767 000036          BNE          QLOOK          ;NO, LOOK FOR NEXT ONE
1220          006772 062700 000014          JSR          PC,QLOAD          ;YES, LOAD ADDR INTO QUEUE
1221          006776 005302          QLOOK: ADD          #14,R0
1222          007000 001367          DEC          R2          ;ALL ENTRIES CHECKED?
1223          007002 005767 007416          BNE          QCHACT
1224          007006 001424          TST          AQQ          ;IS THERE ANYTHING IN QUEUE?
1225          007010 005267 010722          BEQ          QMTRN          ;NO, THIS IS NOT A VALID PASS
1226          007014 012767 100000 010652 QRTN: INC          RCTXPS          ;YES, INCR PASS NO.
1227          007022          MOV          #B15, TXMEV          ;SET XMTR EVENT FLAG
1228          007026          RECRES          ;RESTORE R0...R5
1229          RETURN
1230
1231 007030 116067 000002 021231 QLOAD: MOVW          2(R0),BKELEM+1          ;GET RCV ADDR INTO HIGH BYTE
1232 007036          CALL          ENQ,<BKELEM,AQQ>          ;PUT IT IN ADDR QUEUE
1233 007056 000207          RTS          PC
1234
1235
1236 007060 005767 010614          QMTRN: TST          PCLGO          ;ARE WE RUNNING?
1237 007064 001753          BEQ          QRTN          ;NO, LEAVE
1238 007066          CALL          PRCTLO          ;STOP PRINTING ANYTHING ELSE
1239 007102          CALL          PNTLIN,<MTQMSG>          ;TELL OPERATOR NO RCVR
1240 007120          CALL          PRESC          ;FAKE CNTRL-C
1241 007134 000727          BR          QRTN          ;RETURN
  
```

.SBTTL DATA GENERATION (RANDOM) ROUTINE

: GENERATE A BUFFER OF DATA AND NOTE THE FOLLOWING CONDITIONS:

: BITS <8:0> OF THE FIRST WORD = MESSAGE LENGTH
: IF THE FIRST WORD IS 170000 OR HIGHER, EXPECT A REJECT
: IF THE MESSAGE LENGTH IS > 600, EXPECT A TRUNCATE
: GENERATE RANDOM DATA USING THE FOLLOWING FORMULA:
: SN = (SN-1 * RANDOM NO.) + RANDOM INCREMENT

: WHERE S = DATA SEED; AND N = A NUMBER FROM 1 TO MESSAGE LENGTH.

: A MESSAGE LENGTH OF 0 IS NOT ALLOWED AND IS CHANGED TO 1.

1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
(1)
(1)
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286

007136
007136
007142
007146
007152
007160
007162
007170
007176
007200
007206
007214
007216
007224
007230
007234
007240
007242
007244
007250
007254
007260
007302
007310
007312
007314
007322
007330
007334

005067 010574
005067 010572
005067 010570
042767 177000 010552
001003
012767 000001 010542
026727 010536 000600
002403
012767 177777 010534
022767 170000 010512
010005
012767 177777 010514
005067 010512
012700 020666
016705 010472
006105
010567 010474
016705 010456
016720 010446
066767 010442 010416
005305
003360
016767 010406 010410
012767 100000 010344
005067 010324

PROC DATGEN
DATGEN:
CLR RCTXPS
CLR RJCTF
CLR TRNKF
BIC #177000,MSGLS
BNE DTCNT
MOV #1,MSGLS
DTCNT: CMP MSGLS,#600
BLT DTGNC
MOV #-1,TRNKF
DTGNC: CMP #170000,DTSEED
BHI DTGNCO
MOV #-1,RJCTF
CLR TRNKF
DTGNCO: MOV #DATBUF,R0
MOV MSGLS,R5
ROL R5
NEG R5
MOV R5,XML
MOV MSGLS,R5
DTCLP: MOV DTSEED,(R0)+
MULP RANM,DTSEED
ADD #1,DTSEED
DEC R5
BGT DTCLP
DTCDON: MOV DTSEED,MSGLS
MOV #15,XML
CLR DATGEV
RETURN

:**ENTRY POINT**
:CLEAR DATA PASS FLAG
:CLEAR REJECT SOFTWARE FLAG
:CLR TRUNCATE SOFTWARE FLAG
:LENGTH NOT TO EXCEED 1000
:IF NON-0, OKAY, CONTINUE.
:IF 0, MAKE IT AT LEAST 1
:IS LENGTH LESS THAN 600?
:NO, SET TRUNCATE SOFTWARE FLAG
:IS DATA SEED -10000 OR MORE?
:YES SET REJECT SOFTWARE FLAG
:AND CLR TRUNCATE SFTWR FLAG
:POINT R0 TO TOP OF BUFFER
:R5 IS MESSAGE LENGTH
:DOUBLE IT
:NEGATE IT
:SAVE IT FOR TRANSMIT MODULE
:R5 IS MSG LENGTH AGAIN
:PUT WORD INTO BUFFER
:GENERATE NEW RANDOM NUMBER
:IF R5 = 0, FINISHED
:GET NEW LENGTH
:SET XMIT EVENT FLAG
:CLEAR DAT GEN EVENT FLAG

1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320

000020

.SBTTL MULTIPLY ROUTINE FOR DATA GENERATION

;UNSGINED MULTIPLY ROUTINE

;CALLED BY 'JSR R4,MLI'

;WITH MULTIPLICAND IN R4 AND MULTIPLIER ON STACK

;PRODUCT RETURNED ON TOP OF STACK

```
MLI:  MOV    (SP)+,R1      ;GET MULTIPLICAND
      MCV    (SP),R3      ;GET MULTIPLIER
      MOV    R4,-(SP)     ;SAVE R4
      MCV    #16.,R4      ;SET UP FOR 16 BIT MULTIPLY
      CLR    R2
      MUL:   ROR    R2      ;SHIFT PRODUCT
      ROR    R3
      BCC    CYCL         ;CHECK IF MULTIPLIER BIT IS 0
      ADD    R1,R2        ;ADD IN MULTIPLICAND
      CYCL:  DEC    R4      ;COUNT LOOP
      BGT    MUL
      MOV    (SP)+,R4      ;RESTORE R4
      ASR    R2            ;ONE LAST SHIFT
      ROR    R3            ;PRODUCT IS IN R3
      MOV    R3,(SP)      ;STORE RESULT ON STACK
      JMP    @P4)+        ;RETURN
```

;MACRO FOR THE ABOVE IS AS FOLLOWS:

```
.MACRO  MULD  A,B
:      MOV    A,-(SP)      ;SAVE A ON STACK
:      MOV    B,R4        ;SAVE B IN R4
:      JSR    R4,MLI      ;PERFORM MULTIPLICATION
:      .WORD  .+2
:      MOV    (SP)+,B      ;PUT PRODUCT INTO B
:      .ENDM
```

.SBTTL TRANSMIT MODULE

```
1322
1323
1324
1325
1326      . . . . .
1327      : CONTROL TRANSMISSION OF RANDOM DATA TO DE-QUEUED RECEIVER ADDRESSES
1328      :
1329      : IF 'PCLGO' IS SET, ('GO' OR 'CONTINUE' SET IT), START TRANSMITTING
1330      : THE DATA CONTAINED IN THE BUFFER 'DATBUF'.
1331      : THE TARGET RECEIVER ADDRESS IS DEQUEUED FROM THE ADDRESS QUEUE.
1332      : THE MESSAGE LENGTH WAS DETERMINED DURING THE GENERATION OF THE DATA.
1333      :
1334      : NOTE THAT BIT 15 OF 'TXMST' MAY BE 1 OR 0 DEPENDING UPON THE USE
1335      : OF THE 'RIB SET' OR 'RIB CLEAR' COMMANDS.
1336
1337 007400 060101      TXMST: .WORD    060101      ;TRANSMITTER COMMAND WORD
1338
1339 007402      PROC    TXMIT
1340      (1)
1341      (1) 007402      TXMIT:      ;**ENTRY POINT**
1342      007402 005767 010272      TST     PCLGO      ;CAN PCL GO?
1343      007406 001455      BEQ     TXRTN      ;IF NOT, RETURN
1344      007410 005067 010260      CLR     TXMEV      ;CLR XMTR EVENT FLAG
1345      007414      BDINIT  XMTR      ;CLR XMTR HARDWARE
1346      007422 005767 006776      TST     A00      ;ANY ELEMENTS IN ADDR QUEUE?
1347      007426 001446      BEQ     TXOUT      ;IF NOT, FILL IT UP AGAIN
1348      007430 016777 010310 006446      MOV     TXML,@TSBC      ;LOAD BYTE COUNT
1349      007436 012777 020666 006442      MOV     #DATBUF,@TSBA      ;LOAD BUS ADDRESS
1350      007444      CALL     DEQ,<FRELEM,A00>      ;DEQUEUE RECEIVER ADDRESS
1351      007464 116767 020575 010260      MOVB    FRELEM+1,CURAD      ;GET ADDRESS TABLE OFFSET
1352      007472 016700 010254      MOV     CURAD,R0      ;FIND ADDRESS TABLE ENTRY
1353      007476 006300      ASL     R0      ;
1354      007500 006300      ASL     R0      ;
1355      007502 010005      MOV     R0,R5      ;
1356      007504 006300      ASL     R0      ;
1357      007506 060500      ADD     R5,R0      ;
1358      007510 062700 020064      ADD     #RADB0,R0      ;
1359      007514 062760 000001 000006      ADD     #1,6(R0)      ;UPDATE DOUBLE WORD
1360      007522 005560 000004      ADC     4(R0)      ;
1361      007526 016777 020532 006342      MOV     FRELEM,@TCR      ;GIVE RECEIVER ADDR TO XMTR
1362      007534 056777 177640 006334      BIS     TXMST,@TCR      ;START TRANSMISSION
1363      007542      TXRTN: RETURN
1364
1365      007544 012767 100000 010104      TXOUT: MOV     #B15,QLDEV      ;SET QUEUE LOAD EVENT FLAG
1366      007552 026727 010160 000004      CMP     RCTXPS,#4      ;IS THIS THE 5TH PASS?
1367      007560 001402      BEQ     TXDGEN      ;IF SO, SET DATA GEN EVENT FLAG
1368      007562 000167 177754      JMP     TXRTN      ;RETURN
1369      007566 012767 100000 010064      TXDGEN: MOV     #B15,DATGEV      ;SET DAT GEN EVENT FLAG
1370      007574 000167 177742      JMP     TXRTN      ;RETURN
```

```

1370      ::::TRANSMITTER INTERRUPT HANDLER
1371      :::: SORT THE CAUSES OF TRANSMITTER INTERRUPTS
1372      :::: AND GO TO THE PROPER HANDLING ROUTINE.
1373
1374
1375      100000      TXER      -      100000
1376      000020      TXBSY     =      20
1377      000200      TXSUC     -      200
1378      000040      TXSOR     =      40
1379      000004      NOWMST    =      4
1380
1381      007600      XMTINT: REGSAV      ::::SAVE R0...R5
1382      007604      042777 000100 006264      BIC      #B06,@TCR      ::::CLR TXM INTR ENABLE
1383      007612      012737 000200 177776      MOV      #200,@MPS      ::::ALLOW RCVR INTR BUT NOT TTY
1384      007620      032777 100000 006252      BIT      #TXER,@TSR      ::::IS THERE A HARDWARE ERROR?
1385      007626      001402      BEQ      XMTIS      ::::NO, CHECK SUC TXF
1386      007630      000167 000114      JMP      ERRINT      ::::HANDLE ERROR
1387      007634      032777 000200 006236      XMTIS: BIT      #TXSUC,@TSR      ::::WAS IT A SUC TRANSFER?
1388      007642      001402      BEQ      XMTSR      ::::NO, CHECK SOFTWARE REJECT
1389      007644      000167 000610      JMP      SUCINT      ::::HANDLE SUC TXF
1390      007650      032777 000040 006222      XMTSR: BIT      #TXSOR,@TSR      ::::WAS MESSAGE REJECTED?
1391      007656      001402      BEQ      XMTPNM      ::::NO,CHECK NOW MASTER
1392      007660      000167 000772      JMP      SORINT      ::::HANDLE REJECT
1393      007664      132777 000004 006220      XMTPNM: BITB     #NOWMST,@TMMRH      ::::IS THIS UNIT NOW MASTER?
1394      007672      001402      BEQ      XMTIB      ::::NO, WAS RECVR BUSY?
1395      007674      000167 001074      JMP      NMINT      ::::YES, HANDLE NOW MASTER INT
1396      007700      032777 000020 006172      XMTIB: BIT      #TXBSY,@TSR      ::::WAS RECEIVER BUSY
1397      007706      001402      BEQ      XMTPRB      ::::NO, INTERRUPT WAS ERRONEOUS
1398      007710      000167 000514      JMP      BSYINT      ::::HANDLE BUSY
1399      007714      012767 100000 007752      XMTPRB: MOV      #B15, TXMEV      ::::SET XMTR EVENT FLAG
1400      007722      ERROT      \N      ::::ERROR:ERRONEOUS INTERRUPT FROM XMTR
1401      (1)
1402      (1)
1403      (1)
1404      007742      REGRES      ::::RESTORE R0...R5
1405      007746      000002      RTI      ::::RETURN

```



```

1404      ;XMTR ERROR INTERRUPT ROUTINE
1405
1406
1407      ; DETERMINE AND REPORT THE CAUSE OF THE TRANSMITTER ERROR.
1408
1409
1410 007750 032777 040000 006122 ERRINT: BIT    #B14,@TSR      ;IS ERROR NON EXISTANT LOCATION?
1411 007756 001412                      BEQ    ERRA          ;
1412 007760                      ERROT  \N                    ;ERROR:NON-EXISTANT LOC ERROR IN XMTR
(1)
(1)
(1)
1413 010000 000167 000402          JMP    ERRX
1414 010004 032777 020000 006066 ERRA:  BIT    #B13,@TSR      ;IS ERROR MEM OFL?
1415 010012 001412                      BEQ    ERRA          ;
1416 010014                      ERROT  \N                    ;ERROR:MEM OFL ERROR IN XMTR
(1)
(1)
(1)
1417 010034 000167 000346          JMP    ERRX
1418 010040 032777 010000 006032 ERRA:  BIT    #B12,@TSR      ;IS ERROR TXM ERROR?
1419 010046 001472                      BEQ    ERRA          ;
1420 010050 017767 006024 007676      MOV    @TSR,RSPC      ;SAVE RESPONSE CODES
1421 010056 042767 177760 007670      BIC    #-20,RSPC     ;REMOVE GARBAGE
1422 010064 022767 000015 007662      CMP    #15,RSPC     ;RSP CODES 11&01?
1423 010072 001012                      BNE    ERRA          ;
1424 010074                      ERROT  \N                    ;ERROR:TXM ERROR. RCVR ACCEPTED A NULL
(1)
(1)
(1)
1425 010114 000167 000266          JMP    ERRX
1426 010120 042767 000003 007626 ERRA:  BIC    #3,RSPC      ;DISCARD RSP A'S
1427 010126 001012                      BNE    ERRA          ;RSB B - 00?
1428 010130                      ERROT  \N                    ;ERROR:TXM ERROR. RCVR HAS GONE OFF LINE
(1)
(1)
(1)
1429 010150 000167 000232          JMP    ERRX
1430 010154 022767 000010 007572 ERRA:  CMP    #10,RSPC     ;RSP B = 10?
1431 010162 001012                      BNE    ERRA          ;
1432 010164                      ERROT  \N                    ;ERROR:TXM ERROR. WORD OR CRC REJECTED
(1)
(1)
(1)
1433 010204 000167 000176          JMP    ERRX
1434 010210                      ERROT  \N                    ;ERROR:MISCELLANEOUS TXM ERROR
(1)
(1)
(1)
1435 010230 000167 000152          JMP    ERRX
1436 010234 032777 004000 005636 ERRA:  BIT    #B11,@TSR      ;MASTER DOWN?
1437 010242 001417                      BEQ    ERRA          ;
1438 010244                      ERROT  \N                    ;ERROR: M A S T E R D O W N
(1)
(1)
(1)

```

;***** XMTR ERROR 10 *****

```

1439 010264 005067 007410 CLR PCLGO ;STOP TRANSMITTER
1440 010270 012767 100000 007406 MOV #B15,SPCEV ;SET SPECIAL EVENT FLAG
1441 010276 000167 000104 JMP ERRX
1442 010302 032777 002000 005570 ERRD: BIT #B10,@TSR ;TIMEOUT ERROR?
1443 010310 001422 BEQ ERRE
1444 010312 032777 100000 005556 BIT #B15,@TCR ;IS 'R.I.B.' SET?
1445 010320 001404 BEQ 1$ ;NO MUST BE WEIRD ERROR.
1446 010322 032777 000020 005550 BIT #TXBSY,@TSR ;YES, IS TDM BUS BUSY?
1447 010330 001010 BNE 2$ ;YES, RECEIVER WASN'T THERE
1448 010332 1$: ERROT \N ;ERROR: ERRONEOUS TIMEOUT IN TRANSMITTER
(1)
(1) ;**** XMTR ERROR 11 ****
(1)
1449 010352 000167 000030 2$: JMP ERRX
1450 010356 032777 001000 005514 ERRE: BIT #B09,@TSR ;TXM SILO OVERRUN?
1451 010364 001410 BEQ ERRX
1452 010366 ERROT \N ;ERROR:SILO OVERRUN ERROR IN XMTR
(1)
(1) ;**** XMTR ERROR 12 ****
(1)
1453 010406 ERRX: BDINIT XMTR ;CLEAR ALL XMTR HARDWARE
1454 010414 012767 100000 007252 MOV #B15,TXMEV ;SET XMTR EVENT FLAG
1455 010422 REGRES ;RESTORE R0...R5
1456 010426 000002 RTI ;RETURN

```

```

1458          ;BUSY INTERRUPT ROUTINE
1459
1460 010430 042777 000020 005442 BSYINT: BIC      #TXBSY,@TSR      ;CLEAR TDM BUS BUSY
1461 010436 012767 100000 007230      MOV      #B15,TXMEV      ;SET TXM EVENT FLAG
1462 010444      BDINIT  XMTR      ;CLEAR HARDWARE
1463 010452      REGRES      ;RESTORE R0...R5
1464 010456 000002      RTI      ;RETURN
1465
1466          ;SUCCESSFUL TRANSFER INTERRUPT ROUTINE
1467
1468
1469 010460 042777 000200 005412 SUCINT: BIC      #TXSUC,@TSR      ;CLEAR SUC TXF
1470 010466 005767 007250      TST      TRMKF      ;IS SFTWR TRUNCATE FLAG SET?
1471 010472 001047      BNE      SUCIA      ;YES, GO CHECK SORE
1472 010474 032777 000040 005376 SUCINY: BIT      #TXSOR,@TSR      ;IS SORE SET?
1473 010502 001414      BEQ      SUCINP      ;NO, CONTINUE
1474 010504      ERROT  \N      ;ERROR:MESSAGE TRUNCATED UNEXPECTEDLY
(1)
(1)
(1)
1475 010524 000421      BR      SHBRT      ;DON'T INC SUC CONNS IF ERROR
1476 010526 042777 000040 005344 SUCINP: BIC      #TXSOR,@TSR      ;CLR SORE
1477 010534 016700 007212      MOV      CURAD,R0      ;PREPARE TO UPDATE STATUS TABLE
1478 010540 006300      ASL      R0      ;GET TABLE ENTRY FOR CURRENT
1479 010542 006300      ASL      R0      ;RECEIVER ADDRESS IN
1480 010544 010005      MOV      R0,R5      ;
1481 010546 006300      ASL      R0      ;ORDER TO UPDATE
1482 010550 060500      ADD      R5,R0      ;
1483 010552 062700 020064      ADD      #RADB0,R0      ;'SUCCESSFUL' ELEMENT
1484 010556 062760 000001 000012      ADD      #1,12(R0)      ;UPDATE DOUBLE WORD
1485 010564 005560 000010      ADC      10(R0)      ;
1486 010570 012767 100000 007076 SHBRT: MOV      #B15,TXMEV      ;SET TRANSMIT EVENT FLAG
1487 010576      BDINIT  XMTR      ;CLEAR HARDWARE
1488 010604      REGRES      ;RESTORE R0...R5
1489 010610 000002      RTI      ;RETURN
1490
1491 010612 032777 000040 005260 SUCIA: BIT      #TXSOR,@TSR      ;IS SORE SET? (SHOULD BE SET)
1492 010620 001011      BNE      SUCCS      ;
1493 010622      ERROT  \N      ;ERROR:MSG FAILED TO BE TRUNCATED
(1)
(1)
(1)
1494 010642 000752      BR      SHBRT      ;DON'T INC SUC CONNS IF ERROR
1495 010644 042777 000040 005226 SUCCS: BIC      #TXSOR,@TSR      ;CLR SORE
1496 010652 000'67 177616      JMP      SUCINY      ;PROCESS SUC TXF AS NORMAL

```

```
1498 ;SOFTWARE REJECT INTERRUPT ROUTINE
1499
1500 010656 005767 007056 SORINT: TST RJCTF ;IS SFTWR REJECT FLAG SET?
1501 010662 001432 BEQ SORIP ;IF NOT, SOMETHING'S WRONG
1502 010664 016700 007062 MOV CURAD,R0 ;PREPARE TO UPDATE STATUS TABLE
1503 010670 006300 ASL R0 ;GET TABLE ENTRY FOR CURRENT
1504 010672 006300 ASL R0 ;RECEIVER ADDRESS IN
1505 010674 010003 MOV R0,R3 ;
1506 010676 006300 ASL R0 ;ORDER TO UPDATE
1507 010700 060300 ADD R3,R0 ;
1508 010702 062700 020064 ADD #RAD80,R0 ;'SUCCESSFUL' ELEMENT
1509 010706 062760 000001 000012 ADD #1,12(R0) ;UPDATE ENTRY
1510 010714 005560 000010 ADC 10(R0)
1511 010720 042777 000040 005152 SORCNT: BIC #TXSOR,@TSR ;CLR SORE
1512 010726 012767 100000 006740 MOV #B15, TXMEV ;SET XMIT EVENT FLAG
1513 010734 BDINIT XMTR ;CLEAR HARDWARE
1514 010742 REGRES ;RESTORE R0...R5
1515 010746 000002 RTI ;RETURN
1516
1517 010750 SORIP: ERROT \N ;ERROR:ERRONEOUS REJECT BY RECEIVER
(1)
(1) ;**** XMTR ERROR 15 ****
(1)
1518 010770 000167 177724 JMP SORCNT ;RETURN
1519
1520
1521 ;NOW MASTER INTERRUPT ROUTINE
1522 ;SET XSPEC EVENT FLAG (XSPEV)
1523 ;AND CLR NOW MST
1524
1525
1526 010774 142777 000004 005110 NMINT: BICB #NOWMST,@TMMRH ;CLEAR NOW MASTER
1527 011002 012767 100000 006676 MOV #B15,XSPCEV ;SET EXTRA-SPECIAL EVENT
1528 011010 052777 000100 005060 BIS #B06,@TCR ;RE-SET INTERRUPT ENABLE
1529 011016 REGRES ;RESTORE R0...R5
1530 011022 000002 RTI ;RETURN
1531
1532
```

.SBTTL RECEIVER MODULE

;CONTROL RECEPTION OF DATA.
; DATA IS STORED IN A BUFFER 'RCBUF'
; FOR LATER CHECKING.
; 'RCBUF' BUFFER IS CLEARED BEFORE RECEPTION.

;ACTIVE IF RCVEV IS -VE

RCWD - B13

. PROC RECV

RECV:

CLR RCVEV
JSR PC,DBFCLR
BDINIT RCVR
MOV #RCBUF,@RDBA
MOV #-1400,@RDBC
MOV #RCWD,@RCR
BIS #B06,@RCR
RETURN

;**ENTRY POINT**
;CLR RCVR EVENT FLAG
;CLEAR RCVR DATA BUFFER FIRST
;CLR HARDWARE (REGARDLESS)
;LOAD RCVR BUS ADDRESS
;LOAD BYTE COUNT FOR 600 WORDS
;SET RCV WORD IN RCVR
;SET RCVR INTERRUPT ENABLE

DBFCLR: REGSAV

MOV #RCBUF,R0
MOV #700,R1
DBCLP: CLR (R0)+
DEC R1
BGT DBCLP
REGRES
RTS PC

;SAVE R0...R5
;R0 POINTS AT BUFFER AREA
;R1 HOLDS BUFFER SIZE
;CLEAR BUFF LOC
;DONE ?
;NO,LOOP
;YES, RESTORE R0...R5
;AND RETURN

1534
1535
1536
1537
1538
1539
1540
1541
1542
1543 020000
1544
1545 011024
(1)
(1) 011024
1546 011024 005067 006652
1547 011030 004767 000040
1548 011034
1549 011042 012777 022726 005056
1550 011050 012777 176400 005046
1551 011056 012777 020000 005032
1552 011064 052777 000100 005024
1553 011072
1554
1555
1556
1557 011074
1558 011100 012700 022726
1559 011104 012701 000700
1560 011110 005020
1561 011112 005301
1562 011114 003375
1563 011116
1564 011122 000207

```

1566      ::RECEIVER INTERRUPT HANDLER.
1567
1568      100000      RCER      =      100000
1569      000400      RCDOR     =      400
1570      000200      RCSUC     =      200
1571      000040      RCRCM     =      40
1572
1573 011124      RCVINT: REGSAV      ::SAVE R0...R5
1574 011130 032777 100000 004752      BIT      #RCER,@RSR      ::IS THERE A HARDWARE ERROR?
1575 011136 01402      BEQ      RCVINA      ::NO,CHECK DATA OUTPUT READY
1576 011140 000167 000100      JMP      RERINT      ::HANDLE ERROR INTERRUPT
1577 011144 032777 000400 004746      RCVINA: BIT      #RCDOR,@RSR      ::IS DATA OUTPUT READY?
1578 011152 001402      BEQ      RCVINB      ::NO, CHECK SUC TXF
1579 011154 000167 000472      JMP      RDOINT      ::HANDLE DATA OUTPUT RDY INTERRUPT
1580 011160 032777 000200 004732      RCVINB: BIT      #RCSUC,@RSR      ::SUCCESSFUL TRANSFER SET?
1581 011166 001402      BEQ      RCVINC      ::NO, CHECK REECT COMPLETED
1582 011170 000167 000536      JMP      RSUINT      ::HANDLE SUC TXF INTERRUPT
1583 011174 032777 000040 004716      RCVINC: BIT      #RCRCM,@RSR      ::REJECT COMPLETED INTERRUPT?
1584 011202 001402      BEQ      RCVIND      ::NO,ERRONEOUS INTERRUPT
1585 011204 000167 000776      JMP      RRJINT      ::HANDLE REJ-COM INTERRUPT
1586 011210      RCVIND: ERROR      \N      ::ERROR:UNKNOWN RECEIVER INTERRUPT OCCURRED
      (1)
      (1)
      (1)
1587 011230 012767 100000 006444      MOV      #B15,RCVEV      ::SET RCVR EVENT FLAG
1588 011236      REGRES      ::RESTORE R0...R5
1589 011242 000002      RTI      ::RETURN

```

```

1591                                ;RCVR ERROR INTERRUPT ROUTINE
1592
1593 011244 032777 040000 004646 RERINT BIT    #B14, @RSR      ;IS ERROR NON EXST LOC ERR?
1594 011252 001412                      BEQ    RERRA
1595 011254                      ERROR   \N      ;ERROR:NON EXST LOC ERR IN RCVR
(1)
(1)                                ;**** RCVR ERROR 17 ****
(1)
1596 011274 000167 000332          JMP    RERRX
1597 011300 032777 020000 004612 RERRA: BIT    #B13, @RSR      ;IS ERROR MEM OFLO?
1598 011306 001412                      BEQ    RERRB
1599 011310                      ERROR   \N      ;ERROR:MEM OFLO ERROR IN RCVR
(1)
(1)                                ;**** RCVR ERROR 20 ****
(1)
1600 011330 000167 000276          JMP    RERRX
1601 011334 032777 010000 004556 RERRB: BIT    #B12, @RSR      ;ANY TRANSMISSION ERRORS?
1602 011342 001457                      BEQ    RERRC
1603 011344 017767 004550 006402      MOV    @RSR, RSPC      ;GET RESPONSE CODES
1604 011352 042767 177760 006374      BIC    #-20, RSPC     ;REMOVE GARBAGE
1605 011360 032767 000003 006366      BIT    #3, RSPC      ;LOOK AT TXM RSP CODES
1606 011366 001012                      BNE    RERRB1
1607 011370                      ERROR   \N      ;ERROR:TXM WENT OFF LINE
(1)
(1)                                ;**** RCVR ERROR 21 ****
(1)
1608 011410 000167 000216          JMP    RERRX
1609 011414 042767 000003 006332 RERRB1: BIC    #3, RSPC      ;REMOVE TXM RSP CODES NOW
1610 011422 022767 000010 006324      CMP    #10, RSPC     ;WAS THERE A CRC ERROR?
1611 011430 001012                      BNE    RERRB2
1612 011432                      ERROR   \N      ;ERROR:RCVR CRC ERROR OCCURRED
(1)
(1)                                ;**** RCVR ERROR 22 ****
(1)
1613 011452 000167 000154          JMP    RERRX
1614 011456                      RERRB2: ERROR   \N      ;ERROR:FIRST WORD RECVD NOT VALID
(1)
(1)                                ;**** RCVR ERROR 23 ****
(1)
1615 011476 000167 000130          JMP    RERRX
1616 011502 032777 004000 004410 RERRC: BIT    #B11, @RSR      ;WAS ERROR A PARITY ERROR?
1617 011510 001412                      BEQ    RERRD
1618 011512                      ERROR   \N      ;ERROR:RCVR DETECTED INVALID PARITY
(1)
(1)                                ;**** RCVR ERROR 24 ****
(1)
1619 011532 000167 000074          JMP    RERRX
1620 011536 032777 002000 004354 RERRD: BIT    #B10, @RSR      ;DID RCVR TIME OUT?
1621 011544 001412                      BEQ    RERRE
1622 011546                      ERROR   \N      ;ERROR:RCVR TIMEOUT ERROR OCCURRED
(1)
(1)                                ;**** RCVR ERROR 25 ****
(1)
1623 011566 000167 000040          JMP    RERRX
1624 011572 032777 001000 004320 RERRE: BIT    #B09, @RSR      ;WAS IT BYTE COUNT OVERFLOW?
1625 011600 001414                      BEQ    RERRX

```

CZ
PC

1626	011602	052777	100000	004306	BIS	#B15,@RCR	:IMMEDIATELY SET REJECT IN RCVR
1627	011610	042777	040000	004300	BIC	#B14,@RCR	:CLEAR RCVR NPR BIT
1628	011616	042777	001000	004274	BIC	#B09,@RSR	:CLR BYTE COUNT OVERFLOW
1629	011624				RERTRN:	REGRES	:RESTORE R0...R5
1630	011630	000002				RTI	:RETURN
1631	011632				RERRX:	BDINIT	:CLEAR RECVR
1632	011640	012767	100000	006034	MOV	#B15,RCVEV	:SET RCV EVENT FLAG
1633	011646	000167	177752		JMP	RERTRN	
1634							


```
1636 ;RCVR DATA OUTPUT RDY INTERRUPT ROUTINE
1637
1638 011652 042777 000100 004236 RDOINT: BIC #B06,@RCR ;CLEAR INTR ENAB
1639 011660 017767 004236 006046 MOV @Rddb,RCSEED ;GET FLAG-WORD FOR DATA SEED
1640 011666 022767 170000 006040 CMP #170000,RCSEED ;LOOK AT DATA SEED
1641 011674 101011 BHI RDOINC ;IF MORE -VE THAN -10000
1642 011676 052777 100000 004212 BIS #B15,@RCR ;SET REJECT IN RCVR
1643 011704 RDORTN: REGRES ;RESTORE R0...R5
1644 011710 052777 000100 004200 BIS #B06,@RCR ;RE-SET INTR ENAB
1645 011716 000002 RTI ;RETURN
1646 011720 052777 040001 004170 RDOINC: BIS #B14+B00,@RCR ;START RECEIVER GOING
1647 011726 000167 177752 JMP RDORTN
1648
1649 ;RCVR SUC TXF INTERRUPT ROUTINE
1650
1651 011732 032777 000040 004160 RSUINT: BIT #RCRCM,@RSR ;WAS RECOM SET?
1652 011740 001403 BEQ RSUCNT ;NO, CONTINUE
1653 011742 042777 000040 004150 BIC #RCRCM,@RSR ;YES, CLR IT AND CONTINUE
1654 011750 016701 005760 RSUCNT: MOV RCSEED,R1 ;GET DATA SEED
1655 011754 042701 177000 BIC #177000,R1 ;MASK MSG LENGTH
1656 011760 001002 BNE RSNT0 ;IF NOT 0, OKAY
1657 011762 012701 000001 MOV #1,R1 ;IF 0, MAKE IT 1
1658 011766 162701 000001 RSNT0: SUB #1,R1 ;ACCOUNT FOR FLAG WORD
1659 011772 020127 000600 CMP R1,#600 ;IS IT GREATER THAN 600?
1660 011776 003402 BLE RSFLSZ ;NO, LEAVE IT ALONE
1661 012000 012701 000600 MOV #600,R1 ;YES, SET IT TO LIMIT (600)
1662 012004 012700 001400 RSFLSZ: MOV #1400,R0
1663 012010 067700 004110 ADD @RDBC,R0 ;DETERMINE # OF BYTES RECEIVED
1664 012014 001451 BEQ RSULV ;NOTHING TO CHECK, EXIT
1665 012016 006200 ASR R0 ;R0 = NO. OF WORDS RECEIVED
1666 012020 020001 CMP R0,R1 ;DID WE RECV ALL THE WORDS?
1667 012022 001412 BEQ RSUTST ;YES, CARRY ON
1668 012024 103456 BLO ERNE ;NOT ENOUGH !
1669 012026 ERROR \N ;ERROR:RCVR GOT TOO MANY WORDS..
(1)
(1)
(1)
1670 012046 000434 BR RSULV ;EXIT IF ERROR
1671 012050 012705 022726 RSUTST: MOV #RCBUF,R5 ;R5 POINTS AT RECVD DATA
1672 012054 RSUGEN: MULP RANM,RCSEED ;GENERATE CHECK WORD
1673 012076 066767 005646 005630 ADD RANK,RCSEED ;COMPARE IT WITH RECV'D WORD
1674 012104 026725 005624 CMP RCSEED,(R5)+
1675 012110 001411 BEQ RSUCHK
1676 012112 ERROR \N ;ERROR:DATA WORD RECV'D WAS BAD
(1)
(1)
(1)
1677 012132 000402 BR RSULV ;EXIT IF ERROR
1678 012134 005300 RSUCHK: DEC R0 ;CHECKED ALL WORDS?
1679 012136 001346 BNE RSUGEN ;NO,CONTINUE
1680 012140 RSULV: BDINIT RCVR ;CLR HARDWARE
1681 012146 REGRES ;RESTORE R0...R5
1682 012152 012767 100000 005522 MOV #B15,RCVEV ;SET RCVR EVENT FLAG
1683 012160 000002 RTI ;RETURN
1684
1685 012162 ERNE: ERROR \N ;ERROR: RCVR GOT TOO FEW WORDS ..
```

```

(1)
(1)
(1)
1686 012202 000167 177732          JMP      RSULV          ;EXIT IF ERROR
1687
1688          ;RCVR REJECT INTERRUPT ROUTINE
1689
1690 012206          RRJINT: BDINIT  RCVR          ;CLEAR HARDWARE
1691 012214 012767 100000 005460      MOV      #B15,RCVEV      ;SET RCVR EVENT FLAG
1692 012222          REGRES          ;RESTORE R0...R5
1693 012226 000002          RTI          ;RETURN
1694
1695          ;KW11 CLOCK INTERRUPT ROUTINE
1696          ;UPDATE TICKS. IF = 60. UPDATE SECONDS.
1697          ;IF SECS = 60, UPDATE MINUTS. IF MINUTS - 60
1698          ;UPDATE HOURS.
1699
1700 012230 042777 000200 003712  CLKINT: BIC      #B07,@LCS      ;JUST IN CASE IT MATTERS.
1701 012236 105267 005400          INCB      TICKS          ;UPDATE TICKS
1702 012242 126727 005374 000074      CMPB     TICKS,#60.      ;GET TO 60 YET?
1703 012250 103424          BLO      CLKEXT          ;NO
1704 012252 105067 005364          CLRB      TICKS          ;YES, CLR TICKS
1705 012256 105267 005361          INCB      SECNDS         ;& UPDATE SECONDS
1706 012262 126727 005355 000074      CMPB     SECNDS,#60.      ;GET 60 SECS?
1707 012270 103414          BLO      CLKEXT          ;NO
1708 012272 105067 005345          CLRB      SECNDS         ;YES, CLR SECNDS
1709 012276 105267 005342          INCB      MINUTS         ;& UPDATE MINUTES
1710 012302 126727 005336 000074      CMPB     MINUTS,#60.      ;GET 60 MINUTES?
1711 012310 103404          BLO      CLKEXT          ;NO
1712 012312 105067 005326          CLRB      MINUTS         ;YES, CLEAR MINUTS
1713 012316 005267 005324          INC      HOURS          ;& UPDATE HOURS
1714 012322 000002          CLKEXT: RTI          ;RETURN

```

```
1716 .SBTTL STATUS MODULE
1717
1718 ; DUMP STATUS TABLE INDICATING ACTIVATED RECEIVERS
1719 ; SHOW NO. OF ATTEMPTED CONNECTIONS TO EACH ACTIVE RECEIVER
1720 ; AND NO. OF SUCCESSFUL CONNECTIONS TO EACH ACTIVE RECEIVER.
1721 ; QUANTITIES ARE IN DECIMAL AND ARE CAPABLE OF EXPANDING TO
1722 ; 655,359,999. ANY QUANTITY HIGHER THAN THAT AMOUNT WILL BE
1723 ; REPRESENTED AS '*****'. RECEIVER ADDRESS NUMBERS ARE
1724 ; PRINTED IN OCTAL
1725 ;
1726 ;
1727 ;
1728 012324 PROC STATUS
(1)
(1) 012324 STATUS: ;**ENTRY POINT**
1729 012324 005767 004274 TST TOO ;IS TTY OUT QUEUE EMPTY?
1730 012330 001402 BEQ 60$
1731 012332 000167 000540 JMP STRTN ;NO, RETURN TO WHEREVER
1732 012336 005767 005326 60$: TST S:PNTR ;IS OUTPUT POINTER AT TABLE?
1733 012342 001504 BEQ STHDR ;NO, GO PRINT HEADER
1734 012344 026727 005320 020664 STCHK: CMP STPNTR,#RADEND ;IS POINTER AT END OF TABLE?
1735 012352 002402 BLT 59$ ;YES, DONE; EXIT
1736 012354 000167 000476 JMP STARXT
1737 012360 005777 005304 59$: TST @STPNTR ;CHECK RCV ADDR ACTIVE FLAG
1738 012364 001002 BNE 58$
1739 012366 000167 000452 JMP STBADV ;ADVANCE TABLE IF FLAG CLR
1740 012372 012700 032054 58$: MOV #STLIN1,R0 ;OUTPUT POINTER FOR OCTJSP
1741 012376 062767 000002 005264 ADD #2,STPNTR
1742 012404 017701 005260 MOV @STPNTR,R1 ;OCTJSP DUMPS DATA IN STPNTR
1743 012410 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1744 012414 004767 020262 JSR PC,OCTJSP
1745 012420 062767 000002 005242 ADD #2,STPNTR
1746 012426 012700 032076 MOV #STLIN2,R0 ;OUTPUT POINTER FOR CDDMG
1747 012432 016701 005232 MOV STPNTR,R1 ;OUTPUT WORD FOR CDDMG
1748 012436 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1749 012442 004767 020434 JSR PC,CDDMG
1750 012446 020027 032123 STYHR: CMP R0,#STLIN2+25 ;BLANK REST OF FIELD OUT
1751 012452 103003 BHIS STYON ;BY INSERTING SPACES
1752 012454 112720 000040 MOVB #' ,(R0)+
1753 012460 000772 BR STYHR
1754 012462 062767 000004 005200 STYON: ADD #4,STPNTR
1755 012470 012700 032124 MOV #STLIN3,R0 ;OUTPUT POINTER FOR CDDMG
1756 012474 016701 005170 MOV STPNTR,R1 ;OUTPUT WORD FOR CDDMG
1757 012500 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1758 012504 004767 020372 JSR PC,CDDMG
1759 012510 020027 032140 STYTHR: CMP R0,#STLIN3+14 ;BLANK REST OF FIELD OUT
1760 012514 103003 BHIS STYONT ;BY INSERTING SPACES
1761 012516 112720 000040 MOVB #' ,(R0)+
1762 012522 000772 BR STYTHR
1763 012524 STYONT: CALL PNTLIN,<STLIN> ;ENQUEUE STATUS LINE FOR OUTPUT
1764 012542 062767 000004 005120 ADD #4,STPNTR ;UPDATE POINTER FOR NEXT LINE
1765 012550 000167 000322 JMP STRTN ;RETURN
1766
1767 012554 132777 000001 003330 STHDR: BITB #1,@TMMRH ;IS MASTER SET HERE?
1768 012562 001407 BEQ 1$ ;NO.
1769 012564 CALL PNTLIN,<THUMST> ;YES, TELL OPERATOR
```

1770	012602	132777	000002	003302	1\$:	BITB	#2,@TMMRH	;WELL, IS SECONDARY SET?
1771	012610	001407				BEQ	2\$;NO.
1772	012612					CALL	PNTLIN,<THUSCN>	;YES, TELL OPERATOR
1773	012630	032767	100000	174542	2\$:	BIT	#B15,TXMST	;IS 'RIB' SET?
1774	012636	001410				BEQ	3\$;NO, TELL OPERATOR
1775	012640					CALL	PNTLIN,<RBSTMG>	;YES, TELL OPERATOR
1776	012656	000407				BR	4\$	
1777	012660				3\$:	CALL	PNTLIN,<RBCLMG>	
1778	012676	005767	004746		4\$:	TST	KWFLG	;GOT A CLOCK?
1779	012702	001444				BEQ	5\$;NO, FORGET ABOUT TIME.
1780	012704	005002				CLR	R2	;SUPPRESS LEADING ZEROS
1781	012706	012700	031741			MOV	#TMLIN1,R0	;OUTPUT POINTER FOR DECPNT
1782	012712	016701	004730			MOV	HOURS,R1	;OUTPUT DATA FOR DECPNT
1783	012716	004767	020002			JSR	PC,DECPNT	;ENQUEUE 'HOURS'
1784	012722	005002				CLR	R2	;SUPPRESS LEADING ZEROS
1785	012724	112720	000072			MOVB	#':,(R0)+	;INSERT COLON
1786	012730	116701	004710			MOVB	MINUTS,R1	;GET 'MINUTES' FOR DECPNT
1787	012734	004767	017764			JSR	PC,DECPNT	;ENQUEUE 'MINUTES'
1788	012740	005002				CLR	R2	;SUPPRESS LEADING ZEROS
1789	012742	112720	000072			MOVB	#':,(R0)+	;INSERT ANOTHER COLON
1790	012746	116701	004671			MOVB	SECNDS,R1	;GET 'SECONDS' FOR DECPNT
1791	012752	004767	017746			JSR	PC,DECPNT	;ENQUEUE 'SECONDS'
1792	012756	005002				CLR	R2	;SUPPRESS LEADING ZEROS
1793	012760	112720	000072			MOVB	#':,(R0)+	;INSERT ANOTHER COLON
1794	012764	116701	004652			MOVB	TICKS,R1	;GET 'TICKS' FOR DECPNT
1795	012770	004767	017730			JSR	PC,DECPNT	;ENQUEUE 'TICKS'
1796	012774	105010				CLRB	(R0)	;ENQUEUE 'O' PRINT TERMINATOR.
1797	012776					CALL	PNTLIN,<ELPSTM>	;PRINT 'ELAPSED TIME'
1798	013014				5\$:	CALL	PNTLIN,<STITLE>	;ENQUEUE STATUS HEADER
1799	013032	012767	020100	004630		MOV	#RADB,STPNTR	;SET POINTER TO TOP OF STAT LIST
1800	013040	000167	000032			JMP	STRN	;RETURN
1801								
1802	013044	062767	000014	004616	STBADV:	ADD	#14,STPNTR	;SET POINTER TO NEXT ACTIVE FLAG
1803	013052	000167	000020			JMP	STRN	
1804								
1805	013056	005067	004604		STARXT:	CLR	STSEV	;CLR STATUS EVENT FLAG
1806	013062					CALL	PRESC	;FAKE CNTRL-C KEY
1807	013076				STRTN:	RETURN		;RETURN

```
1809 .SBTTL TRANSMITTER ERRORS MODULE
1810
1811 ;DUMP A TABLE OF TRANSMITTER ERRORS INCLUDING:
1812 ; ERROR NO., CONCT'D RCVR, AND ERROR COUNT FOR
1813 ; EACH OF THE TRANSMITTER ERRORS.
1814 ;IF THE ERROR COUNT FOR ANY ENTRY IS 0, THAT ERROR NO. IS NOT REPERTED
1815 ;IF, AFTER CHECKING THE ENTIRE XMTR ERROR TABLE, NO ERRORS
1816 ; ARE PRINTED, SIMPLY PRINT (NONE).
1817
1818
1819 013100 PROC TEROS
1820 (1)
1821 (1) 013100 TEROS: ;**ENTRY POINT**
1822 013100 005767 003520 TST TOQ ;IS THE TTY OUTPUT QUEUE EMPTY?
1823 013104 001402 BEQ 60$ ;YES, CONTINUE
1824 013106 000167 000324 JMP TERTN ;NO, EXIT
1825 013112 005767 004600 60$: TST HDRFLG ;HEADER PRINTED YET?
1826 013116 001422 BEQ 1$ ;YES, SKIP IT
1827 013120 005067 004572 CLR HDRFLG ;NO, PRINT IT NOW.
1828 013124 CALL PNTLIN,<TERHDR>
1829 013142 CALL PNTLIN,<TRHLIN>
1830 013160 000167 000252 JMP TERTN ;AND RETURN
1831 013164 026727 004600 000016 1$: CMP ERRO,#16 ;IS THIS ERROR # 16?
1832 013172 001042 BNE 3$ ;NOT YET.
1833 013174 005067 004452 CLR TEREV ;YES, CLEAR XMTR ERROR EVENT
1834 013200 052767 100000 004446 BIS #B15,REREV ;SET RCVR ERROR EVENT
1835 013206 052767 100000 004502 BIS #B15,HDRFLG ;SET HEADER FLAG
1836 013214 012767 000001 004550 MOV #1,ERR1 ;LOAD INITIAL XMTR #
1837 013222 012767 026616 004544 MOV #RERTBL,ERR2 ;POINT RCVR ERR EVENT AT ITS TABLE
1838 013230 005767 004460 TST PRINTD ;DID WE PRINT ANYTHING?
1839 013234 001007 BNE 2$ ;YES, O.K.
1840 013236 CALL PNTLIN,<NONMG> ;NO, PRINT '(NONE)'
1841 013254 2$: CALL PNCRLF ;NEW LINE
1842 013270 005067 004420 CLR PRINTD
1843 013274 000167 000136 JMP TERTN ;RETURN
1844 013300 005777 004470 3$: TST @ERR2 ;ANY ERRORS?
1845 013304 001434 BEQ 4$ ;NO.
1846 013306 012700 032362 MOV #TRRNO,R0 ;READY TO PRINT ERROR NO.
1847 013312 016701 004452 MOV ERRO,R1
1848 013316 004767 017360 JSR PC,OCTJSP ;ERROR NO. IS IN OCTAL
1849 013322 012700 032377 MOV #TRRCN,R0 ;READY TO PRINT RCVR NO.
1850 013326 016701 004440 MOV ERR1,R1
1851 013332 004767 017344 JSR PC,OCTJSP ;RCVR NO. IS ALSO IN OCTAL
1852 013336 012700 032416 MOV #TRERC,R0 ;NOW PRINT ERROR COUNT
1853 013342 017701 004426 MOV @ERR2,R1
1854 013346 004767 017344 JSR PC,DECJSP ;ERROR COUNT IS IN DECIMAL
1855 013352 CALL PNTLIN,<TRELIN> ;ENQUEUE TABLE OUTPUT LINE
1856 013370 052767 100000 004316 BIS #B15,PRINTD ;SET 'PRINTED FLAG'
1857 013376 062767 000002 004370 4$: ADD #2,ERR2 ;UPDATE TABLE POINTER
1858 013404 026727 004362 000037 CMP ERR1,#37 ;ALL RCVR'S DONE?
1859 013412 001404 BEQ 5$ ;YES.
1860 013414 005267 004352 INC ERR1 ;NO, DO NEXT
1861 013420 000167 000012 JMP TERTN
1862 013424 012767 000001 004340 5$: MOV #1,ERR1 ;NEXT ERROR NO.
1863 013432 005267 004332 INC ERRO
1864 013436 TERTN: RETURN
```

```
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873 013440
(1)
(1) 013440
1874 013440 005767 003160
1875 013444 001402
1876 013446 000167 000310
1877 013452 005767 004240
1878 013456 001422
1879 013460 005067 004232
1880 013464
1881 013502
1882 013520 000167 000236
1883 013524 026727 004240 000031 1$:
1884 013532 001034
1885 013534 005067 004114
1886 013540 005767 004150
1887 013544 001007
1888 013546
1889 013564
1890 013600 005067 004110
1891 013604
1892 013620 000167 000136
1893 013624 005777 004144
1894 013630 001434
1895 013632 012700 032517
1896 013636 016701 004126
1897 013642 004767 017034
1898 013646 012700 032534
1899 013652 016701 004114
1900 013656 004767 017020
1901 013662 012700 032553
1902 013666 017701 004102
1903 013672 004767 017020
1904 013676
1905 013714 052767 100000 003772
1906 013722 062767 000002 004044 4$:
1907 013730 026727 004036 000037
1908 013736 001404
1909 013740 005267 004026
1910 013744 000167 000012
1911 013750 012767 000001 004014 5$:
1912 013756 005267 004006
1913 013762
```

PROC REROS

REROS:

60\$:

1\$:

2\$:

3\$:

4\$:

5\$:

RERTN: RETURN

RECEIVER ERRORS MODULE

DUMP A TABLE OF RECEIVER ERRORS, INCLUDING:
ERROR NO., CONCT'D XMTR, AND ERROR COUNT
FOR EACH OF THE RECEIVER ERRORS.
IF THE ERROR COUNT FOR ANY ENTRY IS 0, THAT ERROR NO. IS NOT
REPORTED. IF, AFTER CHECKING THE ENTIRE RCVR ERROR TABLE, NO
ERRORS ARE PRINTED, SIMPLY PRINT '(NONE)'

ENTRY POINT

IS TTY OUTPUT QUEUE EMPTY?
YES, CONTINUE
NO, RETURN
IS HEADER PRINTED YET?
YES, SKIP IT
NO, CLEAR FLAG
AND PRINT HEADER
AND RETURN
DONE LAST ERROR?
NOT YET
YES, CLEAR RCVR ERROR EVENT
DID WE PRINT ANYTHING?
YES, O.K.
NO, PRINT '(NONE)'

WRAP IT UP!

ANY ERRORS?
NO.
READY TO PRINT ERROR NO.
ERROR NO.S ARE IN OCTAL
READY TO PRINT XMTR NO.
XMTR NO.S ARE IN OCTAL
NOW PRINT ERROR COUNT
ERROR COUNTS ARE IN DECIMAL
ENQUEUE TABLE OUTPUT LINE
SET 'PRINTED' FLAG
UPDATE TABLE POINTER
ALL XMTRS DONE?
NO, DO NEXT
YES, NEXT ERROR

```
1915 .SBTTL SUMMARY MODULE
1916
1917 ; DUMP SUMMARY TABLE INDICATING ERROR NUMBERS AND THE LOCATIONS
1918 ; OF THE ERRORS AS WELL AS THE NUMBER OF OCCURRENCES OF EACH OF
1919 ; THE ERRORS. ERRORS WITH 0 OCCURRENCES WILL NOT BE LISTED. THE
1920 ; ENTRY UNDER 'NO. OF OCCURRENCES' IS IN DECIMAL AND IS CAPABLE
1921 ; OF EXPANDING TO 65,528. QUANTITIES OF ERRORS IN EXCESS OF
1922 ; THIS WILL BE LISTED AS 'MXCNT'.
1923 ;
1924
1925
1926 013764 PROC SUMRY
1927 (1)
1928 (1) 013764 SUMRY: ;**ENTRY POINT**
1929 013764 005767 002634 TST TOQ ;IS OUTPUT QUEUE EMPTY?
1930 013770 001126 BNE SMRTN ;NO, RETURN
1931 013772 005767 003674 TST SUMPNT ;IS OUTPUT POINTER AT TABLE?
1932 013776 001472 BEQ SUMHDR ;NO, GO PRINT HEADER
1933 014000 027727 003666 177777 SMCHK: CMP @SUMPNT,#-1 ;IS ERROR # = -1 ?
1934 014006 001507 BEQ SMARXT ;YES,DONE; EXIT
1935 014010 016704 003656 MOV SUMPNT,R4
1936 014014 005764 000004 TST 4(R4) ;CHECK TOTAL COUNT OF THIS ERR
1937 014020 001475 BEQ SMADV ;IF 0, TRY NEXT ERROR
1938 014022 012700 032574 MOV #SMLIN1,R0 ;R0 IS OUTPUT POINTER FOR OCTJSP
1939 014026 011401 MOV (R4),R1 ;R1 IS DATA BUFF FOR OCTJSP
1940 014030 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1941 014034 004767 016642 JSR PC,OCTJSP
1942 014040 012700 032616 MOV #SMLIN2,R0
1943 014044 016401 000002 MOV 2(R4),R1
1944 014050 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1945 014054 004767 016622 JSR PC,OCTJSP
1946 014060 012700 032642 MOV #SMLIN3,R0
1947 014064 016401 000004 MOV 4(R4),R1
1948 014070 020127 177770 CMP R1,#177770 ;REACHED MAX COUNT?
1949 014074 101415 BLOS SMPN ;NO,OK
1950 014076 112720 000115 MOVB #'M,(R0)+ ;YES, PRINT 'MXCNT' IN
1951 014102 112720 000130 MOVB #'X,(R0)+ ; NOTE THAT 'MXCNT'
1952 014106 112720 000103 MOVB #'C,(R0)+ ; ALSO HAS ONLY 5
1953 014112 112720 000116 MOVB #'N,(R0)+ ; ASCII CHARACTERS.
1954 014116 112720 000124 MOVB #'T,(R0)+
1955 014122 000404 BR SMSPN
1956 014124 012702 177777 MOV #-1,R2 ;DON'T COMPRESS BLANKS
1957 014130 004767 016562 SMPN: JSR PC,DECJSP
1958 014134 062767 000006 003512 SMSPN: CALL PNTLIN,<SMLIN> ;ENQUEUE SUM LINE FOR OUTPUT
1959 014152 000167 000062 ADD #6,SUMPNT ;UPDATE POINTER FOR NEXT LINE
1960 014160 000167 000062 JMP SMRTN ;RETURN
1961 014164 SUMHDR: CALL PNTLIN,<SMITLE> ;ENQUEUE SUMMARY HEADER
1962 014202 012767 024726 003462 MOV #ERTBL,SUMPNT ;SET POINTER TO TOP OF SUM LIST
1963 014210 000167 000032 JMP SMRTN ;RETURN
1964 014214 062767 000006 003450 SMADV: ADD #6,SUMPNT ;SET POINTER TO NEXT ENTRY
1965 014222 000167 177552 JMP SMCHK
1966
1967 014226 005067 003430 SMARXT: CLR SUMEV ;CLEAR SUMMARY EVENT FLAG
1968 014232 CALL PRESC ;FAKE CNTRL-C KEY
```

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

MACY11 30A(1052) 20-OCT-78 09:48
SUMMARY MODULE

N 7

PAGE 31-1

SEQ 0091

1969 014246

SMRTN: RETURN

;RETURN


```
1971 .SBTTL ERROR UPDATE ROUTINES
1972 .SBTTL TRANSMITTER ERRORS
1973
1974 ;CALLED BY THE MACRO 'CALL ERRMOD,<P,ERRADR>'
1975
1976 014250 PROC ERRMOD,<ERRNUM,ERRADR>
(3) ERRNUM = 2
(3) ERRADR = 4
(1)
(1) 014250 ERRMOD: ;**ENTRY POINT**
1977 014250 010046 MOV R0,-(SP) ;SAVE R0 ON STACK
1978 014252 010146 MOV R1,-(SP) ; AND R1
1979 014254 010246 MOV R2,-(SP) ; AND R2
1980 014256 010346 MOV R3,-(SP) ;AND R3
1981 014260 017767 001612 003476 MOV @TCR,TEMP ;SAVE ADDRESS OF CONNECTED RCVR
1982 014266 042767 160377 003470 BIC #160377,TEMP
1983 014274 000367 003464 SWAB TEMP ;GET IT IN THE RIGHT BYTE
1984 014300 016500 000002 MOV ERRNUM(R5),R0 ;GET ERROR NUMBER
1985 014304 010067 003456 MOV R0,TEMP1 ;SAVE ERROR NUMBER
1986 014310 006300 ASL R0 ;FIND TABLE ENTRY
1987 014312 010001 MOV R0,R1 ; IN ORDER TO UPDATE
1988 014314 006300 ASL R0 ; ADDRESS FIELD
1989 014316 060100 ADD R1,R0
1990 014320 062700 024720 ADD #ERTBL0,R0
1991 014324 016560 000004 000002 MOV ERRADR(R5),2(R0) ;ENTER ERROR ADDRESS
1992 014332 026027 000004 177771 CMP 4(R0),#177771 ;AT MAX COUNT YET?
1993 014340 103401 BLO ERMIS ;NO, INCREMENT COUNT
1994 014342 000444 BR ERMIS ;YES SKIP UPDATE.
1995 014344 005260 000004 ERMIS: INC 4(R0) ;UPDATE OCCURRENCES
1996 014350 005367 003412 DEC TEMP1 ; (ERR # - 1)
1997 014354 MULT 37,TEMP1,R3 ; (E-1)x37
1998 014420 MULT 2,TEMP1,R3 ; [(E-1)x37]x2
1999 014424 005367 003334 DEC TEMP ; RCVR ADDR - 1
2000 014430 MULT 2,TEMP,R3 ; (R-1)x2
2001 014434 066767 003326 003322 ADD TEMP1,TEMP ; [(E-1)x37]x2 + (R-1)x2
2002 014442 062767 025150 003314 ADD #TERTBL,TEMP ;TEMP = #TERTBL + <[(E-1)x37]x2 + (R-1)x2>
2003 014450 005277 003310 INC @TEMP ;UPDATE TABLE ENTRY FOR THIS ERROR
2004 014454 012603 ERMIS: MOV (SP)+,R3 ;RESTORE R3
2005 014456 012602 MOV (SP)+,R2 ;RESTORE R2
2006 014460 012601 MOV (SP)+,R1 ;RESTORE R1
2007 014462 012600 MOV (SP)+,R0 ;RESTORE R0
2008 014464 RETURN ;RETURN
```

```
2010 .SBTTL RECEIVER ERRORS
2011
2012 ;CALLED BY THE MACRO 'CALL ERRMOR,<P,ERRADR>'
2013
2014 014466 PROC ERRMOR,<ERRNUM,ERRADR>
(3) 000002 ERRNUM = 2
(3) 000004 ERRADR = 4
(1)
(1) 014466 ERRMOR: ;**ENTRY POINT**
2015 014466 010046 MOV R0,-(SP) ;SAVE R0 ON THE STACK
2016 014470 010146 MOV R1,-(SP) ;AND R1
2017 014472 010246 MOV R2,-(SP) ;AND R2
2018 014474 010346 MOV R3,-(SP) ;AND R3
2019 014476 017767 001414 003260 MOV @RCR,TEMP ;GET ADDR OF CONNECTED XMTR
2020 014504 042767 160377 003252 BIC #160377,TEMP
2021 014512 000367 003246 SWAB TEMP ; INTO RIGHT BYTE
2022 014516 016500 000002 MOV ERRNUM(R5),R0 ;GET ERROR NUMBER
2023 014522 010067 003240 MOV R0,TEMP1 ;AND SAVE IT
2024 014526 006300 ASL R0 ; FIND 'SUM' TABLE ENTRY
2025 014530 010001 MOV R0,R1 ; IN ORDER TO UPDATE
2026 014532 006300 ASL R0 ; ADDRESS FIELD
2027 014534 060100 ADD R1,R0 ; AND # OF OCCURRENCES
2028 014536 062700 024720 ADD #ERTBL0,R0
2029 014542 016560 000004 000002 MOV ERRADR(R5),2(R0) ;ENTER ERROR ADDRESS
2030 014550 026027 000004 177771 CMP 4(R0),#177771 ;AT MAX COUNT YET?
2031 014556 103401 BLO ERMINR ;NO, UPDATE ERROR COUNT
2032 014560 000447 BR ERMISR ;YES, SKIP UPDATES
2033 014562 005260 000004 ERMINR: INC 4(R0) ;UPDATE OCCURRENCES
2034 014566 162767 000015 003172 SUB #15,TEMP1 ;ALIGN ERRORS TABLE FOR RCVR
2035 014574 005367 003166 DEC TEMP1 ; ERR # - 1 (E-1)
2036 014600 MULT 37,TEMP1,R3 ; (E-1)X37
2037 014644 MULT 2,TEMP1,R3 ; [(E-1)X37]X2
2038 014650 005367 003110 DEC TEMP ; XMTR ADDR - 1 (T-1)
2039 014654 MULT 2,TEMP,R3 ; (T-1)X2
2040 014660 066767 003102 003076 ADD TEMP1,TEMP ; [(E-1)X37]X2 + (T-1)X2
2041 014666 062767 026616 003070 ADD #RERTBL,TEMP ; TEMP = #RERTBL+[(E-1)X37]X2 + (T-1)X2
2042 014674 005277 003064 INC @TEMP ;UPDATE RCVR ERROR COUNT
2043 014700 012603 ERMISR: MOV (SP)+,R3 ;RESTORE R3
2044 014702 012602 MOV (SP)+,R2 ;RESTORE R2
2045 014704 012601 MOV (SP)+,R1 ;RESTORE R1
2046 014706 012600 MOV (SP)+,R0 ;RESTORE R0
2047 014710 RETURN
```

```
2049 .SBTTL UTILITY ROUTINES
2050 .SBTTL PROCESS AN INPUT CHARACTER FROM THE TTY
2051
2052 014712 PROC TTINP
(1)
(1) 014712 TTINP: ;**ENTRY POINT**
2053
2054 014712 CALL DEQ,<FRELEM,TIQ> ;GET TTY INPUT CHARACTER.
2055 014732 142767 000200 013324 BICB #200,FRELEM ;CLEAR OFF PARITY BIT.
2056 014740 116700 013320 MOVB FRELEM,R0
2057 014744 120027 000141 CMPB R0,#141 ;CONVERT LOWER CASE INPUT
2058 014750 002405 BLT TTINH ; TO REGULAR (UPPER CASE)
2059 014752 120027 000172 CMPB R0,#172 ; LETTERS INSTEAD.
2060 014756 003002 BGT TTINH
2061 014760 042700 000040 BIC #B05,R0 ;CONVERT CHARACTER
2062 014764 010067 013274 TTINH: MOV R0,FRELEM ; IN 'FRELEM'
2063 014770 012701 016034 MOV #CMCHTB,R1 ; TO ITS PROCESSING
2064 014774 004767 015654 JSR PC,PROCHR ; ROUTINE.
2065 015000 001407 BEQ CMMISC ;IF MISC CHAR, CALL MISC PROC.
2066 015002 CALL @R1,<FRELEM> ;CALL PROCESSING ROUTINE.
2067 015016 000407 BR TTRET ;RETURN.
2068
2069 015020 CMMISC: CALL PRMISC,<FRELEM> ;PROCESS MISCELLANEOUS CHARACTER
2070 015036 TTRET: RETURN ;RETURN.
```

```
2072 .SBTTL TTY INPUT CHARACTER PROCESSING ROUTINES
2073
2074 ;
2075 ; PROCESSOR FOR CARRIAGE RETURN
2076 ; THIS CHARACTER SIGNIFIES THE END OF A COMMAND LINE.
2077 ;
2078 015040 PROC PRCR
2079 (1)
2080 (1) 015040 PRCR: ;**ENTRY POINT**
2081 105767 002574 TSTB REQINP ;IF INPUT NOT BEING TYPED,
2082 015044 002015 BGE CRRET ; IGNORE CHARACTER.
2083 015046 112777 000015 001326 MOVB #CR,,@CBUFPT ;PUT CR IN BUFFER.
2084 015054 105367 002561 DECB CMDENT ;SET COMMAND ENTERED FLAG.
2085 015060 105067 002554 CLRB REQINP ;CLEAR INPUT REQUESTED FLAG.
2086 015100 CALL PNCRLF ;ECHO CR & LF.
2087 ;
2088 ; PROCESSOR FOR CONTROL-O CHARACTER.
2089 ; ALL CHARACTERS PRESENTLY IN THE TTY OUTPUT QUEUE ARE THROWN AWAY.
2090 ;
2091 015102 PROC PRCTLO
2092 (1)
2093 (1) 015102 PRCTLO: ;**ENTRY POINT**
2094 015106 005067 001516 CLR TOQ ;THROW AWAY ALL CHARACTERS
2095 015114 016767 001524 001520 MOV TOQ+QBACK,TOQ+QFRONT ; IN TTY OUTPUT QUEUE.
2096 015132 CALL PNTLIN,<CTLOMG> ;ECHO '^O'.
2097 ;
2098 ;
2099 ; PROCESSOR FOR CONTROL S.
2100 ; TEMPORARILY SUSPEND PRINTOUT ON CONSOLE DEVICE
2101 ;
2102 ; PRINTOUT WILL BE RESUMED UPON RECEIPT OF A CNTRL-Q
2103 ; OR ANY OTHER KEYBOARD INTERRUPT.
2104 ;
2105 015134 PROC PRCTLS
2106 (1)
2107 (1) 015134 PRCTLS: ;**ENTRY POINT**
2108 015142 052767 100000 002560 BIS #B15,THLTFL ;SET TTY HALT FLAG
2109 ;
2110 ;
2111 ; PROCESSOR FOR CONTROL Q.
2112 ; RESUME PRINTOUT ON CONSOLE IF TTOUT QUEUE HAS NOT BEEN
2113 ; CLEARED BY SOME OTHER MEANS.
2114 ;
2115 015144 PROC PRCTLQ
2116 (1)
2117 (1) 015144 PRCTLQ: ;**ENTRY POINT**
2118 015152 042767 100000 002550 BIC #B15,THLTFL ;CLEAR TTY HALT FLAG
2119 ;
```

```
2120
2121
2122      :PROCESSOR FOR CONTROL-C.
2123      :ECHO '^C' AND GO TO PRESC ROUTINE.
2124
2125      015154      PROC      PRCTL C
2126      (1)
2127      (1) 015154      PRCTL C:      : **ENTRY POINT**
2128      015154      CALL      PNTLIN,<CTL CMG>      :ECHO '^C'
2129      015172 000167 000016      JMP      PRESC      :GO TO PRESC ROUTINE.
2130
2131      :
2132      :PROCESSOR FOR CONTROL-U.
2133      :INPUT COMMAND BEING TYPED IS DISCARDED.
2134
2135      015176      PROC      PRCTL U
2136      (1)
2137      (1) 015176      PRCTL U:      : **ENTRY POINT**
2138      015176      CALL      PNTLIN,<CTL UMG>      :ECHO '^U'.
2139      :DROP INTO PRESC
2140
2141      :
2142      :PROCESSOR FOR CNTRL-C KEY.
2143      :THIS KEY IS USED TO REQUEST INPUT. ALL OTHER EVENTS WILL CEASE
2144      :AFTER THE CURRENT LINE. A 'PCL>' WILL BE ECHOED ON THE TTY. THE
2145      :OPERATOR IS NOW FREE TO TYPE IN A COMMAND.
2146
2147      015214      PROC      PRESC
2148      (1)
2149      (1) 015214      PRESC:      : **ENTRY POINT**
2150      005767 002430      TST      KWFLG      :GOT A CLOCK?
2151      015220 001402      BEQ      14$      :NO.
2152      015222 005077 000722      CLR      @LCS      :STOP THE CLOCK
2153      015226 105367 002406      14$: DEC B REQINP      :SET INPUT REQUESTED FLAG.
2154      015232 005067 002430      CLR      STSEV      :CLR STATUS EVENT
2155      015236 005067 002420      CLR      SUMEV      :CLR SUMMARY EVENT
2156      015242 005067 002404      CLR      TEREV      :CLR XMTR ERROR EVENT
2157      015246 005067 002402      CLR      REREV      :AND RCVR ERROR EVENT
2158      015252 005067 002422      CLR      PCLGO      :STOP XMTR
2159      015253      CALL      ENQ,<A.P,TOQ>      :ECHO 'PCL>'.
2160      015276      CALL      ENQ,<A.C,TOQ>
2161      015316      CALL      ENQ,<A.L,TOQ>
2162      015336      CALL      ENQ,<A.PR,TOQ>
2163      015356 012767 016176 001016      MOV      #CMDBUF,CBUFPT      :INITIALIZE COMMAND BUF PTR.
2164      015364 105067 002251      CLRB      CMDENT      :CLEAR COMMAND ENTERED FLAG.
2165      015370      RETURN      :RETURN.
2166
2167      :
2168      :PROCESS RUB OUT KEY.
2169      :LAST CHARACTER IN COMMAND BUFFER IS DELETED. A '^' IS ECHOED.
```

```
2170
2171 015372          PROC   PRDEL
(1)
(1) 015372          PRDEL:      ;**ENTRY POINT**
2172 015372 105767 002242      TSTB   REQINP      ;IF INPUT NOT REQUESTED,
2173 015376 002034              BGE     PDRET      ; THEN IGNORE RUBOUT CHAR.
2174 015400 026727 000776 0'6176  CMP    CBUFPT,#CMDBUF ;IF AT BEGINNING OF BUFFER,
2175 015406 003016              BGT     1$         ; THEN ISSUE NEW 'PCL>'
2176 015410              CALL    PNCRLF
2177 015424              CALL    PRESC
2178 015440 000167 000024      JMP     PDRET
2179 015444 005367 000732      1$:    DEC    CBUFPT      ;DECREMENT BUFFER POINTER.
2180 015450              CALL    ENQ,<A.BKSL,10Q>      ;ECHO A '\'.
2181 015470      PDRET:    RETURN      ;RETURN.
2182
2183
2184      :
2185      : PROCESSOR FOR MISCELLANEOUS CHARACTERS.
2186      : THE CHARACTER IS APPENDED TO THE COMMAND BUFFER.
2187      :
2188
2189
2190 015472          PROC   PRMISC,<CHAR>
(3) 000002          CHAR   -      2
(1)
(1) 015472          PRMISC:      ;**ENTRY POINT**
2191 015472 105767 002142      TSTB   REQINP      ;IF INPUT NOT REQUESTED,
2192 015476 002034              BGE     PMRET      ; THEN JUST IGNORE CHARACTER.
2193 015500 117500 000002      MOV     @CHAR(R5),R0
2194 015504 120027 000040      CMP     R0,#'
2195 015510 002403              BLT     PM176      ;IF CHARACTER
2196 015512 120027 000140      CMP     R0,#140      ; IS NONPRINTING
2197 015516 002402              BLT     PM187      ; THEN CHANGE
2198 015520 012700 000077      PM176: MOV    #'?,R0      ; IT TO
2199 015524 026727 000652 016401 PM187: CMP    CBUFPT,#CBFEND-1 ; ASCII '?'.
2200 015532 001416              BEQ     PMRET      ;IF WE ARE AT END OF BUFFER,
2201 015534 110077 000642      MOV     R0,@CBUFPT ; THEN JUST RETURN.
2202 015540 005267 000636      INC     CBUFPT      ;PUT CHARACTER INTO BUFFER
2203 015544 110067 012516      MOV     R0,BKELEM ;UPDATE BUFFER POINTER.
2204 015550              CALL    ENQ,<BKELEM,10Q>      ;ECHO CHARACTER.
2205 015570      PMRET:    RETURN      ;RETURN.
```

```

2207          .SBTTL  TTY OUTPUT HANDLERS
2208
2209
2210          :
2211          : OUTPUT A CHARACTER TO THE TELETYPE IF IT IS READY.
2212          :
2213 015572          PROC  TTOUT
2214          (1)
2215          (1) 015572          TTOUT:          ;**ENTRY POINT**
2216          015572 105777 000344          TSTB  @TTXCSR          ;IF DEVICE IS NOT READY,
2217          015576 002020          BGE  TORET          ; THEN JUST RETURN.
2218          015600 005767 002116          TST  THLTFL          ;IS TTY HALTED (CNTRL-S)?
2219          015604 001402          BEQ  1$          ;NO, O.K. TO PRINT
2220          015606 000167 000026          JMP  TORET          ;YES, DON'T DO ANYTHING.
2221          015612          1$:          CALL  DEQ,<FRELEM,TOQ>          ;GET NEXT CHAR TO TYPE.
2222          015632 116777 012426 000304          MOVB  FRELEM,@TTXBUF          ;OUTPUT IT.
2223          015640          TORET:          RETURN.          ;RETURN.
2224
2225          .SBTTL  TTY INPUT INTERRUPT PROCESSORS
2226
2227          TTIINT:          ;**INTERRUPT ENTRY POINT**
2228          015642 005067 002054          CLR  THLTFL          ;**CLEAR TTY HALT FLAG ON INPUT
2229          015646          REGSAV          ;**SAVE R0 - R5.
2230          015652 117767 000262 012414          MOVB  @TTXBUF,INTTMP          ;**GET INPUT CHARACTER.
2231          015660          CALL  ENQ,<INTTMP,TIQ>          ;**PUT IT IN TTY INPUT QUEUE.
2232          015700          REGRES          ;**RESTORE R0 - R5.
2233          015704 000002          RTI          ;**RETURN.
2234

```

```

2236          .SBTTL MESSAGE PRINT ROUTINE
2237
2238          :
2239          : ENQUEUE CHARACTERS STARTING AT MESSAG IN TTY OUTPUT
2240          : QUEUE TOO UNTIL A ZERO BYTE IS ENCOUNTERED. ENQUEUE A CR & LF
2241          : INSTEAD OF THE ZERO, & THEN EXIT FROM THE ROUTINE.
2242          :
2243
2244 015706      PROC      PNTLIN,<MESSAG>
(3)           000002      MESSAG - 2
(1)
2245 015706      PNTLIN:      ;**ENTRY POINT**
015706 016546 000002      MOV      MESSAG(R5),-(SP)      ;GET ADDRESS OF MESSAGE.
015712 117667 000000 012346 PLODEC: MOVB     @ (SP),BKELEM      ;GET A CHARACTER.
015720 001412      BEQ      PLCLRF      ;IF NULL, APPEND CR & LF.
015722      CALL      ENQ,<BKELEM,TOQ>      ;ENQUEUE CHAR FOR TTY OUTPUT.
015742 005216      INC      @SP      ;POINT TO NEXT CHARACTER.
015744 000762      BR      PLODEC      ;PROCESS IT.
2251
2252 015746 005026      PLCLRF: CLR      (SP)+      ;POP ADDR FROM STACK.
2253 015750 000400      BR      PL219      ;APPEND CR & LF.
2254
2255
2256          :
2257          : ENQUEUE A CR & LF IN TTY OUTPUT QUEUE.
2258          :
2259 015752      PROC      PNCRLF
(1)
(1) 015752      PNCRLF:      ;**ENTRY POINT**
015752      PL219: CALL      ENQ,<A.CR,TOQ>      ;ENQUEUE A CR.
015772      CALL      ENQ,<A.LF,TOQ>      ;ENQUEUE A LF.
016012      CALL      ENQ,<ZERO,TOQ>      ;ENQUEUE A NULL FILL CHAR.
2263 016032      RETURN      ;RETURN.

```



```
2265 .SBTTL DATA AREAS
2266
2267 ; TABLE ASSOCIATING SPECIAL TTY INPUT CHARACTERS WITH THEIR
2268 ; PROCESSING ROUTINES.
2269 CMCHTB:
2270 016034 000003 015154 .WORD CTL.C,PRCTL
2271 016040 000012 015040 A.LF: .WORD LF.,PRCR
2272 016044 000015 015040 A.CR: .WORD CR.,PRCR
2273 016050 000017 015102 .WORD CTL.O,PRCTLO
2274 016054 000021 015144 .WORD CTL.Q,PRCTLQ
2275 016060 000023 015134 .WORD CTL.S,PRCTLS
2276 016064 000025 015176 .WORD CTL.U,PRCTLU
2277 016070 000177 015372 .WORD RUBOUT,PRDEL
2278 016074 177777 .WORD -1
2279
2280 ;DEVICE ADDRESS TABLES:
2281
2282
2283 016076 164200 TCR: .WORD PCLTXM
2284 016100 164202 TSR: .WORD PCLTXM+2
2285 016102 164204 TSDB: .WORD PCLTXM+4
2286 016104 164206 TSBC: .WORD PCLTXM+6
2287 016106 164210 TSBA: .WORD PCLTXM+10
2288 016110 164212 TMMR: .WORD PCLTXM+12
2289 016112 164213 TMMRH: .WORD PCLTXM+13
2290 016114 164214 TSCRC: .WORD PCLTXM+14
2291 016116 164220 RCR: .WORD PCLRCV
2292 016120 164222 RSR: .WORD PCLRCV+2
2293 016122 164224 RDDB: .WORD PCLRCV+4
2294 016124 164226 RDBC: .WORD PCLRCV+6
2295 016126 164230 RDBA: .WORD PCLRCV+10
2296 016130 164234 RDCRC: .WORD PCLRCV+14
2297
2298 016132 000170 TXMVEC: .WORD 170
2299 016134 000174 RCVVEC: .WORD 174
2300
2301 016136 177560 TTRCSR: .WORD TTDEV
2302 016140 177562 TTRBUF: .WORD TTDEV+2
2303 016142 177564 TTCSR: .WORD TTDEV+4
2304 016144 177566 TTBUF: .WORD TTDEV+6
2305 016146 000060 TTVECT: .WORD TTVCTR
2306 016150 177546 LCS: .WORD 177546
2307
2308 ; VECTOR ADDRESS.
2309 ; KW11-I. LINE CLOCK ADDR
2310
2311 ; CHARACTER CONSTANTS. THESE ARE DEFINED AS WORDS SO THAT THEY MAY
2312 ; BE ENQUEUED.
2313 A.BKSL: .WORD '\ ;ASCII '\
2314 ZERO: .WORD 0 ;ASCII NULL
2315 A.P: .WORD 'P ;ASCII 'P'
2316 A.C: .WORD 'C ;ASCII 'C'
2317 A.L: .WORD 'L ;ASCII 'L'
2318 A.PR: .WORD '> ;ASCII '>' (PROMPT)
2319
2320 .EVEN
2321 ;DEVICE ADDRESS AND VECTOR VARIABLES
```

```
2321 ;CHANGE THESE LOCATIONS TO MODIFY ALL DEVICE ADDRESSES AND VECTORS
2322 ;FOR PCL11.
2323
2324 016166 164200 TXDEV: .WORD 164200 ;DEFAULT IS 164200
2325 016170 000170 TXVEC: .WORD 170 ;DEFAULT IS 170
2326 016172 164220 RCDEV: .WORD 164220 ;DEFAULT IS 164220
2327 016174 000174 RCVEC: .WORD 174 ;DEFAULT IS 174
2328
2329 ; BUFFER & MESSAGE AREAS.
2330 016176 000204 CMDBUF: .BLKB 132. ;TTY INPUT COMMAND BUFFER.
2331 016402 016402 CBFEND =
2332 016402 033333 CBUFFP: .WORD 33333 ;CMDBUF BUFFER POINTER.
2333
2334 016404 047536 000 CTLOMG: .ASCIZ "'^O'" ;FOR ECHOING CONTROL CHARACTERS.
2335 016407 136 000125 CTLUMG: .ASCIZ "'^U'"
2336 016412 041536 000 CTLCMG: .ASCIZ "'^C'"
2337
2338 016415 040 051105 047522 RPMSG: .ASCIZ "' ERROR'"
2339 016422 000122
2340 ; QUEUE DEFINITIONS.
2341 .IRP LC <AO, TI, TO>
2342 .EVEN
2343 .LIST
2344 LC'Q: .WORD 0 ;QELEMS
2345 .WORD LC'SIZE ;QSIZE
2346 .WORD LC'AREA, LC'END ;QTOP & QBOT
2347 .WORD 33333, 33333 ;QFRONT & QBACK
2348 LC'AREA: .BLKW LC'SIZE ;LC'Q AREA
2349 LC'END =
2350 .NLIST
2351 .ENDM
2352 ACQ: .WORD 0 ;QELEMS
2353 .WORD AOSIZE ;QSIZE
2354 .WORD AOAREA, AOEND ;QTOP & QBOT
2355 .WORD 33333, 33333 ;QFRONT & QBACK
2356 AOAREA: .BLKW AOSIZE ;AOQ AREA
2357 AOEND =
2358 TIQ: .WORD 0 ;QELEMS
2359 .WORD TISIZE ;QSIZE
2360 .WORD TIAREA, TIEND ;QTOP & QBOT
2361 .WORD 33333, 33333 ;QFRONT & QBACK
2362 TIAREA: .BLKW TISIZE ;TIQ AREA
2363 TIEND =
2364 TOQ: .WORD 0 ;QELEMS
2365 .WORD TOSIZE ;QSIZE
2366 .WORD TOAREA, TOEND ;QTOP & QBOT
2367 .WORD 33333, 33333 ;QFRONT & QBACK
2368 TOAREA: .BLKW TOSIZE ;TOQ AREA
2369 TOEND =
2370
2371 ; FLAG VARIABLES.
2372 ; < 0 ==> TRUE
2373 ; >= 0 ==> FALSE
2374
2375 REQINP: .BYTE 333 ;INPUT REQUEST IS BEING TYPED.
2376 CMDENT: .BYTE 333 ;COMMAND HAS BEEN ENTERED.
```

2358	017642	000	TICKS: .BYTE	0
2359	017643	000	SECNDS: .BYTE	0
2360	017644	000	MINUTS: .BYTE	0
2361		017646		.EVEN
2362	017646	000000	HOURS: .WORD	0
2363	017650	000000	KWFLG: .WORD	0
2364	017652	000000	TEREV: .WORD	0
2365	017654	000000	RELEV: .WORD	0
2366	017656	000000	OLDEV: .WORD	0
2367	017660	000000	DATGEV: .WORD	0
2368	017662	000000	SUMEV: .WORD	0
2369	017664	000000	SUMSV: .WORD	0
2370	017666	000000	STSEV: .WORD	0
2371	017670	000000	STPNTR: .WORD	0
2372	017672	000000	SUMPNT: .WORD	0
2373	017674	000000	TXMEV: .WORD	0
2374	017676	000000	TXMSV: .WORD	0
2375	017700	000000	PCLGO: .WORD	0
2376	017702	000000	RCVEV: .WORD	0
2377	017704	000000	SPCEV: .WORD	0
2378	017706	000000	XSPCEV: .WORD	0
2379	017710	000000	FIRST: .WORD	0
2380	017712	000000	NEXT: .WORD	0
2381	017714	000000	PRINTD: .WORD	0
2382	017716	000000	HDRFLG: .WORD	0
2383	017720	000000	PADFLG: .WORD	0
2384	017722	000000	THLTFL: .WORD	0
2385				
2386			.EVEN	
2387				
2388			: DATA VARIABLES.	
2389	017724	133333	ORIGSD: .WORD	133333
2390	017726	000000	DTSEED: .WORD	0
2391	017730	000333	MLSD: .WORD	333
2392	017732	000000	MSGLS: .WORD	0
2393	017734	000000	RCSEED: .WORD	0
2394	017736	000000	RCTXPS: .WORD	0
2395	017740	000000	RJCTF: .WORD	0
2396	017742	000000	TRNKF: .WORD	0
2397	017744	000000	TXML: .WORD	0
2398	017746	037565	RANM: .WORD	37565
2399	017750	012247	RANK: .WORD	12247
2400	017752	000000	CURAD: .WORD	0
2401	017754	000000	RSPC: .WORD	0
2402	017756	000000	ESCFLG: .WORD	0
2403	017760	000000	OBJCNT: .WORD	0
2404	017762	000000	RSHOLD: .WORD	0
2405	017764	000000	TEMP: .WORD	0
2406	017766	000000	TEMP1: .WORD	0
2407	017770	000000	ERRO: .WORD	0
2408	017772	000000	ERR1: .WORD	0
2409	017774	000000	ERR2: .WORD	0
2410				
2411				
2412			: ADDRESS SILO DATA BUFFER AREA	
2413				

```
2414 017776 000102      ADSILO: .BLKB  66.          ;66. BYTE AREA FOR SILO DATA
2415
2416                      ;RECEIVER ADDRESS TABLE
2417
2418 020100 000001      X      =      1
2419                      RADB:
2420                      .REPT  31.
2421                      .WORD  0          ;ACTIVITY FLAG
2422                      .WORD  X          ;RECEIVER ADDRESS
2423                      .WORD  0,0       ;ATTEMPTS ENTRY
2424                      .WORD  0,0       ;SUCCESSSES ENTRY
2425
2426      X      -      X+1
2427                      .ENDR
2428 020664 020664      RADEND: .WORD  .
2429 020064 020064      RADBO= RADB-14
2430
2431                      ;TRANSMITTER DATA BUFFER:
2432
2433 020666 001020      DATBUF: .BLKW  1020
2434
2435                      ;RECEIVER DATA BUFFER:
2436
2437 022726 001000      RCBUF:  .BLKW  1000
2438
2439                      ;EXERCISER ERROR TABLE
2440
2441 024726 000001      Y      =      1          ;INITIAL ERROR NUMBER
2442                      ERTBL:
2443                      .REPT  N-1
2444                      .WORD  Y          ;ERROR NUMBER
2445                      .WORD  0          ;ERROR ADDRESS
2446                      .WORD  0          ;NO. OF OCCURRENCES SINCE INIT
2447
2448      Y      -      Y+1
2449                      .ENDR
2450                      .WORD  -1          ;LAST ERROR # IS -1
2451 025146 177777
2452 024720
2453      ERTBLO =      ERTBL-6
2454
2455                      ;      DETAILED ERROR TABLES FOR RCVR AND XMTR ERRORS:
2456
2457 025150 000623      TERTBL: .BLKW  37*15          ;RESERVE SPACE FOR XMTR ERRORS
2458
2459 026616 000623      RERTBL: .BLKW  37*15          ;RESERVE SPACE FOR RCVR ERRORS
2460
2461
2462
2463
2464
2465
2466 030264 033333      FRFLEM: .WORD  33333          ;STORAGE FOR DEQUEUED ELEMENT.
2467 030266 033333      BKELEM: .WORD  33333          ;STORAGE FOR ENQUEUED ELEMENT.
2468 030270 033333      TCBFPT: .WORD  33333          ;CMDBUF POINTER USED DURING SCAN
2469 030272 033333      TCBIN:  .WORD  33333          ;BINARY VALUE OF INPUT PARAMETER
```

```

2470 030274 033333          INTIMP: .WORD 33333          ;TEMP STORAGE FOR INTERRUPT PROC
2471
2472          .SBTTL KEYWORD TABLE
2473
2474          ;KEYWORD TABLE ASSOCIATING A COMMAND WITH ITS PROCESSING ROUTINE
2475
2476 030276 004536          CMDTBL: .WORD CPADD
2477 030300          002          003          .BYTE 2,3
2478 030302 042101 020104          .ASCII /ADD /
2479 030306 005704          .WORD CPASS
2480 030310          002          006          .BYTE 2,6
2481 030312 051501 044523 047107          .ASCII /ASSIGN/
2482 030320 004764          .WORD CPCLR
2483 030322          002          005          .BYTE 2,5
2484 030324 046103 040505 020122          .ASCII /CLEAR /
2485 030332 005126          .WORD CPCNT
2486 030334          002          010          .BYTE 2,8
2487 030336 047503 052116 047111          .ASCII /CONTINUE /
          030344 042525
2488 030346 004644          .WORD CPDEL
2489 030350          001          006          .BYTE 1,6
2490 030352 042504 042514 042524          .ASCII /DELETE /
2491 030360 006612          .WORD CPERR
2492 030362          001          006          .BYTE 1,6
2493 030364 051105 047522 051522          .ASCII /ERRORS /
2494 030372 005522          .WORD CPGO
2495 030374          001          002          .BYTE 1,2
2496 030376 047507          .ASCII /GO /
2497 030400 005222          .WORD CPINIT
2498 030402          001          012          .BYTE 1,10
2499 030404 047111 052111 040511          .ASCII /INITIALIZE /
          030412 044514 042532
2500 030416 004130          .WORD CPMAS
2501 030420          001          006          .BYTE 1,6
2502 030422 040515 052123 051105          .ASCII /MASTER /
2503 030430 004400          .WORD CPRANG
2504 030432          002          005          .BYTE 2,5
2505 030434 040522 043516 020105          .ASCII /RANGE /
2506 030442 004306          .WORD CPRIB
2507 030444          002          003          .BYTE 2,3
2508 030446 044522 020102          .ASCII /RIB /
2509 030452 004214          .WORD CPSEC
2510 030454          002          011          .BYTE 2,9
2511 030456 042523 047503 042116          .ASCII /SECONDARY /
          030464 051101 020131
2512 030470 003420          .WORD CPSILO
2513 030472          002          004          .BYTE 2,4
2514 030474 044523 047514          .ASCII /SILO /
2515 030500 005066          .WORD CPSTAT
2516 030502          002          006          .BYTE 2,6
2517 030504 052123 052101 051525          .ASCII /STATUS /
2518 030512 005470          .WORD CPSUM
2519 030514          002          007          .BYTE 2,7
2520 030516 052523 046515 051101          .ASCII /SUMMARY /
          030524 020131
2521 030526 000000 000000          .WORD 0,0
  
```

```

2522
2523
2524 030532 000000          SCTBL: .WORD 0
2525 030534      001      005      .BYTE 1,5
2526 030536 046103 040505 020122  .ASCII /CLEAR /
2527 030544 000001      .WORD 1
2528 030546      001      003      .BYTE 1,3
2529 030550 042523 020124  .ASCII /SET /
2530 030554 000000 000000  .WORD 0,0
2531
2532
2533          .SBTTL SOME MORE ASCII STORAGE:
2534
2535 030560 020040 020040 020052 MDNER: .ASCIIZ / * * * * * MASTER DOWN * * * * */
      030566 020052 020052 020052
      030574 020052 020052 046440
      030602 051501 042524 020122
      030610 047504 047127 020040
      030616 020052 020052 020052
      030624 020052 020052 000052
2536 030632 020040 025052 052040 MSCHNG .ASCIIZ / ** THIS UNIT HAS BECOME 'NEW MASTER' **/
      C30640 044510 020123 047125
      030646 052111 044040 051501
      030654 041040 041505 046517
      030662 020105 047042 053505
      030670 046440 051501 042524
      030676 021122 025040 000052
2537 030704 020040 025052 051452 SYNTAX: .ASCIIZ / ***SYNTAX ERROR***/
      030712 047131 040524 020130
      030720 051105 047522 025122
      030726 025052      000
2538 030731      120 046103 030461 PCLEXM: .ASCIIZ /PCL11 EXERCISER V02 CZPLAB0 (SEP-78)/
      030736 042440 042530 041522
      030744 051511 051105 053040
      030752 031060 020040 055103
      030760 046120 041101 020060
      030766 024040 042523 026520
      030774 034067 000051
2539 031000 042522 052123 051101 RSTMSG: .ASCII /RESTART AT ADDRESS /
      031006 020124 052101 040440
      031014 042104 042522 051523
      031022      040
2540 031023      132 055132 055132 RSADD: .ASCIIZ /ZZZZZZ/
      031030 000132
2541 031032 025052 051440 051117 MTQMSG: .ASCIIZ /** SORRY, I HAVE NO RECEIVER ADDRESSES. **/
      031040 054522 020054 020111
      031046 040510 042526 047040
      031054 020117 042522 042503
      031062 053111 051105 040440
      031070 042104 042522 051523
      031076 051505 020041 025052
      031104      000
2542 031105      124 040522 050120 TRPDMG: .ASCII /TRAPPED TO 4 FROM LOCATION /
      031112 042105 052040 020117
      031120 020064 051106 046517
      031126 046040 041517 052101

```

2543	031134	047511	020116	020116	TRP4AD: .ASCIIZ /NNNNN ./
	031140	047116	047116		
	031146	020440	000		
2544	031151	104	053105	041511	ERTMSG: .ASCIIZ /DEVICE ADDRESS ERROR. USE 'ASSIGN' COMMAND./
	031156	020105	042101	051104	
	031164	051505	020123	051105	
	031172	047522	027122	052440	
	031200	042523	021040	051501	
	031206	044523	047107	020042	
	031214	047503	046515	047101	
	031222	027104	000		
2545	031225	105	042530	041522	EXRST: .ASCIIZ /EXERCISER STARTED/
	031232	051511	051105	051440	
	031240	040524	052122	042105	
	031246	000			
2546	031247	105	042530	041522	EXCNT: .ASCIIZ /EXERCISER CONTINUING/
	031254	051511	051105	041440	
	031262	047117	044524	052516	
	031270	047111	000107		
2547	031274	025052	052052	044510	MSTMG1: .ASCIIZ /***THIS UNIT IS NOT MASTER***/
	031302	020123	047125	052111	
	031310	044440	020123	047516	
	031316	020124	040515	052123	
	031324	051105	025052	000052	
2548	031332	052502	020124	040510	MSTMG2: .ASCIIZ /BUT HAS NOW BEEN MADE SECONDARY./
	031340	020123	047516	020127	
	031346	042502	047105	046440	
	031354	042101	020105	042523	
	031362	047503	042116	051101	
	031370	027131	000		
2549	031373	124	042510	051440	MSTMG3: .ASCIIZ /THE SILO YOU HAVE JUST LOADED WILL BE/
	031400	046111	020117	047531	
	031406	020125	040510	042526	
	031414	045040	051525	020124	
	031422	047514	042101	042105	
	031430	053440	046111	020114	
	031436	042502	000		
2550	031441	125	042523	020104	MSTMG4: .ASCIIZ /USED IF YOU CLEAR THE CURRENT MASTER./
	031446	043111	054440	052517	
	031454	041440	042514	051101	
	031462	052040	042510	041440	
	031470	051125	042522	052116	
	031476	046440	051501	042524	
	031504	027122	000		
2551	031507	124	042510	051440	MSTMG5: .ASCIIZ /THE SILO HAS BEEN PADDED WITH ADDRESS '0'/'
	031514	046111	020117	040510	
	031522	020123	042502	047105	
	031530	050040	042101	042504	
	031536	020104	044527	044124	
	031544	040440	042104	042522	
	031552	051523	021040	021060	
	031560	000			
2552	031561	124	044510	020123	THUMST: .ASCIIZ /THIS UNIT IS -MASTER-/
	031566	047125	052111	044440	
	031574	020123	046455	051501	
	031602	042524	026522	000	

2553	031607	124	044510	020123	THUSCN: .ASCIIZ /THIS UNIT IS -SECONDARY-/ 031614 047125 052111 044440 031622 020123 051455 041505 031630 047117 040504 054522 031636 000055
2554	031640	051042	041111	020042	RBSTMG: .ASCIIZ /'RIB' IS -SET-/ 031646 051511 026440 042523 031654 026524 000
2555	031657	042	044522	021102	RBCLMG: .ASCIIZ /'RIB' IS -CLEAR-/ 031664 044440 020123 041455 031672 042514 051101 000055
2556	031700	046105	050101	042523	ELPSTM: .ASCII /ELAPSED TIME (HRS:MIN:SEC:TIK).../ 031706 020104 044524 042515 031714 024040 051110 035123 031722 044515 035116 042523 031730 035103 044524 024513 031736 027056 056
2557	031741	060	030072	030072	TMLIN1: .ASCIIZ /0:0:0:0 / 031746 030072 020040 020040 031754 020040 020040 000040
2558	031762	041522	051126	040440	STITLE: .ASCIIZ /RCVR ADDRESS CONNECTION ATTEMPTS SUCCESSFUL CONNECTIONS/ 031770 042104 042522 051523 031776 020040 047503 047116 032004 041505 044524 047117 032012 040440 052124 046505 032020 052120 020123 051440 032026 041525 042503 051523 032034 052506 020114 047503 032042 047116 041505 044524 032050 047117 000123
2559	032054				STLIN:
2560	032054	047116	047116	020040	STLIN1: .ASCII /NNNN /
2561	032062	020040	020040	020040	
	032070	020040	020040	020040	
2562	032076	047116	047116	020040	STLIN2: .ASCII /NNNN /
	032104	020040	020040	020040	
	032112	020040	020040	020040	
	032120	020040	020040		
2563	032124	047116	047116	020040	STLIN3: .ASCIIZ /NNNN /
	032132	020040	020040	020040	
	032140	000040			
2564					
2565					
2566	032142	051105	047522	020122	SMTITLE: .ASCIIZ /ERROR NUMBER ERROR ADDRESS NO. OF OCCURRENCES/ 032150 052516 041115 051105 032156 020040 020040 051105 032164 047522 020122 042101 032172 051104 051505 020123 032200 020040 047040 027117 032206 047440 020106 041517 032214 052503 051122 047105 032222 042503 000123
2567					
2568	032226	020040	025052	047040	NOERMG: .ASCIIZ / ** NO ERRORS TO REPORT YET **/ 032234 020117 051105 047522

	032242	051522	052040	020117	
	032250	042522	047520	052122	
	032256	054440	052105	025040	
	032264	000052			
2569	032266	051124	047101	046523	TERHDR: .ASCII /TRANSMITTER ERRORS:/
	032274	052111	042524	020122	
	032302	051105	047522	051522	
	032310	000072			
2570	032312	051105	047522	020122	TRHLIN: .ASCII /ERROR NO. CONCTD RCVR ERROR COUNT/
	032320	047516	020056	020040	
	032326	041440	047117	052103	
	032334	020104	041522	051126	
	032342	020040	020040	051105	
	032350	047522	020122	047503	
	032356	047125	000124		
2571	032362				TRELIN:
2572	032362	047116	047116	020116	TRRNO: .ASCII /NNNNN /
	032370	020040	020040	020040	
	032376	040			
2573	032377	116	047116	047116	TTRCN: .ASCII /NNNNN /
	032404	020040	020040	020040	
	032412	020040	020040		
2574	032416	047116	047116	020040	TRERC: .ASCII /NNNN /
	032424	000040			
2575	032426	042522	042503	053111	RERHDR: .ASCII /RECEIVER ERRORS:/
	032434	051105	042440	051122	
	032442	051117	035123	000	
2576	032447	105	051122	051117	RRHLIN: .ASCII /ERROR NO. CONCTD XMTR ERROR COUNT/
	032454	047040	027117	020040	
	032462	020040	047503	041516	
	032470	042124	054040	052115	
	032476	020122	020040	042440	
	032504	051122	051117	041440	
	032512	052517	052116	000	
2577	032517				RCELIN:
2578	032517	116	047116	047116	RCRNO: .ASCII /NNNNN /
	032524	020040	020040	020040	
	032532	020040			
2579	032534	047116	047116	020116	RCTRN: .ASCII /NNNNN /
	032542	020040	020040	020040	
	032550	020040	040		
2580	032553	116	047116	020116	RCERC: .ASCII /NNNN /
	032560	020040	000		
2581	032563	040	024040	047516	NONMG: .ASCII / (NONE)/
	032570	042516	000051		
2582	032574				SMLIN:
2583	032574	047116	047116	020040	SMLIN1: .ASCII /NNNN /
	032602	020040	020040	020040	
	032610	020040	020040	020040	
2584	032616	047116	047116	020040	SMLIN2: .ASCII /NNNN /
	032624	020040	020040	020040	
	032632	020040	020040	020040	
	032640	020040			
2585	032642	047116	047116	020040	SMLIN3: .ASCII /NNNN /
	032650	020040	000		
2586					

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

MACY11 30A(1052) 20-OCT-78 09:48
SOME MORE ASCII STORAGE:

F 9
PAGE 38--9

SEQ 0109

2587

```

2589          032654          .EVEN
2590          .SBTTL AUXILIARY ROUTINES
2591          .SBTTL CHARACTER PROCESSOR
2592
2593          : CONVERT A CHARACTER TO ITS PROCESSING ROUTINE ADDRESS BASED
2594          : UPON A TABLE OF ENTRIES IN THE FOLLOWING FORM:
2595          :
2596          :          I-----I-----I
2597          :          I (UNUSED) I CHARACTER I
2598          :          I-----I-----I
2599          :          I PROCESSING ROUTINE ADDR I
2600          :          I-----I-----I
2601
2602          : THE TABLE MUST BE ARRANGED IN ASCENDING ORDER OF CHARACTER VALUES.
2603
2604          : THE TABLE ENDS WITH A DUMMY ENTRY FOR CHARACTER FF (HEXADECIMAL).
2605
2606          : ON ENTRY:  R0 = CHARACTER
2607          :                R1 = TABLE ADDRESS
2608
2609          : CALL:      JSR      PC,PROCHR
2610
2611          : ON RETURN: R0 = CHARACTER
2612          :                R1 = PROCESSING ROUTINE ADDRESS, IF ANY
2613          :                Z = 1 ==> CHARACTER NOT IN TABLE
2614
2615          032654          PROCHR:          **ENTRY POINT**
2616          032654 121100  PCLOOK: CMPB      @R1,R0          :COMPARE TABLE CHAR
2617
2618          032656 001404          BEQ      PCCALL          : WITH ARG. CHAR.
2619
2620          032660 101006          BHI      PCQUIT          : IF SAME, RETURN PROC.
2621
2622          032662 062701 000004          ADD      #4,R1          : ROUTINE ADDR.
2623
2624          032666 000772          BR      PCLOOK          : IF >, THEN ARG. CHAR
2625
2626          032670 016101 000002  PCCALL: MOV      2(R1),R1          : NOT IN TABLE.
2627
2628          032674 000207          RTS      PC          : IF <, POINT TO NEXT
2629
2630          032676 005001          PCQUIT: CLR      R1          : TABLE ENTRY.
2631          032700 000207          RTS      PC          : TRY AGAIN.
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000

```

```

2634 .SBTTL BINARY TO ASCII CONVERSION
2635
2636 : BINARY TO ASCII CONVERSION
2637 :
2638 : LOCAL MACROS
2639 :
2640 : SETUP CONVERSION CONTROL WORD ON STACK
2641 :
2642 : STCVT RADIX,WIDTH,SIGNED,COMPR,BLANKS
2643 :
2644 : WHERE:
2645 :
2646 :     RADIX=NUMERIC VALUE SPECIFYIN CONVERSION RADIX
2647 :     WIDTH=NUMERIC VALUE FROM 1 TO 7 SPECIFYING FIELD WIDTH
2648 :     SIGNED=SIGN OR MAGNITUDE FLAG. ASCII STRING 'SIGN' SPECI-
2649 :     FIES SIGNED CONVERSION. ANYTHING ELSE SPECIFIES MAGNI-
2650 :     TUDE.
2651 :
2652 :     COMPR=COMPRESS LEADING ZEROS FLAG. ASCII STRING 'COMPRES' SPE-
2653 :     CIFIES COMPRESSION OF LEADING ZEROS. ANYTHING ELSE MEANS
2654 :     INCLUDE LEADING ZEROS OR SPACES IN CONVERSION.
2655 :
2656 :     BLANK REPLACE LEADING ZEROS WITH BLANKS (SPACES). ASCII STRING
2657 :     'BLANKS' MEANS BLANK REPLACEMENT IF ZERO COMPRESS IS DIS-
2658 :     ABLED. ANYTHING ELSE SPECIFIES ZERO PADDING.
2659 :
2660 :
2661 : .MACRO STCVT RADIX,WIDTH,SIGN,COMPR,BLANK
2662 $BLK 0
2663 $SGN 0
2664 $SUP 1*1000
2665 .IF IDN <BLANK>,<BLANKS>
2666 $BLK 1*2000
2667 .ENDC
2668 .IF IDN <SIGN>,<SIGNED>
2669 $SGN 1*400
2670 .ENDC
2671 .IF IDN <COMPR>,<COMPRES>
2672 $SUP 0*1000
2673 .ENDC
2674 MOV #<WIDTH*4000>.$BLK $SGN $SUP!RADIX,-(SP)
2675 .ENDM

```

2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732

032702
032702
032706 000411

032710
032710
032714 000406

032716
032716
032722 000403

032724
032724
032730 000400

032732
032732 005702
032734 001002

```

; INPUTS:
;
; R0 ADDRESS TO STORE FIRST BYTE IN OUTPUT STRING
; R1 NUMBER TO BE CONVERTED
; R2=ZERO COMPRESSION INDICATOR
;   IF R2 EQ 0, THEN SUPPRESS ZEROS
;   IF R2 NE 0, THEN DO NOT SUPPRESS ZEROS.
;
; IF CBTA IS CALLED, THEN R2 MUST CONTAIN THE FOLLOWING INFORMATION
;
; LOW BYTE=CONVERSION RADIX (2-10.)
; BIT 8=MAGNITUDE/SIGNED CONVERSION (1 SIGNED)
;
; BIT 9 =ZERO COMPRESS FLAG (0-COMPRESS LEADING ZEROS)
;
; BIT 10=BLANK FILL FLAG (1=REPLACE LEADING ZEROS WITH BLANKS
;   IF ZERO COMPRESS DISABLED, 0-ZERO FILL).
;
; BITS 11-15=FIELD WIDTH (1-32)
;
; OUTPUTS:
;
; R0-ADDRESS OF NEXT BYTE AFTER LAST DIGIT STORED.
;
; IF THE CONVERTED DIGIT EXCEEDS 9, THE RESULT IS BIASED TO FALL
; IN THE RANGE A - Z
;
; CONVERT 6 DIGIT OCTAL TO ASCII MAGNITUDE
;
OCTJSP:
;   STCVT  8.,6.,MAGN,NOCOMP,BLANKS      ;PUSH CONVERSION PARAMETERS
;   BR      SETCN                        ;CONVERT TO ASCII
;
; CONVERT 6 DIGIT OCTAL TO ASCII (ZERO COMPR)
;
OCTPNT:
;   STCVT  8.,6.,MAGN,COMPRES,NOBLKANK
;   BR      SETCN
;
; CONVERT 5 DIGIT DECIMAL TO ASCII MAGNITUDE
;
DECJSP:
;   STCVT  10.,5.,MAGN,NOCOMP,BLANKS
;   BR      SETCN
;
; CONVERT 5 DIGIT DECIMAL TO ASCII (ZERO COMPR)
;
DECPNT:
;   STCVT  10.,5.,MAGN,COMPRES,NOBLANK
;   BR      SETCN
;
SETCN:
;   TST     R2                          ;SUPPRESS ZEROS?
;   BNE     20$                        ;IF NE, NO

```

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

J 9
MACY11 30A(1052) 20-OCT-78 09:48 PAGE 41-1
BINARY TO ASCII CONVERSION

SEQ 0113

2733 032736 042716 001000
2734 032742
2735 032742 012602

20\$: BIC #1*1000,(SP)
MOV (SP+,R2

;ENABLE ZERO SUPPRESS
;SET CONTROL WORD

CZ
PC

.SBTTL GENERAL BINARY TO ASCII CONVERSION

```
2737
2738
2739 032744          CBTA:
2740 032744 004567 000406      JSR      R5,SAVRG          ;SAVE THE NON-VOLATILE REGISTERS
2741 032750 110205          MOV      R2,R5              ;COPY RADIX BYTE
2742 032752 000302          SWAB     R2                  ;POSITION REMAINING TO LOW BYTE
2743 032754 106202          ASRB     R2                  ;SHIFT OFF MAG. FLAG
2744 032756 103005          BCC      10$                ;UNSIGNED IF C IS CLR
2745 032760 005701          TST      R1                  ;POSITIVE VALUE?
2746 032762 100003          BPL      10$                ;IF PL, YES
2747 032764 005401          NEG      R1                  ;MAKE VALUE POSITIVE
2748 032766 112720 000055    MOV      #'-(R0)+          ;INSERT A MINUS SIGN
2749 032772          10$:
2750 032772 010004          MOV      R0,R4              ;COPY STRING POINTER
2751 032774 000241          CLC                      ;CLEAR CARRY
2752 032776 106002          RORB     R2                  ;SHIFT OFF SUPPR FLAG
2753 033000 006002          ROR      R2                  ;TRANSFER TO R2
2754 033002 006003          ROR      R3                  ;GET BLANK/ZERO PAD FLAG
2755 033004 105003          CLRB     R3                  ;CLEAR COUNT BYTE
2756 033006 150203          BISB     R2,R3              ;TRANSFER COUNT BYTE
2757 033010 105002          CLRB     R2                  ;CLEAR FILL BYTE
2758 033012 152702 000060    BISB     #'0,R2           ;SET FILL BYTE
2759 033016 010100          MOV      R1,R0              ;DIVIDEND TO R0
2760 033020          1$:
2761 033020 010501          MOV      R5,R1              ;SET CONVERSION RADIX
2762 033022 004767 000272    JSR      PC,DIV            ;DIVIDE EM UP
2763 033026 020127 000011    CMP      R1,#9.           ;RESULT EXCEED NUMERICS?
2764 033032 101402          BLOS     15$                ;IF LOS, NO
2765 033034 062701 000007    ADD      #'7,R1           ;BIAS TO FALL IN ALPHA
2766 033040          15$:
2767 033040 060201          ADD      R2,R1              ;ADD CHARACTER BIAS
2768 033042 010146          MOV      R1,-(SP)           ;SAVE CHARACTER
2769 033044 105303          DECB     R3                  ;DECREMENT CHARACTER COUNT
2770 033046 003412          BLE      3$                  ;IF LE NO DIGITS LEFT
2771 033050 005700          TST      R0                  ;ZERO QUOTIENT
2772 033052 001006          BNE      2$                  ;IF NE, YES, GO AGAIN
2773 033054 005702          TST      R2                  ;SUPPRESS ZEROS
2774 033056 100006          BPL      3$                  ;IF PL, YES, ALL DONE
2775 033060 005703          TST      R3                  ;SUBSTITUTE BLANKS?
2776 033062 100002          BPL      2$                  ;IF PL, NO
2777 033064 042702 000020    BIC      #'20,R2          ;CONVERT FILL TO BLANK
2778 033070          2$:
2779 033070 004767 177724    JSR      PC,1$              ;DIVIDE AGAIN
2780 033074          3$:
2781 033074 112624          MOV      (SP)+,(R4)+        ;STORE A DIGIT
2782 033076 010400          MOV      R4,R0              ;STORE TERMINAL ADDRESS
2783 033100          RETURN
2784
```

```
2786 .SBTTL DOUBLE PRECISION BINARY TO ASCII
2787 : CONVERT A DOUBLE PRECISION UNSIGNED QUANTITY TO DECIMAL ASCII
2788 :
2789 : LOCAL MACROS
2790 :
2791 : SET ASCII CONVERSION PARAMETERS
2792 :
2793 : CBTAS RADIX,WIDTH,SIGN,BLANK
2794 :
2795 : WHERE:
2796 :
2797 :     RADIX CONVERSION RADIX
2798 :
2799 :     WIDTH=FIELD WIDTH
2800 :
2801 :     SIGN-'SIGNED' FOR SIGNED CONVERSION. ANYTHING ELSE IMPLIES
2802 :         UNSIGNED CONVERSION
2803 :
2804 :     BLANK-'BLANKS' TO CONVERT LEADING ZEROS TO BLANKS. ANYTHING ELSE
2805 :         IMPLIES NO CONVERSION OF ZEROS .
2806 :
2807 : .MACRO CBTAS RADIX,WIDTH,SIGN,BLANK
2808 $BLKS=0
2809 $SGNS=0
2810 .IF IDN <BLANK>,<BLANKS>
2811 $BLKS=1*2000
2812 .ENDC
2813 .IF IDN <SIGN>,<SIGNED>
2814 $SGNS=1*400
2815 .ENDC
2816 MOV #<WIDTH*400>,$SGNS;$BLKS;RADIX,R5
2817 TST R2
2818 BEQ .+4
2819 BIS #1*1000,R5
2820 .ENDM
2821
2822 : INPUTS:
2823 :
2824 :     R0=POINTER TO ASCII OUTPUT STRING
2825 :     R1=ADDRESS OF DOUBLE PRECISION VALUE
2826 :     R2=ZERO COMPRESS FLAG
2827 :
2828 :
2829 CDDMG:
2830 033102 JSR R5,SAVRG ;SAVE THE NON-VOLATILE REGISTERS
2831 033106 MOV R0,R3 ;COPY THE STRING POINTER
2832 033110 MOV #10000.,R4 ;SET DIVISOR
2833 033114 CBTAS 10.,0,NOSIGN,BLANKS ;SET CONVERSION PARAMETERS
2834 033130 CMP (R1)+,R4 ;TEST FOR OVERFLOW
2835 033132 BHIS 40$ ;IF HIS, OVERFLOW
2836 033134 MOV (R1),R2 ;GET LOW PART OF NUMBER
2837 033136 MOV -(R1),R1 ;GET HIGH PART OF NUMBER
2838 033140 MOV R4,R0 ;COPY DIVISOR
2839 033142 JSR PC,DDIV ;DO DOUBLE PREC. DIVIDE
2840 033146 MOV R0,-(SP) ;SAVE REMAINDER
2841 033150 MOV R2,R1 ;COPY QUOTIENT
```

2829	033102			
2830	033102	004567	000250	
2831	033106	010003		
2832	033110	012704	023420	
2833	033114			
2834	033130	022104		
2835	033132	103042		
2836	033134	011102		
2837	033136	014101		
2838	033140	010400		
2839	033142	004767	000110	
2840	033146	010046		
2841	033150	010201		


```
2842 033152 001011      BNE      11$      ;IF NE, SOMETHING TO PRINT
2843 033154 012702 000005      MOV      #5,R2      ;OTHERWISE, FILL FIELD WITH BLANKS
2844 033160 112723 000040      21$:      MOVB     #'',(R3)+
2845 033164 005302      DEC      R2
2846 033166 001374      BNE      21$
2847 033170 052705 003000      BIS      #3000,R5      ;DISABLE BLANK SUPPRESSION
2848 033174 000411      BR       20$
2849 033176 012702 024000      11$:      MOV      #5*4000,R2      ;SET FIELD WIDTH
2850 033202 004767 000020      JSR      PC,30$      ;OUTPUT HIGH ORDER DIGITS
2851 033206 052705 001000      BIS      #1*1000,R5      ;DISABLE ZERO COMPRESS
2852 033212 042705 002000      BIC      #1*2000,R5      ;DISABLE BLANKS
2853 033216 010003      31$:      MOV      R0,R3      ;SET STRING POINTER
2854 033220      20$:
2855 033220 012601      MOV      (SP)+,R1      ;GET LOW ORDER VALUE
2856 033222 012702 020000      MOV      #4*4000,R2      ;SET FIELD WIDTH
2857 033226      30$:
2858 033226 010300      MOV      R3,R0      ;GET STRING POINTER
2859 033230 050502      BIS      R5,R2      ;INCLUDE RADIX & BLANK SUPPRESS
2860 033232 004767 177506      JSR      PC,CBTA      ;CONVERT TO ASCII
2861 033236 000406      BR       60$      ;EXIT
2862 033240      40$:
2863 033240 012702 000011      MOV      #9.,R2      ;GET COUNT
2864 033244      50$:
2865 033244 112720 000052      MOVB     #'*,(R0)+      ;FILL FIELD WITH ASTERISKS
2866 033250 005302      DEC      R2
2867 033252 001374      BNE      50$      ; 'SOB R2,50$'
2868 033254      60$:
2869 033254      RETURN
```

.SBTTL DOUBLE PRECISION DIVIDE ROUTINE

: INPUTS:

```
: R2=LOW ORDER OF DIVIDEND
: R1=HIGH ORDER OF DIVIDEND
: R0=DIVISOR (15 BITS UNSIGNED)
```

: OUTPUTS:

```
: R2=LOW ORDER OF QUOTIENT
: R1=HIGH ORDER OF QUOTIENT
: R0=REMAINDER
```

DDIV:

```
2888 033256      MOV      R3,-(SP)      ;SAVE R3
2889 033256 010346      MOV      #32.,R3      ;SET ITERATION COUNT IN R3
2890 033260 012703 000040      MOV      R0,-(SP)      ;STACK DIVISOR
2891 033264 010046      CLR      R0      ;SET REMAINDER TO 0
2892 033266 005000      1$:
2893 033270      ASL      R2      ;SHIFT THE ENTIRE DIVIDEND
2894 033270 006302      ROL      R1      ;... ONE BIT TO THE LEFT AND...
2895 033272 006101      ROL      R0      ;... INTO THE REMAINDER
2896 033274 006100      CMP      R0,(SP)      ;IS REMAINDER GE DIVISOR?
2897 033276 020016
```

```

2898 033300 103402      BLO      2$      ;NO, SKIP TO ITERATION CONTROL
2899 033302 161600      SUB      (SP),R0  ;YES, SUB DIVISOR OUT
2900 033304 005202      INC      R2      ;AND INCR THE QUOTIENT
2901 033306      2$:      DEC      R3      ;REPEAT AS LONG AS NECESSARY
2902 033306 005303      BGT      1$      ;PURGE DIVISOR FROM STACK
2903 033310 003367      TST      (SP)+    ;RESTORE R3
2904 033312 005726      MOV      (SP)+,R3
2905 033314 012603      RETURN
2906 033316
2907
2908
2909      .SBTTL  INTEGER DIVIDE MAGNITUDE NUMBERS
2910
2911      :
2912      : INPUTS:
2913      :
2914      : R0=DIVIDEND
2915      : R1=DIVISOR
2916      :
2917      : OUTPUTS:
2918      :
2919      : QUOTIENT IS RETURNED IN R0
2920      : REMAINDER IS RETURNED IN R1
2921      :
2922      :
2923      : DIV:
2924      : MOV      #20,-(SP)      ;SET LOOP COUNT
2925      : MOV      R1,-(SP)      ;SAVE DIVISOR FOR SUBTRACTS
2926      : CLR      R1            ;CLEAR REMAINDER
2927      : 30$:
2928      : ASL      R0            ;DOUBLE LEFT SHIFT.
2929      : ROL      R1
2930      : CMP      R1,(SP)      ;SUBTRACT OUT DIVISOR
2931      : BLO      40$          ;IF LO, NO
2932      : SUB      (SP),R1      ;SUBTRACT OUT DIVISOR
2933      : INC      R0            ;ADD IN LOW BIT
2934      : 40$:
2935      : DEC      2(SP)         ;DECREMENT REPEAT COUNT
2936      : BGT      30$          ;IF GT, MORE TO GO
2937      : 50$:
2938      : CMP      (SP)+,(SP)+   ;CLEAN STACK
2939      : RETURN
2940
2941      : SAVE/RESTORE NONVOLATILE REGISTERS
2942
2943      :
2944      : SAVRG: MOV      R4,-(SP)      ;SAVE R4 & R3
2945      : MOV      R3,-(SP)
2946      : MOV      R5,-(SP)
2947      : MOV      6(SP),R5      ;PUT RETURN ADDRESS ON STACK
2948      : JSR      PC,@(SP)+    ;RETRIEVE REAL R5
2949      : MOV      (SP)+,R3      ;CALL THE CALLER
2950      : MOV      (SP)+,R4      ;RESTORE NON VOLATILE REGISTERS
2951      : MOV      (SP)+,R5
2952      : RETURN

```

```
2954 .SBTTL QUEUE HANDLING ROUTINES
2955 ; THIS MODULE CONTAINS 2 SUBROUTINES, ENQ & DEQ, TO ENQUEUE & DEQUEUE
2956 ; WORDS, RESPECTIVELY, IN A FIRST-IN-FIRST-OUT LIST.
2957 ;
2958 .LIST MEB
2959
2960
2961 033402 PROC ENQ,<ITEM,QUEUE>
(3) 000002 ITEM - 2
(3) 000004 QUEUE = 4
(1)
(1) 033402 ENQ: ;**ENTRY POINT**
2962 ;
2963 ; APPEND ITEM (A WORD) TO THE FIRST-IN-FIRST-OUT LIST QUEUE .
2964 ;
2965 033402 016504 000004 MOV QUEUE(R5),R4 ;GET QUEUE ADDRESS.
2966 033406 021464 000002 CMP @R4,QSIZE(R4) ;IF QUEUE IS FULL,
2967 033412 002020 BGE NQFULL ; SIGNAL TRAGIC ERROR.
2968 033414 017574 000002 000012 MOV @ITEM(R5),@QBACK(R4) ;PUT ITEM AT BACK OF QUEUE.
2969 033422 062764 000002 000012 ADD #2,QBACK(R4) ;UPDATE BACK POINTER.
2970 033430 026464 000012 000006 CMP QBACK(R4),QBOT(R4) ;
2971 033436 103403 BLO NQNOWP ;
2972 033440 016464 000004 000012 MOV QTOP(R4),QBACK(R4) ;
2973 033446 005214 NQNOWP: INC @R4 ;INCREMENT NO. OF ELEMENTS.
2974 033450 000241 CLC ;INDICATE SUCCESSFUL ENQ.
2975 033452 NQRET: RETURN ;RETURN.
(1) 033452 000207 RTS PC
2976
2977 033454 000261 NQFULL: SEC ;INDICATE UNSUCCESSFUL ENQ.
2978 033456 000775 BR NQRET ;IGNORE ITEM & RETURN.
2979
2980
2981 033460 PROC DEQ,<ITEM,QUEUE>
(3) 000002 ITEM - 2
(3) 000004 QUEUE = 4
(1)
(1) 033460 DEQ: ;**ENTRY POINT**
2982 ;
2983 ; REMOVE A WORD ENTRY FROM THE FIRST-IN-FIRST-OUT LIST QUEUE &
2984 ; STORE IT AT ITEM .
2985 ;
2986 033460 016504 000004 MOV QUEUE(R5),R4 ;GET QUEUE ADDRESS.
2987 033464 005714 TST @R4 ;IF QUEUE IS EMPTY,
2988 033466 001420 BEQ DQEMP ; SIGNAL TRAGIC ERROR.
2989 033470 017475 000010 000002 MOV @QFRONT(R4),@ITEM(R5) ;RETRIEVE FRONT ELEMENT.
2990 033476 062764 000002 000010 ADD #2,QFRONT(R4) ;UPDATE FRONT POINTER.
2991 033504 026464 000010 000006 CMP QFRONT(R4),QBOT(R4) ;
2992 033512 103403 BLO DQNOWP ;
2993 033514 016464 000004 000010 MOV QTOP(R4),QFRONT(R4) ;
2994 033522 005314 DQNOWP: DEC @R4 ;DECREMENT NO. OF ELEMENTS.
2995 033524 000241 CLC ;INDICATE SUCCESSFUL DEQ.
2996 033526 DQRET: RETURN ;RETURN.
(1) 033526 000207 RTS PC
2997
2998 033530 000261 DQEMP: SEC ;INDICATE UNSUCCESSFUL DEQ.
2999 033532 012775 177777 000002 MOV #-1,@ITEM(R5) ;SET ITEM TO ALL ONES.
```

CZPLABO PCL11 EXERCISER V02
PCLEXR.P11 12-SEP-78 15:13

MACY11 30A(1052) 20-OCT-78 09:48
QUEUE HANDLING ROUTINES

C 10

PAGE 44-1

SEQ 0119

3000 033540 000772

BR

DQRET

;RETURN.

```

3002      ; SUBROUTINE TO SCAN AN INPUT COMMAND & CALL ITS
3003      ; PROCESSING ROUTINE.
3004
3005          .LIST   MEB
3006
3007
3008          .MACRO  SPAN    REG,CHAR,?L
3009
3010      ;   THIS MACRO SCANS THE STRING OF CHARACTERS STARTING AT
3011      ;   @REG UNTIL IT FINDS ONE NOT EQUAL TO CHAR.  REG IS
3012      ;   SET POINTING TO THAT CHARACTER.
3013
3014      L:      CMPB    (REG)+,CHAR
3015              BEQ     L
3016              DEC     REG
3017
3018              .ENDM
3019
3020          .MACRO  BREAK   REG,CHRSET,?HH,?JJ
3021
3022      ;   THIS MACRO SCANS THE STRING STARTING AT @REG UNTIL
3023      ;   IT FINDS A CHARACTER THAT IS A MEMBER OF CHRSET.
3024      ;   REG IS SET POINTING TO THAT CHARACTER.
3025
3026      ;   EACH MEMBER OF CHRSET IS AN ADDRESSABLE QUANTITY.
3027
3028
3029      HH:
3030          .IRP    LS,<CHRSET>
3031          CMPB    @REG,LS
3032          BEQ     JJ
3033          .ENDM
3034          INC     REG
3035          BR      HH
3036
3037      JJ:
3038          .ENDM
3039
3040
3041          .MACRO  SYNCLS  CHAR,CLASS,?CC,?DD,?EE,?FF,?GG
3042
3043      ;   THIS MACRO DETERMINES THE SYNTACTIC CLASS OF AN
3044      ;   OBJECT BEGINNING WITH CHAR.  THE CLASS IS RETURNED
3045      ;   IN CLASS AS FOLLOWS:
3046      ;       CLASS  =      0 (WORD) IF CHAR = (A,....,Z,-)
3047      ;                   2 (NUMBER) IF CHAR = (0,....,9)
3048      ;                   6 (END OF LINE) IF CHAR = CARRIAGE RETURN
3049      ;                   4 (CHARACTER STRING) OTHERWISE
3050
3051          CMPB    CHAR,#'A
3052          BLT     EE
3053          CMPB    CHAR,#'Z
3054          BGT     DD
3055      CC:      CLR     CLASS
3056              BR      GG
3057

```

3058	DD:	MOV	#4,CLASS
3059		BR	GG
3060			
3061	EE:	CMPB	CHAR,#'0
3062		BLT	FF
3063		CMPB	CHAR,#'9
3064		BGT	DD
3065		MOV	#2,CLASS
3066		BR	GG
3067			
3068	FF:	CMPB	CHAR,#'-
3069		BEQ	CC
3070		CMPB	CHAR,#15
3071		BNE	DD
3072		MOV	#6,CLASS
3073	GG:		
3074			
3075		.ENDM	

```
3077 .SBTTL COMMAND PROCESSOR INITIATING ROUTINE
3078
3079 033542 PROC COMMAND,<INPLIN,KWTABL>
(3) 000002 INPLIN = 2
(3) 000004 KWTABL = 4
(1)
(1) 033542 COMMAND: ;**ENTRY POINT**
3080 ;
3081 ; THIS ROUTINE CAUSES THE COMMAND SPECIFIED BY INPLIN TO BE
3082 ; PROCESSED, AS DESCRIBED BELOW. INPLIN IS A STRING OF ASCII
3083 ; CHARACTERS ENDED BY A CARRIAGE RETURN CODE.
3084 ;
3085 ; - INPLIN IS LEXICALLY SCANNED USING THE LXSCAN ROUTINE.
3086 ; - THE 1ST OBJECT IS CONVERTED TO A PROCESSING ROUTINE ADDRESS
3087 ; USING THE KEYWD ROUTINE & THE KEYWORD TABLE KWTABL
3088 ; SUPPLIED BY THE CALLING PROGRAM.
3089 ; - THE PROCESSING ROUTINE IS CALLED WITH THE OUTPUT FROM LXSCAN
3090 ; ON THE STACK STARTING AT 2(SP). (THE RETURN ADDRESS OF THIS
3091 ; CALL OCCUPIES THE TOP WORD OF THE STACK.)
3092 ; - THE LXSCAN OUTPUT IS REMOVED FROM THE STACK.
3093 ;
3094 ; IF LXSCAN OR KEYWD OR THE PROCESSING ROUTINE RETURN AN ERROR
3095 ; CONDITION, THEN C - 1; OTHERWISE, C - 0.
3096 ;
3097 033542 010546 MOV R5,-(SP) ;SAVE PAR LIST POINTER.
3098 033544 010667 000076 MOV SP,CMMARK ;SAVE STACK POINTER.
3099 033550 LXSCAN INPLIN(R5) ;LEXICALLY SCAN INPLIN.
(1) 033550 016501 000002 MOV INPLIN(R5),R1
(1) 033554 004167 000310 JSR R1,LXSCAN
3100 033560 103426 BCS CMRET ;IF ERROR, RETURN WITH C 1.
3101 033562 005716 TST @SP ;HAVE WE ANY OBJECTS?
3102 033564 003005 BGT 1$ ;NO, IGNORE BLANK COMMAND.
3103 033566 012767 177777 164162 MOV #-1,ESCFLG
3104 033574 000167 000036 JMP CMRET
3105 033600 016700 000042 1$: MOV CMMARK,R0 ;YES, DETERMINE ADDRESS
3106 033604 162700 000006 SUB #6,R0 ;OF FIRST OBJECT.
3107 033610 017705 000032 MOV @CMMARK,R5 ;RESTORE PAR LIST POINTER.
3108 033614 KEYWD R0,KWTABL(R5) ;GET ADDR OF COMMAND PROCESSOR.
(1) 033614 010046 MOV R0,-(SP)
(1) 033616 016546 000004 MOV KWTABL(R5),-(SP)
(1) 033622 004767 000024 JSR PC,KEYWD
3109 033626 103403 BCS CMRET ;IF INVALID COMMAND, MAKE
3110 ; ERROR RETURN.
3111 033630 012705 033650 MOV #NULPAR,R5 ;LOAD NULL PAR LIST ADDRESS.
3112 033634 004736 JSR PC,@(SP)+ ;PROCESS COMMAND & POP ADDR.
3113 033636 016706 000004 CMRET: MOV CMMARK,SP ;RESTORE STACK TO ENTRY STATUS.
3114 033642 030026 BIT R0,(SP)+ ;THROW AWAY SAVED R5 (WITHOUT
3115 ; AFFECTING C BIT).
3116 033644 RETURN ;RETURN TO CALLING PROGRAM.
(1) 033644 000207 RTS PC
3117
3118 ; DATA AREAS.
3119 033646 033333 CMMARK: .WORD 33333 ;STORAGE FOR STACK PTR ON ENTRY.
3120 033650 000000 NULPAR: .WORD 0 ;PAR LIST CONTAINING NO PARS.
```

```
3122 .SBTTL KEYWORD PROCESSING ROUTINE
3123
3124 : SUBROUTINE TO DETERMINE THE ADDRESS OF THE PROCESSING
3125 : ROUTINE ASSOCIATED WITH THE KEYWORD REPRESENTED BY
3126 : THE SYNTACTIC OBJECT POINTED TO BY THE ARGUMENT
3127 : SRC. CONVERSION FROM KEYWORD TO ROUTINE ADDRESS
3128 : IS DONE AS DEFINED IN THE KEYWORD TABLE
3129 : POINTED TO BY THE ARGUMENT TABLAD.
3130
3131 : ON ENTRY, THE TOP OF THE STACK IS AS FOLLOWS:
3132 : (SP): RETURN ADDRESS
3133 : 2(SP): TABLAD
3134 : 4(SP): SRC
3135
3136 : CALLING INSTRUCTION: JSR PC,KEYWD
3137
3138 : ON RETURN, THE TOP OF THE STACK IS AS FOLLOWS:
3139 : (SP): ROUTINE ADDRESS, IF KEYWORD IN TABLE; 0 IF NOT.
3140
3141 : IF THE KEYWORD IS IN THE TABLE, C=0 ON RETURN. IF NOT, C 1.
3142
3143 : STACK POINTER OFFSETS
3144 :ROUTAD - 0
3145
3146 :ROUTINE ADDR FOR
3147 : CURRENT TABLE
3148 : ELEMENT.
3149 : ADDR OF INPUT WORD
3150 : #CHAR IN INPUT WORD
3151 : SUBROUTINE RETURN ADDR
3152 : ADDR OF KEYWORD TABLE
3153 : ADDR OF INPUT OBJECT
3154 : RESULT RETURNED
3155
3156 : **ENTRY POINT**
3157 : SAVE REGISTERS.
3158
3159 : ALLOCATE STACK SPACE
3160 : GET ADDR OF INPUT
3161 : OBJECT
3162 : IF OBJECT NOT A WORD
3163 : THEN EXIT: NOT FOUND
3164 : GET # CHAR IN OBJECT.
3165 : STORE ON STACK.
3166 : GET ADDR OF INPUT WORD.
3167 : STORE ON STACK.
3168 : GET ADDR. OF KEYWORD
3169 : TABLE.
3170 : SAVE ROUTINE ADDR OF
3171 : 1ST ELEMENT.
3172 : GET MINIMUM LENGTH
3173 : OF TABLE WORD.
3174 : GET FULL LENGTH.
3175
3176 : IF 0, THEN NO MORE
3177 : TABLE TO SEARCH
3178 : COMPARE INPUT CHAR
3179 : WITH TABLE CHAR.

000002 ADRINP - 2
000004 LENINP - 4
000022 RETURN - 22
000024 TABLAD = 24
000026 SRC = 26
000026 RESULT = 26

KEYWD: REGSAV
JSR R5,REGSAV
SUB #6,SP
MOV SRC(SP),R2

TST (R2)+
BNE NOTHER
MOV (R2)+,R5
MOV R5,LENINP(SP)
MOV @R2,R4
MOV R4,ADRINP(SP)
MOV TABLAD(SP),R3

GTLENS: MOV (R3)+,@SP

MOVB (R3)+,R2
BIC #177400,R2
MOVB (R3)+,R1
BIC #177400,R1
BEQ NOTHER

NXTCH: CMPB (R4)+,(R3)+
```


3177	033734	002427		BLT	NOTHER		; IF <, THEN NO MATCH
3178							; EXISTS.
3179	033736	003005		BGT	NXTWD		; IF >, THEN TRY NEXT
3180							; TABLE WORD.
3181	033740	005305		DEC	R5		; IF INPUT STRING
3182							; EXHAUSTED,
3183	033742	001413		BEQ	THFND		; WE MAY HAVE FOUND
3184							; MATCH.
3185	033744	005301		DEC	R1		; IF MORE CHAR IN TABLE WORD
3186	033746	001371		BNE	NXTCH		; TO TEST, GO & TEST THEM.
3187	033750	005201		INC	R1		
3188	033752	060103		NXTWD:	ADD	R1,R3	; GET ADDR OF NEXT
3189	033754	042703	000001		BIC	#1,R3	; TABLE ENTRY.
3190	033760	016604	000002		MOV	ADRINP(SP),R4	; POINT TO BEGINNING
3191							; OF INPUT WORD.
3192	033764	016605	000004		MOV	LENINP(SP),R5	; GET LENGTH OF INPUT WORD.
3193	033770	000750			BR	GTLENS	; GET LENGTHS OF TABLE WORD.
3194							
3195	033772	026602	000004	THFND:	CMP	LENINP(SP),R2	; IF LEN(INP.WD) < MIN LEN (TABLE
3196	033776	002406			BLT	NOTHER	; WORD), WORD IS NOT IN TABLE.
3197	034000	011666	000026		MOV	(SP),RESULT(SP)	; SAVE ROUTINE ADDR. OF
3198							; MATCH.
3199	034004	062706	000006		ADD	#6,SP	; FREE LOCAL STACK SPACE.
3200	034010	000241			CLC		; CLEAR CARRY BIT.
3201	034012	000405			BR	KWEXIT	
3202							
3203							
3204	034014	005066	000026				; WORD IS NOT IN TABLE. SET RESULT TO 0 & SET Z BIT ON.
3205	034020	062706	000006	NOTHER:	CLR	RESULT(SP)	; CLEAR RESULT.
3206	034024	000261			ADD	#6,SP	; FREE LOCAL STACK SPACE.
3207	034026				SEC		; SET CARRY BIT.
(1)	034026	004567	000020	KWEXIT:	REGRES		; RESTORE REGISTERS.
3208	034032	012616			JSR	R5,REGRES	
3209	034034	000207			MOV	(SP)+,@SP	; POP AN ARGUMENT.
					RTS	PC	; RETURN.

3211		
3212		
3213		
3214		
3215	034036	
3216	034036	010446
3217	034040	010346
3218	034042	010246
3219	034044	010146
3220	034046	010046
3221	034050	000115
3222		
3223		
3224		
3225		
3226		
3227		
3228		
3229	034052	
3230	034052	030026
3231	034054	012600
3232	034056	012601
3233	034060	012602
3234	034062	012603
3235	034064	012604
3236	034066	000205

```

.SBTL  REGISTER SAVE & RESTORE ROUTINES

; SUBROUTINE TO SAVE R0 - R5 ON STACK
; CALLING SEQUENCE:      JSR      R5,REGSAV
REGSAV:                                     ;**ENTRY POINT**
      MOV      R4,-(SP)
      MOV      R3,-(SP)
      MOV      R2,-(SP)
      MOV      R1,-(SP)
      MOV      R0,-(SP)
      JMP      @R5

; SUBROUTINE TO RESTORE R0-R5 FROM STACK
; THE CONDITION CODE BITS IN THE PS ARE DESTROYED,
; EXCEPT FOR THE CARRY BIT, WHICH IS PRESERVED.

; CALLING SEQUENCE:      JSR      R5,REGRES
REGRES:                                     ;**ENTRY POINT**
      BIT      R0,(SP)+
      MOV      (SP)+,R0
      MOV      (SP)+,R1
      MOV      (SP)+,R2
      MOV      (SP)+,R3
      MOV      (SP)+,R4
      RTS      R5
      ;THROW AWAY OLD R5 VALUE.

```

```

3238 .SBTTL LEXICAL SCAN ROUTINE
3239
3240 ; PERFORM LEXICAL SCAN OF INPUT COMMAND IN
3241 ; BUFFER.
3242
3243 ; THREE CLASSES OF SYNTACTIC OBJECTS ARE RECOGNIZED:
3244 ; 1. WORD: A STRING OF CHARACTERS BEGINNING WITH
3245 ; A LETTER & TERMINATED WITH A
3246 ; BLANK OR CARRIAGE RETURN.
3247 ; 2. NUMBER: A STRING OF OCTETS TERMINATED WITH A
3248 ; BLANK OR CARRIAGE RETURN, OR A STRING OF DIGITS
3249 ; TERMINATED WITH A DOT.
3250 ; 3. CHARACTER STRING: A STRING SURROUNDED BY 2 INSTANCES
3251 ; (1 ON EACH END) OF A SPECIAL CHARACTER.
3252
3253 ; SYNTACTIC OBJECTS ARE SEPARATED BY 1 OR MORE BLANKS.
3254 ; THE COMMAND IS ENDED BY A CARRIAGE RETURN.
3255
3256 ; THIS LEXICAL SCANNER DETERMINES THE LOCATIONS OF THE
3257 ; SYNTACTIC OBJECTS & DETERMINES THEIR CLASSES.
3258 ; NUMBERS ARE CONVERTED TO THEIR BINARY VALUES.
3259
3260 ; AFTER THE LEXICAL SCAN IS PERFORMED, EACH
3261 ; SYNTACTIC OBJECT WILL BE REPRESENTED ON THE STACK
3262 ; AS A 3 WORD QUANTITY AS FOLLOWS:
3263 ; TOP WORD = SYNTACTIC CLASS:
3264 ; 0==>WORD
3265 ; 2==>NUMBER
3266 ; 4==>CHARACTER STRING
3267 ; 2ND WORD - LENGTH OF OBJECT IN CHARACTERS; NOT
3268 ; SIGNIFICANT FOR NUMBERS; DOES NOT
3269 ; INCLUDE SURROUNDING DELIMITERS FOR CHAR.
3270 ; STRINGS.
3271 ; 3RD WORD = VALUE OF NUMBER, OR POINTER TO 1ST
3272 ; LETTER OF KEYWORD OR 1ST SIGNIFICANT
3273 ; CHARACTER OF STRING (IE: NOT SUR-
3274 ; ROUNDING DELIMITER)
3275
3276 ; AT THE END OF THE LEXICAL SCAN THE STACK WILL BE
3277 ; ARRANGED AS FOLLOWS (N NO. OF SYNTACTIC OBJECTS):
3278 ;
3279 ; |-----|
3280 ; | N | <--- SP
3281 ; |-----|
3282 ; | NTH OBJECT |
3283 ; |-----|
3284 ; | : |
3285 ; |-----|
3286 ; | 1ST OBJECT |
3287 ; |-----|
3288 ; WHERE EACH OBJECT IS REPRESENTED AS ABOVE.
3289
3290 ; CALLING SEQUENCE: R1=BUFFER ADDRESS
3291 ; JSR R1,LXSCAN
3292 ; ON RETURN, REGISTERS 0-5 ARE UNDEFINED.
3293 ; EXCEPT FOR THE ABOVE TABLE, THE STACK IS AS IT
; WAS BEFORE ENTRY TO THE ROUTINE.

```

```
3294
3295
3296
3297
3298
3299
3300 034070
3301 034070 005002
3302 034072 012605
3303 034074
(1) 034074 122527 000040
(1) 034100 001775
(1) 034102 005305
3304 034104
(1) 034104 121527 000101
(1) 034110 002410
(1) 034112 121527 000132
(1) 034116 003002
(1) 034120 005004
(1) 034122 000424
(1) 034124 012704 000004
(1) 034130 000421
(1) 034132 121527 000060
(1) 034136 002406
(1) 034140 121527 000071
(1) 034144 003367
(1) 034146 012704 000002
(1) 034152 000410
(1) 034154 121527 000055
(1) 034160 001757
(1) 034162 121527 000015
(1) 034166 001356
(1) 034170 012704 000006
3305 034174 000174 034200
3306
3307 034200 034210
3308 034202 034250
3309 034204 034414
3310 034206 034454
3311
3312 034210 010503
3313 034212
(2) 034212 121527 000040
(2) 034216 001405
(2) 034220 121527 000015
(2) 034224 001402
(1) 034226 005205
(1) 034230 000770
3314 034232 010346
3315 034234 160503
3316 034236 005403
3317 034240 010346
3318 034242 010446
3319
3320 034244 005202
3321 034246 000712
```

```

; IF LXSCAN DETECTS AN ERROR CONDITION, IT RETURNS WITH THE CARRY (C)
; BIT SET; OTHERWISE IT IS CLEAR. AT THE MOMENT, THE
; ONLY ERROR CONDITION DETECTED BY LXSCAN IS A STRING WHICH IS MISSING
; ITS CLOSING DELIMITER.

LXSCAN:
        CLR      R2
        MOV      (SP)+,R5
NXTTEL: SPAN     R5,<#>
64$:    CMPB     (R5)+,#'
        BEQ      64$
        DEC      R5
        SYNCLS   @R5,R4
        CMPB     @R5,#'A
        BLT      67$
        CMPB     @R5,#'Z
        BGT      66$
65$:    CLR      R4
        BR       69$
66$:    MOV      #4,R4
        BR       69$
67$:    CMPB     @R5,#'0
        BLT      68$
        CMPB     @R5,#'9
        BGT      66$
        MOV      #2,R4
        BR       69$
68$:    CMPB     @R5,#'-
        BEQ      65$
        CMPB     @R5,#15
        BNE      66$
        MOV      #6,R4
        JMP      @SJT(R4)

SJT:    .WORD    SCWORD
        .WORD    SCNO
        .WORD    SCCHAR
        .WORD    SCEOL

SCWORD: MOV      R5,R3
        BREAK    R5,<#>,<#15>>
        CMPB     @R5,#'
        BEQ      65$
        CMPB     @R5,#15
        BEQ      65$
        INC      R5
        BR       64$
        MOV      R3,-(SP)
        SUB      R5,R3
PLAC:   NEG      R3
        MOV      R3,-(SP)
        MOV      R4,-(SP)

LXINCN: INC      R2
        BR       NXTTEL

; **ENTRY POINT**
; CLEAR #OBJECTS.
; GET ADDR. OF COMMAND BUFFER.
; R5 = ADDR (1ST NONBLANK CHAR)

; DETERMINE SYNTACTIC CLASS.

; PROCESS OBJECT

; WORD PROCESSOR.
; NUMBER PROCESSOR.
; CHARACTER STRING PROCESSOR.
; END OF LINE PROCESSOR.

; R3 = START ADDR OF OBJECT
; R5 = ADDR (NEXT BLANK OR CR)

; PUSH ADDR OF OBJECT ONTO STACK.
; R3 = (-LENGTH OF OBJECT)
; NEGATE TO GET LENGTH.
; PUSH LENGTH ONTO STACK.
; PUSH SYNTACTIC CLASS ONTO
; STACK
; INCREMENT
; GO TO SCAN NEXT ELEMENT
```

3322									
3323	034250	005000		SCNO:	CLR	R0			:CLEAR ACCUMULATED NO.
3324	034252	010504			MOV	R5,R4			:SAVE POINTER TO 1ST DIGIT.
3325	034254	121527	000060	SNXTDG:	CMPB	@R5,#'0			:IF CHAR < '0'
3326	034260	002415			BLT	SNTDIG			: THEN TREAT AS DELIMITER.
3327	034262	121527	000071		CMPB	@R5,#'9			:IF CHAR > '9'
3328	034266	003012			BGT	SNTDIG			: THEN TREAT AS DELIMITER.
3329	034270				MULT	10.,R0,R3			:MULTIPLY PREVIOUS DIGITS BY 10.
(2)	034270	006300			ASL	R0			
(2)	034272	010003			MOV	R0,R3			
(2)	034274	006300			ASL	R0			
(2)	034276	006300			ASL	R0			
(2)	034300	060300			ADD	R3,R0			
3330	034302	112503			MOVB	(R5)+,R3			
3331	034304	142703	000060		BICB	#60,R3			:CLEAR TOP BITS OF ASCII CODE.
3332	034310	060300			ADD	R3,R0			:ADD DIGIT.
3333	034312	000760			BR	SNXTDG			:GET NEXT DIGIT.
3334									
3335	034314	121527	000040	SNTDIG:	CMPB	@R5,#'			:IF DELIMITER = SPACE, TRY
3336	034320	001413			BEQ	TRYOCT			: CONVERTING OCTAL NO.
3337	034322	121527	000015		CMPB	@R5,#15			:IF DELIMITER = CR, TRY
3338	034326	001410			BEQ	TRYOCT			: CONVERTING OCTAL NO.
3339	034330	122527	000056		CMPB	(R5)+,#'.			:IF DELIMITER IS NOT DOT,
3340	034334	001052			BNE	LXERR			: THEN SIGNAL LXSCAN ERROR.
3341	034336	010046		PUSHNO:	MOV	R0,-(SP)			:PUT CONVERTED NO. ON STACK.
3342	034340	005046			CLR	-(SP)			:SET OBJECT LENGTH TO ZERO.
3343	034342	012746	000002		MOV	#2,-(SP)			:SET OBJECT CLASS TO 2
3344									: (NUMBER).
3345	034346	000736			BR	LXINCN			:INCREMENT OBJ. COUNT &
3346									: SCAN NEXT OBJ.
3347									
3348	034350	005000		TRYOCT:	CLR	R0			:CLEAR ACCUMULATED NO.
3349	034352	121427	000060	ONXTDG:	CMPB	@R4,#'0			:IF CHAR < '0'
3350	034356	002413			BLT	ODELIM			: THEN TREAT AS DELIMITER.
3351	034360	121427	000067		CMPB	@R4,#'7			:IF CHAR > '7'
3352	034364	003010			BGT	ODELIM			: THEN TREAT AS DELIMITER.
3353	034366				MULT	8.,R0			:MULTIPLY PREVIOUS DIGITS BY 8.
(2)	034366	006300			ASL	R0			
(2)	034370	006300			ASL	R0			
(2)	034372	006300			ASL	R0			
3354	034374	112403			MOVB	(R4)+,R3			:GET CHARACTER.
3355	034376	142703	000060		BICB	#60,R3			:CLEAR TOP BITS OF ASCII CODE.
3356	034402	060300			ADD	R3,R0			:ADD DIGIT.
3357	034404	000762			BR	ONXTDG			:GET NEXT DIGIT.
3358									
3359	034406	020405		ODELIM:	CMP	R4,R5			:IF NOT AT END OF NO.(DUE TO
3360	034410	001024			BNE	LXERR			: '8' OR '9') , LXSCAN ERROR.
3361	034412	000751			BR	PUSHNO			:GENERATE OBJECT FOR NUMBER.
3362									
3363	034414	112500		SCCHAR:	MOVB	(R5)+,R0			:R0 = DELIMITER OF STRING.
3364	034416	010503			MOV	R5,R3			:R3 = ADDR (1ST CHAR OF
3365									: STRING ITSELF)
3366	034420				BREAK	R5,<R0,#15>			:R5 = ADDR (NEXT DELIM OR CR).
(2)	034420	121500			CMPB	@R5,R0			
(2)	034422	001405			BEQ	65\$			
(2)	034424	121527	000015		CMPB	@R5,#15			

(2)	034430	001402		BEQ	65\$	
(1)	034432	005205		INC	R5	
(1)	034434	000771		BR	64\$	
3367	034436	121527	000015	(MPB	@R5,#15	;WAS CR FOUND BEFORE DELIMITER?
3368	034442	001407		BEQ	LXERR	;YES, PROCESS ERROR
3369	034444	010346		MOV	R3,-(SP)	;PUSH ADDR. OF STRING ONTO
3370						; STACK.
3371	034446	160503		SUB	R5,R3	;R3 = (-LENGTH OF STRING EXCL.
3372						; DELIMITER)
3373	034450	005205		INC	R5	;R5 = ADDR (CHAR AFTER CLOSING
3374						; DELIMITER).
3375	034452	000671		BR	PLAC	;FINISH OFF PROCESSING STRING.
3376						
3377	034454	010246		SCEOL: MOV	R2,-(SP)	;PUSH NO. OF OBJECTS ONTO STACK
3378	034456	000241			CLC	;INDICATE NO ERROR.
3379	034460	000111		LXIT: JMP	@R1	;RETURN TO CALLING PROGRAMME.
3380						
3381				; PROCESS LEXICAL SCAN ERROR .		
3382	034462	000261		LXERR: SEC		;INDICATE ERROR OCCURRED .
3383	034464	005302			DEC R2	;CLEAN UP STACK & EXIT .
3384	034466	002774			BLT LXIT	; ..
3385	034470	062706	000006		ADD #6,SP	; ..
3386	034474	000772			BR LXERR	; ..
3387						
3388						
3389		000001			.END	

[illegible]

CPALP	004552	771#	786			
CPARTN	004634	788#	790			
CPASS	005704	1023#	2479			
CPCLR	004764	834#	2482			
CPCLRC	005014	843#	850			
CPCLRT	005056	854#	856			
CPCLNRT	005212	888	890#	892		
CPCNT	005126	881#	2485			
CPDEL	004644	794#	2488			
CPDLP	004660	799#	819			
CPDRTN	004754	821#	823			
CPERR	006612	1177#	2491			
CPERTN	006730	1188	1194#	1197		
CPGO	005522	982#	2494			
CPINIT	005222	905#	2497			
CPINRT	005460	947#	949			
CPMAST	004130	649#	2500			
CPMLP	003520	583#	599			
CPMOK	004144	651	654#			
CPMRET	004212	653	663#			
CPRANG	004400	723#	2503			
CPRFIL	004502	747#	751			
CPRIB	004306	694#	2506			
CPROK	004322	696	699#			
CPRRET	004376	698	711#			
CPRRTN	004526	753#	756			
CPSEC	004214	669#	2509			
CPSEX	004076	633#	636	642		
CPSILO	003420	565#	2512			
CPSLV	004034	621	627#			
CPSOK	004230	671	674#			
CPSRET	004300	673	684#	688		
CPSRTN	005116	868#	870			
CPSTAT	005066	862#	2515			
CPSUM	005470	956#	2518			
CPSURT	005512	961#	963			
CRRET	015100	2080	2085#			
CR.	= 000015	333#	2081	2272		
CTLCMG	016412	2126	2336#			
CTLDMG	016404	2094	2334#			
CTLUMG	016'07	2135	2335#			
CTL.C	= 000003	336#	2270			
CTL.O	= 000017	334#	2273			
CTL.Q	= 000021	337#	2274			
CTL.S	= 000023	338#	2275			
CTL.U	= 000025	335#	2276			
CURAD	017752	1349*	1350	1477	1502	2400#
CYCL	007362	1302	1304#			
DATBUF	020666	1272	1347	2433#		
DATGEN	007136	506	1258#			
DATGEV	017660	504	946*	1285*	1367*	2367#
DBCLP	011110	1560#	1562			
DBFCLR	011074	1547	1557#			
DDIV	033256	2839	2888#			
DECJSP	032716	1852	1903	1955	2720#	
DECPNT	032724	1783	1787	1791	1795	2726#

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

CZPLABO PCL11 EXERCISER V02 PCLEXR.P11 12-SEP-78 15:13		MACY11 30A(1052) 20-OCT-78 09:48 PAGE 50-8 CROSS REFERENCE TABLE -- USER SYMBOLS													SEQ 0138	
		1028	1033	1038	1041	1049	1054	1057	1065	1070	1073	1081	1086	1089		
SYNXR	006316	1100#														
TABLAD=	000024	3150#	3165													
TCBFPT	030270	2458#														
TCBIN	030272	2469#														
TCR	016076	388*	867*	910*	1343*	1359*	1360*	1382*	1444	1453*	1462*	1487*	1513*	1528*		
TEMP	017764	1981	2283#													
TEMP1	017766	1981*	1982*	1983*	1999*	2000*	2001*	2002*	2003*	2019*	2020*	2021*	2038*	2039*		
TEREV	017652	2040*	2041*	2042*	2405#											
TERHOR	032266	1985*	1996*	1997*	1998*	2001	2023*	2034*	2035*	2036*	2037*	2040	2406#			
TEROS	013100	525	943*	1189*	1831*	2154*	2364#									
TERTBL	025150	1826	2569#													
TERTN	013436	527	1819#													
THFND	033772	925	1180	1192	2002	2458#										
THLTFL	017722	1822	1828	1841	1859	1862#										
THUMST	031561	3183	3195#													
THUSCN	031607	2107*	2117*	2216	2228*	2384#										
TIAREA	016554	1769	2552#													
TICKS	017642	1772	2553#													
TIEND =	016624	2350#														
TIQ	016540	913*	989*	1701*	1702	1704*	1794	2358#								
TISIZE=	000024	2350#														
TMLIN1	031741	433*	492	2054	2231	2350#										
TMMR	016110	306#	2350													
TMMRH	016112	1781	2557#													
TOAREA	016640	398*	614*	2288#												
TOEND -	017640	400*	607*	620	626*	631*	639*	658	661*	678	682*	1393	1526*	1767		
TOQ	016624	1770	2289#													
TORET	015640	2350#														
TOSIZE=	000400	2350#														
TRAP4	002702	433*	498	1729	1820	1874	1927	2092*	2093*	2157	2158	2159	2160	2180		
TRELIN	032362	2204	2219	2248	2260	2261	2262	2350#								
TRERC	032416	2215	2218	2221#												
TRHLIN	032312	307#	2350													
TRNKF	017742	465	479#													
TRPDMG	031105	1853	2571#													
TRP4AD	031140	1850	2574#													
TRRCN	032377	1827	2570#													
TRRNO	032362	471*	1261*	1267*	1271*	1470	2396#									
TRYOCT	034350	484	2542#													
TSBA	016106	481	2543#													
TSBC	016104	1847	2573#													
TSCRC	016114	1844	2572#													
TSDB	016102	3336	3338	3348#												
TSR	016100	396*	1347*	2287#												
TTDEV -	177560	394*	1346*	2286#												
TTIINT	015642	402*	2290#													
TTINH	014764	392*	2285#													
TTINP	014712	390*	1384	1387	1390	1396	1410	1414	1418	1420	1436	1442	1446	1450		
TTOUT	015572	1460*	1469*	1472	1476*	1491	1495*	1511*	2284#							
TTPRIO-	000004	295#	2301	2302	2303	2304										
		437	2227#													
		2058	2060	2062#												
		494	2052#													
		500	2213#													
		297#	443													

TTRBUF	016140	2230	2302#											
TTRCSR	016136	467*	540*	2301#										
TTRET	015036	2067	2070#											
TTVCTR=	000060	296#	2305											
TTVECT	016146	437*	442	2305#										
TTXBUF	016144	2220*	2304#											
TTXCSR	016142	2214	2303#											
TXBSY =	000020	1376#	1396	1446	1460									
TXDEV	016166	387	1090*	1093*	2324#									
TXDGEN	007566	1365	1367#											
TXER =	100000	1375#	1384											
TXMEV	017674	507	940*	1226*	1284*	1342*	1399*	1454*	1461*	1486*	1512*	2373#		
TXMIT	007402	509	1339#											
TXML	017744	1276*	1346	2397#										
TXMST	007400	703	709*	915*	1337#	1360	1773							
TXMSV	017676	2374#												
TXMVEC	016132	403*	435*	438	2298#									
TXOUT	007544	1345	1363#											
TXPRIO=	000005	302#	439											
TXRTN	007542	1341	1361#	1366	1368									
TXSOR =	000040	1378#	1390	1472	1476	1491	1495	1511						
TXSUC -	000200	1377#	1387	1469										
TXVEC	016170	403	1074*	1094*	2325#									
TXVECT-	000170	301#												
X	000040	2418#	2427#											
XMTIB	007700	1394	1396#											
XMTINT	007600	435	1381#											
XMTIS	007634	1385	1387#											
XMTPNM	007664	1391	1393#											
XMTPRB	007714	1397	1399#											
XMTSR	007650	1388	1390#											
XSPCEV	017706	473*	522	1161*	1527*	2378#								
XSPEC	006536	524	1157#											
Y	- 000031	2442#	2450#											
ZERO	016154	2262	2312#											
Zx1	- 000000	565#	578#	649#	669#	694#	723#	766#	794#	834#	840#	862#	881#	905#
		909#	917#	956#	982#	1023#	1110#	1137#	1157#	1177#	1212#	1258#	1339#	1545#
		1728#	1819#	1873#	1926#	1976#	1997#	1998#	2000#	2014#	2036#	2037#	2039#	2052#
		2078#	2091#	2106#	2116#	2125#	2134#	2147#	2171#	2190#	2213#	2244#	2259#	2961#
		2981#	3079#	3329#	3353#									
		578#	1997#	1998#	2000#	2036#	2037#	2039#	3329#	3353#				
Zx2	- 000000	2709#	2715#	2721#	2727#									
\$BLK	= 000000	2833#												
\$BLKS	= 002000													
\$SGN	= 000000	2709#	2715#	2721#	2727#									
\$SGNS	= 000000	2833#												
\$SUP	- 000000	2709#	2715#	2721#	2727#									
\$TR	002550	453	460#											
.	034476	370#	374	376#	381#	382	383	1279	1400	1412	1416	1424	1428	1432
		1434	1438	1448	1452	1474	1493	1517	1586	1595	1599	1607	1612	1614
		1618	1622	1669	1672	1676	1685	2330#	2331	2350#	2361#	2414#	2428	2433#
		2438#	2458#	2461#	2589#	2833								

BDINIT	229#	910	1343	1453	1462	1487	1513	1548	1631	1680	1690				
BREAK	3020#	3313	3366												
CALL	129#	444	445	446	447	452	474	475	484	494	497	500	503	506	509
	512	515	518	521	524	527	530	537	622	623	624	625	629	885	886
	916	995	996	997	1113	1115	1116	1119	1138	1139	1140	1145	1158	1159	1160
	1186	1232	1238	1239	1240	1348	1400	1412	1416	1424	1428	1432	1434	1438	1448
	1452	1474	1493	1517	1586	1595	1599	1607	1612	1614	1618	1622	1669	1676	1685
	1763	1769	1772	1775	1777	1797	1798	1806	1826	1827	1838	1839	1853	1880	1881
	1888	1889	1891	1904	1956	1960	1968	2054	2066	2069	2084	2094	2126	2135	2157
	2158	2159	2160	2176	2177	2180	2204	2219	2231	2248	2260	2261	2262		
CBTAS	2807#	2833													
ERROR	260#	1586	1595	1599	1607	1612	1614	1618	1622	1669	1676	1685			
ERROT	248#	1400	1412	1416	1424	1428	1432	1434	1438	1448	1452	1474	1493	1517	
HEDING	200#														
KEYWD	44#	656	676	701	3108										
LXSCAN	69#	3099													
MULP	217#	1279	1672												
MULT	151#	578	1997	1998	2000	2036	2037	2039	3329	3353					
PROC	85#	565	649	669	694	723	766	794	834	840	862	881	905	909	917
	956	982	1023	1110	1137	1157	1177	1212	1258	1339	1545	1728	1819	1873	1926
	1976	2014	2052	2078	2091	2106	2116	2125	2134	2147	2171	2190	2213	2244	2259
	2961	2981	3079												
REGRES	280#	542	1227	1401	1455	1463	1488	1514	1529	1563	1588	1629	1643	1681	1692
	2232	3207													
REGSAV	274#	536	1213	1381	1557	1573	2229	3155							
RETURN	110#	633	663	684	711	753	788	821	854	868	890	947	961	1001	1101
	1120	1146	1162	1194	1228	1286	1361	1553	1807	1862	1913	1969	2008	2047	2070
	2085	2095	2108	2118	2163	2181	2205	2221	2263	2783	2869	2906	2939	2952	2975
	2996	3116													
SPAN	3008#	3303													
STCVT	2661#	2709	2715	2721	2727										
SYNCLS	3041#	3304													

. ABS. 034476 000

ERRORS DETECTED: 0

PCLEXR,PCLEXR/CR/NL:TTM:ME<PCLEXR
 RUN-TIME: 11 15 1 SECONDS
 RUN-TIME RATIO: 137/28 4.8
 CORE USED: 10K (19 PAGES)