

PCL11-A,B

PCL11 EXERCISER VO2C
CZPLAC0

AH-E260C-MC

COPYRIGHT 78-79

FICHE 1 OF 1

SEP 1979

digital

MADE IN USA

The main body of the document consists of a grid of 150 small, illegible data tables or charts, arranged in 10 columns and 15 rows. Each cell in the grid contains a small table with multiple columns and rows of text, which is too small to read. The tables appear to be organized in a structured manner, possibly representing a series of related data points or exercises. The overall appearance is that of a technical manual or a data reference sheet.

I D E N T I F I C A T I O N

PROGRAM CODE: AC-E259C-MC
PROGRAM NAME: CZPLACO PCL11 EXERCISER V02C
DATE CREATED: 21-SEP-76
UPDATED (V02): 13-MAR-78
MODIFIED (V02A): 11-SEP-78 BY D.WIENS
(V02C) 07-JUN-79 BY D.WIENS
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) 1978,1979 BY DIGITAL EQUIPMENT OF CANADA, LIMITED.

PARALLEL COMMUNICATIONS LINK (PCL11) EXERCISER

	TABLE OF CONTENTS	PAGE
	-----	----
1	GENERAL	1-1
1.1	GENERAL DESCRIPTION	1-1
1.2	REVISION HISTORY	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 CONTENTS (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, ALL PCL11 TRANSMITTERS INVOLVED WILL BEGIN SENDING RANDOM DATA PATTERNS TO EACH RECEIVER THEY HAVE 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 UPON 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 REVISION HISTORY

1.2.1 VERSION V02

1.2.1.1 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.

1.2.1.2 UPPER/LOWER CASE INPUT (SEE SEC. 5.4)
EITHER UPPER OR LOWER CASE ALPHA CHARACTERS ARE ACCEPTED BY V-2 OF THE EXERCISER. WHICHEVER IS TYPED IN, THE UPPER CASE VERSION IS ECHOED.

1.2.1.3 ERRORS COMMAND (SEE SEC. 3.2.12)
THE 'ERRORS' COMMAND HAS BEEN ADDED TO THE EXERCISER TO GIVE MORE DETAIL ABOUT THE ERROR CONDITIONS.

1.2.1.4 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.

1.2.1.5 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.

1.2.1.6 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.

1.2.1.7 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.

1.2.1.8 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.

1.2.1.9 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.

1.2.1.10 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 \$.

1.2.1.11 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.

1-3

1.2.1.12 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.

1.2.1.13 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.

1.2.2 VERSION 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 CONFUSION ON SYSTEMS WHERE PDP-11'S ARE ATTACHED TO VAX/1170'S. (IT IS ONLY THE DIAGNOSTIC SUPERVISOR ON THE 'VAX' THAT IS ALLOWED TO USE THE PROMPT '\$').

1.2.3 VERSION V02C

1.2.3.1 ALIGN THE VERSION LETTER WITH THE ONE IN THE PRODUCT CODE, MAKING THIS V02C RATHER THAN V02B.

1.2.3.2 CHANGE THE HANDLING OF RECEIVER 'BYTE COUNT OVERFLOW' INTERRUPT SO THAT A SUCCESSFUL TRANSFER INTERRUPT ISN'T LOST. THIS PREVENTS THE EXERCISER FROM HANGING UP A RECEIVER.

1.2.3.3 MODIFY THE COUNT IN THE DATA GENERATION TO ACCOUNT FOR THE FIRST WORD IN THE RECEIVER NOT BEING 'NPR'D'. THE COUNT (MAX WITHOUT EXPECTING A TRUNCATE) IS NOW #602 OCTAL.

1.2.3.4 MODIFY THE MESSAGE PRINTED WHEN THERE ARE NO ERRORS TO REMOVE THE WORD 'YET'. MESSAGE NOW READS:
 'NO ERRORS TO REPORT'

1.2.3.5 MODIFY THE MESSAGE PRINTED IF THE EXERCISER IS STARTED BEFORE BEING GIVEN ANY RECEIVER ADDRESSES. MESSAGE NOW READS:
 'NO RECEIVERS SELECTED'
 ALSO, DO A <CR>, <LF> BEFORE IT SO THAT THE 'GO' DOESN'T GET OVERWRITTEN.

1.2.3.6 REMOVE THE CODE WHICH USED TO COMPUTE THE RESTART ADDRESS BASED ON WHETHER THE DEVICE ADDRESSES WERE VALID. RESTART ADDRESS IS NOW ALWAYS '204'.

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,TIK)...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
OF ATTEMPTS. ^I 1

SEQ 0008

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 **

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, THY 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:

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

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 INDICATION THAT THE 'MASTER SET' COMMAND WAS OR WASN'T SUCCESSFUL.

3.2.3 SECONDARY

[S]ECONDARY [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] [S]ET OR [C]LEAR

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.

3.2.5 RANGE

[R]ANGE [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

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

ADD A B C

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 AN EFFICIENT LIST OF THOSE RECEIVER ADDRESSES WHICH ARE TO BE COMMUNICATED WITH BY THIS TRANSMITTER.

3.2.7 DELETE

[DELETE] D [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]NITIALIZE

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

[SU]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"

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

GO -

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 ABLE TO RUN AND WILL PRINT:

*** NO RECEIVERS SELECTED ***

3.2.15 CONTINUE

[C]ONTINUE

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:

*** NO RECEIVERS SELECTED ***

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 FROM 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 (CZPLBCO) 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 ARE MORE THAN ONE PCL11 UNIT HOSTED BY ONE PDP-11 AND EACH ONE MUST BE TESETD "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.

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.
 - 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).

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

I PRCTLC I
I PRDEL I
I PRCTLS I
I PRCTLQ I
I PRCHR-LF I
I PRCTLU I
I PRCTLO I

---[A]

I
TOQ NOT
EMPTY

I DEQUEUE TOQ I
I TO TTY IF I
I READY I

---[A]

I
QLDEV
EVENT SET

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

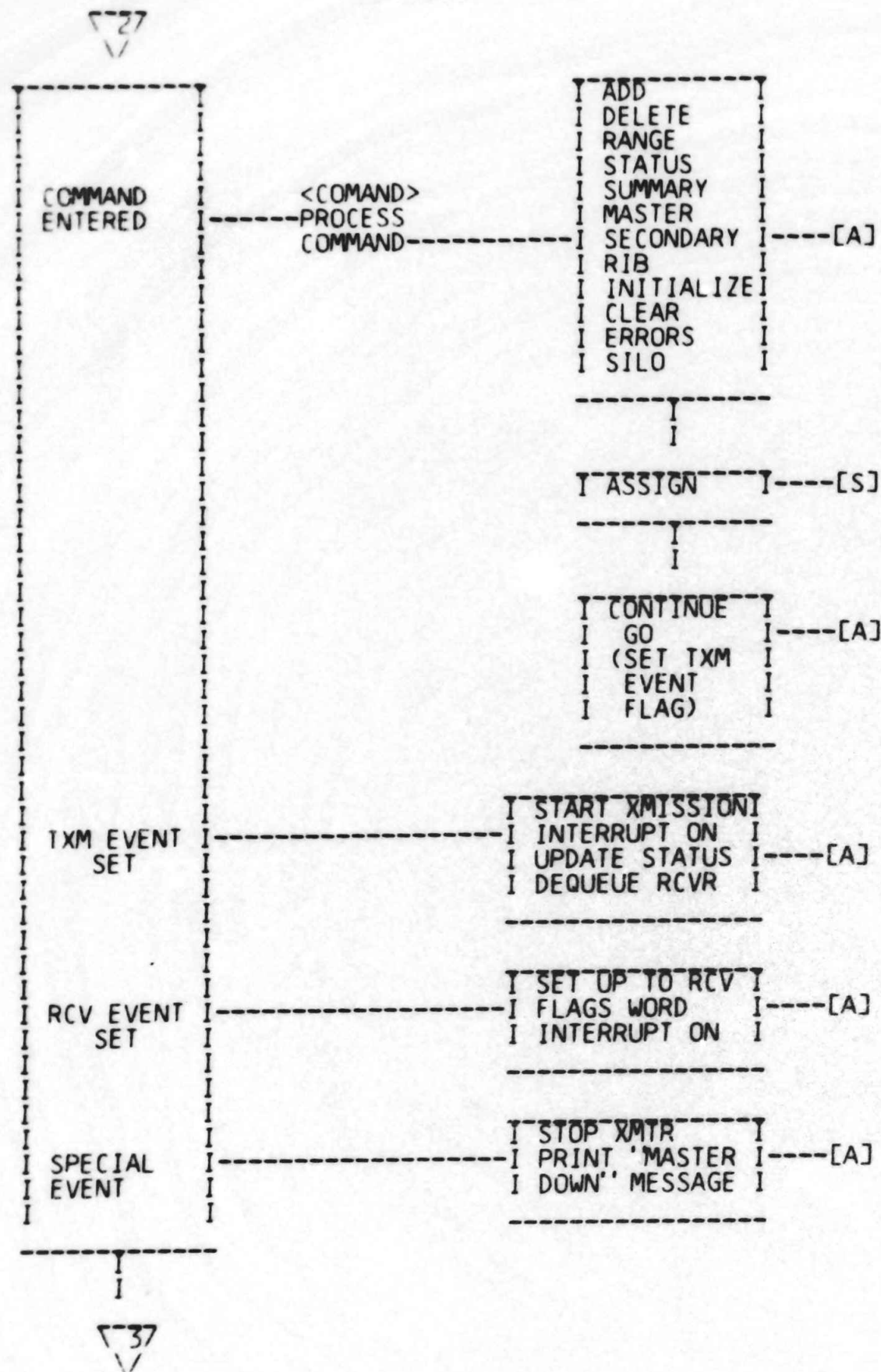
---[A]

I
DATGEN
EVENT SET

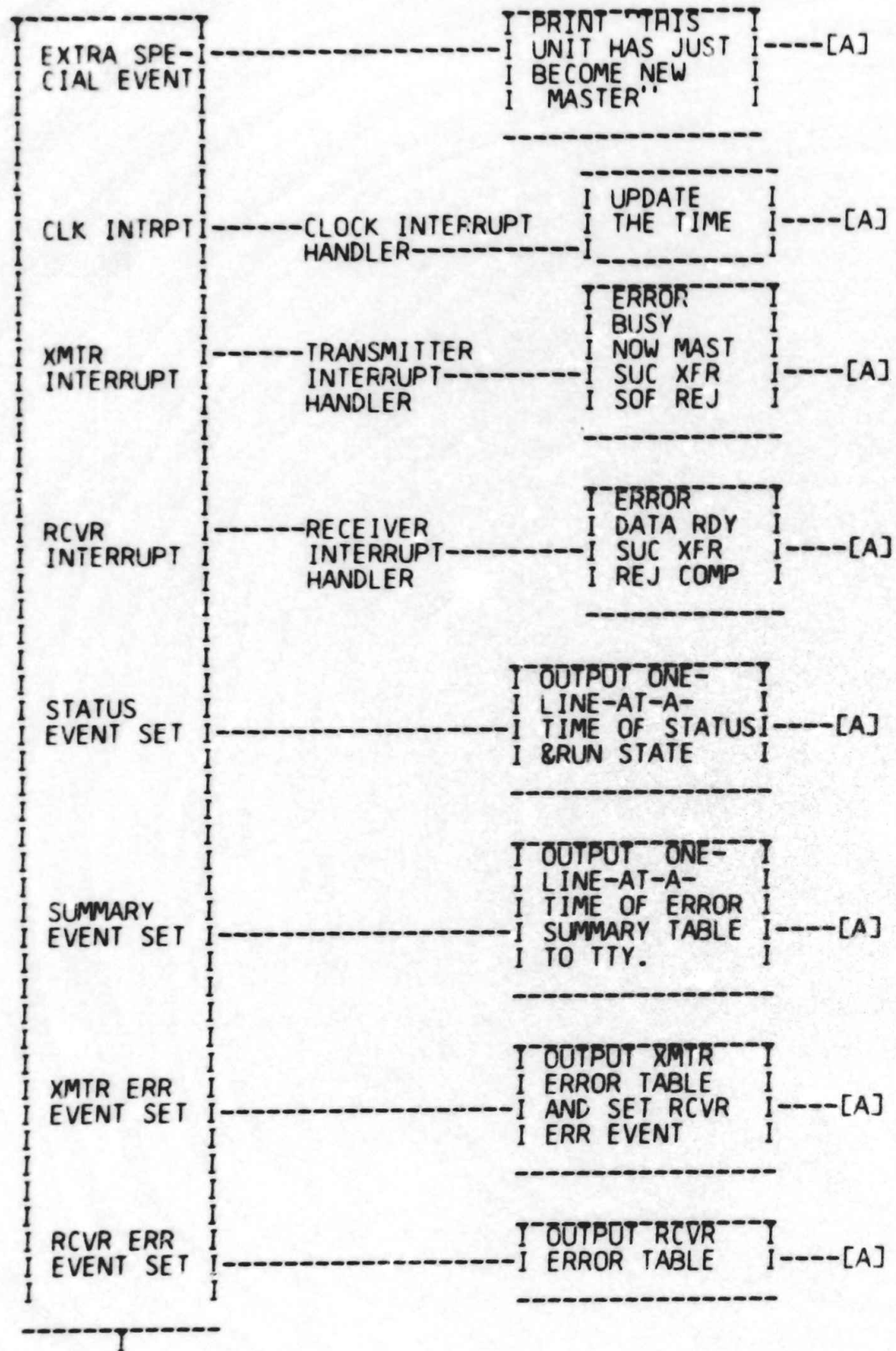
I GENERATE NEW I
I RANDOM DATA I
I TABLE FOR NEXT I
I 5 CYCLES. I

---[A]





37
V



i
[A]

J 3

SEQ 0035

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	34456

LOCATIONS TO CHANGE FOR
DIFFERENT DEVICE ADDRESSES

DEVICE	CHANGE LOCATION
KEYBOARD STATUS	16146
KEYBOARD DATA	16150
TTY STATUS	16152
TTY DATA	16154
KEYBOARD VECTOR	16156
LINE CLK STATUS	16160
XMTR PRIORITY	2326
RCVR PRIORITY	2340
KBD PRIORITY	2352
USEFUL LOCATIONS	ADDRESS
XMTR DATA BUFFER	20676
RCVR DATA BUFFER	22736
DATA SEED IN TRANSMITTER	17736
RANDOM MULTIPLIER	17756
RANDOM INCREMENT	17760

286	DEFINITIONS AND DEVICE INFO
369	PCL11 EXERCISER MAIN PROCEDURE
486	MAIN LOOP
543	COMMAND PROCESSORS:
544	COMMAND PROC. FOR SILO (LOAD)
641	COMMAND PROC. FOR MASTER, SECONDARY AND R.I.B.
710	COMMAND PROC. FOR RANGE
755	COMMAND PROC. FOR ADD AND DELETE
822	COMMAND PROC. FOR CLEAR, STATUS, AND CONTINUE
891	COMMAND PROC. FOR INIT, SUMMARY, AND GO
1005	COMMAND PROC. FOR "ASSIGN"
1100	COMMAND DECODER AND PROCESSOR
1196	RECEIVER ADDRESS QUEUE LOADER ROUTINE
1240	DATA GENERATION (RANDOM) ROUTINE
1285	MULTIPLY ROUTINE FOR DATA GENERATION
1319	TRANSMIT MODULE
1531	RECEIVER MODULE
1714	STATUS MODULE
1807	TRANSMITTER ERRORS MODULE
1862	RECEIVER ERRORS MODULE
1913	SUMMARY MODULE
1969	ERROR UPDATE ROUTINES
1970	TRANSMITTER ERRORS
2008	RECEIVER ERRORS
2047	UTILITY ROUTINES
2048	PROCESS AN INPUT CHARACTER FROM THE TTY
2070	TTY INPUT CHARACTER PROCESSING ROUTINES
2206	TTY OUTPUT HANDLERS
2224	TTY INPUT INTERRUPT PROCESSORS
2235	MESSAGE PRINT ROUTINE
2264	DATA AREAS
2471	KEYWORD TABLE
2532	SOME MORE ASCII STORAGE:
2588	AUXILIARY ROUTINES
2589	CHARACTER PROCESSOR
2632	BINARY TO ASCII CONVERSION
2735	GENERAL BINARY TO ASCII CONVERSION
2784	DOUBLE PRECISION BINARY TO ASCII
2871	DOUBLE PRECISION DIVIDE ROUTINE
2907	INTEGER DIVIDE MAGNITUDE NUMBERS
2952	QUEUE HANDLING ROUTINES
3075	COMMAND PROCESSOR INITIATING ROUTINE
3120	KEYWORD PROCESSING ROUTINE
3209	REGISTER SAVE & RESTORE ROUTINES
3236	LEXICAL SCAN ROUTINE

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

.TITLE CZPLACO PCL11 EXERCISER V02C
.IDENT /0003/ ;DRCMAC.MAC 6-JAN-76
.NLIST TTM
.NLIST ME

: 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

57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112

```
JSR PC,KEYWD  
.ENDM ;KEYWD
```

LXSCAN MACRO

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

```
.MACRO LXSCAN STR  
MOV STR,R1  
JSR R1,LXSCAN  
.ENDM ;LXSCAN
```

PROC MACRO

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

```
.MACRO PROC PNAME,PARAMS  
ZX1 = 0  
.IRP ZX2 <PARAMS>  
ZX1 = ZX1+2  
.IRP ZX3 \ZX1  
ZX2 = ZX3  
.NLIST  
.ENDM  
.ENDM ;ZX2
```

```
.LIST  
PNAME: ;**ENTRY POINT**  
.NLIST  
.ENDM ;PROC
```

RETURN MACRO

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

```
.MACRO RETURN ANSWR
```

113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168

```

    .IF      NB      ANSWR
    MOV      ANSWR,R0
  .ENDC

    RTS      PC
  .ENDM     ;RETURN

; -----
;
;      CALL MACRO
;
; THIS MACRO CALLS A PROCEDURE, SUBR, WITH AN ARGUMENT LIST SPECIFIED
; BY ARGS.  ARGS IS A LIST OF ADDRESSES WHICH WILL BE INCLUDED
; IN THE ASSEMBLED EXPANSION OF THIS MACRO.
;
; THE CALLING SEQUENCE GENERATED IS FORTRAN COMPATIBLE.  R5 IS LEFT
; INTACT THROUGH THE EXECUTION OF THIS MACRO.  OTHER REGISTERS
; ARE DESTROYED.
;
;      .MACRO CALL  SUBR,ARGS,?PLIST,?ZXCALL
;      JSR      R5,ZXCALL
PLIST: BR      ZXCALL
  .IF      NB      <ARGS>
    .WORD    ARGS
  .ENDC
ZXCALL: JSR      PC,SUBR
        MOV      (SP)+,R5
;
;      .ENDM     ;CALL
;
; -----
;
; MACRO TO MULTIPLY A NUMBER BY A CONSTANT.
; THE NUMBER IN DST IS MULTIPLIED BY THE VALUE OF THE EXPRESSION
; CONST; THE RESULT IS LEFT IN DST.  A TEMPORARY LOCATION MAY
; BE SPECIFIED AT WORK WHICH WILL BE USED IN THE MACRO EXPANSION
; IF NECESSARY.  IF WORK IS NOT SPECIFIED & A TEMPORARY LOCATION
; IS NEEDED, A STACK ELEMENT WILL BE ALLOCATED (& SUBSEQUENTLY
; DEALLOCATED) FOR THE PURPOSE.  THE MACRO GENERATES A SERIES OF
; SHIFT & ADD INSTRUCTIONS IN-LINE TO ACCOMPLISH THE MULTIPLICATION.
;
;      .MACRO MULT  CONST,DST,WORK
ZX1   =      0          ;FLAG: 0==>LEAST SIG 1-BIT NOT TESTED
;          ;          YET; 1==>OPPOSITE.
ZX2   =      CONST     ;COPY CONSTANT FOR SHIFTING.
  .IF      Z          ZX2
    CLR      DST
    .MEXIT
  .ENDC

  .REPT    16.
  .IF      NZ          ZX2&1          ;IF BIT 0 = 1
ZX2    =      ZX2/2      ;SHIFT CONSTANT RIGHT 1 POSITION.
  .IF      Z          ZX1          ;IF LEAST SIG 1-BIT
  .IF      NZ          ZX2          ;IF NOT MOST SIG 1-BIT
  .IF      NB         WORK         ;IF WORK SPECIFIED
  
```

169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224

```

      MOV     DST,WORK           ;ELSE IF WORK BLANK
    .IFF
      MOV     DST,-(SP)         ;END [.IF NB WORK]
    .ENDC
    .IFF
      .MEXIT                    ;ELSE IF MOST SIG 1-BIT
    .ENDC
    ZX1 = 1                      ;END [.IF NZ ZX2]
    .IFF
      .IF     NZ     ZX2        ;INDICATE NO LONGER LEAST SIG 1-BIT.
      .IF     NB     WORK      ;ELSE IF NOT LEAST SIG 1-BIT
      ADD     DST,WORK        ;IF NOT MOST SIG 1-BIT
      .IFF
      ADD     DST,WORK        ;IF WORK SPECIFIED
      .IFF
      ADD     DST,@SP         ;ELSE IF WORK BLANK
    .ENDC
    .IFF
      ADD     DST,@SP         ;END [.IF NB WORK]
    .ENDC
    .IFF
      .IF     NB     WORK      ;ELSE IF MOST SIG 1-BIT
      ADD     WORK,DST        ;IF WORK SPECIFIED
      .IFF
      ADD     (SP)+,DST       ;ELSE IF WORK BLANK
    .ENDC
    .MEXIT                      ;END [.IF NB WORK]
    .ENDC
    .ENDC
    .IFF
      ZX2 = ZX2/2              ;END [.IF NZ ZX2]
    .ENDC
      ASL     DST              ;END [.IF Z ZX1]
    .ENDM
    .ENDM :MULT                ;ELSE IF BIT 0 = 0
                                ;SHIFT CONSTANT RIGHT 1 POSITION.
                                ;END [.IF NZ ZX2&1]
                                ;END [.REPT 16.] LOOP.
  ; -----
  .MACRO HEDING NAM,VER,EDIT,PATCH
  .TITLE HEDING
  .IDENT /VER'EDIT'PATCH/
  .CSECT HEDING
  .GLOBL HEDING,HEDLEN
HEDING: .ASCII /NAM'VER'-'EDIT'/
  .IF B PATCH
  .BYTE 40
  .IFF
  .ASCII /PATCH/
  .ENDC
HEDLEN = .-HEDING
  .ENDM :HEDING
;MULTIPLY MACRO FOR UNSIGNED MULTIPLY ROUTINE
  .MACRO MULP 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
  
```

225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280

000001

```
.ENDM ;MULP

;BOARD INIT MACRO FOR CLEARING PCL HARDWARE
;BOARD INIT RECEIVER OR TRANSMITTER.

.MACRO BDINIT DEV
.NLIST
.IF IDN <DEV>,<XMTR>
BIS #B01,@TCR
.IFF
.IF IDN <DEV>,<RCVR>
BIS #B01,@RCR
.IFF .ERROR ;BAD ARGUMENT FOR BDINIT
.ENDC
.ENDC
.LIST
.ENDM

N = 1 ;INITIAL ERROR NUMBER
;ERROR MACROS

.MACRO ERROT P
ERADR =
CALL ERRMOD,<P,ERADR>
N = N+1
.LIST
;***** XMTR ERROR P *****

.NLIST
.ENDM

.MACRO ERROR P
ERADR =
CALL ERRMOR,<P,ERADR>
N = N+1
.LIST
;***** RCVR ERROR P *****

.NLIST
.ENDM

;REGISTER SAVE MACRO

.MACRO REGSAV
JSR R5,REGSAV
.ENDM

;REGISTER RESTORE MACRO
```

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 ^{H 4} PAGE 1-5

SEQ 0046

281
282
283
284

.MACRO REGRES
JSR R5,REGRES
.ENDM

.SBTTL DEFINITIONS AND DEVICE INFO

.IDENT '02'

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

: VARIABLE SYMBOL DEFINITIONS.

: DEVICE DEFAULT INFORMATION.

177560
000060
000004
164200
164220
000174
000170
000005
000005

TTDEV = 177560
TTVCTR = 60
TTPRIO = 4
PCLTXM = 164200
PCLRCV = 164220
RCVECT = 174
TXVECT = 170
TXPRIO = 5
RCPRIO = 5

:ADDR OF RCSR FOR TTY.
:INPUT VECTOR ADDR FOR TTY.
:PRIORITY LEVEL FOR TTY.

: QUEUE SIZES.

000024
000400
000040

TISIZE = 20.
TOSIZE = 256.
AOSIZE = 32.

:TIQ SIZE.
:TOQ SIZE.
:ADR QUEUE SIZE

000000
000002
000004
000006
000010
000012

QELEMS = 0
QSIZE = 2
QTOP = 4
QBOT = 6
QFRONT = 10
QBACK = 12

:#ELEMENTS PRESENTLY IN QUEUE.
:#ELEMENTS IN QUEUE SPACE.
:ADDR OF 1ST WORD OF QUEUE SPACE
:=QTOP+(QSIZE*2)
:ADDR OF FRONT ELEMENT OF QUEUE.
:ADDR OF BACK ELEMENT OF QUEUE.

: REGISTER DEFINITIONS.

000000
000001
000002
000003
000004
000005
000006
000007
177776
177570

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:

000012
000015
000017
000025
000003
000021
000023
000177

LF. = 12
CR. = 15
CTL.O = 17
CTL.U = 25
CTL.C = 3
CTL.Q = 21
CTL.S = 23
RUBOUT = 177

:CR OR LF TO END LINE.
:..
: ^O TO THROW AWAY TTY OUTPUT.
: ^U TO DELETE LINE.
: CNTRL-C TO START INPUT. (^C)
: CNTRL-Q TO RESUME PRINTOUT
: CNTRL-S TO SUSPEND PRINTOUT
: RUB OUT TO DELETE CHARACTER.

286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341

CZPLACO PCL11 EXERCISER VC2C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 J 4
DEFINITIONS AND DEVICE INFO PAGE 2-1

SEQ 0048

342
343
344 100000
345 004000
346 000200
347 000100

: DEVICE BIT DEFINITIONS.
ERR = 100000
BUSY = 4000
DONE = 200
INTENB = 100

;BIT DEFINITIONS:

349				
350				
351	100000	B15	=	100000
352	040000	B14	=	40000
353	020000	B13	=	20000
354	010000	B12	=	10000
355	004000	B11	=	4000
356	002000	B10	=	2000
357	001000	B09	=	1000
358	000400	B08	=	400
359	000200	B07	=	200
360	000100	B06	=	100
361	000040	B05	=	40
362	000020	B04	=	20
363	000010	B03	=	10
364	000004	B02	=	4
365	000002	B01	=	2
366	000001	B00	=	1
367				

```

369          .SBTTL PCL11 EXERCISER MAIN PROCEDURE
370
371          .ENABL ABS
372          000000          =          0
373          000200          .REPT 128.
374          .WORD          .+2,0          ;TRAP CATCHERS
375
376          .ENDR
377
378          =          200
379          000200 000167 001574          JMP          PCLEX          ;PROGRAM STARTS AT 200
380          000204 012706 002000          MOV          #STKTOP,SP          ;AND RESTARTS AT 204
381          000210 000167 002342          JMP          PCREST
382
383          002000          .STKTOP =          2000
384          002000          =
385
386          002000          PCLEX:          ;*****START HERE*****
387          002000 012706 002000          MOV          #STKTOP,SP          ;SET STACK POINTER TO TOP
388          002004 016700 014166          MOV          TXDEV,R0          ;PREPARE TO GENERATE TXM ADDR
389          002010 010067 014072          MOV          R0,TCR          ;GENERATE TCR ADDRESS
390          002014 062700 000002          ADD          #2,R0
391          002020 010067 014064          MOV          R0,TSR          ;GENERATE TSR ADDRESS
392          002024 062700 000002          ADD          #2,R0
393          002030 010067 014056          MOV          R0,TSDB          ;GENERATE TSDB ADDRESS
394          002034 062700 000002          ADD          #2,R0
395          002040 010067 014050          MOV          R0,TSBC          ;GENERATE TSBC ADDRESS
396          002044 062700 000002          ADD          #2,R0
397          002050 010067 014042          MOV          R0,TSBA          ;GENERATE TSBA ADDRESS
398          002054 062700 000002          ADD          #2,R0
399          002060 010067 014034          MOV          R0,TMMR          ;GENERATE TMMR ADDRESS
400          002064 005200          INC          R0
401          002066 010067 014030          MOV          R0,TMMRH          ;AND TMMR HIGH BYTE
402          002072 005200          INC          R0
403          002074 010067 014024          MOV          R0,TSCRC          ;GENERATE TSCRC ADDRESS
404          002100 016767 014074 014034          MOV          TXVEC,TXMVEC          ;GENERATE TXMVEC ADDRESS
405
406          002106 016700 014070          MOV          RCDEV,R0          ;PREPARE TO GENERATE RCVR ADDR
407          002112 010067 014010          MOV          R0,RCR          ;GENERATE RCR ADDRESS
408          002116 062700 000002          ADD          #2,R0
409          002122 010067 014002          MOV          R0,RSR          ;GENERATE RSR ADDRESS
410          002126 062700 000002          ADD          #2,R0
411          002132 010067 013774          MOV          R0,Rddb          ;GENERATE Rddb ADDRESS
412          002136 062700 000002          ADD          #2,R0
413          002142 010067 013766          MOV          R0,RDBC          ;GENERATE RDBC ADDRESS
414          002146 062700 000002          ADD          #2,R0
415          002152 010067 013760          MOV          R0,RDBA          ;GENERATE RDBA ADDRESS
416          002156 062700 000004          ADD          #4,R0
417          002162 010067 013752          MOV          R0,RDCRC          ;GENERATE RDCRC ADDRESS
418          002166 016767 014012 013750          MOV          RCVEC,RCVVEC          ;GENERATE RCVVEC ADDR

```

```
420 002174          PCRST:          ;*****RESTART HERE *****
421 002174 012706 002000          MOV      #STKTOP,SP          ;RESET STACK POINTER
422 002200 012737 003326 000004  MOV      #ERTRAP,@#4        ;SET UP VECTOR FOR ADDRESS ERROR
423 002206 012737 000340 000006  MOV      #340,@#6
424 002214 105067 015430          CLR      REQINP             ;CLR INPUT REQUEST
425 002220 105067 015425          CLR      CMDENT            ;CLR COMMAND ENTERED FLAG
426 002224 005067 015460          CLR      PCLGO             ;CLR XMTR GO FLAG
427          .IRP              LC      <TI,TO>          ;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) 002230 005067 014314          CLR      TIQ
(1) 002234 016767 014314 014316  MOV      TIQ+QTOP,TIQ+QFRONT
(1) 002242 016767 014306 014312  MOV      TIQ+QTOP,TIQ+QBACK
(1) 002250 005067 014360          CLR      TOQ
(1) 002254 016767 014360 014362  MOV      TOQ+QTOP,TOQ+QFRONT
(1) 002262 016767 014352 014356  MOV      TOQ+QTOP,TOQ+QBACK
434 002270 012737 000340 177776  MOV      #340,@#PS          ;DISABLE INTERRUPTS
435 002276 012777 007566 013636  MOV      #XMTINT,@TXMVEC    ;:::SET UP XMTR INTR VECTOR
436 002304 012777 011112 013632  MOV      #RCVINT,@RCVVEC    ;:::SET UP RCVR INTR VECTOR
437 002312 012777 015652 013636  MOV      #TTIINT,@TTVECT    ;:::SET UP TTY INTR VECTOR
438 002320 016700 013616          MOV      TXMVEC,R0
439 002324 012760 000240 000002  MOV      #TXPRIO*32.,2(R0)   ;:::SET TXM PRIORITY
440 002332 016700 013606          MOV      RCVVEC,R0
441 002336 012760 000240 000002  MOV      #RCPRIO*32.,2(R0)   ;:::SET RCVR PRIORITY
442 002344 016700 013606          MOV      TTVECT,R0
443 002350 012760 000200 000002  MOV      #TTPRIO*32.,2(R0)   ;:::SET TTY PRIORITY
444 002356          CALL    PNCRLF
445 002372          CALL    PNCRLF
446 002406          CALL    PNTLIN,<PCLEXM>     ;:::PRINT TITLE MESSAGE
447 002424          CALL    PNCRLF             ;:::<CR>&<LF>
448 002440          CALL    PNTLIN,<RSTMSG>    ;:::ENQUEUE RESTART ADDR MSG
449 002456          CALL    PRINIT            ;:::INITIALIZE TRANSMITTER
450 002472 012737 002536 000004  PRST:  MOV      #STR,@#4      ;:::SET UP TO TEST FOR CLOCK
451 002500 005067 015154          CLR      KWFLG             ;:::CLEAR KW11 FLAG
452 002504 005777 013450          TST      @LCS              ;:::ANY CLOCK?
453 002510 012767 177777 015142  MOV      #-1,KWFLG          ;:::YES, SET KW11 FLAG
454 002516 012737 012224 000100  MOV      #CLKINT,@#100      ;:::SET UP CLK VECTOR
455 002524 012737 000340 000102  MOV      #340,@#102
456 002532 000167 000020          JMP      PCPEST             ;:::AND CONTINUE
457 002536 022626          $STR:  CMP      (SP)+,(SP)+   ;:::NO CLOCK, CLEAR STACK
458 002540 005067 015114          CLR      KWFLG             ;:::CLR KW11 FLAG
459 002544 012737 000102 000100  MOV      #102,@#100         ;:::SET UP TO TRAP HALT
460 002552 005037 000102          CLR      @#102
461 002556 005037 177776          PCREST: CLR @#PS
462 002562 012737 002670 000004  MOV      #TRAP4,@#4
463 002570 052767 100000 015114  BIS      #B15,RCVEV
464 002576 052777 000100 013342  BIS      #B06,@ATTRCSR
465 002604 005067 015156          CLR      ESCFLG
466 002610 005067 015052          CLR      QLDEV
467 002614 005067 015130          CLR      RJCTF
468 002620 005067 015126          CLR      TRNKF
469 002624 005067 015064          CLR      SPCEV
;:::ALLOW INTERRUPTS
;:::CHANGE TRAP VECTOR FOR ERROR
;:::SET RCVR EVENT FLAG
;:::SET TTY KBD INTR ENAB
;:::CLEAR CNTRL-C FLAG
;:::CLR QUEUE LOAD EVENT FLAG
;:::CLEAR REJECT FLAG
;:::CLEAR TRUNCATE FLAG
;:::CLEAR MST DWN EVENT
```

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 5-1
PCL11 EXERCISER MAIN PROCEDURE

SEQ 0052

470	002630	005067	015062
471	002634		
472	002650		
473	002664	000167	000060
474			
475			
476	002670	011667	015000
477	002674	162767	000002
478	002702	012700	031127
479	002706	016701	014762
480	002712	004767	027746
481	002716		
482	002734	012706	002000
483	002740	005037	177776
484	002744	000167	177664

```

RTRYA: CLR XSPCEV
        CALL PNCRLF
        CALL PRESC
        JMP PCLOOP

```

```

;CLR NOW MST EVENT
;ENQUEUE CR, & LF
;ENTER COMMAND MODE!
;GO TO MAIN LOOP

```

```

014772 TRAP4: MOV (SP),SUMSV
          SUB #2,SUMSV
          MOV #TRP4AD,R0
          MOV SUMSV,R1
          JSR PC,OCTJSP
          CALL PNTLIN,<TRPDMG>
          MOV #STKTOP,SP
          CLR @#PS
          JMP RTRYA

```

```

;SAVE TRAP ADDRESS
;ALIGN IT -2
;SHOW OCT CONV RTN RIGHT ADDRESS
; FOR ASCII CHARS.

```

;PRINT TRAP MESSAGE

;LOWER PRIORITY

.SBTTL MAIN LOOP

```
486
487
488
489 002750 005767 013574 PCLOOP: TST TIQ ;IS TTY INPUT QUEUE EMPTY?
490 002754 0014J6 BEQ NXT0 ;YES, TEST ANOTHER FLAG
491 002756 CALL TTINP ;NO, PROCESS A CHARACTER
492 002772 105767 014653 NXT0: TSTB CMDENT ;HAS A COMMAND BEEN ENTERED?
493 002776 100006 BPL NXT1 ;NO, TEST ANOTHER FLAG
494 003000 CALL COMENT ;YES, PROCESS COMMAND
495 003014 005767 013614 NXT1: TST TOQ ;IS TTY OUTPUT QUEUE EMPTY?
496 003020 001406 BEQ NXT2 ;YES, TEST ANOTHER FLAG
497 003022 CALL TTOUT ;NO, OUTPUT A CHAR IF DEV RDY
498 003036 005767 014624 NXT2: TST QLDEV ;IS ADDR QUEUE EMPTY?
499 003042 100006 BPL NXT3 ;NO, TEST ANOTHER FLAG
500 003044 CALL ADQLD ;YES, LOAD ADDR QUEUE
501 003060 005767 014604 NXT3: TST DATGEV ;IS DATA GEN FLAG SET?
502 003064 100006 BPL NXT4 ;NO, TEST ANOTHER FLAG
503 003066 CALL DATGEN ;YES, GENERATE NEW RANDOM DATA
504 003102 005767 014576 NXT4: TST TXMEV ;IS XMTR EVENT FLAG SET?
505 003106 100006 BPL NXT5 ;NO, TEST ANOTHER FLAG
506 003110 CALL TXMIT ;YES, ENTER XMIT MODULE
507 003124 005767 014562 NXT5: TST RCVEV ;IS RCVR EVENT FLAG SET?
508 003130 100006 BPL NXT6 ;NO, TEST ANOTHER FLAG
509 003132 CALL RECV ;YES, ENTER RCVR MODULE
510 003146 005767 014542 NXT6: TST SPCEV ;IS SPECIAL EVENT FLAG SET?
511 003152 100006 BPL NXT7 ;NO, TEST ANOTHER FLAG
512 003154 CALL SPEC ;YES, HANDLE SPECIAL EVENT
513 003170 005767 014502 NXT7: TST STSEV ;IS STATUS EVENT FLAG SET?
514 003174 100006 BPL NXT8 ;NO, TEST ANOTHER FLAG
515 003176 CALL STATUS ;YES, OUTPUT STATUS
516 003212 005767 014454 NXT8: TST SUMEV ;IS SUMMARY EVENT FLAG SET?
517 003216 100006 BPL NXT9 ;NO, TEST ANOTHER FLAG
518 003220 CALL SUMRY ;YES, OUTPUT ERROR SUMMARY
519 003234 005767 014456 NXT9: TST XSPCEV ;IS EXTRA-SPECIAL EVENT SET?
520 003240 100006 BPL NXT10 ;YES, HANDLE NOW MASTER.
521 003242 CALL XSPEC ;IS XMTR ERROR EVENT SET?
522 003256 005767 014400 NXT10: TST TEREV ;NO, TEST ANOTHER FLAG
523 003262 100006 BPL NXT11 ;YES, OUTPUT XMTR ERROR TABLE
524 003264 CALL TEROS ;IS RCVR ERROR EVENT SET?
525 003300 005767 014360 NXT11: TST REREV ;NO
526 003304 100006 BPL NXT12 ;YES, OUTPUT RCVR ERROR TABLE
527 003306 CALL REROS ;STAY IN MAIN LOOP
528 003322 000167 177422 NXT12: JMP PCLOOP
529
530 ;TRAP TO 4 HANDLER
531
532
533 003326 ERTRAP: REGSAV ;:::SAVE R0...R5
534 003332 CALL PNTLIN,<ERTMSG> ;:::PRINT TRAP MESSAGE
535 003350 005067 014336 CLR RCVEV ;:::CLEAR RCVR EVENT FLAG
536 003354 005067 014330 CLR PCLGO ;:::CLR PCL GO FLAG
537 003360 052777 000100 012560 BIS #B06,@TTRCSR ;:::SET TTY KBD INTR ENAB
538 003366 005037 177776 CLR @#PS ;:::DROP CP PRIORITY
539 003372 REGRES
540 003376 012706 002000 MOV #STKTOP,SP ;FIX STACK
541 003402 000167 177226 JMP RTRYA ;ENTER COMMAND MODE
```

543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
(1)
(1)
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596

003406
003406 012701 020006
003412 016602 000002
003416 005302
003420 001002
003422 000167 000446
003426 010267 014276
003432 020227 000062
003436 101402
003440 000167 000422
003444 010267 014320
003450 010200
003452 005300
003454
003464 060600
003466 062700 000004
003472 016067 000004 014220
003500 012767 000000 014214
003506 021027 000002
003512 001402
003514 000167 000346
003520 026067 000004 014174
003526 001004
003530 112721 000000
003534 005267 014230
003540 116011 000004
003544 001002
003546 000167 000314
003552 122711 000037
003556 103002
003560 000167 000302
003564 112167 014132
003570 062700 177772
003574 005302
003576 001343

```

.SBTTL COMMAND PROCESSORS:
.SBTTL COMMAND PROC. FOR SILO (LOAD)

: LOAD OR CLEAR THE TRANSMITTER ADDRESS (HARDWARE) SILO.
: IF THERE ARE NO ARGUMENTS, CLEAR THE ADDRESS SILO AND SET
: AUTO ADDRESS. IF THERE ARE ARGUMENTS, LOAD THE TOTAL NUMBER
: OF ARGUMENTS (INCLUDING PAD VALUES) INTO THE SILO AS MANY TIMES
: AS THE TOTAL WILL GO INTO 50. LOCATIONS.

: IF ANY 2 SEQUENTIAL ARGUMENTS ARE THE SAME, SEPARATE THEM WITH
: A PAD VALUE OF '0'. IF THE FIRST ARGUMENT IS THE SAME AS THE LAST
: ARGUMENT, INSERT A PAD VALUE OF '0' AFTER THE LAST ARGUMENT.

: ARGUMENTS HIGHER THAN 37 (OCTAL) WILL NOT BE ACCEPTED
: ARGUMENTS LOWER THAN 1 WILL NOT BE ACCEPTED.

: ARGUMENTS MUST BE NUMERIC. DECIMAL ARGUMENTS MUST BE FOLLOWED BY
: A DECIMAL POINT. (18.)
PROC CPSILO

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$: CMPB #37,(R1)
BHIS 5$
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
    
```

597	003600	026767	014116	014112		CMP	NEXT,FIRST		:IS LAST OBJECT = FIRST?
598	003606	001004				BNE	6\$:NO, O.K.
599	003610	112721	000000			MOVB	#0,(R1)+		:YES, INSERT A PAD VALUE
600	003614	005267	014150			INC	OBJCNT		:AND KEEP OBJECT COUNT UP TO DATE
601	003620	026727	014144	000062	6\$:	CMP	OBJCNT,#50.		:OBJECT COUNT GOTTEN TOO BIG?
602	003626	101402				BLOS	7\$:NO, O.K.
603	003630	000167	000232			JMP	SYNRTM		:YES, ERROR
604	003634	152777	000060	012260	7\$:	BISB	#B05+B04,@TMMRH		:PREPARE TO LOAD ADDR SILO
605	003642	012700	000062			MOV	#50,R0		:HOLD SILO SIZE
606	003646	026767	014116	014054		CMP	OBJCNT,PADFLG		:IS OBJECT COUNT DIFFERENT THAN START?
607	003654	001002				BNE	SILLD		:YES, LEAVE SOMETHING IN PADFLG (SET)
608	003656	005067	014046		8\$:	CLR	PADFLG		:NO, CLEAR PAD FLAG
609	003652	016702	014102		SILLD:	MOV	OBJCNT,R2		:GET NO. OF OBJECTS
610	003666	012701	020006			MOV	#ADSILO,R1		:GET OBJECT BUFFER
611	003672	112177	012222		SLOLP:	MOVB	(R1)+,@TMMR		:GET AN OBJECT INTO SILO
612	003676	005302				DEC	R2		:LOADED ALL OBJECTS?
613	003700	001374				BNE	SLOLP		:NO, CONTINUE
614	003702	166700	014062			SUB	OBJCNT,R0		:YES, IS THAT ALL THAT'LL FIT?
615	003706	020067	014056			CMP	R0,OBJCNT		
616	003712	002363				BGE	SILLD		:IF NOT LOAD THEM AGAIN
617	003714	132777	000001	012200		BITB	#1,@TMMRH		:SEE IF I AM MASTER
618	003722	001037				BNE	CPSLV		:IF SO TURN ON SILO AND EXIT
619	003724					CALL	PNTLIN,<MSTMG1>		:PRINT 'THIS UNIT IS NOT MASTER
620	003742					CALL	PNTLIN,<MSTMG2>		: BUT HAS BEEN MADE SECONDARY
621	003760					CALL	PNTLIN,<MSTMG3>		: THE SILO YOU HAVE JUST FILLED
622	003776					CALL	PNTLIN,<MSTMG4>		: WILL BE USED IF YOU CLEAR THE
623	004014	152777	000002	012100		BISB	#B01,@TMMRH		: THE CURRENT MASTER"
624	004022	005767	013702		CPSLV:	TST	PADFLG		:HAS SILO BEEN PADDED?
625	004026	001407				BEQ	9\$:NO, CARRY ON.
626	004030					CALL	PNTLIN,<MSTMG5>		:YES, TELL OPERATOR.
627	004046	012767	177777	013712	9\$:	MOV	#-1,ESCFLG		:FLAG COMENT TO ISSUE NEW 'PCL>'.
628	004054	142777	000020	012040		BICB	#B04,@TMMRH		:CLR AUTO ADDR
629	004062	000241				CLC			:CLEAR 'C' FOR NON ERROR RETURN
630	004064				CPSEX:	RETURN			
631									
632	004066	000261			SYNRTM:	SEC			:SET 'C' BIT FOR SYNTAX ERROR
633	004070	000167	177770			JMP	CPSEX		:AND RETURN
634									
635									
636	004074	152777	000060	012020	CLEAV:	BISB	#B05+B04,@TMMRH		:RESET AUTO ADDR & CLR SILO
637	004102	012767	177777	013656		MOV	#-1,ESCFLG		:FLAG COMENT TO ISSUE NEW 'PCL>'.
638	004110	000241				CLC			:CLEAR 'C' IN CASE IT WAS SET
639	004112	000167	177746			JMP	CPSEX		:RETURN

.SBTTL COMMAND PROC. FOR MASTER, SECONDARY AND R.I.B.

```

641
642
643 :PROCESSOR FOR 'MASTER SET (OR CLEAR)' COMMAND
644 : SET MASTER, OR CLEAR MASTER
645
646 004116 PROC CPMAS
(1)
(1) 004116 CPMAS: : **ENTRY POINT**
647 004116 022766 000002 000002 CMP #2,2(SP) :GET # OF ARGUMENTS
648 004124 001402 BEQ CPMOK :OKAY IF 2
649 004126 000261 SEC
650 004130 000423 BR CPMRET :OTHERWISE, SYNTAX ERROR
651 004132 010603 CPMOK: MOV SP,R3 :GET 2ND WORD OF COMMAND
652 004134 062703 000004 ADD #4,R3
653 004140 KEYWD R3,#SCTBL :PROCESS IT
654 004152 103446 BCS KWDERT :SYNTAX ERROR IF 'C' SET
655 004154 117702 011742 MOVB @TMMRH,R2 :GET OLD TMMR
656 004160 142702 000001 BICB #1,R2 :REMOVE OLD STATE OF MASTER
657 004164 052602 BIS (SP)+,R2 :SET NEW STATE OF MASTER
658 004166 110277 011730 MOVB R2,@TMMRH :LOAD NEW TMMR
659 004172 012767 177777 013566 MOV #-1,ESCFLG :FLAG COMENT TO ISSUE NEW 'PCL>'.
660 004200 CPMRET: RETURN :EXIT

```

```

661
662 :PROCESSOR FOR 'SECONDARY SET (OR CLEAR)' COMMAND
663 : SET SECONDARY, OR CLEAR SECONDARY
664
665
666 004202 PROC CPSEC
(1)
(1) 004202 CPSEC: : **ENTRY POINT**
667 004202 022766 000002 000002 CMP #2,2(SP) :ARE THERE 2 ARGUMENTS?
668 004210 001402 BEQ CPSOK :IF YES, PROCEED
669 004212 000261 SEC
670 004214 000424 BR CPSRET :OTHERWISE, SYNTAX ERROR
671 004216 010603 CPSOK: MOV SP,R3 :GET 2ND WORD OF COMMAND
672 004220 062703 000004 ADD #4,R3
673 004224 KEYWD R3,#SCTBL :PROCESS IT
674 004236 103414 BCS KWDERT :SYNTAX ERROR IF 'C' SET
675 004240 117702 011656 MOVB @TMMRH,R2 :GET OLD TMMR
676 004244 142702 000002 BICB #2,R2 :REMOVE OLD STATE OF SECONDARY
677 004250 006316 ASL (SP)
678 004252 052602 BIS (SP)+,R2 :SET NEW STATE OF SECONDARY
679 004254 110277 011642 MOVB R2,@TMMRH :LOAD NEW TMMR
680 004260 012767 177777 013500 MOV #-1,ESCFLG :FLAG COMENT TO ISSUE NEW 'PCL>'.
681 004266 CPSRET: RETURN :EXIT

```

```

682
683
684 004270 032600 KWDERT: BIT (SP)+,R0 :POP BAD WORD OFF STACK
685 004272 000775 BR CPSRET :EXIT

```

```

686
687
688 :PROCESSOR FOR 'RIB SET' OR 'RIB CLEAR'
689 : SET 'RIB' BIT IN XMTR COMMAND WORD LOCATION 'TXMST', OR CLEAR IT
690
691 004274 PROC CPRIB
(1)

```

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 8-1
COMMAND PROC. FOR MASTER, SECONDARY AND R.I.B.

F 5

SEQ 0057

```
(1) 004274          CPRI8:          : **ENTRY POINT**
692 004274 022766 000002 000002   CMP      #2,2(SP)      : 2 ARGUMENTS?
693 004302 001402          BEQ      CPROK          : YES, O.K.
694 004304 000261          SEC
695 004306 000426          BR       CPRET          : OTHERWISE, ERROR RETURN
696 004310 010603          CPROK: MOV     SP,R3     : GET 2ND WORD OF COMMAND
697 004312 062703 000004     ADD     #4,R3
698 004316          KEYWD   R3,#SCTBL     : PROCESS IT
699 004330 103757          BCS     KWDERT        : IF ERROR, SHOW IT.
700 004332 016702 003030     MOV     TXMST,R2      : GET OLD STATE OF TCR
701 004336 042702 100000     BIC     #B15,R2       : CLEAR IMAGE OF RIB
702 004342 000241          CLC
703 004344 006016          ROR     (SP)          : GET NEW STATE OF RIB
704 004346 006016          ROR     (SP)
705 004350 052602          BIS     (SP)+,R2     : SET IT IN IMAGE
706 004352 010267 003010     MOV     R2,TXMST     : LOAD NEW STATE OF TCR
707 004356 012767 177777 013402   MOV     #-1,ESCFLG   : FLAG COMENT TO ISSUE NEW "PCL>"
708 004364          CPRET: RETURN      : BACK TO CALLER
```

.SBTTL COMMAND PROC. FOR RANGE

710
 711
 712
 713
 714
 715
 716
 717
 718
 719

```

: SET RANGE OF RECEIVER ADDRESSES THAT THIS TRANSMITTER SHOULD TALK TO.
:
: ACCEPT A LOWER AND UPPER RECEIVER ADDRESS (IN THAT ORDER).
: GENERATE A LIST (SET ACTIVE FLAGS) OF RCVR ADDRESSES FROM LOW ADDRESS
: TO HIGH ADDRESS INCLUSIVE.
  
```

720 004366
 (1)

PROC CPRANG

(1) 004366
 721 004366 016600 000002
 722 004372 022700 000003
 723 004376 001047
 724 004400 016600 000004
 725 004404 022700 000002
 726 004410 001042
 727 004412 016601 000010
 728 004416 016600 000012
 729 004422 022700 000002
 730 004426 001033
 731 004430 016602 000016
 732 004434 020102
 733 004436 101427
 734 004440 022701 000037
 735 004444 103424
 736 004446 010200
 737 004450 001422
 738 004452 006300
 739 004454 006300
 740 004456 010003
 741 004460 006300
 742 004462 060300
 743 004464 062700 020074
 744 004470 012710 177777
 745 004474 062700 000014
 746 004500 005202
 747 004502 020201
 748 004504 003771
 749 004506 012767 177777 013252
 750 004514
 751
 752 004516 000261
 753 004520 000167 177770

CPRANG:

CPRFIL:

CPRRTN:

SYNRTN: SEC
 JMP

```

MOV 2(SP),R0
CMP #3,R0
BNE SYNRTN
MOV 4(SP),R0
CMP #2,R0
BNE SYNRTN
MOV 10(SP),R1
MOV 12(SP),R0
CMP #2,R0
BNE SYNRTN
MOV 16(SP),R2
CMP R1,R2
BLOS SYNRTN
CMP #37,R1
BLO SYNRTN
MOV R2,R0
BEQ SYNRTN
ASL R0
ASL R0
MOV R0,R3
ASL R0
ADD R3,R0
ADD #RADBO,R0
MOV #-1,(R0)
ADD #14,R0
INC R2
CMP R2,R1
BLE CPRFIL
MOV #-1,ESCF LG
  
```

```

: **ENTRY POINT**
: GET # OF OBJECTS
: 3 OBJECTS?
: IF NOT, INDICATE SYNTAX ERROR
: GET CLASS OF FIRST OBJECT
: IS IT CLASS 2?
: IF NOT, INDICATE SYNTAX ERROR
: GET 'TO' ARGUMENT INTO R1
: GET CLASS OF SECOND OBJECT
: IS IT CLASS 2?
: IF NOT, INDICATE SYNTAX ERROR
: GET 'FROM' ARGUMENT INTO R2
: IS 'FROM' LOWER THAN 'TO'?
: IF NOT, INDICATE SYNTAX ERROR
: IS 'TO' > 37?
: IF SO, INDICATE SYNTAX ERROR
: FIND 'FROM' ENTRY
: IF 'FROM'=0, SYNTAX ERROR

: R0 CONTAINS 'FROM' ADDRESS
: SET RCV ADDR ACTIVE FLAG.
: UPDATE TABLE POINTER
: INCREMENT 'FROM' ADDRESS
: UNTIL EQUAL TO 'TO' ADDRESS
: EXIT WHEN COMPLETE.
: FLAG COMMENT TO ISSUE NEW 'PCL>'

: SET 'C' BIT TO INDICATE SYNTAX ERROR
  
```

.SBTTL COMMAND PROC. FOR ADD AND DELETE

```

: ADD A RECEIVER ADDRESS TO THE LIST (SET ACTIVE FLAG) OF RCVR ADDRESSES
:
: ACCEPT ARGUMENTS (AT LEAST 1) CHECK THAT THEY FALL IN A RANGE FROM
: 1 TO 37; THEN SET THE CORRESPONDING ACTIVE FLAGS IN THE STAT
: TABLE.
  
```

```

755
756
757
758
759
760
761
762
763 004524          PROC      CPADD
(1)
(1) 004524          CPADD:      ;**ENTRY POINT**
764 004524 016602 000002      MOV      2(SP),R2      ;GET # OF ARGUMENTS INTO R2
765 004530 005302              DEC      R2            ;IGNORE KEYWORD OBJECT
766 004532 010600              MOV      SP,R0        ;GENERATE ADDRESS OF ARGUMENTS
767 004534 062700 000004      ADD      #4,R0        ;POINT R0 AT 1ST OBJECT CLASS
768 004540 021027 000002      CPALP:  CMP      (R0),#2  ;IS CLASS = 2?
769 004544 001027              BNE     SYNTRA        ;NO, INDICATE SYNTAX ERROR
770 004546 016001 000004      MOV      4(R0),R1    ;GET OBJECT INTO R1
771 004552 001424              BEQ     SYNTRA        ;IF IT'S 0, SYNTAX ERROR
772 004554 022701 000037      CMP      #37,R1
773 004560 103421              BLO     SYNTRA        ;IF IT'S >37, SYNTAX ERROR
774 004562 006301              ASL     R1            ;FIND THE TABLE ELEMENT
775 004564 006301              ASL     R1
776 004566 010103              MOV     R1,R3
777 004570 006301              ASL     R1
778 004572 060301              ADD     R3,R1
779 004574 062701 020074      ADD     #RADB0,R1
780 004600 012711 177777      MOV     #-1,(R1)     ;SET ACTIVE FLAG FOR THIS TABLE ENTRY
781 004604 062700 000006      ADD     #6,R0        ;SET UP FOR NEXT ARGUMENT
782 004610 005302              DEC     R2            ;ARE WE DONE?
783 004612 001352              BNE     CPALP        ;NO, CONTINUE.
784 004614 012767 177777 013144  MOV     #-1,ESCFLG   ;FLAG COMENT TO ISSUE NEW 'PCL>'
785 004622
786 004624 000261              CPARTN: RETURN
787 004626 000167 177770      SYNTRA: SEC
788
789
790
791 004632          PROC      CPDEL
(1)
(1) 004632          CPDEL:      ;**ENTRY POINT**
792 004632 016602 000002      MOV      2(SP),R2      ;GET # OF ARGUMENTS INTO R2
793 004636 005302              DEC      R2            ;IGNORE KEYWORD OBJECT
794 004640 010600              MOV      SP,R0        ;GENERATE ADDRESS OF ARGUMENTS
795 004642 062700 000004      ADD      #4,R0        ;POINT R0 AT 1ST OBJECTS CLASS
796 004646 021027 000002      CPDLP:  CMP      (R0),#2  ;IS CLASS =2
797 004652 001034              BNE     SYNTRD       ;NO, INDICATE SYNTAX ERROR
798 004654 016001 000004      MOV      4(R0),R1    ;GET OBJECT INTO R1
799 004660 001431              BEQ     SYNTRD       ;IF IT'S 0, SYNTAX ERROR
800 004662 022701 000037      CMP      #37,R1
801 004666 103426              BLO     SYNTRD       ;IF IT'S >37, SYNTAX ERROR
802 004670 006301              ASL     R1            ;FIND TABLE ELEMENT
803 004672 006301              ASL     R1
804 004674 010103              MOV     R1,R3
805 004676 006301              ASL     R1
806 004700 060301              ADD     R3,R1
  
```

;DELETE RECEIVER ADDRESSES AND ASSOCIATED "ATTEMPTS" AND "SUCCESSSES".

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 10-1
COMMAND PROC. FOR ADD AND DELETE

SEQ 0060

807	004702	062701	020074		ADD	#RADB0,R1	
808	004706	005011			CLR	(R1)	;CLEAR ACTIVE FLAG AT TABLE LOCATION
809	004710	062701	000004		ADD	#4,R1	;AND OTHER ENTRIES...
810	004714	005021			CLR	(R1)+	
811	004716	005021			CLR	(R1)+	
812	004720	005021			CLR	(R1)+	
813	004722	005011			CLR	(R1)	
814	004724	062700	000006		ADD	#6,R0	;SET UP FOR NEXT ARGUMENT
815	004730	005302			DEC	R2	;ARE WE DONE?
816	004732	001345			BNE	CPDLP	;NO, CONTINUE
817	004734	012767	177777	013024	MOV	#-1,ESCFLG	;FLAG COMENT TO ISSUE NEW 'PCL>'
818	004742				CPDRTN:	RETURN	
819	004744	000261			SYNTRD:	SEC	;SET 'C' BIT FOR SYNTAX ERROR
820	004746	000167	177770		JMP	CPDRTN	

822
823
824
825
826
827
828
829
830
831
(1)
(1)
832
833
834
835
836
837
(1)
(1)
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
(1)
(1)
860
861
862
863
864
865
866
867
868
869
870
871

.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
PROC    CPCLR
; CLEAR STATUS TABLE

```

```

CPCLR:
; **ENTRY POINT**
; GET NUMBER OF OBJECTS INTO R2
; ARE THERE ANY ARGUMENTS?
; IF SO, INDICATE SYNTAX ERROR
; FLAG COMMENT TO ISSUE NEW 'PCL>'
MOV     2(SP),R2
CMP     #1,R2
BNE     SYNTRC
MOV     #-1,ESCF LG
PROC    PRCLR

```

```

PRCLR:
; **ENTRY POINT**
; SAVE COUNT OF TABLE ELEMENTS
; SET UP TO CLR ACTIVE FLAGS
; CLEAR ALL ELEMENTS OF ENTRY
MOV     #37,R1
MOV     #RADB,R0
CPCLRC: CLR     (R0)
ADD     #4,R0
CLR     (R0)+
CLR     (R0)+
CLR     (R0)+
CLR     (R0)+
DEC     R1
BNE     CPCLRC
CLR     A00
MOV     A00+QTOP,A00+QFRONT
MOV     A00+QTOP,A00+QBACK
CPCLRT: RETURN
SYNTRC: SEC
JMP     CPCLRT
; DONE?
; IF NOT, CONTINUE
; INITIALIZE ADDR QUEUE TO EMPTY
; AND LEAVE
; SET 'C' BIT FOR SYNTAX ERROR

```

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

```

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

```

; CONTINUE EXERCISING. DO NOT AFFECT THE STATUS TABLE

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 11-1
COMMAND PROC. FOR CLEAR, STATUS, AND CONTINUE

SEQ 0062

872
873
874
875
876
877
878
(1)
879
880
881
882
883
884
885
886
887
888
889

005114
005114
005114 022766 000001 000002
005122 001027
005124 012767 100000 012556
005132
005146
005164 005767 012470
005170 001403
005172 012777 000100 010760
005200
005202 000261
005204 000167 177770

: DO NOT AFFECT THE ERROR TABLE
: DO NOT AFFECT THE RCVR ADDRESS QUEUE
: SIMPLY SET PCLGO
:

PROC CPCNT
CPCNT:
CMP #1,2(SP)
BNE SYNTCN
MOV #B15,PCLGO
CALL PNCRLF
CALL PNTLIN,<EXCNT>
TST KWFLG
BEQ CPCNRT
MOV #B06,@LCS
CPCNRT: RETURN
SYNTCN: SEC
JMP CPCNRT

;**ENTRY POINT**
:ARE THERE ANY ARGUMENTS?
:IF SO, INDICATE SYNTAX ERROR
:SET PCL GO FLAG
:PRINT CR & LF
:PRINT 'EXERCISER CONTINUING'
:GOT A CLOCK?
:NO
:RE-ENABLE CLOCK.
:SET 'C' BIT FOR SYNTAX ERROR

.SBTTL COMMAND PROC. FOR INIT, SUMMARY, AND GO

891
892
893
894
895
896
897
898
899
900
901
902
(1)
(1)
903
904
905
906
(1)
(1)
907
908
909
910
911
912
913
914
(1)
(1)
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940

005210
005210 022766 000001 000002
005216 001114
005220 012767 177777 012540
005226
005226
005234 105067 012413
005240 105067 012410
005244 105067 012402
005250 005067 012402
005254 042767 100000 002104
005262
005276
005276 012700 024736
005302 022710 177777
005306 001406
005310 005060 000004
005314 062700 000006
005320 000167 177756
005324 012700 001350
005330 012701 025160
005334 012721 000000
005340 005300
005342 001374
005344 012700 000037
005350 012701 020110
005354 062701 000004
005360 005021
005362 005021
005364 005021
005366 005021
005370 005300
005372 001370
005374 016767 012334 012334
005402 016767 012332 012332
005410 005067 012270
005414 005067 012252
005420 005067 012252
005424 005067 012232

:TO USE CALL MACRO, ENTER AT PRINIT AND PARCLR
: CLEAR STATUS TABLE, SUMMARY TABLE, ERROR TABLE
: CLEAR 'RIB' IN XMTR COMMAND WORD 'TXMST'
: CLEAR CLOCK DATA
: CLEAR TRANSMITTER HARDWARE
: INITIALIZE DATA PATTERN
:

PROC CPINIT
CPINIT:
CMP #1,2(SP)
BNE SYNTIN
MOV #-1,ESCFLG
PROC PRINT
BDINIT XMTR
CLRB SECNDS
CLRB MINUTS
CLRB TICKS
CLR HOURS
BIC #B15,TXMST
CALL PRCLR
PROC PARCLR
PARCLR:
MOV #ERTBL,R0
ERTBCL: CMP #-1,(R0)
BEQ ERTCD
CLR 4(R0)
ADD #6,R0
JMP ERTBCL
ERTCD: MOV #37*30,R0
MOV #TERTBL,R1
1\$: MOV #0,(R1)+
DEC R0
BNE 1\$
MOV #37,R0
MOV #RADB,R1
2\$: ADD #4,R1
CLR (R1)+
CLR (R1)+
CLR (R1)+
CLR (R1)+
DEC R0
BNE 2\$
MOV ORIGSD,DTSEED
MOV MLSD,MSGLSD
CLR TXMEV
CLR SUMEV
CLR STSEV
CLR TEREV

ENTRY POINT
:ARE THERE ANY ARGUMENTS?
:IF SO, INDICATE SYNTAX ERROR
:FLAG COMENT TO ISSUE NEW 'PCL>'

ENTRY POINT
:CLEAR XMTR.
:CLEAR SECONDS (TIME)
:CLEAR MINUTES
:CLEAR TICKS
:CLEAR HOURS
:CLEAR RIB IN COMMAND WORD FOR XMTR
:CLR STATUS TABLE & ADDR QUEUE

ENTRY POINT
:CLEAR ERROR SUMMARY TABLE
:IS ERR NUM = -1?
:IF SO, DONE CLEARING
:OTHERWISE, CLR OCURRENCES
:STEP TO NEXT ENTRY
:AND CONTINUE
:ALSO CLEAR ERROR DATA
:FROM DETAILED ERR TABLE

:CLEAR ATTEMPS & SUCCESSES
:FROM STATUS TABLE

:RESTORE ORIGINAL DATA SEED
:AND MSG LENGTH SEED
:CLR XMIT EVENT FLAG
:CLR SUMMARY EVENT FLAG
:CLR STATUS EVENT FLAG

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 12-1
COMMAND PROC. FOR INIT, SUMMARY, AND GO

941 005430 005067 012230
942 005434 005067 012306
943 005440 052767 100000
944 005446
945 005450 000261
946 005452 000167 177770

012222

CLR REREV
CLR RCTXPS
BIS #B15,DATGEV
CPINRT: RETURN
SYNTIN: SEC
JMP CPINRT

;CLR XMTR & RCVR ERROR EVENTS
;CLEAR DATA PASS NO.
;SET DATA GEN FLAG
;SET 'C' BIT FOR SYNTAX ERROR

; SET THE SUMMARY EVENT FLAG SO THAT THE ERROR SUMMARY TABLE WILL BE PRINTED

953 005456

PROC CPSUM

(1)
(1) 005456
954 005456 022766 000001 000002
955 005464 001006
956 005466 012767 100000 012176
957 005474 005067 012202
958 005500
959 005502 000261
960 005504 000167 177770

CPSUM:
CMP #1,2(SP)
BNE SYNTSM
MOV #B15,SUMEV
CLR SUMPNT
CPSURT: RETURN
SYNTSM: SEC
JMP CPSURT

ENTRY POINT
;ARE THERE ANY ARGUMENTS?
;IF SO, INDICATE SYNTAX ERROR
;SET SUMMARY EVENT FLAG
;0 SUMPNT INDICATES HEADER FIRST
;SET 'C' BIT FOR SYNTAX ERROR

961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978

: START THE EXERCISER
: CLEAR 'ATTEMPTS' AND 'SUCCESSSES' IN STATUS TABLE
: CLEAR ERROR TABLES
: CAUSE RCVR ADDRESS QUEUE TO BE LOADED
: SET PCLGO
: IF THE RCVR ADDRESS QUEUE IS EMPTY AFTER BEING LOADED,
: INDICATE THIS BY PRINTING THE FOLLOWING MESSAGE:
: ***NO RECEIVERS SELECTED.**
: AND RETURN TO COMMAND INPUT MODE.

979 005510

PROC CPGO

(1)
(1) 005510
980 005510 022766 000001 000002
981 005516 001062
982 005520 005067 012222
983 005524 005067 010704
984 005530 105067 012117
985 005534 105067 012114
986 005540 105067 012106
987 005544 005067 012106
988 005550 016767 010664 010666
989 005556 016767 010656 010662
990 005564 052767 100000 012074
991 005572 012767 177777 012110
992 005600

CPGO:
CMP #1,2(SP)
BNE SYNTAX
CLR RCTXPS
CLR AOQ
CLRB SECNDS
CLRB MINUTS
CLRB TICKS
CLR HOURS
MOV AOQ+QTOP,AOQ+QFRONT
MOV AOQ+QTOP,AOQ+QBACK
BIS #B15,QLDEV
MOV #-1,PCLGO
CALL PARCLR

ENTRY POINT
;ARE THERE ANY ARGUMENTS?
;IF SO, INDICATE SYNTAX ERROR
;CLEAR PASS NO.
;INIT ADDR QUEUE TO EMPTY
;CLEAR SECONDS REG.
;CLEAR MINUTES REG.
;CLEAR TICKS REG.
;CLEAR HOURS REG.
;SET ADDR QUEUE LOAD EVENT
;SET PCL TO GO
;CLEAR STATUS & ERRORS (NOT RCVR)

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 12-2
COMMAND PROC. FOR INIT, SUMMARY, AND GO

SEQ 0065

```

993 005614          CALL  PNCRLF          ;PRINT CR, LF
994 005630          CALL  PNTLIN,<EXRST> ;PRINT 'EXERCISER STARTED'.
995 005646 005767 012006 TST  KWFLG          ;GOT A CLOCK?
996 005652 001403          BEQ  STRTRT        ;NO.
997 005654 012777 000100 010276 MOV  #B06,@LCS      ;ENABLE CLOCK INTR.
998 005662          STRTRT: RETURN        ;EXIT
999
1000 005664 000261          SYNTAX: SEC          ;SET 'C' TO INDICATE SYNTAX ERROR
1001 005666 000167 177770          JMP  STRTRT        ;AND EXIT
1002
1003

```

.SBTTL COMMAND PROC. FOR 'ASSIGN'

: ASSIGN XMTR ADDR & VECTOR AND RCVR ADDR & VECTOR.
 : THE ASSIGN COMMAND MAY HAVE 0, 1, 2, 3 OR 4 ARGUMENTS:

- 0 = ASSIGN STANDARD ADDRESSES & VECTORS TO RCVR & XMTR
- 1 = ARGUMENT IS ASSUMED TO BE XMTR ADDRESS
- 2 = ARGUMENTS ARE ASSUMED TO BE : XMTR ADDR, XMTR VECTOR
- 3 = ARGUMENTS ARE ASSUMED TO BE: XMTR ADDR, XMTR VECT, RCVR ADDR
- 4 = ARGUMENTS IN THE FOLLOWING ORDER:

: XMTR ADDRESS, XMTR VECTOR, RCVR ADDRESS, AND RCVR VECTOR.
 : CAUTION MUST BE EXERCISED WHEN ASSIGNING ADDRESSES TO GET THEM
 : IN THE CORRECT SEQUENCE AND ORDER.

1005
 1006
 1007
 1008
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 1016
 1017
 1018
 1019
 1020
 (1)
 (1)
 1021
 1022
 1023
 1024
 1025
 1026
 1027
 1028
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047
 1048
 1049
 1050
 1051
 1052
 1053
 1054
 1055
 1056
 1057
 1058

005672
 005672
 005672 010600
 005674 016001 000002
 005700 020127 000005
 005704 101402
 005706 000167 000372
 005712 001030
 005714 016002 000004
 005720 022702 000002
 005724 001402
 005726 000167 000352
 005732 062700 000006
 005736 016002 000002
 005742 032702 000003
 005746 001402
 005750 000167 000330
 005754 020227 000774
 005760 101402
 005762 000167 000316
 005766 010267 010212
 005772 005301
 005774 020127 000004
 006000 001030
 006002 016002 000004
 006006 022702 000002
 006012 001402
 006014 000167 000264
 006020 062700 000006
 006024 016002 000002
 006030 032702 000017
 006034 001402
 006036 000167 000242
 006042 020227 164000
 006046 103002
 006050 000167 000230
 006054 010267 010122
 006060 005301
 006062 020127 000003
 006066 001030

PROC CPASS
 CPASS:
 1\$:
 2\$:
 3\$:
 4\$:
 PART1:
 1\$:
 2\$:
 3\$:
 PART2:

MOV SP,R0
 MOV 2(R0),R1
 CMP R1,#5
 BLOS 1\$
 JMP SYNXR
 BNE PART1
 MOV 4(R0),R2
 CMP #2,R2
 BEQ 2\$
 JMP SYNXR
 ADD #6,R0
 MOV 2(R0),R2
 BIT #3,R2
 BEQ 3\$
 JMP SYNXR
 CMP R2,#774
 BLOS 4\$
 JMP SYNXR
 MOV R2,RCVEC
 DEC R1
 CMP R1,#4
 BNE PART2
 MOV 4(R0),R2
 CMP #2,R2
 BEQ 1\$
 JMP SYNXR
 ADD #6,R0
 MOV 2(R0),R2
 BIT #17,R2
 BEQ 2\$
 JMP SYNXR
 CMP R2,#164000
 BHIS 3\$
 JMP SYNXR
 MOV R2,RCDEV
 DEC R1
 CMP R1,#3
 BNE PART3

ENTRY POINT
 : COPY LIST POINTER
 : GET NO. OF ARGUMENTS
 : 4 ARGUMENTS?
 : ERROR TOO MANY ARGUMENTS
 : LESS THAN 4, PARTIAL ASSIGNMENT
 : GET CLASS OF OBJECT
 : IF IT'S A NUMBER, O.K.
 : ERROR WRONG CLASS
 : NOW GET ACTUAL NUMBER
 : IS IT ON VECTOR BOUNDARY?
 : YES
 : ERROR NOT ON VECTOR BOUNDARY
 : IS IT WITHIN VECTOR AREA?
 : YES
 : ERROR, OUTSIDE VECTOR AREA
 : SAVE AS RCVR VECTOR
 : ARE THERE ONLY 3 ARGS?
 : NO, MUST BE FEWER
 : GET CLASS OF OBJECT
 : IF IT'S A NUMBER, O.K.
 : ERROR, OBJECT NOT A NUMBER
 : NOW, GET THE NUMBER
 : CHECK IF ITS A VALID ADDR
 : IT IS. O.K.
 : ERROR INVALID ADDRESS
 : IS IT IN THE ADDRESS FIELD?
 : YES, O.K.
 : ERROR. ADDR = OUTSIDE DEVICE AREA
 : SAVE AS RCVR ADDRESS
 : ARE THERE ONLY 2 ARGS?
 : NO, MUST BE ONLY 1.

1059	006070	016002	000004		MOV	4(R0),R2	;GET CLASS OF OBJECT	
1060	006074	022702	000002		CMP	#2,R2		
1061	006100	001402			BEQ	1\$;IF IT'S A NUMBER, O.K.	
1062	006102	000167	000176		JMP	SYNXR	;ERROR, WRONG CLASS	
1063	006106	062700	000006	1\$:	ADD	#6,R0	;NOW GET THE NUMBER	
1064	006112	016002	000002		MOV	2(R0),R2		
1065	006116	032702	000003		BIT	#3,R2	;IS IT ON VECTOR BOUNDARY?	
1066	006122	001402			BEQ	2\$;YES	
1067	006124	000167	000154		JMP	SYNXR	;ERROR, NOT ON VECTOR BOUNDARY	
1068	006130	020227	000774	2\$:	CMP	R2,#774	;IS IT WITHIN VECTOR AREA?	
1069	006134	101402			BLOS	3\$;YES	
1070	006136	000167	000142		JMP	SYNXR	;ERROR, OUTSIDE VECTOR AREA	
1071	006142	010267	010032	3\$:	MOV	R2, TXVEC	;SAVE AS XMTR VECTOR	
1072	006146	005301			DEC	R1		
1073	006150	020127	000002	PART3:	CMP	R1,#2	;ONLY 1 ARGUMENT?	
1074	006154	001027			BNE	PART4	;NO, MUST BE NONE.	
1075	006156	016002	000004		MOV	4(R0),R2	;GET CLASS OF OBJECT	
1076	006162	022702	000002		CMP	#2,R2		
1077	006166	001402			BEQ	1\$;IF IT'S A NUMBER, O.K.	
1078	006170	000167	000110		JMP	SYNXR	;ERROR, WRONG CLASS	
1079	006174	062700	000006	1\$:	ADD	#6,R0	;NOW GET THE NUMBER	
1080	006200	016002	000002		MOV	2(R0),R2		
1081	006204	032702	000017		BIT	#17,R2	;CHECK FOR VALID ADDRESS	
1082	006210	001402			BEQ	2\$;O.K.	
1083	006212	000167	000066		JMP	SYNXR		
1084	006216	020227	164000	2\$:	CMP	R2,#164000	;IS IT IN ADDRESS FIELD?	
1085	006222	103002			BHIS	3\$;YES	
1086	006224	000167	000054		JMP	SYNXR	;ERROR, OUTSIDE ADDRESS FIELD	
1087	006230	010267	007742	3\$:	MOV	R2, TXDEV	;SAVE AS XMTR ADDRESS	
1088	006234	020127	000001	PART4:	CMP	R1,#1	;NO ARGUMENTS?	
1089	006240	001014			BNE	ASEXT	;WE'RE DONE!	
1090	006242	012767	164200	007726	MOV	#164200, TXDEV	;NO ARGUMENTS, LOAD DEFAULTS.	
1091	006250	012767	000170	007722	MOV	#170, TXVEC		
1092	006256	012767	164220	007716	MOV	#164220, RCDEV		
1093	006264	012767	000174	007712	MOV	#174, RCVEC		
1094	006272	012737	000340	177776	ASEXT:	MOV	#340, @#PS	;RAISE PROC. PRIORITY
1095	006300	000167	173474		JMP	PCLEX	;GO START EXERCISER OVER.	
1096								
1097	006304	000261			SYNXR:	SEC	;SET 'C' TO FLAG ERROR	
1098	006306					RETURN	;RETURN	

.SBTTL COMMAND DECODER AND PROCESSOR

1100
 1101
 1102
 1103
 1104
 1105
 1106
 1107 006310
 (1)
 (1) 006310
 1108 006310 105067 011335
 1109 006314 005067 011446
 1110 006320
 1111 006340 103015
 1112 006342
 1113 006360
 1114 006374 005767 011366
 1115 006400 001406
 1116 006402
 1117 006416
 1118
 1119
 1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1130
 1131
 1132
 1133
 1134 006420
 (1)
 (1) 006420
 1135 006420
 1136 006434
 1137 006452
 1138 006466 005067 011222
 1139 006472 005067 011212
 1140 006476 005067 011174
 1141 006502 005067 011164
 1142 006506
 1143 006522

:CONVERT COMMAND TO IT'S PROCESSING ROUTINE ADDRESS
 :IF C BIT IS SET, INDICATE SYNTAX ERROR
 :RETURN TO COMMAND INPUT MODE
 :IF ESCFLG <> 0, CALL PRESC (CNTRL-C)

PROC COMENT

COMENT: : **ENTRY POINT**
 CLR B CMDENT : CLEAR COMMAND ENTERED FLAG
 CLR ESCFLG : CLR COMMAND INPUT MODE FLAG
 CALL COMAND,<CMDBUF,CMDTBL> : CONVERT COMMAND
 BCC CMDRTN : IF C = 0, EXIT
 CALL PNTLIN,<SYNTAX> : PRINT 'SYNTAX ERROR' IF C SET
 CALL PRESC : GIVE ANOTHER 'PCL>'
 CMDRTN: TST ESCFLG : SHOULD WE FAKE CNTRL-C?
 BEQ CMDRET : NO, EXIT
 CALL PRESC : YES, GIVE A 'PCL>'
 CMDRET: RETURN : RETURN TO MAIN LOOP

:PROCESSOR FOR SPECIAL EVENT
 :INDICATE 'NO MASTER' AND CLEAR EVENTS
 :THIS EVENT IS CALLED BY THE MAIN LOOP AFTER
 : THE OCCURRENCE OF A MASTER DOWN INTERRUPT
 :THE RESULT OF THIS EVENT IS THAT THE PROGRAM WILL
 : PRINT MASTER DOWN IN THE FOLLOWING FASION:

***** MASTER DOWN *****

:THEN CLEAR ACTIVE EVENT FLAGS BY ENTERING COMMAND MODE.
 :TO RECOVER FROM THIS STATE AND RESUME EXERCISING:
 : (A) SET MASTER ON ANY ONE PCL11
 : (B) TYPE THE COMMAND 'CONTINUE'

PROC SPEC

SPEC: : **ENTRY POINT**
 CALL PNCRLF : ENQUEUE CR & LF
 CALL PNTLIN,<MDNER> : ENQUEUE MASTER DOWN MESSAGE
 CALL PNCRLF : ANOTHER CR & LF
 CLR SPCEV : CLR SPECIAL EVENT FLAG
 SPERTN: CLR PCLGO : DON'T ALLOW TXM EVENT
 CLR STSEV : DISCONTINUE STATUS EVENT
 CLR SUMEV : DISCONTINUE SUM EVENT
 CALL PRESC : FAKE CNTRL-C KEY
 RETURN

```
1145 :PROCESSOR FOR 'EXTRA' SPECIAL EVENT
1146 :TO INDICATE WHEN THIS UNIT HAS JUST BECOME
1147 :MASTER AS A RESULT OF PREVIOUSLY HAVING
1148 :SECONDARY SET AND THE CURRENT BUS MASTER
1149 :HAVING JUST DROPPED MASTER.
1150 :WHEN 'NOW MASTER' INTERRUPT OCCURS, THE
1151 :CONSOLE DEVICE WILL PRINT
1152 :   '** THIS UNIT HAS BECOME NEW MASTER **'
1153
1154 006524          PROC   XSPEC
(1)
(1) 006524          XSPEC:          ;**ENTRY POINT**
1155 006524          CALL   PNCRLF          ;ENQUEUE CR & LF
1156 006540          CALL   PNTLIN,<MSCHNG> ;ENQUEUE NEW MASTER MESSAGE
1157 006556          CALL   PNCRLF          ;ANOTHER CR & LF
1158 006572 005067 011120 CLR   XSPCEV          ;CLR XSPEC EVENT FLAG
1159 006576          RETURN          ;RETURN TO MAIN LOOP
1160
1161 :PROCESSOR FOR 'DETAILED' ERROR TABLE PRINTOUT COMMAND 'ERRORS'
1162
1163 :THIS ROUTINE WILL INITIALIZE THE TEMPORARY LOCATIONS REQUIRED
1164 :BY THE XMTR AND RCVR ERROR EVENTS. IT WILL THEN QUICKLY CHECK THE
1165 :RCVR AND XMTR ERROR TABLES FOR ANY ENTRIES. IF NONE EXIST, THE MESSAGE
1166 :   '**NO ERRORS TO REPORT**'
1167 :   WILL BE PRINTED ON THE CONSOLE AND THAT IS ALL.
1168 :   IF ANY ENTRIES WERE FOUND, THE XMTR ERROR EVENT FLAG 'TEREV' WILL BE SET
1169 :   WHICH WILL CAUSE THE XMTR ERRORS AND THE RCVR ERRORS (IF ANY) TO
1170 :   BE PRINTED.
1171
1172
1173
1174 006600          PROC   CPERR
(1)
(1) 006600          CPERR:          ;**ENTRY POINT**
1175 006600 022766 000001 000002 CMP   #1,2(SP)          ;ANY ARGUMENTS?
1176 006606 001044          BNE   SYNRR          ;ERROR IF THERE ARE.
1177 006610 012700 025160          MOV   #TERTBL,R0          ;POINT AT XMTR TABLE
1178 006614 012701 001350          MOV   #37*30,R1          ;SET TABLE LENGTH COUNTER
1179 006620 005720          1$:   TST   (R0)+          ;ANY ENTRIES?
1180 006622 001016          BNE   SERFL          ;YES, CALL ERROR MODULE
1181 006624 005301          DEC   R1          ;NO, CHECK WHOLE TABLE
1182 006626 001374          BNE   1$
1183 006630          CALL   PNTLIN,<NOERMG> ;PRINT 'NO ERRORS'
1184 006646 012767 177777 011112 MOV   #-1,ESCFLG      ;FLAG COMENT TO ISSUE 'PCL>'
1185 006654 000167 000036          JMP   CPERTN          ;AND RETURN
1186 006660 012767 100000 010774 SERFL: MOV   #B15,TEREV ;SET XMTR EVENT FLAG
1187 006666 012767 000001 011104 MOV   #1,ERR0          ;INITIATE ERROR #
1188 006674 012767 000001 011100 MOV   #1,ERR1          ;INITIATE RCVR ADDR
1189 006702 012767 025160 011074 MOV   #TERTBL,ERR2    ;INITIATE TABLE POINTER
1190 006710 052767 100000 011010 BIS   #B15,HDRFLG     ;SET HEADER FLAG
1191 006716          CPERTN: RETURN
1192
1193          SYNRR:  SEC
1194 006720 000261          JMP   CPERTN          ;SET 'C' BIT TO INDICATE SYNTAX ERR
1194 006722 000167 177770
```

.SBTTL RECEIVER ADDRESS QUEUE LOADER ROUTINE

1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
(1)
(1)
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238

006726
006726
006732 005067 010730
006736 012702 000037
006742 012700 020110
006746 022710 177777
006752 001002
006754 004767 000036
006760 062700 000014
006764 005302
006766 001367
006770 005767 007440
006774 001424
006776 005267 010744
007002 012767 100000 010674
007010
007014
007016 116067 000002 021253
007024
007044 000207
007046 005767 010636
007052 001753
007054
007070
007106
007122 000727

: LOAD THE RECEIVER ADDRESS SOFTWARE QUEUE
:
: THE TRANSMITTER EVENT WILL PICK RCVR ADDRESSES FROM THE QUEUE
: UNTIL IT IS EMPTY.
:
: IF THE QUEUE IS LOADED BUT STILL REMAINS EMPTY, AND PCLGO IS SET,
: THIS MEANS THE OPERATOR IS TRYING TO RUN BUT HASN'T ENTERED
: ANY RECEIVER ADDRESSES (USING 'RANGE' OR 'ADD' ETC.)
:

PROC ADQLD
ADQLD: REGSAV
CLR QLDEV
MOV #37,R2
MOV #RADB,R0
QCHACT: CMP #-1,(R0)
BNE QLOOK
JSR PC,QLOAD
QLOOK: ADD #14,R0
DEC R2
BNE QCHACT
TST AQQ
BEQ QMTRN
INC RCTXPS
QRTN: MOV #B15, TXMEV
REGRES
RETURN
: **ENTRY POINT**
: SAVE R0...R5
: CLR QUEUE LOAD EVENT FLAG
: POSITION COUNTER
: R0 POINTS AT TABLE
: IS ACTIVE FLAG SET IN TABLE?
: NO, LOOK FOR NEXT ONE
: YES, LOAD ADDR INTO QUEUE
:
: ALL ENTRIES CHECKED?
:
: IS THERE ANYTHING IN QUEUE?
: NO, THIS IS NOT A VALID PASS
: YES, INCR PASS NO.
: SET XMTR EVENT FLAG
: RESTORE R0...R5
:
: GET RCV ADDR INTO HIGH BYTE
: PUT IT IN ADDR QUEUE
:
: ARE WE RUNNING?
: NO, LEAVE
: STOP PRINTING ANYTHING ELSE
: TELL OPERATOR NO RCVR
: FAKE CNTRL-C
: RETURN
QLOAD: MOV B 2(R0),BKELEM+1
CALL ENQ,<BKELEM,AQQ>
RTS PC
QMTRN: TST PCLGO
BEQ QRTN
CALL PRCTLO
CALL PNTLIN,<MTQMSG>
CALL PRESC
BR QRTN

.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 > 602, 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.

1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
(1)
(1)
1256
1257
1258
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

007124
007124
007124
007130
007134
007140
007146
007150
007156
007164
007166
007174
007202
007204
007212
007216
007222
007226
007230
007232
007236
007242
007246
007270
007276
007300
007302
007310
007316
007322

005067 010616
005067 010614
005067 010612
042767 177000
001003
012767 000001
026727 010560
002403
012767 177777
022767 170000
101005
012767 177777
005067 010534
012700 020676
016705 010514
006105
005405
010567 010516
016705 010500
016720 010470
066767 010464
005305
003360
016767 010430
012767 100000
005067 010346

010574
010564
000602
010556
010534
010536
010440
010432
010366

DATGEN:
DTCNT:
DTGNC:
DTGNCO:
DTCLP:
DTCDON:

PROC DATGEN
CLR RCTXPS
CLR RJCTF
CLR TRNKF
BIC #177000,MSGLSD
BNE DTCNT
MOV #1,MSGLSD
DTCNT: CMP MSGLSD,#602
BLT DTGNC
MOV #-1,TRNKF
DTGNC: CMP #170000,DTSEED
BHI DTGNCO
MOV #-1,RJCTF
DTGNCO: MOV #DATBUF,R0
MOV MSGLSD,R5
ROL R5
NEG R5
MOV R5,TXML
MOV MSGLSD,R5
DTCLP: MOV DTSEED,(R0)+
MULP RANM,DTSEED
ADD RANK,DTSEED
DEC R5
BGT DTCLP
DTCDON: MOV DTSEED,MSGLSD
MOV #B15,TXMEV
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 602?

: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

.SBTTL MULTIPLY ROUTINE FOR DATA GENERATION

1285
1286
1287
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

007324 012601
007326 011603
007330 010446
007332 012704 000020
007336 005002
007340 006002
007342 006003
007344 103001
007346 060102
007350 005304
007352 003372
007354 012604
007356 006202
007360 006003
007362 010316
007364 000134

```
:UNSIGNED 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  
      MOV    (SP),R3      ;GET MULTIPLIER  
      MOV    R4,-(SP)     ;SAVE R4  
      MOV    #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  @R4+         ;RETURN
```

```
:MACRO FOR THE ABOVE IS AS FOLLOWS:  
: .MACRO  MULP  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

1319
 1320
 1321
 1322
 1323
 1324
 1325
 1326
 1327
 1328
 1329
 1330
 1331
 1332
 1333
 1334
 1335
 1336
 (1)
 (1)
 1337
 1338
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1358
 1359
 1360
 1361
 1362
 1363
 1364
 1365

007366 060101
 007370
 007370 005767 010314
 007374 001455
 007376 005067 010302
 007402
 007410 005767 007020
 007414 001446
 007416 016777 010332 006470
 007424 012777 020676 006464
 007432
 007452 116767 020617 010302
 007460 016700 010276
 007464 006300
 007466 006300
 007470 010005
 007472 006300
 007474 060500
 007476 062700 020074
 007502 062760 000001 000006
 007510 005560 000004
 007514 016777 020554 006364
 007522 056777 177640 006356
 007530
 007532 012767 100000 010126
 007540 026727 010202 000004
 007546 001402
 007550 000167 177754
 007554 012767 100000 010106
 007562 000167 177742

: CONTROL TRANSMISSION OF RANDOM DATA TO DE-QUEUED RECEIVER ADDRESSES
 : IF 'PCLGO' IS SET, ('GO' OR 'CONTINUE' SET IT), START TRANSMITTING
 : THE DATA CONTAINED IN THE BUFFER 'DATBUF'.
 : THE TARGET RECEIVER ADDRESS IS DEQUEUED FROM THE ADDRESS QUEUE.
 : THE MESSAGE LENGTH WAS DETERMINED DURING THE GENERATION OF THE DATA.
 : NOTE THAT BIT 15 OF 'TXMST' MAY BE 1 OR 0 DEPENDING UPON THE USE
 : OF THE 'RIB SET' OR 'RIB CLEAR' COMMANDS.

TXMST: .WORD 060101 ;TRANSMITTER COMMAND WORD

PROC TXMIT

TXMIT: ;**ENTRY POINT**
 TST PCLGO ;CAN PCL GO?
 BEQ TXRTN ;IF NOT, RETURN
 CLR TXMEV ;CLR XMTR EVENT FLAG
 BDINIT XMTR ;CLR XMTR HARDWARE
 TST AQQ ;ANY ELEMENTS IN ADDR QUEUE?
 BEQ TXOUT ;IF NOT, FILL IT UP AGAIN
 MOV TXML,@TSBC ;LOAD BYTE COUNT
 MOV #DATBUF,@TSBA ;LOAD BUS ADDRESS
 CALL DEQ,<FRELEM,AQQ> ;DEQUEUE RECEIVER ADDRESS
 MOVB FRELEM+1,CURAD ;GET ADDRESS TABLE OFFSET
 MOV CURAD,R0 ;FIND ADDRESS TABLE ENTRY
 ASL R0 ; IN ORDER TO UPDATE
 ASL R0 ; 'ATTEMPTS'
 MOV R0,R5
 ADD R5,R0
 ADD #RADB0,R0
 ADD #1,6(R0) ;UPDATE DOUBLE WORD
 ADC 4(R0)
 MOV FRELEM,@TCR ;GIVE RECEIVER ADDR TO XMTR
 BIS TXMST,@TCR ;START TRANSMISSION

TXRTN: RETURN

TXOUT: MOV #B15,QLDEV ;SET QUEUE LOAD EVENT FLAG
 CMP RCTXPS,#4 ;IS THIS THE 5TH PASS?
 BEQ TXDGEN ;IF SO, SET DATA GEN EVENT FLAG
 JMP TXRTN ;RETURN
 TXDGEN: MOV #B15,DATGEV ;SET DAT GEN EVENT FLAG
 JMP TXRTN ;RETURN

1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397

007566
007572 042777 000100 006306
007600 012737 000200 177776
007606 032777 100000 006274
007614 001402
007616 000167 000114
007622 032777 000200 006260
007630 001402
007632 000167 000610
007636 032777 000040 006244
007644 001402
007646 000167 000772
007652 132777 000004 006242
007660 001402
007662 000167 001074
007666 032777 000020 006214
007674 001402
007676 000167 000514
007702 012767 100000 007774
007710

(1)
(1)
(1)
007730
007734 000002

;;; TRANSMITTER INTERRUPT HANDLER
;;; SORT THE CAUSES OF TRANSMITTER INTERRUPTS
;;; AND GO TO THE PROPER HANDLING ROUTINE.

TXER = 100000
TXBSY = 20
TXSUC = 200
TXSOR = 40
NOWMST = 4

XMTINT: REGSAV
BIC #B06,@TCR
MOV #200,@#PS
BIT #TXER,@TSR
BEQ XMTIS
JMP ERRINT
XMTIS: BIT #TXSUC,@TSR
BEQ XMTSR
JMP SUCINT
XMTSR: BIT #TXSOR,@TSR
BEQ XMTPNM
JMP SORINT
XMTPNM: BITB #NOWMST,@TMMRH
BEQ XMTIB
JMP NMINT
XMTIB: BIT #TXBSY,@TSR
BEQ XMTPRB
JMP BSYINT
XMTPRB: MOV #B15, TXMEV
ERROT \N

;;; SAVE R0...R5
;;; CLR TXM INTR ENABLE
;;; ALLOW RCVR INTR BUT NOT TTY
;;; IS THERE A HARDWARE ERROR?
;;; NO, CHECK SUC TXF
;;; HANDLE ERROR
;;; WAS IT A SUC TRANSFER?
;;; NO, CHECK SOFTWARE REJECT
;;; HANDLE SUC TXF
;;; WAS MESSAGE REJECTED?
;;; NO, CHECK NOW MASTER
;;; HANDLE REJECT
;;; IS THIS UNIT NOW MASTER?
;;; NO, WAS RECVR BUSY?
;;; YES, HANDLE NOW MASTER INT
;;; WAS RECEIVER BUSY
;;; NO, INTERRUPT WAS ERRONEOUS
;;; HANDLE BUSY
;;; SET XMTR EVENT FLAG
;;; ERROR: ERRONEOUS INTERRUPT FROM XMTR

***** XMTR ERROR 1 *****

REGRES
RTI

;;; RESTORE R0...R5
;;; RETURN

```
1401 ;XMTR ERROR INTERRUPT ROUTINE
1402
1403
1404 ; DETERMINE AND REPORT THE CAUSE OF THE TRANSMITTER ERROR.
1405
1406
1407 007736 032777 040000 006144 ERRINT: BIT #B14,@TSR ;IS ERROR NON EXISTANT LOCATION?
1408 007744 001412 BEQ ERRA ;ERROR:NON-EXISTANT LOC ERROR IN XMTR
1409 007746 ERROT \N
(1) ;**** XMTR ERROR 2 ****
(1)
(1)
1410 007766 000167 000402 006110 ERRRA: JMP ERRX ;IS ERROR MEM OFL?
1411 007772 032777 020000 BEQ #B13,@TSR
1412 010000 001412 ERROT \N ;ERROR:MEM OFL ERROR IN XMTR
1413 010002
(1) ;**** XMTR ERROR 3 ****
(1)
(1)
1414 010022 000167 000346 006054 ERRB: JMP ERRX ;IS ERROR TXM ERROR?
1415 010026 032777 010000 BEQ #B12,@TSR
1416 010034 001472 MOV ERRC ;SAVE RESPONSE CODES
1417 010036 017767 006046 007720 @TSR,RSPC ;REMOVE GARBAGE
1418 010044 042767 177760 007712 BIC #-20,RSPC ;RSP CODES 11&01?
1419 010052 022767 000015 007704 CMP #15,RSPC
1420 010060 001012 BNE ERRA1
1421 010062 ERROT \N ;ERROR:TXM ERROR. RCVR ACCEPTED A NULL
(1) ;**** XMTR ERROR 4 ****
(1)
(1)
1422 010102 000167 000266 007650 ERRA1: JMP ERRX ;DISCARD RSP A'S
1423 010106 042767 000003 BIC #3,RSPC ;RSB B = 00?
1424 010114 001012 BNE ERRA2
1425 010116 ERROT \N ;ERROR:TXM ERROR. RCVR HAS GONE OFF LINE
(1) ;**** XMTR ERROR 5 ****
(1)
(1)
1426 010136 000167 000232 007614 ERRA2: JMP ERRX ;RSP B = 10?
1427 010142 022767 000010 CMP #10,RSPC
1428 010150 001012 BNE ERRA3
1429 010152 ERROT \N ;ERROR:TXM ERROR. WORD OR CRC REJECTED
(1) ;**** XMTR ERROR 6 ****
(1)
(1)
1430 010172 000167 000176 ERRA3: JMP ERRX ;ERROR:MISCELLANEOUS TXM ERROR
1431 010176 ERROT \N
(1) ;**** XMTR ERROR 7 ****
(1)
(1)
1432 010216 000167 000152 005660 ERRC: JMP ERRX ;MASTER DOWN?
1433 010222 032777 004000 BEQ #B11,@TSR
1434 010230 001417 ERROT \N ;ERROR: M A S T E R D O W N !
1435 010232
(1) ;**** XMTR ERROR 10 ****
(1)
(1)
```

```

1436 010252 005067 007432          CLR      PCLGO          ;STOP TRANSMITTER
1437 010256 012767 100000 007430  MOV      #B15,SPCEV    ;SET SPECIAL EVENT FLAG
1438 010264 000167 000104          JMP      ERRX
1439 010270 032777 002000 005612  ERRD:   BIT      #B10,@TSR ;TIMEOUT ERROR?
1440 010276 001422          BEQ      ERRE
1441 010300 032777 100000 005600  BIT      #B15,@TCR    ;IS 'R.I.B.' SET?
1442 010306 001404          BEQ      1$           ;NO MUST BE WEIRD ERROR.
1443 010310 032777 000020 005572  BIT      #TXBSY,@TSR  ;YES, IS TDM BUS BUSY?
1444 010316 001010          BNE      2$           ;YES, RECEIVER WASN'T THERE!
1445 010320          ERROT   \N          ;ERROR: ERRONEOUS TIMEOUT IN TRANSMITTER
(1)
(1)
(1)
1446 010340 000167 000030 005536  2$:    JMP      ERRX          ;**** XMTR ERROR 11 ****
1447 010344 032777 001000          BIT      #B09,@TSR   ;TXM SILO OVERRUN?
1448 010352 001410          BEQ      ERRX
1449 010354          ERROT   \N          ;ERROR:SILO OVERRUN ERROR IN XMTR
(1)
(1)
(1)
1450 010374          ERRX:   BDINIT   XMTR   ;CLEAR ALL XMTR HARDWARE
1451 010402 012767 100000 007274  MOV      #B15,TXMEV  ;SET XMTR EVENT FLAG
1452 010410          REGRES
1453 010414 000002          RTI                ;RESTORE R0...R5
                          ;RETURN

```

```

1455                                     ;BUSY INTERRUPT ROUTINE
1456
1457 010416 042777 000020 005464 BSYINT: BIC      #TXBSY,@TSR      ;CLEAR TDM BUS BUSY
1458 010424 012767 100000 007252      MOV      #B15, TXMEV    ;SET TXM EVENT FLAG
1459 010432      BDINIT  XMTR      ;CLEAR HARDWARE
1460 010440      REGRES      ;RESTORE R0...R5
1461 010444 000002      RTI      ;RETURN
1462
1463                                     ;SUCCESSFUL TRANSFER INTERRUPT ROUTINE
1464
1465
1466 010446 042777 000200 005434 SUCINT: BIC      #TXSUC,@TSR      ;CLEAR SUC TXF
1467 010454 005767 007272      TST      TRNKF      ;IS SFTWR TRUNCATE FLAG SET?
1468 010460 001047      BNE      SUCIA      ;YES, GO CHECK SORE
1469 010462 032777 000040 005420 SUCINY: BIT      #TXSOR,@TSR    ;IS SORE SET?
1470 010470 001414      BEQ      SUCINP      ;NO, CONTINUE
1471 010472      ERROT      \N      ;ERROR:MESSAGE TRUNCATED UNEXPECTEDLY
(1)
(1)
(1)
1472 010512 000421      BR      SHBRT      ;DON'T INC SUC CONNS IF ERROR
1473 010514 042777 000040 005366 SUCINP: BIC      #TXSOR,@TSR    ;CLR SORE
1474 010522 016700 007234      MOV      CURAD,R0    ;PREPARE TO UPDATE STATUS TABLE
1475 010526 006300      ASL      R0      ;GET TABLE ENTRY FOR CURRENT
1476 010530 006300      ASL      R0      ;RECEIVER ADDRESS IN
1477 010532 010005      MOV      R0,R5
1478 010534 006300      ASL      R0
1479 010536 060500      ADD      R5,R0
1480 010540 062700 020074      ADD      #RADBO,R0
1481 010544 062760 000001 000012      ADD      #1,12(R0)
1482 010552 005560 000010      ADC      10(R0)
1483 010556 012767 100000 007120 SHBRT: MOV      #B15, TXMEV    ;SET TRANSMIT EVENT FLAG
1484 010564      BDINIT  XMTR      ;CLEAR HARDWARE
1485 010572      REGRES      ;RESTORE R0...R5
1486 010576 000002      RTI      ;RETURN
1487
1488 010600 032777 000040 005302 SUCIA: BIT      #TXSOR,@TSR    ;IS SORE SET? (SHOULD BE SET)
1489 010606 001011      BNE      SUCCS
1490 010610      ERROT      \N      ;ERROR:MSG FAILED TO BE TRUNCATED
(1)
(1)
(1)
1491 010630 000752      BR      SHBRT      ;DON'T INC SUC CONNS IF ERROR
1492 010632 042777 000040 005250 SUCCS: BIC      #TXSOR,@TSR    ;CLR SORE
1493 010640 000167 177616      JMP      SUCINY      ;PROCESS SUC TXF AS NORMAL
    
```

```
1495 ;SOFTWARE REJECT INTERRUPT ROUTINE
1496
1497 010644 005767 007100 SORINT: TST RJCTF ;IS SFTWR REJECT FLAG SET?
1498 010650 001432 BEQ SORIP ;IF NOT, SOMETHING'S WRONG
1499 010652 016700 007104 MOV CURAD,R0 ;PREPARE TO UPDATE STATUS TABLE
1500 010656 006300 ASL R0 ;GET TABLE ENTRY FOR CURRENT
1501 010660 006300 ASL R0 ; RECIIVER ADDRESS IN
1502 010662 010003 MOV R0,R3 ;
1503 010664 006300 ASL R0 ; ORDER TO UPDATE
1504 010666 060300 ADD R3,R0 ;
1505 010670 062700 020074 ADD #RADB0,R0 ; 'SUCCESSFUL' ELEMENT
1506 010674 062760 000001 000012 ADD #1,12(R0) ;UPDATE ENTRY
1507 010702 005560 000010 ADC 10(R0)
1508 010706 042777 000040 005174 SORCNT: BIC #TXSOR,@TSR ;CLR SORR
1509 010714 012767 100000 006762 MOV #B15, TXMEV ;SET XMIT EVENT FLAG
1510 010722 BDINIT XMTR ;CLEAR HARDWARE
1511 010730 REGRES ;RESTORE R0...R5
1512 010734 000002 RTI ;RETURN
1513
1514 010736 SORIP: ERROT \N ;ERROR:ERRONEOUS REJECT BY RECEIVER
(1) ;**** XMTR ERROR 15 ****
(1)
(1)
1515 010756 000167 177724 JMP SORCNT ;RETURN
1516
1517
1518 ;NOW MASTER INTERRUPT ROUTINE
1519 ;SET XSPEC EVENT FLAG (XSPEV)
1520 ;AND CLR NOW MST
1521
1522
1523 010762 142777 000004 005132 NMINT: BICB #NOWMST,@TMMRH ;CLEAR NOW MASTER
1524 010770 012767 100000 006720 MOV #B15,XSPCEV ;SET EXTRA-SPECIAL EVENT
1525 010776 052777 000100 005102 BIS #B06,@TCR ;RE-SET INTERRUPT ENABLE
1526 011004 REGRES ;RESTORE R0...R5
1527 011010 000002 RTI ;RETURN
1528
1529
```

.SBTTL RECEIVER MODULE

1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
(1)
(1)
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561

020000
011012
011012
011012 005067 006674
011016 004757 000040
011022
011030 012777 022736 005100
011036 012777 176400 005070
011044 012777 020000 005054
011052 052777 000100 005046
011060
011062
011066 012700 022736
011072 012701 000700
011076 005020
011100 005301
011102 003375
011104
011110 000207

: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,DBF CLR
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

DBF CLR: 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


```
1563  
1564  
1565          100000  
1566          000400  
1567          000200  
1568          000040  
1569  
1570 011112  
1571 011116 042777 000100 005002  
1572 011124 032777 100000 004776  
1573 011132 001402  
1574 011134 000167 000100  
1575 011140 032777 000400 004762  
1576 011146 001402  
1577 011150 000167 000500  
1578 011154 032777 000200 004746  
1579 011162 001402  
1580 011164 000167 000536  
1581 011170 032777 000040 004732  
1582 011176 001402  
1583 011200 000167 000776  
1584 011204  
      (1)  
      (1)  
      (1)  
1585 011224 012767 100000 006460  
1586 011232  
1587 011236 000002
```

:::RECEIVER INTERRUPT HANDLER.

```
RCER = 100000  
RCDOR = 400  
RCSUC = 200  
RCRCM = 40
```

```
RCVINT: REGSAV  
      BIC #B06,@RCR  
      BIT #RCER,@RSR  
      BEQ RCVINA  
      JMP RERINT  
RCVINA: BIT #RCDOR,@RSR  
      BEQ RCVINB  
      JMP RDOINT  
RCVINB: BIT #RCSUC,@RSR  
      BEQ RCVINC  
      JMP RSUINT  
RCVINC: BIT #RCRCM,@RSR  
      BEQ RCVIND  
      JMP RRJINT  
RCVIND: ERROR \N
```

```
:::SAVE R0...R5  
:::CLEAR RCVR INTR ENABLE  
:::IS THERE A HARDWARE ERROR?  
:::NO,CHECK DATA OUTPUT READY  
:::HANDLE ERROR INTERRUPT  
:::IS DATA OUTPUT READY?  
:::NO, CHECK SUC TXF  
:::HANDLE DATA OUTPUT RDY INTERRUPT  
:::SUCCESSFUL TRANSFER SET?  
:::NO, CHECK REJECT COMPLETED  
:::HANDLE SUC TXF INTERRUPT  
:::REJECT COMPLETED INTERRUPT?  
:::NO,ERRONEOUS INTERRUPT  
:::HANDLE REJ-COM INTERRUPT  
:::ERROR:UNKNOWN RECEIVER INTERRUPT OCCURRED
```

***** RCVR ERROR 16 *****

```
:::SET RCVR EVENT FLAG  
:::RESTORE R0...R5  
:RETURN
```

```
1589 ;RCVR ERROR INTERRUPT ROUTINE
1590
1591 011240 032777 040000 004662 RERINT: BIT #B14,@RSR ;IS ERROR NON EXST LOC ERR?
1592 011246 001412 BEQ RERRA ;ERROR:NON EXST LOC ERR IN RCVR
1593 011250 ERROR \N ;**** RCVR ERROR 17 ****
(1)
(1)
(1)
1594 011270 000167 000340 RERRA: JMP RERRX ;IS ERROR MEM OFLO?
1595 011274 032777 020000 004626 BEQ #B13,@RSR ;ERROR:MEM OFLO ERROR IN RCVR
1596 011302 001412 RERRB RERRB ;**** RCVR ERROR 20 ****
1597 011304 ERROR \N
(1)
(1)
(1)
1598 011324 000167 000304 RERRB: JMP RERRX ;ANY TRANSMISSION ERRORS?
1599 011330 032777 010000 004572 BEQ RERRC ;GET RESPONSE CODES
1600 011336 001457 MOV @RSR,RSPC ;REMOVE GARBAGE
1601 011340 017767 004564 006416 BIC #-20,RSPC ;LOOK AT TXM RSP CODES
1602 011346 042767 177760 006410 BIT #3,RSPC
1603 011354 032767 000003 006402 BNE RERRB1 ;ERROR:TXM WENT OFF LINE
1604 011362 001012 ERROR \N ;**** RCVR ERROR 21 ****
1605 011364
(1)
(1)
(1)
1606 011404 000167 000224 RERRB1: JMP RERRX ;REMOVE TXM RSP CODES NOW
1607 011410 042767 000003 006346 BIC #3,RSPC ;WAS THERE A CRC ERROR?
1608 011416 022767 000010 006340 CMP #10,RSPC
1609 011424 001012 BNE RERRB2 ;ERROR:RCVR CRC ERROR OCCURRED
1610 011426 ERROR \N ;**** RCVR ERROR 22 ****
(1)
(1)
(1)
1611 011446 000167 000162 RERRB2: JMP RERRX ;ERROR:FIRST WORD RECVD NOT VALID
1612 011452 ERROR \N ;**** RCVR ERROR 23 ****
(1)
(1)
(1)
1613 011472 000167 000136 RERRC: JMP RERRX ;WAS ERROR A PARITY ERROR?
1614 011476 032777 004000 004424 BIT #B11,@RSR ;ERROR:RCVR DETECTED INVALID PARITY
1615 011504 001412 BEQ RERRD ;**** RCVR ERROR 24 ****
1616 011506 ERROR \N
(1)
(1)
(1)
1617 011526 000167 000102 RERRD: JMP RERRX ;DID RCVR TIME OUT?
1618 011532 032777 002000 004370 BIT #B10,@RSR ;ERROR:RCVR TIMEOUT ERROR OCCURRED
1619 011540 001412 BEQ RERRR ;**** RCVR ERROR 25 ****
1620 011542 ERROR \N
(1)
(1)
(1)
1621 011562 000167 000046 RERRR: JMP RERRX ;WAS IT BYTE COUNT OVERFLOW?
1622 011566 032777 001000 004334 BIT #B09,@RSR
1623 011574 001417 BEQ RERRX
```

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 26-1
RECEIVER MODULE

SEQ 0082

1624	011576	042777	040000	004322
1625	011604	042777	001000	004316
1626	011612	052777	100000	004306
1627	011620	052777	000100	004300
1628	011626			
1629	011632	000002		
1630	011634			
1631	011642	012767	100000	006042
1632	011650	000167	177752	
1633				

```

BIC #B14,@RCR
BIC #B09,@RSR
BIS #B15,@RCR
BIS #B06,@RCR
RERTRN: REGRES
RTI
RERRX: BDINIT RCVR
MOV #B15,RCVEV
JMP RERTRN

```

```

:CLR RCVR NPR
:CLR RCVR BYTE COUNT OVERFLOW
:SET 'REJECT' IN RECEIVER
:RE-ENABLE INTERRUPT
:RESTORE R0...R5
:RETURN
:CLEAR RECVR
:SET RCV EVENT FLAG

```

```

*NEW*
*NEW*
*NEW*
*NEW*

```

```

1635 ;RCVR DATA OUTPUT RDY INTERRUPT ROUTINE
1636
1637 011654 017767 004252 006062 RDOINT: MOV @Rddb,RCSEED ;GET FLAG-WORD FOR DATA SEED
1638 011662 022767 170000 006054 CMP #170000,RCSEED ;LOOK AT DATA SEED
1639 011670 101011 BHI RDOINC ;IF MORE -VE THAN -10000
1640 011672 052777 100000 004226 BIS #B15,@RCR ; SET REJECT IN RCVR
1641 011700 RDORTN: REGRES ;RESTORE RO...R5
1642 011704 052777 000100 004214 BIS #B06,@RCR ;RE-SET INTR ENAB
1643 011712 000002 RTI ;RETURN
1644 011714 052777 040001 004204 RDOINC: BIS #B14+B00,@RCR ;START RECEIVER GOING
1645 011722 000167 177752 JMP RDORTN
1646
1647 ;RCVR SUC TXF INTERRUPT ROUTINE
1648
1649 011726 032777 000040 004174 RSUINT: BIT #RCRCM,@RSR ;WAS RECOM SET?
1650 011734 001403 BEQ RSUCNT ;NO, CONTINUE
1651 011736 042777 000040 004164 BIC #RCRCM,@RSR ;YES, CLR IT AND CONTINUE
1652 011744 016701 005774 RSUCNT: MOV RCSEED,R1 ;GET DATA SEED
1653 011750 042701 177000 BIC #177000,R1 ;MASK MSG LENGTH
1654 011754 001002 BNE RSNT0 ;IF NOT 0, OKAY
1655 011756 012701 000001 MOV #1,R1 ;IF 0, MAKE IT 1
1656 011762 162701 000001 RSNT0: SUB #1,R1 ;ACCOUNT FOR FLAG WORD
1657 011766 020127 000600 CMP R1,#600 ;IS IT GREATER THAN 600?
1658 011772 003402 BLE RSFLSZ ;NO, LEAVE IT ALONE
1659 011774 012701 000600 MOV #600,R1 ;YES, SET IT TO LIMIT (600)
1660 012000 012700 001400 RSFLSZ: MOV #1400,R0 ;DETERMINE # OF BYTES RECEIVED
1661 012004 067700 004124 ADD @RDBC,R0 ;NOTHING TO CHECK, EXIT
1662 012010 001451 BEQ RSULV ;RO = NO. OF WORDS RECEIVED
1663 012012 006200 ASR R0 ;DID WE RECV ALL THE WORDS?
1664 012014 020001 CMP R0,R1 ;YES, CARRY ON
1665 012016 001412 BEQ RSUTST ;NOT ENOUGH !
1666 012020 103456 BLO ERNE ;ERROR:RCVR GOT TOO MANY WORDS!!
1667 012022 ERROR \N
(1)
(1)
(1)
1668 012042 000434 BR RSULV ;EXIT IF ERROR
1669 012044 012705 022736 RSUTST: MOV #RCBUF,R5 ;R5 POINTS AT RECDV DATA
1670 012050 RSUGEN: MULP RANM,RCSEED ;GENERATE CHECK WORD
1671 012072 066767 005662 005644 ADD RANK,RCSEED ;COMPARE IT WITH RECV'D WORD
1672 012100 026725 005640 CMP RCSEED,(R5)+
1673 012104 001411 BEQ RSUCHK ;ERROR:DATA WORD RECV'D WAS BAD
1674 012106 ERROR \N
(1)
(1)
(1)
1675 012126 000402 BR RSULV ;EXIT IF ERROR
1676 012130 005300 RSUCHK: DEC R0 ;CHECKED ALL WORDS?
1677 012132 001346 BNE RSUGEN ;NO,CONTINUE
1678 012134 RSULV: BDINIT RCVR ;CLR HARDWARE
1679 012142 REGRES ;RESTORE RO...R5
1680 012146 012767 100000 005536 MOV #B15,RCVEV ;SET RCVR EVENT FLAG
1681 012154 000002 RTI ;RETURN
1682
1683 ERNE: ERROR \N ;ERROR: RCVR GOT TOO FEW WORDS !!
(1)

```

```
(1)
(1)
1684 012176 000167 177732          JMP    RSULV          ;**** RCVR ERROR 30 ****
1685
1686          ;RCVR REJECT INTERRUPT ROUTINE
1687
1688 012202          RRJINT: BDINIT  RCVR          ;CLEAR HARDWARE
1689 012210 012767 100000 005474      MOV    #B:15,RCVEV    ;SET RCVR EVENT FLAG
1690 012216          REGRES          ;RESTORE R0...R5
1691 012222 000002          RTI              ;RETURN
1692
1693          ;KW11 CLOCK INTERRUPT ROUTINE
1694          ;UPDATE TICKS. IF = 60. UPDATE SECONDS.
1695          ;IF SECS = 60, UPDATE MINUTS. IF MINUTS = 60
1696          ;UPDATE HOURS.
1697
1698 012224 042777 000200 003726      CLKINT: BIC    #B07,@LCS    ;JUST IN CASE IT MATTERS.
1699 012232 105267 005414          INCB    TICKS            ;UPDATE TICKS
1700 012236 126727 005410 000074      CMPB   TICKS,#60.        ;GET TO 60 YET?
1701 012244 103424          BLO    CLKEXT            ;NO
1702 012246 105067 005400          CLRB   TICKS            ;YES, CLR TICKS
1703 012252 105267 005375          INCB   SECNDS           ;& UPDATE SECONDS
1704 012256 126727 005371 000074      CMPB   SECNDS,#60.      ;GET 60 SECS?
1705 012264 103414          BLO    CLKEXT            ;NO
1706 012266 105067 005361          CLRB   SECNDS           ;YES, CLR SECNDS
1707 012272 105267 005356          INCB   MINUTS           ;& UPDATE MINUTES
1708 012276 126727 005352 000074      CMPB   MINUTS,#60.     ;GET 60 MINUTES?
1709 012304 103404          BLO    CLKEXT            ;NO
1710 012306 105067 005342          CLRB   MINUTS           ;YES, CLEAR MINUTS
1711 012312 005267 005340          INC    HOURS            ;& UPDATE HOURS
1712 012316 000002          CLKEXT: RTI              ;RETURN
```

.SBTTL STATUS MODULE

```

: DUMP STATUS TABLE INDICATING ACTIVATED RECEIVERS
: SHOW NO. OF ATTEMPTED CONNECTIONS TO EACH ACTIVE RECEIVER
: AND NO. OF SUCCESSFUL CONNECTIONS TO EACH ACTIVE RECEIVER.
: QUANTITIES ARE IN DECIMAL AND ARE CAPABLE OF EXPANDING TO
: 655,359,999. ANY QUANTITY HIGHER THAN THAT AMOUNT WILL BE
: REPRESENTED AS '*****'. RECEIVER ADDRESS NUMBERS ARE
: PRINTED IN OCTAL
:
:

```

1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 (1)
 (1)
 1727
 1728
 1729
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1738
 1739
 1740
 1741
 1742
 1743
 1744
 1745
 1746
 1747
 1748
 1749
 1750
 1751
 1752
 1753
 1754
 1755
 1756
 1757
 1758
 1759
 1760
 1761
 1762
 1763
 1764
 1765
 1766
 1767

012320
 012320
 012320
 012324
 012326
 012332
 012336
 012340
 012346
 012350
 012354
 012360
 012362
 012366
 012372
 012400
 012404
 012410
 012414
 012422
 012426
 012432
 012436
 012442
 012446
 012450
 012454
 012456
 012464
 012470
 012474
 012500
 012504
 012510
 012512
 012516
 012520
 012536
 012544
 012550
 012556
 012560

005767 004310
 001402
 000167 000540
 005767 005342
 001504
 026727 005334
 002402
 000167 000476
 005777 005320
 001002
 000167 000452
 012700 032043
 062767 000002
 017701 005274
 012702 177777
 004767 020250
 062767 000002
 012700 032065
 016701 005246
 012702 177777
 004767 020422
 020027 032112
 103003
 112720 000040
 000772
 062767 000004
 012700 032113
 016701 005204
 012702 177777
 004767 020360
 020027 032127
 103003
 112720 000040
 000772
 012520
 062767 000004
 000167 000322
 132777 000001
 001407

020674
 005300
 005256
 005214
 005134
 003344

```

PROC STATUS
STATUS:
TST T00
BEQ 60$
JMP STRTN
60$: TST STPNTR
BEQ STHDR
STCHK: CMP STPNTR,#RADEND
BLT 59$
JMP STARXT
59$: TST @STPNTR
BNE 58$
JMP STBADV
58$: MOV #STLIN1,R0
ADD #2,STPNTR
MOV @STPNTR,R1
MOV #-1,R2
JSR PC,OCTJSP
ADD #2,STPNTR
MOV #STLIN2,R0
MOV STPNTR,R1
MOV #-1,R2
JSR PC,CDDMG
STYHR: CMP R0,#STLIN2+25
BHIS STYON
MOVB #' ,(R0)+
BR STYHR
STYON: ADD #4,STPNTR
MOV #STLIN3,R0
MOV STPNTR,R1
MOV #-1,R2
JSR PC,CDDMG
STYTHR: CMP R0,#STLIN3+14
BHIS STYONT
MOVB #' ,(R0)+
BR STYTHR
STYONT: CALL PNTLIN,<STLIN>
ADD #4,STPNTR
JMP STRTN
STHDR: BITB #1,@TMMRH
BEQ 1$
CALL PNTLIN,<THUMST>

```

```

: **ENTRY POINT**
: IS TTY OUT QUEUE EMPTY?
: NO, RETURN TO WHEREVER
: IS OUTPUT POINTER AT TABLE?
: NO, GO PRINT HEADER
: IS POINTER AT END OF TABLE?
: YES, DONE; EXIT
: CHECK RCV ADDR ACTIVE FLAG
: ADVANCE TABLE IF FLAG CLR
: OUTPUT POINTER FOR OCTJSP
: OCTJSP DUMPS DATA IN STPNTR
: DON'T COMPRESS BLANKS
: OUTPUT POINTER FOR CDDMG
: OUTPUT WORD FOR CDDMG
: DON'T COMPRESS BLANKS
: BLANK REST OF FIELD OUT
: BY INSERTING SPACES
: OUTPUT POINTER FOR CDDMG
: OUTPUT WORD FOR CDDMG
: DON'T COMPRESS BLANKS
: BLANK REST OF FIELD OUT
: BY INSERTING SPACES
: ENQUEUE STATUS LINE FOR OUTPUT
: UPDATE POINTER FOR NEXT LINE
: RETURN
: IS MASTER SET HERE?
: NO.
: YES, TELL OPERATOR

```

1768	012576	132777	000002	003316	1\$:	BITB	#2,@TMMRH	:WELL, IS SECONDARY SET?
1769	012604	001407				BEQ	2\$:NO.
1770	012606					CALL	PNTLIN,<THUSCN>	:YES, TELL OPERATOR
1771	012624	032767	100000	174534	2\$:	BIT	#B15,TXMST	:IS 'RIB' SET?
1772	012632	001410				BEQ	3\$:NO, TELL OPERATOR
1773	012634					CALL	PNTLIN,<RBSTMG>	:YES, TELL OPERATOR
1774	012652	000407				BR	4\$	
1775	012654				3\$:	CALL	PNTLIN,<RBCLMG>	
1776	012672	005767	004762		4\$:	TST	KWFLG	:GOT A CLOCK?
1777	012676	001444				BEQ	5\$:NO, FORGET ABOUT TIME.
1778	012700	005002				CLR	R2	:SUPPRESS LEADING ZEROS
1779	012702	012700	031730			MOV	#TMLIN1,R0	:OUTPUT POINTER FOR DECPNT
1780	012706	016701	004744			MOV	HOURS,R1	:OUTPUT DATA FOR DECPNT
1781	012712	004767	017770			JSR	PC,DECPNT	:ENQUEUE 'HOURS'
1782	012716	005002				CLR	R2	:SUPPRESS LEADING ZEROS
1783	012720	112720	000072			MOVB	#':,(R0)+	:INSERT COLON
1784	012724	116701	004724			MOVB	MINUTS,R1	:GET 'MINUTES' FOR DECPNT
1785	012730	004767	017752			JSR	PC,DECPNT	:ENQUEUE 'MINUTES'
1786	012734	005002				CLR	R2	:SUPPRESS LEADING ZEROS
1787	012736	112720	000072			MOVB	#':,(R0)+	:INSERT ANOTHER COLON
1788	012742	116701	004705			MOVB	SECNDS,R1	:GET 'SECONDS' FOR DECPNT
1789	012746	004767	017734			JSR	PC,DECPNT	:ENQUEUE 'SECONDS'
1790	012752	005002				CLR	R2	:SUPPRESS LEADING ZEROS
1791	012754	112720	000072			MOVB	#':,(R0)+	:INSERT ANOTHER COLON
1792	012760	116701	004666			MOVB	TICKS,R1	:GET 'TICKS' FOR DECPNT
1793	012764	004767	017716			JSR	PC,DECPNT	:ENQUEUE 'TICKS'
1794	012770	105010				CLRB	(R0)	:ENQUEUE '0' PRINT TERMINATOR.
1795	012772					CALL	PNTLIN,<ELPSTM>	:PRINT 'ELAPSED TIME'
1796	013010				5\$:	CALL	PNTLIN,<STITLE>	:ENQUEUE STATUS HEADER
1797	013026	012767	020110	004644		MOV	#RADB,STPNTR	:SET POINTER TO TOP OF STAT LIST
1798	013034	000167	000032			JMP	STRTN	:RETURN
1799								
1800	013040	062767	000014	004632	STBADV:	ADD	#14,STPNTR	:SET POINTER TO NEXT ACTIVE FLAG
1801	013046	000167	000020			JMP	STRTN	
1802								
1803	013052	005067	004620		STARXT:	CLR	STSEV	:CLR STATUS EVENT FLAG
1804	013056					CALL	PRESC	:FAKE CNTRL-C KEY
1805	013072				STRTN:	RETURN		:RETURN

.SBTTL TRANSMITTER ERRORS MODULE

1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
(1)
(1)
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860

013074
013074
013074
013100
013102
013106
013112
013114
013120
013136
013154
013160
013166
013170
013174
013202
013210
013216
013224
013230
013232
013250
013264
013270
013274
013300
013302
013306
013312
013316
013322
013326
013332
013336
013342
013346
013364
013372
013400
013406
013410
013414
013420
013426
013432

005767 003534
001402
000167 000324
005767 004614
001422
005067 004606
000167 000252
026727 004614 000016 1\$:
001042
005067 004466
052767 100000 004462
052767 100000 004516
012767 000001 004564
012767 026626 004560
005767 004474
001007
005067 004434
000167 000136
005777 004504
001434
012700 032345
016701 004466
004767 017346
012700 032362
016701 004454
004767 017332
012700 032401
017701 004442
004767 017332
052767 100000 004332
062767 000002 004404 4\$:
026727 004376 000037
001404
005267 004366
000167 000012
012767 000001 004354 5\$:
005267 004346

000016 1\$:
004462
004516
004564
004560
004434
000136
004504
004332
004404 4\$:
000037

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

PROC TERS
TEROS:
60\$:
1\$:
2\$:
3\$:
4\$:
5\$:
TERTN:
RETURN

TST TOQ
BEQ 60\$
JMP TERTN
TST HDRFLG
BEQ 1\$
CLR HDRFLG
CALL PNTLIN,<TERHDR>
CALL PNTLIN,<TRHLIN>
JMP TERTN
CMP ERRO,#16
BNE 3\$
CLR TEREV
BIS #B15,REREV
BIS #B15,HDRFLG
MOV #1,ERR1
MOV #RERTBL,ERR2
TST PRINTD
BNE 2\$
CALL PNTLIN,<NONMG>
CALL PNCRLF
CLR PRINTD
JMP TERTN
TST @ERR2
BEQ 4\$
MOV #TRRNO,R0
MOV ERRO,R1
JSR PC,OCTJSP
MOV #TRRCN,R0
MOV ERR1,R1
JSR PC,OCTJSP
MOV #TRERC,R0
MOV @ERR2,R1
JSR PC,DECJSP
CALL PNTLIN,<TRELIN>
BIS #B15,PRINTD
ADD #2,ERR2
CMP ERR1,#37
BEQ 5\$
INC ERR1
JMP TERTN
MOV #1,ERR1
INC ERRO

:**ENTRY POINT**
:IS THE TTY OUTPUT QUEUE EMPTY?
:YES, CONTINUE
:NO, EXIT
:HEADER PRINTED YET?
:YES, SKIP IT
:NO, PRINT IT NOW.

:AND RETURN
:IS THIS ERROR # 16?
:NOT YET.
:YES, CLEAR XMTR ERROR EVENT
:SET RCVR ERROR EVENT
:SET HEADER FLAG
:LOAD INITIAL XMTR #
:POINT RCVR ERR EVENT AT ITS TABLE
:DID WE PRINT ANYTHING?
:YES, O.K.
:NO, PRINT ``(NONE)''
:NEW LINE

:RETURN
:ANY ERRORS?
:NO.
:READY TO PRINT ERROR NO.

:ERROR NO. IS IN OCTAL
:READY TO PRINT RCVR NO.

:RCVR NO. IS ALSO IN OCTAL
:NOW PRINT ERROR COUNT

:ERROR COUNT IS IN DECIMAL
:ENQUEUE TABLE OUTPUT LINE
:SET 'PRINTED FLAG'
:UPDATE TABLE POINTER
:ALL RCVR'S DONE?
:YES.
:NO, DO NEXT

:NEXT ERROR NO.

.SBTTL RECEIVER ERRORS MODULE

1862
1863
1864
1865
1866
1867
1868
1869
1870

: 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)'

1871 013434

PROC REROS

(1)

REROS:

1872 013434 005767 003174

TST TOQ

: **ENTRY POINT**
: IS TTY OUTPUT QUEUE EMPTY?

1873 013440 001402

BEQ 60\$

: YES, CONTINUE

1874 013442 000167 000310

JMP RERTN

: NO, RETURN

1875 013446 005767 004254

60\$:

TST HDRFLG

: IS HEADER PRINTED YET?

1876 013452 001422

BEQ 1\$

: YES, SKIP IT

1877 013454 005067 004246

CLR HDRFLG

: NO, CLEAR FLAG

1878 013460

CALL PNTLIN,<RERHDR>

: AND PRINT HEADER

1879 013476

CALL PNTLIN,<RRHLIN>

: AND RETURN

1880 013514 000167 000236

JMP RERTN

: DONE LAST ERROR?

1881 013520 026727 004254 000031 1\$:

CMP ERRO,#31

: NOT YET

1882 013526 001034

BNE 3\$

: YES, CLEAR RCVR ERROR EVENT

1883 013530 005067 004130

CLR REREV

: DID WE PRINT ANYTHING?

1884 013534 005767 004164

TST PRINTD

: YES, O.K.

1885 013540 001007

BNE 2\$

: NO, PRINT '(NONE)'

1886 013542

CALL PNTLIN,<NONMG>

1887 013560

2\$:

CALL PNCRLF

: WRAP IT UP!

1888 013574 005067 004124

CLR PRINTD

1889 013600

CALL PRESC

: ANY ERRORS?

1890 013614 000167 000136

JMP RERTN

: NO.

1891 013620 005777 004160

3\$:

TST @ERR2

: READY TO PRINT ERROR NO.

1892 013624 001434

BEQ 4\$

1893 013626 012700 032502

MOV #RCRNO,R0

: ERROR NO.S ARE IN OCTAL

1894 013632 016701 004142

MOV ERRO,R1

: READY TO PRINT XMTR NO.

1895 013636 004767 017022

JSR PC,OCTJSP

: XMTR NO.S ARE IN OCTAL

1896 013642 012700 032517

MOV #RCTRN,R0

: NOW PRINT ERROR COUNT

1897 013646 016701 004130

MOV ERR1,R1

: ERROR COUNTS ARE IN DECIMAL

1898 013652 004767 017006

JSR PC,OCTJSP

: ENQUEUE TABLE OUTPUT LINE

1899 013656 012700 032536

MOV #RCERC,R0

: SET 'PRINTED' FLAG

1900 013662 017701 004116

MOV @ERR2,R1

: UPDATE TABLE POINTER

1901 013666 004767 017006

JSR PC,DECJSP

: ALL XMTRS DONE?

1902 013672

CALL PNTLIN,<RCELIN>

1903 013710 052767 100000 004006

BIS #B15,PRINTD

: NO, DO NEXT

1904 013716 062767 000002 004060

4\$:

ADD #2,ERR2

: YES, NEXT ERROR

1905 013724 026727 004052 000037

CMP ERR1,#37

1906 013732 001404

BEQ 5\$

1907 013734 005267 004042

INC ERR1

1908 013740 000167 000012

JMP RERTN

1909 013744 012767 000001 004030

5\$:

MOV #1,ERR1

1910 013752 005267 004022

INC ERRO

RERTN: RETURN

.SBTTL SUMMARY MODULE

1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924 013760
(1)
(1) 013760
1925 013760 005767 002650
1926 013764 001126
1927 013766 005767 003710
1928 013772 001472
1929 013774 027727 003702 177777 SMCHK:
1930 014002 001507
1931 014004 016704 003672
1932 014010 005764 000004
1933 014014 001475
1934 014016 012700 032557
1935 014022 011401
1936 014024 012702 177777
1937 014030 004767 016630
1938 014034 012700 032601
1939 014040 016401 000002
1940 014044 012702 177777
1941 014050 004767 016610
1942 014054 012700 032625
1943 014060 016401 000004
1944 014064 020127 177770
1945 014070 101415
1946 014072 112720 000115
1947 014076 112720 000130
1948 014102 112720 000103
1949 014106 112720 000116
1950 014112 112720 000124
1951 014116 000404
1952 014120 012702 177777
1953 014124 004767 016550
1954 014130
1955 014146 062767 000006 003526
1956 014154 000167 000062
1957
1958 014160
1959 014176 012767 024736 003476
1960 014204 000167 000032
1961
1962 014210 062767 000006 003464
1963 014216 000167 177552
1964
1965 014222 005067 003444
1966 014226

: DUMP SUMMARY TABLE INDICATING ERROR NUMBERS AND THE LOCATIONS
: OF THE ERRORS AS WELL AS THE NUMBER OF OCCURRENCES OF EACH OF
: THE ERRORS. ERRORS WITH 0 OCCURRENCES WILL NOT BE LISTED. THE
: ENTRY UNDER 'NO. OF OCCURRENCES' IS IN DECIMAL AND IS CAPABLE
: OF EXPANDING TO 65,528. QUANTITIES OF ERRORS IN EXCESS OF
: THIS WILL BE LISTED AS 'MXCNT'.
:

PROC SUMRY
SUMRY:
TST TOQ
BNE SMRTN
TST SUMPNT
BEQ SUMHDR
CMP @SUMPNT,#-1
BEQ SMARXT
MOV SUMPNT,R4
TST 4(R4)
BEQ SMADV
MOV #SMLIN1,R0
MOV (R4),R1
MOV #-1,R2
JSR PC,OCTJSP
MOV #SMLIN2,R0
MOV 2(R4),R1
MOV #-1,R2
JSR PC,OCTJSP
MOV #SMLIN3,R0
MOV 4(R4),R1
CMP R1,#177770
BLOS SMPN
MOVB #'M,(R0)+
MOVB #'X,(R0)+
MOVB #'C,(R0)+
MOVB #'N,(R0)+
MOVB #'T,(R0)+
BR SMSPN
MOV #-1,R2
SMPN: JSR PC,DECJSP
SMPN: CALL PNTLIN,<SMLIN>
ADD #6,SUMPNT
JMP SMRTN
SUMHDR: CALL PNTLIN,<SMTITLE>
MOV #ERTBL,SUMPNT
JMP SMRTN
SMADV: ADD #6,SUMPNT
JMP SMCHK
SMARXT: CLR SUMEV
CALL PRESC

:**ENTRY POINT**
:IS OUTPUT QUEUE EMPTY?
:NO, RETURN
:IS OUTPUT POINTER AT TABLE?
:NO, GO PRINT HEADER
:IS ERROR # = -1 ?
:YES,DONE; EXIT

:CHECK TOTAL COUNT OF THIS ERR
:IF 0, TRY NEXT ERROR
:R0 IS OUTPUT POINTER FOR OCTJSP
:R1 IS DATA BUFF FOR OCTJSP
:DON'T COMPRESS BLANKS

:DON'T COMPRESS BLANKS

:REACHED MAX COUNT?
:NO,OK
:YES, PRINT 'MXCNT' IN
: NOTE THAT 'MXCNT'
: ALSO HAS ONLY 5
: ASCII CHARACTERS.

:DON'T COMPRESS BLANKS

:ENQUEUE SUM LINE FOR OUTPUT
:UPDATE POINTER FOR NEXT LINE
:RETURN

:ENQUEUE SUMMARY HEADER
:SET POINTER TO TOP OF SUM LIST
:RETURN

:SET POINTER TO NEXT ENTRY

:CLEAR SUMMARY EVENT FLAG
:FAKE CNTRL-C KEY

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

M 7
MACY11 30A(1052) 25-JUN-79 08:52 PAGE 31-1
SUMMARY MODULE

SEQ 0090

1967 014242

SMRTN: RETURN

;RETURN

```

1969          .SBTTL ERROR UPDATE ROUTINES
1970          .SBTTL TRANSMITTER ERRORS
1971
1972          ;CALLED BY THE MACRO 'CALL ERRMOD,<P,ERADR>'
1973
1974          014244          PROC      ERRMOD,<ERRNUM,ERRADR>
(3)          000002          ERRNUM   =      2
(3)          000004          ERRADR   =      4
(1)
(1)          014244          ERMIS:
1975          014244          010046          MOV      R0,-(SP)          ;**ENTRY POINT**
1976          014246          010146          MOV      R1,-(SP)          ;SAVE R0 ON STACK
1977          014250          010246          MOV      R2,-(SP)          ; AND R1
1978          014252          010346          MOV      R3,-(SP)          ; AND R2
1979          014254          017767          001626          003512          MOV      @TCR,TEMP          ;AND R3
1980          014262          042767          160377          003504          MOV      @TCR,TEMP          ;SAVE ADDRESS OF CONNECTED RCVR
1981          014270          000367          003500          BIC      #160377,TEMP          ;GET IT IN THE RIGHT BYTE
1982          014274          016500          000002          SWAB     TEMP          ;GET ERROR NUMBER
1983          014300          010067          003472          MOV      ERRNUM(R5),R0          ;SAVE ERROR NUMBER
1984          014304          006300          ASL      R0          ;FIND TABLE ENTRY
1985          014306          010001          MOV      R0,R1          ; IN ORDER TO UPDATE
1986          014310          006300          ASL      R0          ; ADDRESS FIELD
1987          014312          060100          ADD      R1,R0
1988          014314          062700          024730          ADD      #ERTBL0,R0
1989          014320          016560          000004          000002          MOV      ERRADR(R5),2(R0)          ;ENTER ERROR ADDRESS
1990          014326          026027          000004          177771          CMP      4(R0),#177771          ;AT MAX COUNT YET?
1991          014334          103401          BLO     ERMIS          ;NO, INCREMENT COUNT
1992          014336          000444          BR      ERMIS          ;YES SKIP UPDATE.
1993          014340          005260          000004          ERMIS:  INC      4(R0)          ;UPDATE OCCURRENCES
1994          014344          005367          003426          DEC      TEMP1          ; (ERR # - 1)
1995          014350          MULT     37,TEMP1,R3          ; (E-1)X37
1996          014414          MULT     2,TEMP1,R3          ; [(E-1)X37]X2
1997          014420          005367          003350          DEC      TEMP          ; RCVR ADDR - 1
1998          014424          MULT     2,TEMP,R3          ; (R-1)X2
1999          014430          066767          003342          003336          ADD      TEMP1,TEMP          ; [(E-1)X37]X2 + (R-1)X2
2000          014436          062767          025160          003330          ADD      #TERTBL,TEMP          ;TEMP = #TERTBL + <[(E-1)X37]X2 + (R-1)X2>
2001          014444          005277          003324          INC      @TEMP          ;UPDATE TABLE ENTRY FOR THIS ERROR
2002          014450          012603          ERMIS:  MOV      (SP)+,R3          ;RESTORE R3
2003          014452          012602          MOV      (SP)+,R2          ;RESTORE R2
2004          014454          012601          MOV      (SP)+,R1          ;RESTORE R1
2005          014456          012600          MOV      (SP)+,R0          ;RESTORE R0
2006          014460          RETURN          ;RETURN

```

```
2008 .SBTTL RECEIVER ERRORS
2009
2010 ;CALLED BY THE MACRO 'CALL ERRMOR,<P,ERRADR>'
2011
2012 014462 PROC ERRMOR,<ERRNUM,ERRADR>
(3) 000002 ERRNUM = 2
(3) 000004 ERRADR = 4
(1)
(1) 014462 ERRMOR: ;**ENTRY POINT**
2013 014462 010046 MOV R0,-(SP) ;SAVE R0 ON THE STACK
2014 014464 010146 MOV R1,-(SP) ;AND R1
2015 014466 010246 MOV R2,-(SP) ;AND R2
2016 014470 010346 MOV R3,-(SP) ;AND R3
2017 014472 017767 001430 003274 MOV @RCR,TEMP ;GET ADDR OF CONNECTED XMTR
2018 014500 042767 160377 003266 BIC #160377,TEMP
2019 014506 000367 003262 SWAB TEMP ; INTO RIGHT BYTE
2020 014512 016500 000002 MOV ERRNUM(R5),R0 ;GET ERROR NUMBER
2021 014516 010067 003254 MOV R0,TEMP1 ;AND SAVE IT
2022 014522 006300 ASL R0 ; FIND 'SUM' TABLE ENTRY
2023 014524 010001 MOV R0,R1 ; IN ORDER TO UPDATE
2024 014526 006300 ASL R0 ; ADDRESS FIELD
2025 014530 060100 ADD R1,R0 ; AND # OF OCCURRENCES
2026 014532 062700 024730 ADD #ERTBLO,R0
2027 014536 016560 000004 000002 MOV ERRADR(R5),2(R0) ;ENTER ERROR ADDRESS
2028 014544 026027 000004 177771 CMP 4(R0),#177771 ;AT MAX COUNT YET?
2029 014552 103401 BLO ERMINR ;NO, UPDATE ERROR COUNT
2030 014554 000447 BR ERMISR ;YES, SKIP UPDATES
2031 014556 005260 000004 ERMINR: INC 4(R0) ;UPDATE OCCURRENCES
2032 014562 162767 000015 003206 SUB #15,TEMP1 ;ALIGN ERRORS TABLE FOR RCVR
2033 014570 005367 003202 DEC TEMP1 ; ERR # - 1 (E-1)
2034 014574 MULT 37,TEMP1,R3 ; (E-1)x37
2035 014640 MULT 2,TEMP1,R3 ; [(E-1)x37]x2
2036 014644 005367 003124 DEC TEMP ; XMTR ADDR - 1 (T-1)
2037 014650 MULT 2,TEMP,R3 ; (T-1)x2
2038 014654 066767 003116 003112 ADD TEMP1,TEMP ; [(E-1)x37]x2 + (T-1)x2
2039 014662 062767 026626 003104 ADD #RERTBL,TEMP ; TEMP = #RERTBL+[(E-1)x37]x2 + (T-1)x2
2040 014670 005277 003100 INC @TEMP ;UPDATE RCVR ERROR COUNT
2041 014674 012603 ERMISR: MOV (SP)+,R3 ;RESTORE R3
2042 014676 012602 MOV (SP)+,R2 ;RESTORE R2
2043 014700 012601 MOV (SP)+,R1 ;RESTORE R1
2044 014702 012600 MOV (SP)+,R0 ;RESTORE R0
2045 014704 RETURN
```

```
2047 .SBTTL UTILITY ROUTINES
2048 .SBTTL PROCESS AN INPUT CHARACTER FROM THE TTY
2049
2050 014706 PROC TTINP
(1)
(1) 014706 TTINP: ;**ENTRY POINT**
2051
2052 014706 CALL DEQ,<FRELEM,TIQ> ;GET TTY INPUT CHARACTER.
2053 014726 142767 000200 013340 BICB #200,FRELEM ;CLEAR OFF PARITY BIT.
2054 014734 116700 013334 MOVB FRELEM,R0
2055 014740 120027 000141 CMPB R0,#141 ;CONVERT LOWER CASE INPUT
2056 014744 002405 BLT TTINH ; TO REGULAR (UPPER CASE)
2057 014746 120027 000172 CMPB R0,#172 ; LETTERS INSTEAD.
2058 014752 003002 BGT TTINH
2059 014754 042700 000040 BIC #B05,R0 ;CONVERT CHARACTER
2060 014760 010067 013310 TTINH: MOV R0,FRELEM ; IN 'FRELEM'
2061 014764 012701 016044 MOV #CMCHTB,R1 ; TO ITS PROCESSING
2062 014770 004767 015642 JSR PC,PROCHR ; ROUTINE.
2063 014774 001407 BEQ CMMISC ;IF MISC CHAR, CALL MISC PROC.
2064 014776 CALL @R1,<FRELEM> ;CALL PROCESSING ROUTINE.
2065 015012 000407 BR TTRET ;RETURN.
2066
2067 015014 CMMISC: CALL PRMISC,<FRELEM> ;PROCESS MISCELLANEOUS CHARACTER
2068 015032 TTRET: RETURN ;RETURN.
```

.SBTTL TTY INPUT CHARACTER PROCESSING ROUTINES

```
2070
2071
2072
2073
2074
2075
2076 015034
(1)
(1) 015034
2077 015034 105767 002610
2078 015040 002015
2079 015042 112777 000015 001342
2080 015050 105367 002575
2081 015054 105067 002570
2082 015060
2083 015074
2084
2085
2086
2087
2088
2089 015076
(1)
(1) 015076
2090 015076 005067 001532
2091 015102 016767 001540 001534
2092 015110
2093 015124
2094 015142
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105 015144
(1)
(1) 015144
2106 015144 052767 100000 002560
2107 015152
2108
2109
2110
2111
2112
2113
2114
2115 015154
(1)
(1) 015154
2116 015154 042767 100000 002550
2117 015162
```

```

:
: PROCESSOR FOR CARRIAGE RETURN
: THIS CHARACTER SIGNIFIES THE END OF A COMMAND LINE.
:
: PROC PRCR
PRCR:
: **ENTRY POINT**
: IF INPUT NOT BEING TYPED,
: IGNORE CHARACTER.
: PUT CR IN BUFFER.
: SET COMMAND ENTERED FLAG.
: CLEAR INPUT REQUESTED FLAG.
: ECHO CR & LF.
: RETURN.
TSTB REQINP
BGE CRRET
MOVB #CR,@CBUFPT
DECB CMDENT
CLRB REQINP
CALL PNCRLF
CRRET: RETURN

:
: PROCESSOR FOR CONTROL-O CHARACTER.
: ALL CHARACTERS PRESENTLY IN THE TTY OUTPUT QUEUE ARE THROWN AWAY.
:
: PROC PRCTLO
PRCTLO:
: **ENTRY POINT**
: THROW AWAY ALL CHARACTERS
: IN TTY OUTPUT QUEUE.
: PRINT <CR> AND <LF>
: ECHO '^O'.
: RETURN.
CLR TOQ
MOV TOQ+QBACK,TOQ+QFRONT
CALL PNCRLF
CALL PNTLIN,<CTLOMG>
RETURN

:
: PROCESSOR FOR CONTROL S.
: TEMPORARILY SUSPEND PRINTOUT ON CONSOLE DEVICE
:
: PRINTOUT WILL BE RESUMED UPON RECEIPT OF A CNTRL-Q
: OR ANY OTHER KEYBOARD INTERRUPT.
:
: PROC PRCTLS
PRCTLS:
: **ENTRY POINT**
: SET TTY HALT FLAG
BIS #B15,THLTFL
RETURN

:
: PROCESSOR FOR CONTROL Q.
: RESUME PRINTOUT ON CONSOLE IF TTOUT QUEUE HAS NOT BEEN
: CLEARED BY SOME OTHER MEANS.
:
: PROC PRCTLQ
PRCTLQ:
: **ENTRY POINT**
: CLEAR TTY HALT FLAG
BIC #B15,THLTFL
RETURN
```

2118
2119
2120
2121
2122
2123
2124
(1)
(1)
2125
2126
2127
2128
2129
2130
2131
2132
2133
(1)
(1)
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
(1)
(1)
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167

015164
015164
015164
015202 000167 000016

015206
015206
015206

015224
015224
015224 005767 002430
015230 001402
015232 005077 000722
015236 105367 002406
015242 005067 002430
015246 005067 002420
015252 005067 002404
015256 005067 002402
015262 005067 002422
015266
015306
015326
015346
015366 012767 016206 001016
015374 105067 002251
015400

```

:
: PROCESSOR FOR CONTROL-C.
: ECHO '^C' AND GO TO PRESC ROUTINE.
:
PROC PRCTL
PRCTL:
CALL PNTLIN,<CTLCMG>
JMP PRESC
: **ENTRY POINT**
: ECHO '^C'
: GO TO PRESC ROUTINE.

:
: PROCESSOR FOR CONTROL-U.
: INPUT COMMAND BEING TYPED IS DISCARDED.
:
PROC PRCTLU
PRCTLU:
CALL PNTLIN,<CTLUMG>
: **ENTRY POINT**
: ECHO '^U'
: DROP INTO PRESC

:
: PROCESSOR FOR CNTRL-C KEY.
: THIS KEY IS USED TO REQUEST INPUT. ALL OTHER EVENTS WILL CEASE
: AFTER THE CURRENT LINE. A 'PCL' WILL BE ECHOED ON THE TTY. THE
: OPERATOR IS NOW FREE TO TYPE IN A COMMAND.
:
PROC PRESC
PRESC:
TST KWFLG
BEQ 14$
CLR @LCS
14$:
DECB REQINP
CLR STSEV
CLR SUMEV
CLR TEREV
CLR REREV
CLR PCLGO
CALL ENQ,<A.P,TOQ>
CALL ENQ,<A.C,TOQ>
CALL ENQ,<A.L,TOQ>
CALL ENQ,<A.PR,TOQ>
MOV #CMDBUF,CBUFPT
CLRB CMDENT
RETURN
: **ENTRY POINT**
: GOT A CLOCK?
: NO.
: STOP THE CLOCK
: SET INPUT REQUESTED FLAG.
: CLR STATUS EVENT
: CLR SUMMARY EVENT
: CLR XMTR ERROR EVENT
: AND RCVR ERROR EVENT
: STOP XMTR
: ECHO 'PCL'

: INITIALIZE COMMAND BUF PTR.
: CLEAR COMMAND ENTERED FLAG.
: RETURN.

:
: PROCESS RUB OUT KEY.
: LAST CHARACTER IN COMMAND BUFFER IS DELETED. A '^' IS ECHOED.

```



```

2168 ;
2169 ;
2170 015402 PROC PRDEL
(1)
(1) 015402 PRDEL: ;**ENTRY POINT**
2171 015402 105767 002242 TSTB REQINP ;IF INPUT NOT REQUESTED,
2172 015406 002034 BGE PDRET ; THEN IGNORE RUBOUT CHAR.
2173 015410 026727 000776 016206 CMP CBUFPT,#CMDBUF ;IF AT BEGINNING OF BUFFER,
2174 015416 003016 BGT 1$ ; THEN ISSUE NEW 'PCL>'
2175 015420 CALL PNCRLF
2176 015434 CALL PRESC
2177 015450 000167 000024 JMP PDRET
2178 015454 005367 000732 1$: DEC CBUFPT ;DECREMENT BUFFER POINTER.
2179 015460 CALL ENQ,<A.BKSL,TOQ> ;ECHO A '^'.
2180 015500 PDRET: RETURN ;RETURN.
2181
2182
2183 ;
2184 ; PROCESSOR FOR MISCELLANEOUS CHARACTERS.
2185 ; THE CHARACTER IS APPENDED TO THE COMMAND BUFFER.
2186 ;
2187
2188
2189 015502 PROC PRMISC,<CHAR>
(3) CHAR = 2
(1) 000002
(1) 015502 PRMISC: ;**ENTRY POINT**
2190 015502 105767 002142 TSTB REQINP ;IF INPUT NOT REQUESTED,
2191 015506 002034 BGE PMRET ; THEN JUST IGNORE CHARACTER.
2192 015510 117500 000002 MOVB @CHAR(R5),R0
2193 015514 120027 000040 CMPB R0,#'
2194 015520 002403 BLT PM176 ;IF CHARACTER
2195 015522 120027 000140 CMPB R0,#140 ; IS NONPRINTING
2196 015526 002402 BLT PM187 ; THEN CHANGE
2197 015530 012700 000077 PM176: MOV #'?,R0 ; IT TO
2198 015534 026727 000652 016411 PM187: CMP CBUFPT,#CBFEND-1 ; ASCII '?'.
2199 015542 001416 BEQ PMRET ;IF WE ARE AT END OF BUFFER,
2200 015544 110077 000642 MOVB R0,@CBUFPT ; THEN JUST RETURN.
2201 015550 005267 000636 INC CBUFPT ;PUT CHARACTER INTO BUFFER
2202 015554 110067 012516 MOVB R0,BKELEM ;UPDATE BUFFER POINTER.
2203 015560 CALL ENQ,<BKELEM,TOQ> ;ECHO CHARACTER.
2204 015600 PMRET: RETURN ;RETURN.
  
```

2206 .SBTTL TTY OUTPUT HANDLERS
2207
2208
2209
2210 : OUTPUT A CHARACTER TO THE TELETYPE IF IT IS READY.
2211 :
2212 PROC TTOUT

(1)
(1) 015602 TTOUT: : **ENTRY POINT**
2213 015602 105777 000344 TSTB @TTXCSR : IF DEVICE IS NOT READY,
2214 015606 002020 BGE TORET : THEN JUST RETURN.
2215 015610 005767 002116 TST THLTFI : IS TTY HALTED (CNTRL-S)?
2216 015614 001402 BEQ 1\$: NO, O.K. TO PRINT
2217 015616 000167 000026 JMP TORET : YES, DON'T DO ANYTHING.
2218 015622 1\$: CALL DEQ,<FRELEM,TOQ> : GET NEXT CHAR TO TYPE.
2219 015642 116777 012426 000304 MOVB FRELEM,@TTXBUF : OUTPUT IT.
2220 015650 TORET: RETURN : RETURN.

2221
2222
2223
2224 .SBTTL TTY INPUT INTERRUPT PROCESSGRS

2225
2226 015652 TTIINT: : **INTERRUPT ENTRY POINT**
2227 015652 005067 002054 CLR THLTFI : CLEAR TTY HALT FLAG ON INPUT
2228 015656 REGSAV : SAVE R0 - R5.
2229 015662 117767 000262 012414 MOVB @TTRBUF,INTIMP : GET INPUT CHARACTER.
2230 015670 CALL ENQ,<INTIMP,TIQ> : PUT IT IN TTY INPUT QUEUE.
2231 015710 REGRES : RESTORE R0 - R5.
2232 015714 000002 RTI : RETURN.
2233

.SBTTL MESSAGE PRINT ROUTINE

```
2235  
2236  
2237  
2238  
2239  
2240  
2241  
2242  
2243 015716 000002  
  (3)  
  (1)  
2244 015716 016546 000002  
2245 015722 117667 000000 012346  
2246 015730 001412  
2247 015732  
2248 015752 005216  
2249 015754 000762  
2250  
2251 015756 005026  
2252 015760 000400  
2253  
2254  
2255  
2256  
2257  
2258 015762  
  (1)  
  (1) 015762  
2259 015762  
2260 016002  
2261 016022  
2262 016042
```

```
      ENQUEUE CHARACTERS STARTING AT MESSAG IN TTY OUTPUT  
      QUEUE TOQ UNTIL A ZERO BYTE IS ENCOUNTERED. ENQUEUE A CR & LF  
      INSTEAD OF THE ZERO, & THEN EXIT FROM THE ROUTINE.  
  
      PROC      PNTLIN,<MESSAG>  
      =        2  
  
      PNTLIN:  MOV      MESSAG(R5),-(SP)      ;**ENTRY POINT**  
                @ (SP),BKELEM              ;GET ADDRESS OF MESSAGE.  
      PLODEC:  MOV      @ (SP),BKELEM        ;GET A CHARACTER.  
                BEQ     PLCRLF              ;IF NULL, APPEND CR & LF.  
                CALL    ENQ,<BKELEM,TOQ>    ;ENQUEUE CHAR FOR TTY OUTPUT.  
                INC     @SP                 ;POINT TO NEXT CHARACTER.  
                BR      PLODEC              ;PROCESS IT.  
  
      PLCRLF:  CLR     (SP)+                 ;POP ADDR FROM STACK.  
                BR      PL219               ;APPEND CR & LF.  
  
      ENQUEUE A CR & LF IN TTY OUTPUT QUEUE.  
  
      PROC      PNCRLF  
  
      PNCRLF:  CALL    ENQ,<A.CR,TOQ>        ;**ENTRY POINT**  
      PL219:  CALL    ENQ,<A.LF,TOQ>        ;ENQUEUE A CR.  
                CALL    ENQ,<ZERO,TOQ>     ;ENQUEUE A LF.  
                RETURN                       ;ENQUEUE A NULL FILL CHAR.  
                RETURN                       ;RETURN.
```

.SBTTL DATA AREAS

2264
2265
2266
2267
2268 016044
2269 016044 000003 015164
2270 016050 000012 015034
2271 016054 000015 015034
2272 016060 000017 015076
2273 016064 000021 015154
2274 016070 000023 015144
2275 016074 000025 015206
2276 016100 000177 015402
2277 016104 177777
2278
2279
2280
2281
2282 016106 164200
2283 016110 164202
2284 016112 164204
2285 016114 164206
2286 016116 164210
2287 016120 164212
2288 016122 164213
2289 016124 164214
2290 016126 164220
2291 016130 164222
2292 016132 164224
2293 016134 164226
2294 016136 164230
2295 016140 164234
2296
2297 016142 000170
2298 016144 000174
2299
2300 016146 177560
2301 016150 177562
2302 016152 177564
2303 016154 177566
2304 016156 000060
2305 016160 177546
2306
2307
2308
2309
2310 016162 000134
2311 016164 000000
2312 016166 000120
2313 016170 000103
2314 016172 000114
2315 016174 000076
2316
2317
2318
2319

: TABLE ASSOCIATING SPECIAL TTY INPUT CHARACTERS WITH THEIR
: PROCESSING ROUTINES.

CMCHTB:
.WORD CTL.C,PRCTLC
A.LF: .WORD LF.,PRCR
A.CR: .WORD CR.,PRCR
.WORD CTL.O,PRCTLO
.WORD CTL.Q,PRCTLO
.WORD CTL.S,PRCTLS
.WORD CTL.U,PRCTLU
.WORD RUBOUT,PRDEL
.WORD -1

:DEVICE ADDRESS TABLES:

TCR: .WORD PCLTXM
TSR: .WORD PCLTXM+2
TSDB: .WORD PCLTXM+4
TSBC: .WORD PCLTXM+6
TSBA: .WORD PCLTXM+10
TMMR: .WORD PCLTXM+12
TMMRH: .WORD PCLTXM+13
TSCRC: .WORD PCLTXM+14
RCR: .WORD PCLRCV
RSR: .WORD PCLRCV+2
RDDDB: .WORD PCLRCV+4
RDBC: .WORD PCLRCV+6
RDBA: .WORD PCLRCV+10
RDCRC: .WORD PCLRCV+14

TXMVEC: .WORD 170
RCVVEC: .WORD 174

TTRCSR: .WORD TTDEV
TTRBUF: .WORD TTDEV+2
TTXCSR: .WORD TTDEV+4
TTXBUF: .WORD TTDEV+6
TTVECT: .WORD TTVCTR
LCS: .WORD 177546

:VECTOR ADDRESS.
:KW11-L LINE CLOCK ADDR

: CHARACTER CONSTANTS. THESE ARE DEFINED AS WORDS SO THAT THEY MAY
: BE ENQUEUED.

A.BKSL: .WORD '\':ASCII '\'
ZERO: .WORD 0:ASCII NULL
A.P: .WORD 'P':ASCII 'P'
A.C: .WORD 'C':ASCII 'C'
A.L: .WORD 'L':ASCII 'L'
A.PR: .WORD '>':ASCII '>' (PROMPT)

.EVEN

:DEVICE ADDRESS AND VECTOR VARIABLES

```
2320 ;CHANGE THESE LOCATIONS TO MODIFY ALL DEVICE ADDRESSES AND VECTORS
2321 ;FOR PCL11.
2322
2323 016176 164200 TXDEV: .WORD 164200 ;DEFAULT IS 164200
2324 016200 000170 TXVEC: .WORD 170 ;DEFAULT IS 170
2325 016202 164220 RCDEV: .WORD 164220 ;DEFAULT IS 164220
2326 016204 000174 RCVEC: .WORD 174 ;DEFAULT IS 174
2327
2328 ; BUFFER & MESSAGE AREAS.
2329 016206 000204 CMDBUF: .BLKB 132. ;TTY INPUT COMMAND BUFFER.
2330 016412 033333 CBFEND = ;CMDBUF BUFFER POINTER.
2331 016412 033333 CBUFPT: .WORD 33333
2332 ;FOR ECHOING CONTROL CHARACTERS.
2333 016414 047536 000 CTLOMG: .ASCIZ "'^O'" ;"
2334 016417 136 000125 CTLUMG: .ASCIZ "'^U'" ;"
2335 016422 041536 000 CTLCMG: .ASCIZ "'^C'" ;"
2336
2337 016425 040 051105 047522 RPMSG: .ASCIZ "' ERROR'"
2338 ; QUEUE DEFINITIONS.
2339 .IRP LC <AQ, TI, TO>
2340 .EVEN
2341 .LIST
2342 LC'Q: .WORD 0 ;QELEMS
2343 .WORD LC'SIZE ;QSIZE
2344 .WORD LC'AREA, LC'END ;QTOP & QBOT
2345 .WORD 33333, 33333 ;QFRONT & QBACK
2346 LC'AREA: .BLKW LC'SIZE ;LC'Q AREA
2347 LC'END = .
2348 .NLIST
2349 .ENDM
2350 AQQ: .WORD 0 ;QELEMS
2351 .WORD AOSIZE ;QSIZE
2352 .WORD AOAREA, AOEND ;QTOP & QBOT
2353 .WORD 33333, 33333 ;QFRONT & QBACK
2354 AOAREA: .BLKW AOSIZE ;AQQ AREA
2355 AOEND = .
2356 TIQ: .WORD 0 ;QELEMS
2357 .WORD TISIZE ;QSIZE
2358 .WORD TIAREA, TIEND ;QTOP & QBOT
2359 .WORD 33333, 33333 ;QFRONT & QBACK
2360 TIAREA: .BLKW TISIZE ;TIQ AREA
2361 TIEND = .
2362 TOQ: .WORD 0 ;QELEMS
2363 .WORD TOSIZE ;QSIZE
2364 .WORD TOAREA, TOEND ;QTOP & QBOT
2365 .WORD 33333, 33333 ;QFRONT & QBACK
2366 TOAREA: .BLKW TOSIZE ;TOQ AREA
2367 TOEND = .
2368 ; FLAG VARIABLES.
2369 ; < 0 ==> TRUE
2370 ; >=0 ==> FALSE
2371
2372 REQINP: .BYTE 333 ;INPUT REQUEST IS BEING TYPED.
2373 CMDENT: .BYTE 333 ;COMMAND HAS BEEN ENTERED.
2374
```

2357	017652	000	TICKS:	.BYTE	0
2358	017653	000	SECNDS:	.BYTE	0
2359	017654	000	MINUTS:	.BYTE	0
2360		017656		.EVEN	
2361	017656	000000	HOURS:	.WORD	0
2362	017660	000000	KWFLG:	.WORD	0
2363	017662	000000	TEREV:	.WORD	0
2364	017664	000000	REREV:	.WORD	0
2365	017666	000000	QLDEV:	.WORD	0
2366	017670	000000	DATGEV:	.WORD	0
2367	017672	000000	SUMEV:	.WORD	0
2368	017674	000000	SUMSV:	.WORD	0
2369	017676	000000	STSEV:	.WORD	0
2370	017700	000000	STPNTR:	.WORD	0
2371	017702	000000	SUMPNT:	.WORD	0
2372	017704	000000	TXMEV:	.WORD	0
2373	017706	000000	TXMSV:	.WORD	0
2374	017710	000000	PCLGO:	.WORD	0
2375	017712	000000	RCVEV:	.WORD	0
2376	017714	000000	SPCEV:	.WORD	0
2377	017716	000000	XSPCEV:	.WORD	0
2378	017720	000000	FIRST:	.WORD	0
2379	017722	000000	NEXT:	.WORD	0
2380	017724	000000	PRINTD:	.WORD	0
2381	017726	000000	HDRFLG:	.WORD	0
2382	017730	000000	PADFLG:	.WORD	0
2383	017732	000000	THLTFL:	.WORD	0
2384					
2385				.EVEN	
2386					
2387				: DATA VARIABLES.	
2388	017734	133333	ORIGSD:	.WORD	133333
2389	017736	000000	DTSEED:	.WORD	0
2390	017740	000333	MLSD:	.WORD	333
2391	017742	000000	MSGLSD:	.WORD	0
2392	017744	000000	RCSEED:	.WORD	0
2393	017746	000000	RCTXPS:	.WORD	0
2394	017750	000000	RJCTF:	.WORD	0
2395	017752	000000	TRNKF:	.WORD	0
2396	017754	000000	TXML:	.WORD	0
2397	017756	037565	RANM:	.WORD	37565
2398	017760	012247	RANK:	.WORD	12247
2399	017762	000000	CURAD:	.WORD	0
2400	017764	000000	RSFC:	.WORD	0
2401	017766	000000	ESCFLG:	.WORD	0
2402	017770	000000	OBJCNT:	.WORD	0
2403	017772	000000	RSHOLD:	.WORD	0
2404	017774	000000	TEMP:	.WORD	0
2405	017776	000000	TEMP1:	.WORD	0
2406	020000	000000	ERRO:	.WORD	0
2407	020002	000000	ERR1:	.WORD	0
2408	020004	000000	ERR2:	.WORD	0
2409					
2410					
2411					
2412					

;ADDRESS SILO DATA BUFFER AREA

```
2413 020006 000102 ADSILO: .BLKB 66. ;66. BYTE AREA FOR SILO DATA
2414
2415 ;RECEIVER ADDRESS TABLE
2416
2417 000001 X = 1
2418 020110 RADB:
2419 000037 .REPT 31.
2420 .WORD 0 ;ACTIVITY FLAG
2421 .WORD X ;RECEIVER ADDRESS
2422 .WORD 0,0 ;ATTEMPTS ENTRY
2423 .WORD 0,0 ;SUCCESSSES ENTRY
2424
2425 X = X+1
2426 .ENDR
2427 020674 020674 RADEND: .WORD .
2428 020074 RADBO= RADB-14
2429
2430 ;TRANSMITTER DATA BUFFER:
2431
2432 020676 001020 DATBUF: .BLKW 1020
2433
2434 ;RECEIVER DATA BUFFER:
2435
2436 RCBUF: .BLKW 1000
2437 022736 001000
2438
2439 ;EXERCISER ERROR TABLE
2440
2441 000001 Y = 1 ;INITIAL ERROR NUMBER
2442 024736 ERTBL:
2443 000030 .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 025156 177777 .WORD -1 ;LAST ERROR # IS -1
2451
2452 024730 ERTBLO = ERTBL-6
2453
2454 ; DETAILED ERROR TABLES FOR RCVR AND XMTR ERRORS:
2455
2456 025160 000623 TERTBL: .BLKW 37*15 ;RESERVE SPACE FOR XMTR ERRORS
2457
2458 026626 000623 RERTBL: .BLKW 37*15 ;RESERVE SPACE FOR RCVR ERRORS
2459
2460
2461
2462
2463
2464
2465 030274 033333 FRELEM: .WORD 33333 ;STORAGE FOR DEQUEUED ELEMENT.
2466 030276 033333 BKELEM: .WORD 33333 ;STORAGE FOR ENQUEUED ELEMENT.
2467 030300 033333 TCBFPT: .WORD 33333 ;CMDBUF POINTER USED DURING SCAN
2468 030302 033333 TCBIN: .WORD 33333 ;BINARY VALUE OF INPUT PARAMETER
```

```

2469 030304 033333          INTTMP: .WORD 33333          ;TEMP STORAGE FOR INTERRUPT PROC
2470
2471                      .SBTTL  KEYWORD TABLE
2472
2473          ;KEYWORD TABLE ASSOCIATING A COMMAND WITH ITS PROCESSING ROUTINE
2474
2475 030306 004524          CMDTBL: .WORD  CPADD
2476 030310          002          003          .BYTE  2,3
2477 030312 042101 020104          .ASCII /ADD /
2478 030316 005672          .WORD  CPASS
2479 030320          002          006          .BYTE  2,6
2480 030322 051501 044523 047107          .ASCII /ASSIGN/
2481 030330 004752          .WORD  CPCLR
2482 030332          002          005          .BYTE  2,5
2483 030334 046103 040505 020122          .ASCII /CLEAR /
2484 030342 005114          .WORD  CPCNT
2485 030344          002          010          .BYTE  2,8.
2486 030346 047503 052116 047111          .ASCII /CONTINUE/
2487 030354 042525          .WORD  CPDEL
2488 030356 004632          .BYTE  1,6
2489 030360          001          006          .ASCII /DELETE/
2490 030362 042504 042514 042524          .WORD  CPERR
2491 030370 006600          .BYTE  1,6
2492 030372          001          006          .ASCII /ERRORS/
2493 030374 051105 047522 051522          .WORD  CPGO
2494 030402 005510          .BYTE  1,2
2495 030404          001          002          .ASCII /GO/
2496 030406 047507          .WORD  CPINIT
2497 030410 005210          .BYTE  1,10.
2498 030412          001          012          .ASCII /INITIALIZE/
2499 030414 047111 052111 040511
2500 030422 044514 042532          .WORD  CPMAS
2501 030426 004116          .BYTE  1,6
2502 030430          001          006          .ASCII /MASTER/
2503 030432 040515 052123 051105          .WORD  CPRANG
2504 030440 004366          .BYTE  2,5
2505 030442          002          005          .ASCII /RANGE /
2506 030444 040522 043516 020105          .WORD  CPRIB
2507 030452 004274          .BYTE  2,3
2508 030454          002          003          .ASCII /RIB /
2509 030456 044522 020102          .WORD  CPSEC
2510 030462 004202          .BYTE  2,9.
2511 030464          002          011          .ASCII /SECONDARY /
2512 030466 042523 047503 042116
2513 030474 051101 020131          .WORD  CPSILO
2514 030500 003406          .BYTE  2,4
2515 030502          002          004          .ASCII /SILO/
2516 030504 044523 047514          .WORD  CPSTAT
2517 030510 005054          .BYTE  2,6
2518 030512          002          006          .ASCII /STATUS/
2519 030514 052123 052101 051525          .WORD  CPSUM
2520 030522 005456          .BYTE  2,7
2521 030524          002          007          .ASCII /SUMMARY /
2522 030526 052523 046515 051101
2523 030534 020131          .WORD  0,0
2524 030536 000000 000000
  
```


2521						
2522						
2523	030542	000000			SCTBL: .WORD	0
2524	030544	001	005		.BYTE	1,5
2525	030546	046103	040505	020122	.ASCII	/CLEAR /
2526	030554	000001			.WORD	1
2527	030556	001	003		.BYTE	1,3
2528	030560	042523	020124		.ASCII	/SET /
2529	030564	000000	000000		.WORD	0,0
2530						
2531					.SBTTL	SOME MORE ASCII STORAGE:
2532						
2533						
2534	030570	020040	020040	020052	MDNER: .ASCIZ	/ * * * * * MASTER DOWN * * * * * /
	030576	020052	020052	020052		
	030604	020052	020052	046440		
	030612	051501	042524	020122		
	030620	047504	047127	020040		
	030626	020052	020052	020052		
	030634	020052	020052	000052		
2535	030642	020040	025052	052040	MSCHNG: .ASCIZ	/ ** THIS UNIT HAS BECOME 'NEW MASTER' **/
	030650	044510	020123	047125		
	030656	052111	044040	051501		
	030664	041040	041505	046517		
	030672	020105	047042	053505		
	030700	046440	051501	042524		
	030706	021122	025040	000052		
2536	030714	020040	025052	051452	SYNTAX: .ASCIZ	/ ***SYNTAX ERROR***/
	030722	047131	040524	020130		
	030730	051105	047522	025122		
	030736	025052	000			
2537	030741	120	046103	030461	PCLEXM: .ASCIZ	/PCL11 EXERCISER V02C CZPLACO (JUN-79)/
	030746	042440	042530	041522		
	030754	051511	051105	053040		
	030762	031060	020103	041440		
	030770	050132	040514	030103		
	030776	020040	045050	047125		
	031004	033455	024471	000		
2538	031011	122	051505	040524	RSTMSG: .ASCIZ	/RESTART AT ADDRESS 204/
	031016	052122	040440	020124		
	031024	042101	051104	051505		
	031032	020123	030062	000064		
2539	031040	025052	047040	020117	MTQMSG: .ASCIZ	/** NO RECEIVERS SELECTED **/
	031046	042522	042503	053111		
	031054	051105	020123	042523		
	031062	042514	052103	042105		
	031070	025040	000052			
2540	031074	051124	050101	042520	TRPDMG: .ASCII	/TRAPPED TO 4 FROM LOCATION /
	031102	020104	047524	032040		
	031110	043040	047522	020115		
	031116	047514	040503	044524		
	031124	047117	040			
2541	031127	116	047116	047116	TRP4AD: .ASCIZ	/NNNNN !/
	031134	020040	000041			
2542	031140	042504	044526	042503	ERTMSG: .ASCIZ	/DEVICE ADDRESS ERROR. USE 'ASSIGN' COMMAND./
	031146	040440	042104	042522		

	031154	051523	042440	051122	
	031162	051117	020056	051525	
	031170	020105	040442	051523	
	031176	043511	021116	041440	
	031204	046517	040515	042116	
	031212	000056			
2543	031214	054105	051105	044503	EXRST: .ASCIZ /EXERCISER STARTED/
	031222	042523	020122	052123	
	031230	051101	042524	000104	
2544	031236	054105	051105	044503	EXCNT: .ASCIZ /EXERCISER CONTINUING/
	031244	042523	020122	047503	
	031252	052116	047111	044525	
	031260	043516	000		
2545	031263	052	025052	044124	MSTMG1: .ASCIZ /***THIS UNIT IS NOT MASTER***/
	031270	051511	052440	044516	
	031276	020124	051511	047040	
	031304	052117	046440	051501	
	031312	042524	025122	025052	
	031320	000			
2546	031321	102	052125	044040	MSTMG2: .ASCIZ /BUT HAS NOW BEEN MADE SECONDARY./
	031326	051501	047040	053517	
	031334	041040	042505	020116	
	031342	040515	042504	051440	
	031350	041505	047117	040504	
	031356	054522	000056		
2547	031362	044124	020105	044523	MSTMG3: .ASCIZ /THE SILO YOU HAVE JUST LOADED WILL BE/
	031370	047514	054440	052517	
	031376	044040	053101	020105	
	031404	052512	052123	046040	
	031412	040517	042504	020104	
	031420	044527	046114	041040	
	031426	000105			
2548	031430	051525	042105	044440	MSTMG4: .ASCIZ /USED IF YOU CLEAR THE CURRENT MASTER./
	031436	020106	047531	020125	
	031444	046103	040505	020122	
	031452	044124	020105	052503	
	031460	051122	047105	020124	
	031466	040515	052123	051105	
	031474	000056			
2549	031476	044124	020105	044523	MSTMG5: .ASCIZ /THE SILO HAS BEEN PADDED WITH ADDRESS '0'/'
	031504	047514	044040	051501	
	031512	041040	042505	020116	
	031520	040520	042104	042105	
	031526	053440	052111	020110	
	031534	042101	051104	051505	
	031542	020123	030042	000042	
2550	031550	044124	051511	052440	THUMST: .ASCIZ /THIS UNIT IS -MASTER-/
	031556	044516	020124	051511	
	031564	026440	040515	052123	
	031572	051105	000055		
2551	031576	044124	051511	052440	THUSCN: .ASCIZ /THIS UNIT IS -SECONDARY-/
	031604	044516	020124	051511	
	031612	026440	042523	047503	
	031620	042116	051101	026531	
	031626	000			
2552	031627	042	044522	021102	RBSTMG: .ASCIZ /'RIB' IS -SET-/

	031634	044440	020123	051455	
2553	031642	052105	000055		
	031646	051042	041111	020042	RBCLMG: .ASCIZ /'RIB' IS -CLEAR-/
	031654	051511	026440	046103	
2554	031662	040505	026522	000	
	031667	105	040514	051520	ELPSTM: .ASCII /ELAPSED TIME (HRS:MIN:SEC:TIK).../
	031674	042105	052040	046511	
	031702	020105	044050	051522	
	031710	046472	047111	051472	
	031716	041505	052072	045511	
2555	031724	027051	027056		
	031730	035060	035060	035060	TMLIN1: .ASCIZ /0:0:0:0 /
	031736	020060	020040	020040	
2556	031744	020040	020040	000	
	031751	122	053103	020122	STITLE: .ASCIZ /RCVR ADDRESS CONNECTION ATTEMPTS SUCCESSFUL CONNECTIONS/
	031756	042101	051104	051505	
	031764	020123	041440	047117	
	031772	042516	052103	047511	
	032000	020116	052101	042524	
	032006	050115	051524	020040	
	032014	052523	041503	051505	
	032022	043123	046125	041440	
	032030	047117	042516	052103	
	032036	047511	051516	000	
2557					
2558	032043				STLIN:
2559	032043	116	047116	020116	STLIN1: .ASCII /NNNN /
	032050	020040	020040	020040	
	032056	020040	020040	020040	
	032064	040			
2560	032065	116	047116	020116	STLIN2: .ASCII /NNNN /
	032072	020040	020040	020040	
	032100	020040	020040	020040	
	032106	020040	020040	040	
2561	032113	116	047116	020116	STLIN3: .ASCIZ /NNNN /
	032120	020040	020040	020040	
	032126	020040	000		
2562					
2563					
2564	032131	105	051122	051117	SMTTLE: .ASCIZ /ERROR NUMBER ERROR ADDRESS NO. OF OCCURRENCES/
	032136	047040	046525	042502	
	032144	020122	020040	042440	
	032152	051122	051117	040440	
	032160	042104	042522	051523	
	032166	020040	020040	047516	
	032174	020056	043117	047440	
	032202	041503	051125	042522	
	032210	041516	051505	000	
2565					
2566	032215	040	025040	020052	NOERMG: .ASCIZ / ** NO ERRORS TO REPORT **/
	032222	047516	042440	051122	
	032230	051117	020123	047524	
	032236	051040	050105	051117	
	032244	020124	025052	000	
2567	032251	124	040522	051516	TERHDR: .ASCIZ /TRANSMITTER ERRORS:/
	032256	044515	052124	051105	

2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630

.EVEN
.SBTTL AUXILIARY ROUTINES
.SBTTL CHARACTER PROCESSOR

: CONVERT A CHARACTER TO ITS PROCESSING ROUTINE ADDRESS BASED
: UPON A TABLE OF ENTRIES IN THE FOLLOWING FORM:

(UNUSED)	CHARACTER
PROCESSING ROUTINE ADDR	

: THE TABLE MUST BE ARRANGED IN ASCENDING ORDER OF CHARACTER VALUES.
: THE TABLE ENDS WITH A DUMMY ENTRY FOR CHARACTER FF (HEXADECIMAL).

: ON ENTRY: R0 = CHARACTER
: R1 = TABLE ADDRESS

: CALL: JSR PC,PROCHR

: ON RETURN: R0 = CHARACTER
: R1 = PROCESSING ROUTINE ADDRESS, IF ANY
: Z = 1 ==> CHARACTER NOT IN TABLE

PROCHR: ;**ENTRY POINT**
PCLOOK: CMPB @R1,R0 ;COMPARE TABLE CHAR
; WITH ARG. CHAR.
; IF SAME, RETURN PROC.
; ROUTINE ADDR.
; IF >, THEN ARG. CHAR
; NOT IN TABLE.
; IF <, POINT TO NEXT
; TABLE ENTRY.
; TRY AGAIN.
; R1 = PROCESSING
; ROUTINE ADDRESS.
; RETURN: Z BIT IS
; OFF.
PCCALL: MOV 2(R1),R1
RTS PC
PCQUIT: CLR R1 ;R1 = 0
RTS PC ;RETURN: Z BIT IS ON.

.SBTTL BINARY TO ASCII CONVERSION

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

: BINARY TO ASCII CONVERSION

: LOCAL MACROS

: SETUP CONVERSION CONTROL WORD ON STACK

: STCVT RADIX,WIDTH,SIGNED,COMPR,BLANKS

: WHERE:

RADIX=NUMERIC VALUE SPECIFYING CONVERSION RADIX
WIDTH=NUMERIC VALUE FROM 1 TO 7 SPECIFYING FIELD WIDTH
SIGNED=SIGN OR MAGNITUDE FLAG. ASCII STRING 'SIGN' SPECI-
FIES SIGNED CONVERSION. ANYTHING ELSE SPECIFIES MAGNI-
TUDE.

COMPR=COMPRESS LEADING ZEROS FLAG. ASCII STRING 'COMPRES' SPE-
CIFIES COMPRESSION OF LEADING ZEROS. ANYTHING ELSE MEANS
INCLUDE LEADING ZEROS OR SPACES IN CONVERSION.

BLANK=REPLACE LEADING ZEROS WITH BLANKS (SPACES). ASCII STRING
'BLANKS' MEANS BLANK REPLACEMENT IF ZERO COMPRESS IS DIS-
ABLED. ANYTHING ELSE SPECIFIES ZERO PADDING.

.MACRO STCVT RADIX,WIDTH,SIGN,COMPR,BLANK

\$BLK=0

\$SGN=0

\$SUP=1*1000

.IF

IDN <BLANK>,<BLANKS>

\$BLK=1*2000

.ENDC

.IF

IDN <SIGN>,<SIGNED>

\$SGN=1*400

.ENDC

.IF

IDN <COMPR>,<COMPRES>

\$SUP=0*1000

.ENDC

MOV

#<WIDTH*4000>!\$BLK!\$SGN!\$SUP!RADIX,-(SP)

.ENDM

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 032664
2707 032664
2708 032670 000411
2709
2710
2711
2712 032672
2713 032672
2714 032676 000406
2715
2716
2717
2718 032700
2719 032700
2720 032704 000403
2721
2722
2723
2724 032706
2725 032706
2726 032712 000400
2727
2728 032714
2729 032714 005702
2730 032716 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

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 H 9 PAGE 41-1
BINARY TO ASCII CONVERSION

SEQ 0111

2731 032720 042716 001000
2732 032724
2733 032724 012602

20\$: BIC #1*1000,(SP) ;ENABLE ZERO SUPPRESS
MOV (SP)+,R2 ;SET CONTROL WORD


```

2735                                     .SBTTL GENERAL BINARY TO ASCII CONVERSION
2736
2737 032726                                CBTA:
2738 032726 004567 000406                JSR     R5,SAVRG                ;SAVE THE NON-VOLATILE REGISTERS
2739 032732 110205                        MOVSB  R2,R5                  ;COPY RADIX BYTE
2740 032734 000302                        SWAB   R2                    ;POSITION REMAINING TO LOW BYTE
2741 032736 106202                        ASRB   R2                    ;SHIFT OFF MAG. FLAG
2742 032740 103005                        BCC    10$                   ;UNSIGNED IF C IS CLR
2743 032742 005701                        TST    R1                    ;POSITIVE VALUE?
2744 032744 100003                        BPL    10$                   ;IF PL, YES
2745 032746 005401                        NEG    R1                    ;MAKE VALUE POSITIVE
2746 032750 112720 000055                MOVSB  #'-(R0)+              ;INSERT A MINUS SIGN
2747 032754                                10$:
2748 032754 010004                        MOV    R0,R4                ;COPY STRING POINTER
2749 032756 000241                        CLC                                ;CLEAR CARRY
2750 032760 106002                        RORB   R2                    ;SHIFT OFF SUPPR FLAG
2751 032762 006002                        ROR    R2                    ;TRANSFER TO R2
2752 032764 006003                        ROR    R3                    ;GET BLANK/ZERO PAD FLAG
2753 032766 105003                        CLRB   R3                    ;CLEAR COUNT BYTE
2754 032770 150203                        BISB   R2,R3                ;TRANSFER COUNT BYTE
2755 032772 105002                        CLRB   R2                    ;CLEAR FILL BYTE
2756 032774 152702 000060                BISB   #'0,R2               ;SET FILL BYTE
2757 033000 010100                        MOV    R1,R0                ;DIVIDEND TO R0
2758 033002                                1$:
2759 033002 010501                        MOV    R5,R1                ;SET CONVERSION RADIX
2760 033004 004767 000272                JSR    PC,DIV               ;DIVIDE EM UP
2761 033010 020127 000011                CMP    R1,#9                ;RESULT EXCEED NUMERICS?
2762 033014 101402                        BLOS   15$                   ;IF LOS, NO
2763 033016 062701 000007                ADD    #7,R1                ;BIAS TO FALL IN ALPHA
2764 033022                                15$:
2765 033022 060201                        ADD    R2,R1                ;ADD CHARACTER BIAS
2766 033024 010146                        MOV    R1,-(SP)             ;SAVE CHARACTER
2767 033026 105303                        DECB   R3                    ;DECREMENT CHARACTER COUNT
2768 033030 003412                        BLE    3$                    ;IF LE NO DIGITS LEFT
2769 033032 005700                        TST    R0                    ;ZERO QUOTIENT
2770 033034 001006                        BNE    2$                    ;IF NE, YES, GO AGAIN
2771 033036 005702                        TST    R2                    ;SUPPRESS ZEROS
2772 033040 100006                        BPL    3$                    ;IF PL, YES, ALL DONE
2773 033042 005703                        TST    R3                    ;SUBSTITUTE BLANKS?
2774 033044 100002                        BPL    2$                    ;IF PL, NO
2775 033046 042702 000020                BIC    #20,R2               ;CONVERT FILL TO BLANK
2776 033052                                2$:
2777 033052 004767 177724                JSR    PC,1$                ;DIVIDE AGAIN
2778 033056                                3$:
2779 033056 112624                        MOVSB  (SP)+,(R4)+          ;STORE A DIGIT
2780 033060 010400                        MOV    R4,R0                ;STORE TERMINAL ADDRESS
2781 033062
2782

```

```

2784 .SBTTL DOUBLE PRECISION BINARY TO ASCII
2785 : CONVERT A DOUBLE PRECISION UNSIGNED QUANTITY TO DECIMAL ASCII
2786 :
2787 : LOCAL MACROS
2788 :
2789 : SET ASCII CONVERSION PARAMETERS
2790 :
2791 : CBTAS RADIX,WIDTH,SIGN,BLANK
2792 :
2793 : WHERE:
2794 :
2795 :     RADIX=CONVERSION RADIX
2796 :
2797 :     WIDTH=FIELD WIDTH
2798 :
2799 :     SIGN='SIGNED' FOR SIGNED CONVERSION. ANYTHING ELSE IMPLIES
2800 :           UNSIGNED CONVERSION
2801 :
2802 :     BLANK='BLANKS' TO CONVERT LEADING ZEROS TO BLANKS. ANYTHING ELSE
2803 :           IMPLIES NO CONVERSION OF ZEROS .
2804 :

```

```

2805 .MACRO CBTAS RADIX,WIDTH,SIGN,BLANK
2806 $BLKS=0
2807 $SGNS=0
2808 .IF IDN <BLANK>,<BLANKS>
2809 $BLKS=1*2000
2810 .ENDC
2811 .IF IDN <SIGN>,<SIGNED>
2812 $SGNS=1*400
2813 .ENDC
2814 MOV #<WIDTH*400>,$SGNS!$BLKS!RADIX,R5
2815 TST R2
2816 BEQ .+4
2817 BIS #1*1000,R5
2818 .ENDM

```

```

2819 :
2820 : INPUTS:
2821 :
2822 :     RC=POINTER TO ASCII OUTPUT STRING
2823 :     R1=ADDRESS OF DOUBLE PRECISION VALUE
2824 :     R2=ZERO COMPRESS FLAG
2825 :

```

2826					
2827	033064			CDDMG:	JSR R5,SAVRG ;SAVE THE NON-VOLATILE REGISTERS
2828	033064	004567	000250		MOV R0,R3 ;COPY THE STRING POINTER
2829	033070	010003			MOV #10000.,R4 ;SET DIVISOR
2830	033072	012704	023420		CBTAS 10.,0,NOSIGN,BLANKS ;SET CONVERSION PARAMETERS
2831	033076				CMP (R1)+,R4 ;TEST FOR OVERFLOW
2832	033112	022104			BHIS 40\$;IF HIS, OVERFLOW
2833	033114	103042			MOV (R1),R2 ;GET LOW PART OF NUMBER
2834	033116	011102			MOV -(R1),R1 ;GET HIGH PART OF NUMBER
2835	033120	014101			MOV R4,R0 ;COPY DIVISOR
2836	033122	010400			JSR PC,DDIV ;DO DOUBLE PREC. DIVIDE
2837	033124	004767	000110		MOV R0,-(SP) ;SAVE REMAINDER
2838	033130	010046			MOV R2,R1 ;COPY QUOTIENT
2839	033132	010201			

```

2840 033134 001011          BNE      11$          ;IF NE, SOMETHING TO PRINT
2841 033136 012702 000005    MOV      #5,R2        ;OTHERWISE, FILL FIELD WITH BLANKS
2842 033142 112723 000040    21$:    MOVB     #'',(R3)+
2843 033146 005302          DEC      R2
2844 033150 001374          BNE      21$
2845 033152 052705 003000    BIS     #3000,R5      ;DISABLE BLANK SUPPRESSION
2846 033156 000411          BR       20$
2847 033160 012702 024000    11$:    MOV      #5*4000,R2 ;SET FIELD WIDTH
2848 033164 004767 000020    JSR     PC,30$        ;OUTPUT HIGH ORDER DIGITS
2849 033170 052705 001000    BIS     #1*1000,R5    ;DISABLE ZERO COMPRESS
2850 033174 042705 002000    BIC     #1*2000,R5    ;DISABLE BLANKS
2851 033200 010003    31$:    MOV      R0,R3        ;SET STRING POINTER
2852 033202          20$:
2853 033202 012601          MOV      (SP)+,R1     ;GET LOW ORDER VALUE
2854 033204 012702 020000    MOV      #4*4000,R2  ;SET FIELD WIDTH
2855 033210          30$:
2856 033210 010300          MOV      R3,R0        ;GET STRING POINTER
2857 033212 050502          BIS     R5,R2        ;INCLUDE RADIX & BLANK SUPPRESS
2858 033214 004767 177506    JSR     PC,CBTA      ;CONVERT TO ASCII
2859 033220 000406          BR       60$        ;EXIT
2860 033222          40$:
2861 033222 012702 000011    MOV      #9.,R2      ;GET COUNT
2862 033226          50$:
2863 033226 112720 000052    MOVB     #'*,(R0)+   ;FILL FIELD WITH ASTERISKS
2864 033232 005302          DEC      R2
2865 033234 001374          BNE     50$          ; 'SOB R2,50$'
2866 033236          60$:
2867 033236          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
:
  
```

```

2886 033240          DDIV:
2887 033240 010346          MOV      R3,-(SP)    ;SAVE R3
2888 033242 012703 000040    MOV      #32.,R3    ;SET ITERATION COUNT IN R3
2889 033246 010046          MOV      R0,-(SP)    ;STACK DIVISOR
2890 033250 005000          CLR      R0         ;SET REMAINDER TO 0
2891 033252          1$:
2892 033252 006302          ASL     R2           ;SHIFT THE ENTIRE DIVIDEND
2893 033254 006101          ROL     R1           ;... ONE BIT TO THE LEFT AND...
2894 033256 006100          ROL     R0           ;... INTO THE REMAINDER
2895 033260 020016          CMP     R0,(SP)     ;IS REMAINDER GE DIVISOR?
  
```

2896 033262 103402
2897 033264 161600
2898 033266 005202
2899 033270
2900 033270 005303
2901 033272 003367
2902 033274 005726
2903 033276 012603
2904 033300
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921 033302
2922 033302 012746 000020
2923 033306 010146
2924 033310 005001
2925 033312
2926 033312 006300
2927 033314 006101
2928 033316 020116
2929 033320 103402
2930 033322 161601
2931 033324 005200
2932 033326
2933 033326 005366 000002
2934 033332 003367
2935 033334
2936 033334 022626
2937 033336
2938
2939
2940
2941
2942 033340 010446
2943 033342 010346
2944 033344 010546
2945 033346 016605 000006
2946 033352 004736
2947 033354 012603
2948 033356 012604
2949 033360 012605
2950 033362

```
BLO      2$          ;NO, SKIP TO ITERATION CONTROL
SUB      (SP),R0    ;YES, SUB DIVISOR OUT
INC      R2         ;AND INCR THE QUOTIENT

2$:      DEC      R3          ;REPEAT AS LONG AS NECESSARY
BGT      1$          ;PURGE DIVISOR FROM STACK
TST      (SP)+      ;RESTORE R3
MOV      (SP)+,R3
RETURN

.SBTTL  INTEGER DIVIDE MAGNITUDE NUMBERS

:
: INPUTS:
: RO=DIVIDEND
: R1=DIVISOR
:
: OUTPUTS:
: QUOTIENT IS RETURNED IN R0
: REMAINDER IS RETURNED IN R1
:
:
: DIV:
MOV      #20,-(SP)   ;SET LOOP COUNT
MOV      R1,-(SP)   ;SAVE DIVISOR FOR SUBTRACTS
CLR      R1         ;CLEAR REMAINDER

30$:    ASL      R0          ;DOUBLE LEFT SHIFT
ROL      R1
CMP      R1,(SP)    ;SUBTRACT OUT DIVISOR
BLO      40$        ;IF LO, NO
SUB      (SP),R1    ;SUBTRACT OUT DIVISOR
INC      R0         ;ADD IN LOW BIT

40$:    DEC      2(SP)    ;DECREMENT REPEAT COUNT
BGT      30$        ;IF GT, MORE TO GO

50$:    CMP      (SP)+,(SP)+ ;CLEAN STACK
RETURN

; SAVE/RESTORE NONVOLATILE REGISTERS

SAVRG:  MOV      R4,-(SP) ;SAVE R4 &R3
MOV      R3,-(SP)
MOV      R5,-(SP)
MOV      6(SP),R5      ;PUT RETURN ADDRESS ON STACK
JSR      PC,@(SP)+    ;RETRIEVE REAL R5
MOV      (SP)+,R3     ;CALL THE CALLER !!!
MOV      (SP)+,R4     ;RESTORE NON VOLATILE REGISTERS
MOV      (SP)+,R5
RETURN
```

```

2952          .SBTTL  QUEUE HANDLING ROUTINES
2953          : THIS MODULE CONTAINS 2 SUBROUTINES, ENQ & DEQ, TO ENQUEUE & DEQUEUE
2954          : WORDS, RESPECTIVELY, IN A FIRST-IN-FIRST-OUT LIST.
2955          :
2956          .LIST  MEB
2957
2958          PROC  ENQ,<ITEM,QUEUE>
2959 033364      ITEM  = 2
          (3)      000002
          (3)      000004
          (1)
          (1) 033364      ENQ:          ;**ENTRY POINT**
2960          :
2961          : APPEND ITEM (A WORD) TO THE FIRST-IN-FIRST-OUT LIST QUEUE .
2962          :
2963 033364 016504 000004      MOV  QUEUE(R5),R4          ;GET QUEUE ADDRESS.
2964 033370 021464 000002      CMP  @R4,QSIZE(R4)        ;IF QUEUE IS FULL,
2965 033374 002020      BGE  NQFULL          ; SIGNAL TRAGIC ERROR.
2966 033376 017574 000002 000012      MOV  @ITEM(R5),@QBACK(R4)      ;PUT ITEM AT BACK OF QUEUE.
2967 033404 062764 000002 000012      ADD  #2,QBACK(R4)          ;UPDATE BACK POINTER.
2968 033412 026464 000012 000006      CMP  QBACK(R4),QBOT(R4)
2969 033420 103403      BLO  NQNOWP
2970 033422 016464 000004 000012      MOV  QTOP(R4),QBACK(R4)
2971 033430 005214      NQNOWP: INC  @R4          ;INCREMENT NO. OF ELEMENTS.
2972 033432 000241      CLC
2973 033434 000207      NQRET: RETURN          ;INDICATE SUCCESSFUL ENQ.
          (1) 033434 000207      RTS  PC          ;RETURN.
2974
2975 033436 000261      NQFULL: SEC
2976 033440 000775      BR   NQRET          ;INDICATE UNSUCCESSFUL ENQ.
2977          ;IGNORE ITEM & RETURN.
2978
2979 033442      PROC  DEQ,<ITEM,QUEUE>
          (3)      000002
          (3)      000004
          (1)
          (1) 033442      DEQ:          ;**ENTRY POINT**
2980          :
2981          : REMOVE A WORD ENTRY FROM THE FIRST-IN-FIRST-OUT LIST QUEUE &
2982          : STORE IT AT ITEM .
2983          :
2984 033442 016504 000004      MOV  QUEUE(R5),R4          ;GET QUEUE ADDRESS.
2985 033446 005714      TST  @R4          ;IF QUEUE IS EMPTY,
2986 033450 001420      BEQ  DQEMP          ; SIGNAL TRAGIC ERROR.
2987 033452 017475 000010 000002      MOV  @QFRONT(R4),@ITEM(R5)      ;RETRIEVE FRONT ELEMENT.
2988 033460 062764 000002 000010      ADD  #2,QFRONT(R4)          ;UPDATE FRONT POINTER.
2989 033466 026464 000010 000006      CMP  QFRONT(R4),QBOT(R4)
2990 033474 103403      BLO  DQNOWP
2991 033476 016464 000004 000010      MOV  QTOP(R4),QFRONT(R4)
2992 033504 005314      DQNOWP: DEC  @R4          ;DECREMENT NO. OF ELEMENTS.
2993 033506 000241      DQRET: RETURN          ;INDICATE SUCCESSFUL DEQ.
          (1) 033510 000207      RTS  PC          ;RETURN.
2994
2995
2996 033512 000261      DQEMP: SEC
2997 033514 012775 177777 000002      MOV  #-1,@ITEM(R5)          ;INDICATE UNSUCCESSFUL DEQ.
          ;SET ITEM TO ALL ONES.

```

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 N 9
QUEUE HANDLING ROUTINES PAGE 44-1

SEQ 0117

2998 033522 000772

BR DQRET

;RETURN.

3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055

```
: SUBROUTINE TO SCAN AN INPUT COMMAND & CALL ITS  
: PROCESSING ROUTINE.  
      .LIST  MEB  
  
      .MACRO  SPAN  REG,CHAR,?L  
  
: THIS MACRO SCANS THE STRING OF CHARACTERS STARTING AT  
: @REG UNTIL IT FINDS ONE NOT EQUAL TO CHAR.  REG IS  
: SET POINTING TO THAT CHARACTER.  
L:    CMPB   (REG)+,CHAR  
      BEQ    L  
      DEC    REG  
  
      .ENDM  
  
      .MACRO  BREAK  REG,CHRSET,?HH,?JJ  
  
: THIS MACRO SCANS THE STRING STARTING AT @REG UNTIL  
: IT FINDS A CHARACTER THAT IS A MEMBER OF CHRSET.  
: REG IS SET POINTING TO THAT CHARACTER.  
  
: EACH MEMBER OF CHRSET IS AN ADDRESSABLE QUANTITY.  
  
HH:   .IRP   LS,<CHRSET>  
      CMPB  @REG,LS  
      BEQ   JJ  
      .ENDM  
      INC   REG  
      BR    HH  
  
JJ:   .ENDM  
  
      .MACRO  SYNCLS  CHAR,CLASS,?CC,?DD,?EE,?FF,?GG  
  
: THIS MACRO DETERMINES THE SYNTACTIC CLASS OF AN  
: OBJECT BEGINNING WITH CHAR.  THE CLASS IS RETURNED  
: IN CLASS AS FOLLOWS:  
: CLASS = 0 (WORD) IF CHAR = (A.....Z,-)  
:         2 (NUMBER) IF CHAR = (0.....9)  
:         6 (END OF LINE) IF CHAR = CARRIAGE RETURN  
:         4 (CHARACTER STRING) OTHERWISE  
  
      CMPB  CHAR,#'A  
      BLT  EE  
      CMPB  CHAR,#'Z  
      BGT  DD  
CC:     CLR  CLASS  
      BR   GG
```

3056	DD:	MOV	#4,CLASS
3057		BR	GG
3058	EE:	CMPB	CHAR,#'0
3059		BLT	FF
3060		CMPB	CHAR,#'9
3061		BGT	DD
3062		MOV	#2,CLASS
3063		BR	GG
3064	FF:	CMPB	CHAR,#'-
3065		BEQ	CC
3066		CMPB	CHAR,#15
3067		BNE	DD
3068		MOV	#6,CLASS
3069	GG:		
3070			
3071			
3072			
3073			.ENDM


```

3075          .SBTTL  COMMAND PROCESSOR INITIATING ROUTINE
3076
3077 033524    PROC    COMAND,<INPLIN,KWTABL>
(3)          000002    INPLIN =      2
(3)          000004    KWTABL =      4
(1)
(1) 033524    COMAND:          ;**ENTRY POINT**
3078          :
3079          : THIS ROUTINE CAUSES THE COMMAND SPECIFIED BY INPLIN TO BE
3080          : PROCESSED, AS DESCRIBED BELOW. INPLIN IS A STRING OF ASCII
3081          : CHARACTERS ENDED BY A CARRIAGE RETURN CODE.
3082          :
3083          : - INPLIN IS LEXICALLY SCANNED USING THE LXSCAN ROUTINE.
3084          : - THE 1ST OBJECT IS CONVERTED TO A PROCESSING ROUTINE ADDRESS
3085          :   USING THE KEYWD ROUTINE & THE KEYWORD TABLE KWTABL
3086          :   SUPPLIED BY THE CALLING PROGRAM.
3087          : - THE PROCESSING ROUTINE IS CALLED WITH THE OUTPUT FROM LXSCAN
3088          :   ON THE STACK STARTING AT 2(SP). (THE RETURN ADDRESS OF THIS
3089          :   CALL OCCUPIES THE TOP WORD OF THE STACK.)
3090          : - THE LXSCAN OUTPUT IS REMOVED FROM THE STACK.
3091          :
3092          : IF LXSCAN OR KEYWD OR THE PROCESSING ROUTINE RETURN AN ERROR
3093          : CONDITION, THEN C = 1; OTHERWISE, C = 0.
3094          :
3095 033524 010546    MOV      R5,-(SP)          ;SAVE PAR LIST POINTER.
3096 033526 010667 000076    MOV      SP,CMMARK          ;SAVE STACK POINTER.
3097 033532          LXSCAN  INPLIN(R5)          ;LEXICALLY SCAN INPLIN .
(1) 033532 016501 000002    MOV      INPLIN(R5),R1
(1) 033536 004167 000310    JSR     R1,LXSCAN
3098 033542 103426          BCS     CMRET          ;IF ERROR, RETURN WITH C=1.
3099 033544 005716          TST     @SP            ;HAVE WE ANY OBJECTS ?
3100 033546 003005          BGT     1$            ;NO, IGNORE BLANK COMMAND.
3101 033550 012767 177777 164210    MOV     #-1,ESCFLG
3102 033556 000167 000036    JMP     CMRET
3103 033562 016700 000042    1$:    MOV     CMMARK,R0          ;YES, DETERMINE ADDRESS
3104 033566 162700 000006    SUB     #6,R0          ;OF FIRST OBJECT.
3105 033572 017705 000032    MOV     @CMMARK,R5     ;RESTORE PAR LIST POINTER.
3106 033576          KEYWD  R0,KWTABL(R5)          ;GET ADDR OF COMMAND PROCESSOR.
(1) 033576 010046    MOV     R0,-(SP)
(1) 033600 016546 000004    MOV     KWTABL(R5),-(SP)
(1) 033604 004767 000024    JSR     PC,KEYWD
3107 033610 103403          BCS     CMRET          ;IF INVALID COMMAND, MAKE
3108          : ERROR RETURN.
3109 033612 012705 033632    MOV     #NULPAR,R5     ;LOAD NULL PAR LIST ADDRESS.
3110 033616 004736          JSR     PC,@(SP)+      ;PROCESS COMMAND & POP ADDR.
3111 033620 016706 000004    CMRET: MOV     CMMARK,SP ;RESTORE STACK TO ENTRY STATUS.
3112 033624 030026          BIT     R0,(SP)+      ;THROW AWAY SAVED R5 (WITHOUT
3113          : AFFECTING C BIT).
3114 033626          RETURN
(1) 033626 000207    RTS     PC            ;RETURN TO CALLING PROGRAM.
3115
3116          : DATA AREAS.
3117 033630 033333    CMMARK: .WORD 33333    ;STORAGE FOR STACK PTR ON ENTRY.
3118 033632 000000    NULPAR: .WORD 0       ;PAR LIST CONTAINING NO PARS.

```

```

3120          .SBTTL  KEYWORD PROCESSING ROUTINE
3121
3122          : SUBROUTINE TO DETERMINE THE ADDRESS OF THE PROCESSING
3123          : ROUTINE ASSOCIATED WITH THE KEYWORD REPRESENTED BY
3124          : THE SYNTACTIC OBJECT POINTED TO BY THE ARGUMENT
3125          : SRC.  CONVERSION FROM KEYWORD TO ROUTINE ADDRESS
3126          : IS DONE AS DEFINED IN THE KEYWORD TABLE
3127          : POINTED TO BY THE ARGUMENT TABLAD.
3128
3129          : ON ENTRY, THE TOP OF THE STACK IS AS FOLLOWS:
3130          : (SP):  RETURN ADDRESS
3131          : 2(SP):  TABLAD
3132          : 4(SP):  SRC
3133
3134          : CALLING INSTRUCTION:          JSR    PC,KEYWD
3135
3136          : ON RETURN, THE TOP OF THE STACK IS AS FOLLOWS:
3137          : (SP):  ROUTINE ADDRESS, IF KEYWORD IN TABLE; 0 IF NOT.
3138
3139          : IF THE KEYWORD IS IN THE TABLE, C=0 ON RETURN.  IF NOT, C=1.
3140
3141          : STACK POINTER OFFSETS
3142          :ROUTAD =          0          ;ROUTINE ADDR FOR
3143          ;          ;          ;CURRENT TABLE
3144          ;          ;          ;ELEMENT.
3145          ;          ;          ;ADDR OF INPUT WORD
3146          ;          ;          ;#CHAR IN INPUT WORD
3147          ;          ;          ;SUBROUTINE RETURN ADDR
3148          ;          ;          ;ADDR OF KEYWORD TABLE
3149          ;          ;          ;ADDR OF INPUT OBJECT
3150          ;          ;          ;RESULT RETURNED
3151
3152          033634          ADRINP =          2          ;**ENTRY POINT**
3153          033634          LENINP =          4          ;SAVE REGISTERS.
3154          (1) 033634 004567 000160          JSR    R5,REGSAV
3155          033640 162706 000006          SUB    #6,SP          ;ALLOCATE STACK SPACE
3156          033644 016602 000026          MOV    SRC(SP),R2      ;GET ADDR OF INPUT
3157          033650 005722          TST    (R2)+          ;OBJECT
3158          033652 001051          BNE   NOTHER          ;IF OBJECT NOT A WORD
3159          033654 012205          MOV    (R2)+,R5       ;THEN EXIT: NOT FOUND
3160          033656 010566 000004          MOV    R5,LENINP(SP) ;GET # CHAR IN OBJECT.
3161          033662 011204          MOV    @R2,R4        ;STORE ON STACK.
3162          033664 010466 000002          MOV    R4,ADRINP(SP) ;GET ADDR OF INPUT WORD.
3163          033670 016603 000024          MOV    TABLAD(SP),R3 ;STORE ON STACK.
3164          ;          ;          ;GET ADDR. OF KEYWORD
3165          033674 012316          GTLENS: MOV    (R3)+,@SP ;GET ADDR. OF KEYWORD
3166          ;          ;          ;TABLE.
3167          033676 112302          MOVB  (R3)+,R2       ;SAVE ROUTINE ADDR OF
3168          033700 042702 177400          BIC   #177400,R2     ;1ST ELEMENT.
3169          033704 112301          MOVB  (R3)+,R1       ;GET MINIMUM LENGTH
3170          033706 042701 177400          BIC   #177400,R1     ;OF TABLE WORD.
3171          033712 001431          BEQ   NOTHER          ;GET FULL LENGTH.
3172          ;          ;          ;
3173          033714 122423          NXTCH: CMPB  (R4)+,(R3)+ ;IF 0, THEN NO MORE
3174          ;          ;          ;TABLE TO SEARCH
3175          ;          ;          ;COMPARE INPUT CHAR
3176          ;          ;          ;WITH TABLE CHAR.

```

3175	033716	002427		BLT	NOTHER						:IF<, THEN NO MATCH
3176											: EXISTS.
3177	033720	003005		BGT	NXTWD						:IF>, THEN TRY NEXT
3178											: TABLE WORD.
3179	033722	005305		DEC	R5						:IF INPUT STRING
3180											: EXHAUSTED,
3181	033724	001413		BEQ	THFND						: WE MAY HAVE FOUND
3182											: MATCH.
3183	033726	005301		DEC	R1						:IF MORE CHAR IN TABLE WORD
3184	033730	001371		BNE	NXTCH						: TO TEST, GO & TEST THEM.
3185	033732	005201		INC	R1						
3186	033734	060103		NXTWD:	ADD	R1,R3					:GET ADDR OF NEXT
3187	033736	042703	000001		BIC	#1,R3					: TABLE ENTRY.
3188	033742	016604	000002		MOV	ADRINP(SP),R4					:POINT TO BEGINNING
3189											: OF INPUT WORD.
3190	033746	016605	000004		MOV	LENINP(SP),R5					:GET LENGTH OF INPUT WORD.
3191	033752	000750			BR	GTLENS					:GET LENGTHS OF TABLE WORD.
3192											
3193	033754	026602	000004	THFND:	CMP	LENINP(SP),R2					:IF LEN(INP.WD) < MIN LEN (TABLE
3194	033760	002406			BLT	NOTHER					: WORD), WORD IS NOT IN TABLE.
3195	033762	011666	000026		MOV	(SP),RESULT(SP)					:SAVE ROUTINE ADDR. OF
3196											: MATCH.
3197	033766	062706	000006		ADD	#6,SP					:FREE LOCAL STACK SPACE.
3198	033772	000241			CLC						:CLEAR CARRY BIT.
3199	033774	000405			BR	KWEXIT					
3200											
3201											: WORD IS NOT IN TABLE. SET RESULT TO 0 & SET Z BIT ON.
3202	033776	005066	000026	NOTHER:	CLR	RESULT(SP)					:CLEAR RESULT.
3203	034002	062706	000006		ADD	#6,SP					:FREE LOCAL STACK SPACE.
3204	034006	000261			SEC						:SET CARRY BIT.
3205	034010			KWEXIT:	REGRES						:RESTORE REGISTERS.
(1)	034010	004567	000020		JSR	R5,REGRES					
3206	034014	012616			MOV	(SP)+,@SP					:POP AN ARGUMENT.
3207	034016	000207			RTS	PC					:RETURN.

```
3209          .SBTTL REGISTER SAVE & RESTORE ROUTINES
3210
3211          : SUBROUTINE TO SAVE R0 - R5 ON STACK
3212          : CALLING SEQUENCE: JSR R5,REGSAV          ;**ENTRY POINT**
3213          REGSAV:
3214          MOV R4,-(SP)
3215          MOV R3,-(SP)
3216          MOV R2,-(SP)
3217          MOV R1,-(SP)
3218          MOV R0,-(SP)
3219          JMP @R5
3220
3221          : SUBROUTINE TO RESTORE R0-R5 FROM STACK
3222          : THE CONDITION CODE BITS IN THE PS ARE DESTROYED,
3223          : EXCEPT FOR THE CARRY BIT, WHICH IS PRESERVED.
3224
3225          : CALLING SEQUENCE: JSR R5,REGRES
3226          REGRES:
3227          BIT R0,(SP)+          ;**ENTRY POINT**
3228          MOV (SP)+,R0          ;THROW AWAY OLD R5 VALUE.
3229          MOV (SP)+,R1
3230          MOV (SP)+,R2
3231          MOV (SP)+,R3
3232          MOV (SP)+,R4
3233          RTS R5
3234
```

3213	034020	
3214	034020	010446
3215	034022	010346
3216	034024	010246
3217	034026	010146
3218	034030	010046
3219	034032	000115
3227	034034	
3228	034034	030025
3229	034036	012600
3230	034040	012601
3231	034042	012602
3232	034044	012603
3233	034046	012604
3234	034050	000205

.SBTTL LEXICAL SCAN ROUTINE

3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291

: PERFORM LEXICAL SCAN OF INPUT COMMAND IN
: BUFFER.

: THREE CLASSES OF SYNTACTIC OBJECTS ARE RECOGNIZED:
: 1. WORD: A STRING OF CHARACTERS BEGINNING WITH
: A LETTER & TERMINATED WITH A
: BLANK OR CARRIAGE RETURN.
: 2. NUMBER: A STRING OF OCTETS TERMINATED WITH A
: BLANK OR CARRIAGE RETURN, OR A STRING OF DIGITS
: TERMINATED WITH A DOT.
: 3. CHARACTER STRING: A STRING SURROUNDED BY 2 INSTANCES
: (1 ON EACH END) OF A SPECIAL CHARACTER.

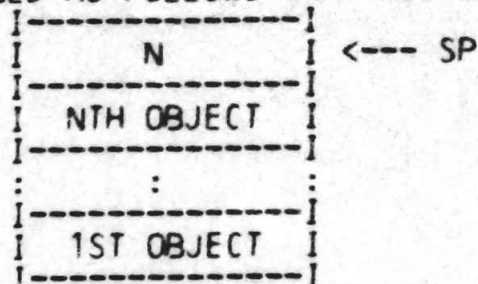
: SYNTACTIC OBJECTS ARE SEPARATED BY 1 OR MORE BLANKS.
: THE COMMAND IS ENDED BY A CARRIAGE RETURN.

: THIS LEXICAL SCANNER DETERMINES THE LOCATIONS OF THE
: SYNTACTIC OBJECTS & DETERMINES THEIR CLASSES.
: NUMBERS ARE CONVERTED TO THEIR BINARY VALUES.

: AFTER THE LEXICAL SCAN IS PERFORMED, EACH
: SYNTACTIC OBJECT WILL BE REPRESENTED ON THE STACK
: AS A 3 WORD QUANTITY AS FOLLOWS:

TOP WORD	=	SYNTACTIC CLASS: 0==>WORD 2==>NUMBER 4==>CHARACTER STRING
2ND WORD	=	LENGTH OF OBJECT IN CHARACTERS; NOT SIGNIFICANT FOR NUMBERS; DOES NOT INCLUDE SURROUNDING DELIMITERS FOR CHAR. STRINGS.
3RD WORD	=	VALUE OF NUMBER, OR POINTER TO 1ST LETTER OF KEYWORD OR 1ST SIGNIFICANT CHARACTER OF STRING (IE: NOT SUR- ROUNDING DELIMITER)

: AT THE END OF THE LEXICAL SCAN THE STACK WILL BE
: ARRANGED AS FOLLOWS (N = NO. OF SYNTACTIC OBJECTS):



: WHERE EACH OBJECT IS REPRESENTED AS ABOVE.

: CALLING SEQUENCE: R1=BUFFER ADDRESS
: JSR R1,LXSCAN
: ON RETURN, REGISTERS 0-5 ARE UNDEFINED.
: EXCEPT FOR THE ABOVE TABLE, THE STACK IS AS IT
: WAS BEFORE ENTRY TO THE ROUTINE.

```

3292
3293
3294
3295
3296
3297
3298 034052
3299 034052 005002
3300 034054 012605
3301 034056
  (1) 034056 122527 000040
  (1) 034062 001775
  (1) 034064 005305
3302 034066
  (1) 034066 121527 000101
  (1) 034072 002410
  (1) 034074 121527 000132
  (1) 034100 003002
  (1) 034102 005004
  (1) 034104 000424
  (1) 034106 012704 000004
  (1) 034112 000421
  (1) 034114 121527 000060
  (1) 034120 002406
  (1) 034122 121527 000071
  (1) 034126 003367
  (1) 034130 012704 000002
  (1) 034134 000410
  (1) 034136 121527 000055
  (1) 034142 001757
  (1) 034144 121527 000015
  (1) 034150 001356
  (1) 034152 012704 000006
3303 034156 000174 034162
3304
3305 034162 034172
3306 034164 034232
3307 034166 034376
3308 034170 034436
3309
3310 034172 010503
3311 034174
  (2) 034174 121527 000040
  (2) 034200 001405
  (2) 034202 121527 000015
  (2) 034206 001402
  (1) 034210 005205
  (1) 034212 000770
3312 034214 010346
3313 034216 160503
3314 034220 005403
3315 034222 010346
3316 034224 010446
3317
3318 034226 005202
3319 034230 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
NXTEL: SPAN R5,<#>
64$:   CMPB (R5)+,#'
        BEQ 64$
        DEC R5
        SYNCLS @R5,R4
        CMPB @R5,#'A
        BLT 67$
        CMPB @R5,#'Z
65$:   BGT 66$
        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
        BNF 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 NXTEL
  
```

```

:**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
  
```

3320									
3321	034232	005000		SCNO:	CLR	R0			:CLEAR ACCUMULATED NO.
3322	034234	010504			MOV	R5,R4			:SAVE POINTER TO 1ST DIGIT.
3323	034236	121527	000060	SNXTDG:	CMPB	@R5,#'0			:IF CHAR <'0'
3324	034242	002415			BLT	SNTDIG			: THEN TREAT AS DELIMITER.
3325	034244	121527	000071		CMPB	@R5,#'9			:IF CHAR >'9'
3326	034250	003012			BGT	SNTDIG			: THEN TREAT AS DELIMITER.
3327	034252				MULT	10.,R0,R3			:MULTIPLY PREVIOUS DIGITS BY 10.
(2)	034252	006300			ASL	R0			
(2)	034254	010003			MOV	R0,R3			
(2)	034256	006300			ASL	R0			
(2)	034260	006300			ASL	R0			
(2)	034262	060300			ADD	R3,R0			
3328	034264	112503			MOVB	(R5)+,R3			
3329	034266	142703	000060		BICB	#60,R3			:CLEAR TOP BITS OF ASCII CODE.
3330	034272	060300			ADD	R3,R0			:ADD DIGIT.
3331	034274	000760			BR	SNXTDG			:GET NEXT DIGIT.
3332									
3333	034276	121527	000040	SNTDIG:	CMPB	@R5,#'			:IF DELIMITER = SPACE, TRY
3334	034302	001413			BEQ	TRYOCT			: CONVERTING OCTAL NO.
3335	034304	121527	000015		CMPB	@R5,#15			:IF DELIMITER = CR, TRY
3336	034310	001410			BEQ	TRYOCT			: CONVERTING OCTAL NO.
3337	034312	122527	000056		CMPB	(R5)+,#'			:IF DELIMITER IS NOT DOT,
3338	034316	001052			BNE	LXERR			: THEN SIGNAL LXSCAN ERROR.
3339	034320	010046		PUSHNO:	MOV	R0,-(SP)			:PUT CONVERTED NO. ON STACK.
3340	034322	005046			CLR	-(SP)			:SET OBJECT LENGTH TO ZERO.
3341	034324	012746	000002		MOV	#2,-(SP)			:SET OBJECT CLASS TO 2
3342									: (NUMBER).
3343	034330	000736			BR	LXINCN			:INCREMENT OBJ. COUNT &
3344									: SCAN NEXT OBJ.
3345									
3346	034332	005000		TRYOCT:	CLR	R0			:CLEAR ACCUMULATED NO.
3347	034334	121427	000060	ONXTDG:	CMPB	@R4,#'0			:IF CHAR <'0'
3348	034340	002413			BLT	ODELIM			: THEN TREAT AS DELIMITER.
3349	034342	121427	000067		CMPB	@R4,#'7			:IF CHAR >'7'
3350	034346	003010			BGT	ODELIM			: THEN TREAT AS DELIMITER.
3351	034350				MULT	8.,R0			:MULTIPLY PREVIOUS DIGITS BY 8.
(2)	034350	006300			ASL	R0			
(2)	034352	006300			ASL	R0			
(2)	034354	006300			ASL	R0			
3352	034356	112403			MOVB	(R4)+,R3			:GET CHARACTER.
3353	034360	142703	000060		BICB	#60,R3			:CLEAR TOP BITS OF ASCII CODE.
3354	034364	060300			ADD	R3,R0			:ADD DIGIT.
3355	034366	000762			BR	ONXTDG			:GET NEXT DIGIT.
3356									
3357	034370	020405		ODELIM:	CMP	R4,R5			:IF NOT AT END OF NO.(DUE TO
3358	034372	001024			BNE	LXERR			: '8' OR '9'), LXSCAN ERROR.
3359	034374	000751			BR	PUSHNO			:GENERATE OBJECT FOR NUMBER.
3360									
3361	034376	112500		SCCHAR:	MOVB	(R5)+,R0			:R0 = DELIMITER OF STRING.
3362	034400	010503			MOV	R5,R3			:R3 = ADDR (1ST CHAR OF
3363									: STRING ITSELF)
3364	034402				BREAK	R5,<R0,#15>			:R5 = ADDR (NEXT DELIM OR CR).
(2)	034402	121500			CMPB	@R5,R0			
(2)	034404	001405			BEQ	65\$			
(2)	034406	121527	000015		CMPB	@R5,#15			

```
(2) 034412 001402      BEQ    65$
(1) 034414 005205      INC    R5
(1) 034416 000771      BR     64$
3365 034420 121527 000015  CMPB  @R5,#15      ;WAS CR FOUND BEFORE DELIMITER?
3366 034424 001407      BEQ    LXERR        ;YES, PROCESS ERROR
3367 034426 010346      MOV    R3,-(SP)    ;PUSH ADDR. OF STRING ONTO
3368                                ; STACK.
3369 034430 160503      SUB    R5,R3      ;R3 = (-LENGTH OF STRING EXCL.
3370                                ; DELIMITER)
3371 034432 005205      INC    R5          ;R5 = ADDR (CHAR AFTER CLOSING
3372                                ; DELIMITER).
3373 034434 000671      BR     PLAC        ;FINISH OFF PROCESSING STRING.
3374
3375 034436 010246      SCEOL: MOV    R2,-(SP) ;PUSH NO. OF OBJECTS ONTO STACK
3376 034440 000241      CLC                                ;INDICATE NO ERROR
3377 034442 000111      LXIT: JMP    @R1    ;RETURN TO CALLING PROGRAMME
3378
3379                                ; PROCESS LEXICAL SCAN ERROR .
3380 034444 000261      LXERR: SEC                                ;INDICATE ERROR OCCURRED .
3381 034446 005302      DEC    R2          ;CLEAN UP STACK & EXIT .
3382 034450 002774      BLT    LXIT        ; ..
3383 034452 062706 000006  ADD    #6,SP      ; ..
3384 034456 000772      BR     LXERR      ; ..
3385
3386
3387                                .END
```


CPALP	004540	768#	783				
CPARTN	004622	785#	787				
CPASS	005672	1020#	2478				
CPCLR	004752	831#	2481				
CPCLRC	005002	840#	847				
CPCLRT	005044	851#	853				
CPCNRT	005200	885	887#	889			
CPCNT	005114	878#	2484				
CPDEL	004632	791#	2487				
CPDLP	004646	796#	816				
CPDRTN	004742	818#	820				
CPERR	006600	1174#	2490				
CPERTN	006716	1185	1191#	1194			
CPGO	005510	979#	2493				
CPINIT	005210	902#	2496				
CPINRT	005446	944#	946				
CPMAST	004116	646#	2499				
CPMLP	003506	580#	596				
CPMOK	004132	648	651#				
CPMRET	004200	650	660#				
CPRANG	004366	720#	2502				
CPRFIL	004470	744#	748				
CPRIIB	004274	691#	2505				
CPROK	004310	693	696#				
CPRRET	004364	695	708#				
CPRRTN	004514	750#	753				
CPSEC	004202	666#	2508				
CPSEX	004064	630#	633	639			
CPSILO	003406	562#	2511				
CPSLV	004022	618	624#				
CPSOK	004216	668	671#				
CPSRET	004266	670	681#	685			
CPSRTN	005104	865#	867				
CPSTAT	005054	859#	2514				
CPSUM	005456	953#	2517				
CPSURT	005500	958#	960				
CRRET	015074	2078	2083#				
CR. =	000015	335#	2079	2271			
CTL CMG	016422	2125	2335#				
CTL OMG	016414	2093	2333#				
CTL UMG	016417	2134	2334#				
CTL.C =	000003	338#	2269				
CTL.O =	000017	336#	2272				
CTL.Q =	000021	339#	2273				
CTL.S =	000023	340#	2274				
CTL.U =	000025	337#	2275				
CURAD	017762	1346*	1347	1474	1499	2399#	
CYCL	007350	1299	1301#				
DATBUF	020676	1269	1344	2432#			
DATGEN	007124	503	1255#				
DATGEV	017670	501	943*	1282*	1364*	2366#	
DBCLP	011076	1557#	1559				
DBF CLR	011062	1544	1554#				
DDIV	033240	2837	2886#				
DECJSP	032700	1850	1901	1953	2718#		
DECPNT	032706	1781	1785	1789	1793	2724#	

CZPLACO PCL11 EXERCISER V02C
CZPLAC.P11 08-JUN-79 15:55

MACY11 30A(1052) 25-JUN-79 08:52 PAGE 51
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0138

BDINIT	231#	907	1340	1450	1459	1484	1510	1545	1630	1678	1688				
BREAK	3018#	3311	3364												
CALL	131#	444	445	446	447	448	449	471	472	481	491	494	497	500	503
	506	509	512	515	518	521	524	527	534	619	620	621	622	626	882
	883	913	992	993	994	1110	1112	1113	1116	1135	1136	1137	1142	1155	1156
	1157	1183	1229	1235	1236	1237	1345	1397	1409	1413	1421	1425	1429	1431	1435
	1445	1449	1471	1490	1514	1584	1593	1597	1605	1610	1612	1616	1620	1667	1674
	1683	1761	1767	1770	1773	1775	1795	1796	1804	1824	1825	1836	1837	1851	1878
	1879	1886	1887	1889	1902	1954	1958	1966	2052	2064	2067	2082	2092	2093	2125
	2134	2156	2157	2158	2159	2175	2176	2179	2203	2218	2230	2247	2259	2260	2261
CBTAS	2805#	2831													
ERROR	262#	1584	1593	1597	1605	1610	1612	1616	1620	1667	1674	1683			
ERROT	250#	1397	1409	1413	1421	1425	1429	1431	1435	1445	1449	1471	1490	1514	
HEDING	202#														
KEYWD	46#	653	673	698	3106										
LXSCAN	71#	3097													
MULP	219#	1276	1670												
MULT	153#	575	1995	1996	1998	2034	2035	2037	3327	3351					
PROC	87#	562	646	666	691	720	763	791	831	837	859	878	902	906	914
	953	979	1020	1107	1134	1154	1174	1209	1255	1336	1542	1726	1817	1871	1924
	1974	2012	2050	2076	2089	2105	2115	2124	2133	2146	2170	2189	2212	2243	2258
	2959	2979	3077												
REGRES	282#	539	1224	1398	1452	1460	1485	1511	1526	1560	1586	1628	1641	1679	1690
	2231	3205													
REGSAV	276#	533	1210	1378	1554	1570	2228	3153							
RETURN	112#	630	660	681	708	750	785	818	851	865	887	944	958	998	1098
	1117	1143	1159	1191	1225	1283	1358	1550	1805	1860	1911	1967	2006	2045	2068
	2083	2094	2107	2117	2162	2180	2204	2220	2262	2781	2867	2904	2937	2950	2973
	2994	3114													
SPAN	3006#	3301													
STCVT	2659#	2707	2713	2719	2725										
SYNCLS	3039#	3302													

. ABS. 034460 000

ERRORS DETECTED: 0

CZPLAC,CZPLAC/CR=CZPLAC
RUN-TIME: 31 43 3 SECONDS
RUN-TIME RATIO: 462/79=5.8
CORE USED: 10K (19 PAGES)