TI-88 ALEX PROCEDURE

FUNCTIONS: CASSETTE INTERFACE AND GENERAL I/O COMMUNICATION

PRIMARY KEYS: CUE RESPONSES, SWAP

OP CODES: OPS ~~47-52~~ 56-63, 82-83

TEST PROCEDURE:

1. CASSETTE OPS ~~47-52~~ 56-63 IN MANUAL MODE:
   TEST EACH OF THE OP CODES FOR CORRECT OPERATION.
   - 5/81 A. ENTER VALID AND INVALID DATA FOR STARTING AND ENDING POINTS.
   - 5/81 B. CHECK DATA TRANSMITTED OVER I/O LINES FOR WRITES.
   - 5/81 C. MAKE SURE DATA IS STORED PROPERLY ON READS.
   - D. MEASURE DATA RATE FOR EACH OP CODE.
   - 5/81 E. CHECK THAT CUES WORK PROPERLY WITH VALID OR INVALID RESPONSES.

2. CASSETTE OPS ~~47-52~~ 56-63 IN PROGRAM RUN MODE
   - A. CHECK THAT EACH OP CODE OPERATES CORRECTLY IN RUN MODE AND SINGLE STEP PROGRAM MODE.
     1) INPUTS ARE ENTERED PRIOR TO EXECUTION OF THE OP CODE AND THE OP CODE DOES NOT PAUSE FOR INPUTS THROUGH CUE RESPONSES.
     2) CHECK ITEMS A, B, AND C FROM NUMBER 1.
   - B. CHECK TO BE SURE THAT BOTH THE CASSETTE OP CODES AND THE PROGRAM TERMINATE PROPERLY.

3. I/O COMMANDS IN MANUAL MODE
   - 5/81 A. CHECK OP ~~87~~ 82 TO BE SURE THAT 16 BITS FROM THE SWAP REGISTER AND 64 BITS FROM THE DISPALY REGISTER IS TRANSFERRED TO THE I/O LINE.
   - B. CHECK TIMING OF START PULSE AND ONE AND ZERO WAVEFORMS.
   - C. CHECK COMMUNICATION AT VARYING FREQUENCIES OF OUTPUT AND RECEIVING PROCESSORS TO INSURE OPERATION THROUGHOUT PRODUCTION VARIATION.
   - 5/81 D. CHECK OP ~~88~~ 83 TO BE SURE THAT 16 BITS ARE STORED IN THE SWAP REGISTER AND 64 BITS IN THE DISPLAY REGISTER.
   - 5/81 E. CHECK TO SEE THAT AN IDLE CALCULATOR WILL DO AN OP ~~88~~ 83 COMMAND AUTOMATICALLY IF STROBED AT THE PROPER TIME OVER THE I/O LINE.
   - 5/81 F. CHECK THE KEYPUSH COMMAND WHICH CAN BE SENT IN OVER THE I/O LINE TO CAUSE THE CALCULATOR TO EXECUTE A KEY FUNCTION.
   - G. CHECK THAT AN OP ~~88~~ 83 COMMAND IS ATTEMPTED 100 TIMES IF THE I/O LINE IS BUSY AND THEN ABORTED IF THE LINE IS STILL BUSY.

4. I/O COMMANDS IN RUN OR SINGLE STEP PROGRAM MODES.
   CHECK ITEMS A, D, AND G FROM NUMBER 3.

PROBLEMS: THE CASSETTE INTERFACE ALGORITHM IS NOT AVAILBLE FOR THIS TEST. A HARDWARE COMMUNICATION BOX WILL BE USED TO TEST INPUT AND OUTPUT OF I/O DATA. ALTHOUGH SOME TESTING IS POSSIBLE TO VERIFY THE DATA, ANY TIMING PROBLEMS BETWEEN THE CALCULATOR AND CASSETTE INTERFACE WILL NOT NECESSARILY BE FOUND WITH THESE TESTS.

ACH 07/08/80

FUNCTION : MEMORY EXPANSION

PRIMARY KEYS : ON, READ, INV READ

OP CODES : OP 3, OP 41, OPS 46 - 56, OPS 72 - 74

TEST PROCEDURE :

1. BUILD A MEMORY EXPANSION SIMULATOR TO HOOK UP TO THE CURRENT CRAM/CROM SIMULATOR -- KEN LIES

2. USING DIFFERENT COMBINATIONS OF CRAMS AND CROMS IN VARIOUS SLOTS OF THE EXPANSION MEMORY, TURN THE CALCULATOR ON AND OFF AND MAKE SURE THE POWER UP SEQUENCE SEES ALL CHIPS. CHECK THE CHIP INFORMATION IN THE RAM OF THE CD2902 AND RUN OP 3. STORE A UNIQUE NUMBER IN THE FIRST REGISTER OF EACH CRAM TO CHECK POLLING ORDER.

3. TEST OP 3 WITH DIFFERENT AMOUNTS OF TOTAL STEPS. TRY REQUESTING MORE THAN 10000 STEPS, 0 STEPS, AND OTHER PARTITIONS.

4. STORE UNIQUE VALUES IN REGISTERS IN ALL CHIPS TO MAKE SURE EXPANSION MEMORY ADDRESSING WORKS.

5. TRY UPLOAD, DOWNLOAD, NAMING A CRAM, UNNAMING A CRAM, RUNNING A PROGRAM IN A CROM, FINDING THE NAME OF THE CROM IN THE MASTER SLOT, STORING STEPS AND RECALLING STEPS, READING AND WRITING CASSETTES, SST AND RUN MODE IN VARIOUS SLOTS OF THE EXPANDED MEMORY.

PROBLEMS : THERE ARE NOT ENOUGH CRAM/CROM SIMULATORS TO CHECK ALL 16 SLOTS OF EXPANSION MEMORY AT ONCE. THERE ARE NO CASSETTE CAPABILITIES.

LINDA FERRIO 7/11/80

OBSOLETE

FUNCTION : EMULATOR -- A ONE CHIP 16 K NAMED CRAM THAT COMES WITH A PREVIOUSLY MADE CATALOG FOR OUTSIDE USERS TO DEVELOP CUSTOM CROMS ON.

PRIMARY KEYS : INV READ (UPLOAD), READ

OP CODES : OP 72, OP 73, OP 74, OP 75

TEST PROCEDURE :

1. CHANGE HARDWARE TO MAKE A CROM LOOK LIKE A 15 K CRAM TO ALLOW UPLOADING PROGRAMS -- MARK JANDER

3/81 2. UPLOAD PROGRAMS OF VARIOUS SIZES - 8 STEPS, 300 STEPS, 600 STEPS, 1000 STEPS, 2000 STEPS. CHECK THE 600 AND 1200 STEP BOUNDARIES IN THE EMULATOR CAREFULLY. MAKE SURE THE CATALOG ALWAYS KNOWS THERE IS ONLY 1 CHIP.

3/81 3. UPLOAD PROGRAMS NUMBERED BETWEEN 1 AND 99 AND TRY TO FILL UP 16 K TO MAKE SURE IT WILL GO THAT HIGH AND WON'T GO ANY FARTHER.

3/81 4. EXECUTE THE PROGRAMS TO MAKE SURE THEY WROTE CORRECTLY.

5. USE OP 73 TO RENAME THE EMULATOR - SHOULD EXECUTE
3/81 SUCCESSFULLY. USE OP 72 TO ERASE THE EMULATOR - SHOULD ERASE THE FIRST 600 STEPS ONLY AND ADD 148 REGISTERS TO MAIN MEMORY. OP 73 TO NUMBER THE MODULE WILL SET UP THE CATALOG FOR 2 CHIPS AND 10 PROGRAMS ONLY AND SHOULD NOT BE USED.

3/81 6. DOWNLOAD A PROGRAM, EDIT IT, AND UPLOAD INTO THE SAME PROGRAM NUMBER. SHOULD NOT CHANGE ANY OTHER PROGRAMS OR THE CATALOG.

5/81 7. WRITE THE CONTENTS OF THE EMULATOR ONTO A CASSETTE AND READ IT ONTO THE 990 - NOT POSSIBLE

8. USE OP 74 TO COPY THE CROM. ONLY THE FIRST 600 STEPS OF THE EMULATOR SHOULD BE COPIED. - CHANGED

PROBLEMS : THE CROM BEING ALTERED IS ONLY 15 K AND THE ACTUAL - OK NOW
EMULATOR IS 16 K. THERE IS NO CASSETTE TO TEST
WRITING THE EMULATOR AND THE CASSETTE GENERATED
COULD NOT BE READ DIRECTLY ONTO THE 990 EVEN IF WE
COULD MAKE A CASSETTE. MADE CASSETTE, BUT STILL CANT
READ ON 990

LINDA FERRIO 7/11/80

PROCEDURE RESULTS ----- 8/08/80

1. THE HARDWARE WAS CHANGED TO MAKE A CROM LOOK LIKE A 16 K CRAM (SO THE FIRST PROBLEM WAS SOLVED).

2 & 3. LBL A 12345 RTN (LAST STEP = 7)
LBL B 1+1+1+1+ ..... +1= RTN (LAST STEP = 299)
LBL C A+1+1+1+ ..... +1= RTN (LAST STEP = 599)
LBL D A+2+2+2+ ..... 2+ GTO B (LAST STEP = 999)
LBL E 1+1+1+1+ ..... +1= RTN (LAST STEP = 1999)

THE PROGRAMS ABOVE WERE UPLOADED IN THE FOLLOWING ORDER, USING CHIP NUMBER 0, CHIP SELECT 2 (SECONDARY SLOT). AFTER ALL PROGRAMS WERE IN, EACH STEP OF THE CATALOG WAS CHECKED TO MAKE SURE ALL ENTRIES WERE CORRECT. THE 600 AND 1200 STEP BOUNDARIES WERE ALSO CHECKED.

PROGRAM #	SIZE (IN STEPS)	CATALOG ENTRY
3	600	306
10	8	906
11	8	914
15	304	922
22	1000	1226
30	8	2226
50	600	2234
52	2000	2834
57	1000	4834
60	304	5834
61	8	6138
68	600	6146
74	1000	6746
79	2000	7746
83	1000	9746
86	600	10746
90	304	11346
92	2000	11650
95	1000	13650
96	8	14650
97	304	14658
98	1000	14962
99	304(TOO BIG)	15962

PROGRAM 99 WAS TOO BIG TO FIT AND DIDN'T UPLOAD.

4. EACH LABEL IN EACH OF THE PROGRAMS WAS EXECUTED AND WORKED.

5. OPS 72 AND 73 WORK IN THE MASTER SLOT ONLY AND COULD NOT BE TESTED THIS TIME SINCE THE EMULATOR WAS IN THE 2ND SLOT.

6. A PROGRAM WAS DOWNLOADED, EDITED, UPLOADED INTO THE SAME PLACE AND EXECUTED CORRECTLY. NOTHING ELSE WAS CHANGED.

7. CANNOT BE TESTED AS STATED IN THE PROBLEM SECTION.

8. OP 74 WORKED AS EXPECTED.

9. OP 75 TO DETERMINE HOW MANY STEPS WERE LEFT DID NOT GIVE THE CORRECT ANSWER.

PROCEDURE RESULTS ----- 8/20/80

THE FOLLOWING PROGRAMS WERE UPLOADED.

PGM #	SIZE	CHIP #, CHIP SELECT	OP 75 RESULTS
1	2000	3, 2	13694, 01
5	3000	3, 2	10694, 05
11	3000	7, 4	7694, 11
57	3000	2, 4	4694, 57
99	3000	0, 2	1694, 99

4. EACH PROGRAM WAS EXECUTED CORRECTLY.

5. OP 73 WAS USED TO RENAME THE EMULATOR AND SEVERAL PROGRAMS WERE EXECUTED USING THE NEW NAME. OP 72 WORKED AS EXPECTED AND 148 REGISTERS WERE ADDED TO THE TOTAL.

9. THE RESULTS OF OP 75 SHOWN ABOVE ARE CORRECT.

FUNCTION : PROTECTED CROM FEATURE

PRIMARY KEYS : READ, TRACE (FLAG D), EVAL, R/S

OP CODES : OP 1, OP 41, OP 52, OP 74

TEST PROCEDURE :

1. WRITE A PROGRAM ON THE 990 TO GENERATE A PROTECTED CROM FROM THE OBJECT FILE OF AN UNPROTECTED ONE -- ELAINE ACREE

2. USE THE MASTER LIBRARY OBJECT FILE AND CREATE A PROTECTED CROM -- DON O'GRADY

3. 10/80 RUN SEVERAL PROGRAMS OF THE LIBRARY -- DON O'GRADY. ALL MESSAGES, PRINT, AND ANSWERS SHOULD BE EXACTLY THE SAME AS BEFORE BECOMING PROTECTED. BAD DATA SHOULD BE USED TO TEST ERROR CONDITIONS IN ADDITION TO GOOD DATA.

4. 3/81 TRY COMPUTATIONS, RUNNING HARDWIRE FUNCTIONS, AND GENERATING ERRORS IN CUE AND SST MODES. THESE SHOULD TRACE ON THE DISPLAY AND OPERATE AS USUAL WITHOUT DESTROYING THE CUE ADDRESS AND THE PROTECTED FEATURE OF THE CROM.

5. 3/81 RUN PROGRAMS IN THE CROM WITH THE TRACE FLAG ON. ONLY THE KEYS PRESSED FROM THE KEYBOARD OR SECTIONS OF CODE RUN IN MAIN MEMORY (CALLED AS SUBROUTINES FROM THE CROM) SHOULD TRACE (IN DISPLAY AND ON PRINTER).

6. 10/80 TRY READ (DOWNLOAD) AND WRITING TO A CASSETTE. THEY SHOULD NOT OPERATE. THE COPY CRAM OP SHOULD COPY ONLY THE FIRST 600 STEPS. MAKE SURE THAT CRAM IS AS PROTECTED AS THE ORIGINAL. CHANGED

7. 3/81 OP 1 TO SET DEFAULTS SHOULD NOT AFFECT THE PROTECTION OF THE CROM. OP 41 TO SHOW THE NUMBER OF THE MODULE IN THE MASTER SLOT SHOULD WORK AS USUAL. RUNNING EVAL OR USER PROGRAMS IN MAIN MEMORY SHOULD NOT BE AFFECTED BY THE PROTECTED CROM EVEN IN TRACE MODE. OTHER MODULES INSTALLED SHOULD ALSO NOT BE AFFECTED.

8. TRY RUNNING ROUTINES IN THE PROTECTED CROM FROM THE PROMPTING SEQUENCE, THE KEYBOARD, AND AS SUBROUTINES FROM MAIN MEMORY. IN ALL CASES, ONLY THE PROTECTED PART SHOULD NOT TRACE BUT OTHERWISE THE OPERATION SHOULD BE THE SAME AS AN UNPROTECTED CROM. TRY THE SST FUNCTION ON THE MAIN MEMORY ROUTINE THAT CALLS THE PROTECTED CROM. THE WHOLE ROUTINE IN THE CROM SHOULD BE EXECUTED WITH 1 SST.

LINDA FERRIO 7/8/80

BOB GAHL - JULY 15, 1980

FUNCTION: CROM/CRAM USAGE

PRIMARY KEYS: PGM, SBR, LBL, USER-DEFINED AND OTHER KEYS WHICH MAY BE USED AS LABELS, AND READ

OP CODES: OP 49

TEST PROCEDURE:

I. CROM USAGE

5/81 A. CALLING DIFFERENT ROUTINES WITHIN LIBRARY
1. UTILIZE MASTER LIBRARY
2. CODEBREAKER PROGRAM CALLS RANDOM NUMBER GENERATOR TO GENERATE CODE
3. SUCCESSFUL CODE GENERATION WILL SHOW THAT ONE CROM WILL INTERACT WITH ITSELF USING DIFFERENT PROGRAMS

B. CALLING DIFFERENT ROUTINES WITHIN LIBRARY PROGRAM

5/81 1. UTILIZE MASTER LIBRARY
2. GENERALLY ALL PROMPTING IS USED AS A SUBROUTINE
3. IF A PROGRAM WILL CALL ITS TITLE (A SUBROUTINE WITHIN THE PROGRAM), THEN PROGRAMS WILL INTERACT WITH THEMSELVES

C. LIBRARY PROGRAM CALLING PROGRAM IN MAIN MEMORY

5/81 1. UTILIZE MASTER LIBRARY
2. ROOTFINDER PROGRAM CALLS EQUATION IN MAIN MEMORY FOR EVALUATION
3. EVALUATING f(x)= 4 sin x + 1 - x SHOULD SHOW ROOTS AT:
   a. -2.20703125
   b. -0.33984375
   c. 2.69921875
4. a=-3, b=3, e=.01, Δx=.5 (RAD)

II. CRAM USAGE

A. CALLING DIFFERENT ROUTINES WITHIN LIBRARY

5/81 1. LOAD RANDOM NUMBER PROGRAM FROM MASTER TO CRAM AS PROGRAM 1 OF CRAM 99
2. LOAD A PROGRAM WHICH WILL CHANGE RANDOM NUMBER RANGE FROM $0 < x \leq 1$ to $23 < x \leq 32$ AS PROGRAM 2
3. EXECUTE PROGRAM 2 TO GET NEW RANDOM NUMBERS

B. CALLING DIFFERENT ROUTINES WITHIN LIBRARY PROGRAM

3/81 1. USING PROGRAM 2 MENTIONED ABOVE, UTILIZE PROMPTING SUBROUTINES
2. ROUTINES PROPERLY CALLED INDICATE CORRECTNESS

C. LIBRARY PROGRAM CALLING PROGRAM IN MAIN MEMORY

3/81 1. LOAD ROOTFINDER OF MASTER LIBRARY INTO CRAM 99
2. CALL fcn IN I.C.3 FOR EVALUATION USING SAME INITIAL CONDITIONS
3. CORRECT ANSWERS YIELD PROPER FUNCTIONING

D. LIBRARY PROGRAM CALLING PROGRAM IN CROM

7/80 1. UTILIZE PROGRAM 2 IN CRAM
2. CALL RANDOM NUMBER ROUTINE IN CROM LIBRARY 1
3. CORRECT ANSWERS INDICATE CORRECT FUNCTIONING

BOB GAHL - JULY 15, 1980

III. TRY ALL COMBINATIONS OF MASTER & 2ND. SLOT - -
3/81
A. CROM/CROM
B. CRAM/CRAM
C. CROM/CRAM) done
D. CRAM/CROM)
E. NAMES & UNNAMED CRAMS
F. TRY WITH EMPTY SLOTS- unnamed + named crams w/ 1 empty

FUNCTION: USER RESPONSE KEYS

PRIMARY KEYS: YES, NO, UNK, ENT, CONT

OP CODES: OP 4, OP 5, OP 6, OP 7

TEST PROCEDURE:

I. OP 4 (ALL QUE)
5/81
A. MOVE LABELS AROUND TO INSURE BRANCHING CORRECT
B. TRY NON-NUMERIC LABELS
C. TRY NON-EXISTING LABELS
D. TRY CROM/CRAM/MM

II. OPS (Y/N)
5/81
A. INSURE NEXT INSTRUCTION BLOCK EXECUTED ON YES
B. VARY INSTRUCTION BLOCKS
C. CROM/CRAM/MM

III. OP 6 (E/C)
5/81
A. INSURE NEXT INSTRUCTION BLOCK EXECUTED ON ENT
B. VARY INSTRUCTION BLOCKS
C. CROM/CRAM/MM

IV. OP 7 (CONT)
5/81
A. INSURE EXECUTION CONTINUES ON CONT
B. CROM/CRAM/MM

V. CHECK OUT QUE MODE - MAKE SURE PROGRAM COUNTER STAYS THE SAME NO MATTER WHAT KEYS ARE TRIED EXCEPT RST.

FUNCTION: PROMPTING SEQUENCE

PRIMARY KEYS: YES, NO, ENT, CONT, NUMERICS

OP CODES: NONE

TEST PROCEDURE

5/81
I. CALL EXISTING PROGRAMS IN CROM/CRAM
II. CALL PROGRAMS USED FOR UTILITY BUT NOT FOR USER
III. CALL NON-EXISTANT PROGRAMS/MODULES
IV. UTILIZE FUNCTION TESTS 6 & 8
V. CALL A NAMED CRAM FROM THE PROMPTING SEQUENCE

```
COM              TI 88 ALEX CHECKOUT
COM                   DON OGRADY

COM      PART I:          PROTECTED CROMS
COM      RESPOSIBILITY:  OGRADY/FERRIO/ACREE

COM      THIS TEST WILL BE PERFORMED BY PROVIDING LINDA AND
COM      ELAINE WITH ACCESS TO THE MASTER LIBRARY FILE UPON
COM      REQUEST. A RECORD OF KNOWN PROGRAM CAPABILITIES,
COM      LIMITATIONS AND BUGS AT THE TIME OF THE REQUEST WILL
COM      BE COMPILED. AFTER THE "PROTECTED" MASTER LIBRARY
COM 10/80FILE HAS BEEN GENERATED A SIMULATOR TAPE WILL BE
COM      MADE AND THE PROGRAMS TESTED TO VERIFY THAT THEY
COM      PERFORM ACCORDING TO THE PROGRAM CAPABILITIES AND
COM      BUGS LIST COMPILED AT THE TIME OF "PROTECTION."
COM      THE TESTS, WHICH WILL BE PERFORMED BY BOB GAHL
COM      AND MYSELF, WILL CONSIST OF THE SAME SET OF SAMPLE
COM      PROBLEMS USED TO VERIFY CORRECT EXECUTION OF THE
COM      ORIGINAL MASTER LIBRARY

COM      PART II:         USER-DEFINED KEYBOARD CROMS
COM      RESPONSIBILITY:  OGRADY

COM 8/80 THIS TEST WILL BE PERFORMED BY CREATING A SMALL
COM      USER-DEFINED KEYBOARD CROM FROM THE SOURCE FILE
COM      PRESENTED BELOW. THE PURPOSE OF THE PROGRAM IS
COM      TO PERFORM VARIOUS ENGLISH/METRIC UNIT CONVERSIONS:

COM      LBL A    IN    -> CM
COM      LBL B    FT    -> M
COM      LBL C    YD    -> M
COM      LBL D    MI    -> KM
COM      LBL K    CM    -> IN
COM      LBL L    M     -> FT
COM      LBL M    M     -> YD
COM      LBL N    KM    -> MI
COM      LBL P    N.MI  -> MI
COM      LBL Q    F     -> C
COM      LBL R    C     -> F
COM      LBL S    OZ    -> LIT
COM      LBL T    GAL   -> LIT
COM      LBL U    OZ    -> GM
COM      LBL V    LB    -> KG
COM      LBL W    LIT   -> OZ
COM      LBL X    LIT   -> GAL
COM      LBL Y    GM    -> OZ
COM      LBL Z    KG    -> LB

COM      LABEL 0 ( WHICH PERFORMS THE MI -> N.MI
COM      CONVERSION IS PLACED IN PROGRAM 2 AND MAY
COM      BE ACCESSED FROM PROGRAM 1 THROUGH SBL 00.
COM      LABEL 0 SHOULD NOT BE DIRECTLY ACCESSIBLE
COM      AS A USER-DEFINED KEY.

COM      LABEL E IS USED TO OUTPUT THE PROGRAM TITLE
COM      AND EXPLAIN THAT AN OVERLAY IS REQUIRED FOR
COM      PROGRAM OPERATION.  LABELS F-I ARE USED AS
COM      SUBROUTINES IN PROGRAM 1.  LABEL J CALLS
COM      MAIN MEMORY LABEL A AS A SUBROUTINE

COM      THIS TEST WILL BE EXECUTED BY PERFORMING THE
COM      FOLLOWING OPERATIONS FROM BOTH THE KEYBOARD
```

Tested another user defined crom 3/81. LF

```
COM     AND A MAIN MEMORY PROGRAM WITH THE USER-DEFINED
COM     CROM PLACED IN BOTH THE MASTER AND SECONDARY
COM     SLOTS: (NOTE - WHEN THE USER-DEFINED CROM IS IN
COM     THE MASTER SLOT PROGRAMS IN MAIN MEMOR AND THE
COM     SECONDARY SLOT CAN ONLY BE EXECUTED AS SUBROUTINES
COM     OF THE USER-DEFINED MODULE)

COM     I.    CALL THE USER-DEFINED KEYS A-Z  FROM THE
COM           KEYBOARD

COM           A.    WITHOUT SELECTION OF PROGRAM

COM           EXPECTED RESULTS: MASTER SLOT
COM           CALCULATOR SHOULD SEARCH FOR A-Z IN PROGRAM
COM           1 OF THE USER-DEFINED CROM

COM           EXPECTED RESULTS: SECONDARY SLOT
COM           CALCULATOR SHOULD SEARCH FOR A-J IN MAIN
COM           MEMORY AND PERFORM FIRST FUNCTIONS OF
COM           KEYS CORRESPONDNG TO K-Z

COM           B.    WITH SELECTION OF PROGRAM

COM           EXPECTED RESULTS: MASTER SLOT
COM           SHOULD NOT ALLOW SELECTION OF PROGRAM

COM           EXPECTED RESULTS: SECONDARY SLOT
COM           CALCULATOR WILL SEARCH FOR A-J IN SELECTED
COM           PROGRAM AND INDICATE AN INVALID FIELD ERROR
COM           FOR K-Z

COM     II.   CALL THE USER-DEFINED KEYS A-Z FROM A
COM           PROGRAM IN MAIN MEMORY

COM           A.    WITHOUT SELECTION OF PROGRAM

COM           EXPECTED RESULTS: EITHER SLOT
COM           CALCULATOR WILL SEARCH MAIN MEMORY FOR A-J
COM           AND INDICATE AN INVALID FIELD ERROR FOR
COM           K-Z

COM           B.    WITH SELECTION OF PROGRAM

COM           EXPECTED RESULTS: MASTER SLOT
COM           CALCULATOR WILL SEARCH FOR A-J IN PROGRAM 1
COM           OF USER-DEFINED CROM AND INDICATE AN INVALID
COM           FIELD ERROR FOR K-Z

COM           EXPECTED RESULTS: SECONDARY SLOT
COM           CALCULATOR WILL SEARCH FOR A-J IN SELECTED
COM           PROGRAM AND INDICATE AN INVALID FIELD ERROR
COM           FOR K-Z

COM     III.  CALL SBL 00 FROM THE KEYBOARD OR MAIN MEMORY

COM           A.    WITHOUT SELECTION OF PROGRAM

COM           EXPECTED RESULTS: EITHER SLOT
COM           CALCULATOR SHOULD LOOK FOR THE LABEL IN
COM           MAIN MEMORY WHEN CALLED FROM SAME-
COM           OR IN PROGAM 1 OF USER-DEFINED CROM
COM           WHEN CALLED FROM KEYBOARD
```

```
COM           B.    WITH SELECTION OF PROGRAM

COM           EXPECTED RESULTS:  MASTER SLOT
COM           SHOULD NOT ALLOW SELECTION OF PROGRAM
COM           FROM KEYBOARD-
COM           CALCULATOR SHOULD LOOK FOR THE LABEL IN
COM           PROGRAM 1 OF THE USER-DEFINED CROM
COM           WHEN CALLED FROM MAIN MEMORY

COM           EXPECTED RESULTS:  SECONDARY SLOT
COM           CALCULATOR SHOULD LOOK FOR THE LABEL IN
COM           SELECTED PROGRAM

COM      IV.  CALL ABSOLUTE-ADDRESS SUBROUTINE
COM           FROM THE KEYBOARD OR MAIN MEMORY

COM           A.    WITHOUT SELECTION OF PROGRAM

COM           EXPECTED RESULTS:  MASTER SLOT
COM           CALCULATOR SHOULD CALL SUBROUTINE AT
COM           SPECIFIED ADDRESS IN PROGRAM 1 OF
COM           USER-DEFINED CROM WHEN CALLED FROM
COM           KEYBOARD AND IN MAIN MEMORY WHEN
COM           CALLED FROM SAME

COM           EXPECTED RESULTS:  SECONDARY SLOT
COM           CALCULATOR SHOULD CALL SUBROUTINE AT
COM           SPECIFIED ADDRESS IN MAIN MEMORY

COM           B.    WITH SELECTION OF PROGRAM

COM           EXPECTED RESULTS:  MASTER SLOT
COM           CALCULATOR SHOULD CALL SUBROUTINE AT
COM           SPECIFIED ADDRESS IN PROGRAM 1 OF
COM           THE USER-DEFINED CROM WHEN CALLED
COM           FROM MAIN MEMORY
COM           SHOULD NOT ALLOW SELECTION OF PROGRAM
COM           FROM KEYBOARD

COM           EXPECTED RESULTS:  SECONDARY SLOT
COM           CALCULATOR SHOULD CALL SUBROUTINE AT
COM           SPECIFIED ADDRESS IN SELECTED PROGRAM

COM      V.   CALL LABELS D AND E FROM ONBOARD
COM           PROMPT SEQUENCE

COM           EXPECTED RESULTS:  MASTER SLOT
COM           SHOULD PROVIDE ERROR MESSAGE

COM           EXPECTED RESULTS:  SECONDARY SLOT
COM           SAME AS II-B

NAME 01

PAGE 1

COM*****************UNIT CONVERSIONS*************************
LBL E   OP 1
LBL 90  ALPHA C O N V E R S I O N S ? ALPHA
        OP 04 91 92 90 90 GTL 90
LBL 92  0 RTN
```

```
LBL 91  ALPHA E X I T SPACE P R O M P T I N G CNT ALPHA PAU
        ALPHA & SPACE U S E SPACE K E Y B O A R D CNT
        ALPHA PAU ALPHA O V E R L A Y ; CNT ALPHA PAU
        ALPHA T O SPACE S T A R T , CNT ALPHA PAU ALPHA
        P R E S S SPACE FBIT LCN LCK SPACE R LCD LCT PT
        ALPHA OP 04 91 91 91 91 GTL 91

LBL F   ADV
LBL I   UDG PRT IFN UDC PAU RTN
LBL G   STO C B SBL LCC RTN

LBL A       STO B  1 UDF 17 UDG      STO B  2 GTL F
LBL B       STO B  3 UDF 18 UDG      STO B  4 GTL F
LBL C       STO B  5 UDF 19 UDG      STO B  4 GTL F
LBL D       STO B  6 UDF 20 UDG      STO B  7 GTL F
LBL K   RCP STO B  2 UDF 17 UDG RCP STO B  1 GTL F
LBL L   RCP STO B  4 UDF 18 UDG RCP STO B  3 GTL F
LBL M   RCP STO B  4 UDF 19 UDG RCP STO B  5 GTL F
LBL N   RCP STO B  7 UDF 20 UDG RCP STO B  6 GTL F
LBL P   RCP STO B  8 UDF 21 UDG RCP STO B  6 GTL F
LBL Q       STO B  9 UDF 26 UDG      STO B 10 GTL F
LBL R       STO B 10 UDF 27 UDG      STO B  9 GTL F
LBL S       STO B 11 UDF 22 UDG      STO B 12 GTL F
LBL T       STO B 13 UDF 23 UDG      STO B 12 GTL F
LBL U       STO B 11 UDF 24 UDG      STO B 14 GTL F
LBL V       STO B 15 UDF 25 UDG      STO B 16 GTL F
LBL W   RCP STO B 12 UDF 22 UDG RCP STO B 11 GTL F
LBL X   RCP STO B 12 UDF 23 UDG RCP STO B 13 GTL F
LBL Y   RCP STO B 14 UDF 24 UDG RCP STO B 11 GTL F
LBL Z   RCP STO B 16 UDF 25 UDG RCP STO B 15 GTL F

LBL 00  PGM  2    SBL  0    GTL  F

LBL 01  ALPHA LCI LCN          ALPHA RTN
LBL 02  ALPHA LCC LCM          ALPHA RTN
LBL 03  ALPHA LCF LCT          ALPHA RTN
LBL 04  ALPHA LCM              ALPHA RTN
LBL 05  ALPHA LCY LCD          ALPHA RTN
LBL 06  ALPHA LCM LCI          ALPHA RTN
LBL 07  ALPHA LCK LCM          ALPHA RTN
LBL 08  ALPHA LCN SR  LCM LCI  ALPHA RTN
LBL 09  ALPHA *0  F            ALPHA RTN
LBL 10  ALPHA *0  C            ALPHA RTN
LBL 11  ALPHA LCO LCZ          ALPHA RTN
LBL 12  ALPHA LCL LCI LCT      ALPHA RTN
LBL 13  ALPHA LCG LCA LCL      ALPHA RTN
LBL 14  ALPHA LCG LCM          ALPHA RTN
LBL 15  ALPHA LCL LCB          ALPHA RTN
LBL 16  ALPHA LCK LCG          ALPHA RTN

LBL 17  ( *  2.54          ) RTN
LBL 18  ( *   .3048        ) RTN
LBL 19  ( *   .9144        ) RTN
LBL 20  ( *  1.609344      ) RTN
LBL 21  ( *   .86897624    ) RTN
LBL 22  ( *   .0295735296  ) RTN
LBL 23  ( *  3.785411784   ) RTN
LBL 24  ( * 28.34952313    ) RTN
LBL 25  ( *  .45359237     ) RTN

LBL 26  ( ( - 32 ) / 1.8 ) RTN
LBL 27  ( * 1.8 + 32 ) RTN
```

```
LBL J   PGM 0  UDA  RTN

LBL H   5 STO Z Z OP 7
        UDA RTN

PAGE 2

LBL 0   STO  B    6    PGM  1    UDF
        21   PGM  1    UDG  STO  B    8    RTN

END
```

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: PAUSE TIMING (OP 61, OP 62)

DEFINITION: 18.105 - 108

TEST PROCEDURE:

1. In MANUAL MODE:

   0 OP 62 - set pause to lower limit

   9.9 OP 62 - set pause to upper limit

   -1 OP 62, 20 OP 62 - use invalid numbers

3/81 5 OP 61 - set default

   5 ALPHA 7 7 7 7 7 ALPHA OP 62

   - set pause with display covered

   9.9 OP 62 - measure time

2. In RAM

   a) As in 1

   b) Execute Loop (A DSZ) to get time with no pause instruction

   c) Execute Loop (A PAU DSZ) to get time with pause set to 0.

   d) Do c) with pause set to .1.

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: Recall and Load Program Steps

DEFINITION: 18.133 - 136

TEST PROCEDURE:

1. In MANUAL MODE:

   a) OP 47 - Puts the hex-code of the program step pointed to by main PC into display

5/81

   b) OP 48 - Puts the contents of the numeric display register into the program location pointed to by the main PG.

2. OP 47/48

6/81

   a) Check with invalid numbers, fractions, EE-Mode

   b) Check that after execution the main PC is not affected

6/81 3. In RAM (same as 1.)

4. In CROM (same as 1.)

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: Load Program Counter

DEFINITION: 18.129 - 132

TEST PROCEDURE:

1. In MANUAL MODE

5/81 a) OP 46 - puts the contents of the numeric display register in PC

b) OP 46 - check with invalid contents in numeric register, 5/81 with 0 program steps, HEX - MODE, UNNORMALIZED #, fractions in display.

6/81 2. In RAM (same as 1.)

7/80 3. In CROM (same as 1.)

7/11/80
SKIPPER MCDONALD

24

FUNCTION: PROMPTING SEQUENCE

PRIMARY KEYS: TIME

OP CODES:

TEST PROCEDURE:

1. CHECK ALL POSSIBLE PATHS THE PROMPTS MAY TAKE.
2. CHECK TIME ENTRIES:
   - A. CHECK EACH ENTRY FOR NEGATIVE OR EXPONENTIAL ENTRIES.
   - B. HOURS - > 12 IN 12HR MODE.
     > 24 IN 24HR MODE.
   - C. MIN - > 59.
   - D. MONTH - > 12.
   - E. DAY - > 28, 29, 30, 31 FOR THEIR RESPECTIVE MONTH/YEAR COMBINATION.
   - F. CHECK LEAP YEARS < 2100 FOR FEB 28->29.
   - G. CHECK MONTH AND DAY ROLLOVER FOR EACH MONTH FOR A LEAP YEAR AND A NON LEAP YEAR.
3. CHECK TO SEE IF ABOVE WORKS PROPERLY IN PROGRAM MODE.
4. MODULE SELECT

5/81

7/80

6/81

7/11/80
SKIPPER MCDONALD

25

FUNCTIONS: TIME/ALARM/DATE/BUZZER

PRIMARY KEYS:

OP CODES: OP59, 60, 63, 64, 65, 66, 67, 68, 69

TEST PROCEDURE:

1. CHECK EACH OP FOR THE CORRECT PROMPT AND RESPONSE.
2. ALARM CHECK
   - A. INVALID ENTRIES (SEE TIME SET TESTS)
   - B. CHECK THAT ALARM GOES OFF ONLY IN THE AM IF SET FOR AM. LIKEWISE IN PM.
   - C. DO "B" AGAIN ONLY SWITCH BETWEEN 12 & 24 AND 24 & 12 BEFORE THE ALARM SHOULD SOUND.
3. MAKE SURE TONES SOUND FOR ONLY THE PROPER TIMES.
4. CHECK FOR EACH COMBINATION OF TONES TO THE EXCLUSION OF OTHERS.
5. PROGRAM MODE - REPETE TEST IN PROGRAM MODE.
6. CHECK THAT ERROR BEEP OCCURS FOR ALL ERRORS AND DOES NOT OCCUR ON ERRORS WHICH DO NOT HALT THE PROGRAM

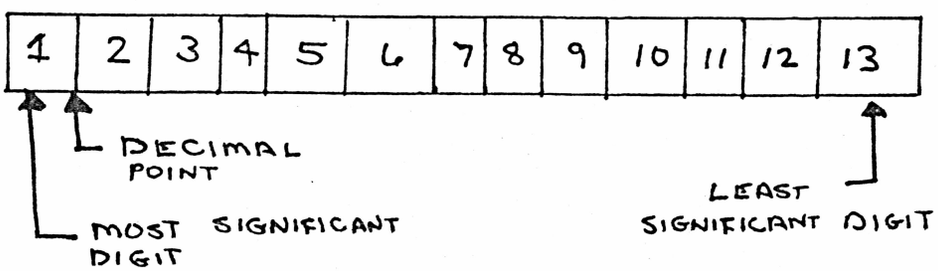
3/81

From the Desk of
ELAINE ACREE

8/29/80

RE: ALEX REPORTS - ACCURACY

ALL VALUES ARE CONSIDERED TO BE 13 DIGIT VALUES. THEREFORE, THE "LAST" DIGIT IS THE 13TH DIGIT OR LEAST SIGNIFICANT DIGIT.



WORST CASE RELATIVE ERROR (NOT ABSOLUTE ERROR MAY BE DETERMINED AS FOLLOWS:

	AFFECT DIGITS	WORST CASE RELATIVE ERROR
LAST DIGIT	13	$9 \times 10^{-12} \approx 10^{-11}$
LAST 2 DIGITS	12, 13	$99 \times 10^{-12} \approx 10^{-10}$
LAST 3 DIGITS	11, 12, 13	$999 \times 10^{-12} \approx 10^{-9}$
LAST 4 DIGITS	10, 11, 12, 13	$9999 \times 10^{-12} \approx 10^{-8}$

TEXAS INSTRUMENTS
INCORPORATED
P. O. Box 10508 ● Mail Station 5893 ● Lubbock, Texas 79408
(806) 741-3301

ALEX REPORT 8/5/80 E. S. ACREE

OPERATIONS: FACTORIALS

TEST #1: FACTORIALS 0! TO 69! IN PROGRAM MODE

RESULTS

1) FACTORIALS 0! TO 20! ARE CORRECT

2) FACTORIALS 21! TO 69! WHICH HAVE 16 OR MORE SIGNIFICANT DIGITS (TRAILING ZEROES EXCLUDED) ARE INCORRECT IN THE LAST DIGITS

FACTORIAL RANGE	NO. OF DIGITS LOST	RANGE OF DIGITS LOST
21! - 32!, 34!, 35! 39!, 42!, 45!, 48!, 51!, 55!, 59! 64!	1 (13TH DIGIT)	1 TO 9
33! 36! 38! 40! 41! 43! 44! 46! 47! 49! 50! 52! - 54! 56! - 58! 60! - 63! 65! - 69!	2 (12TH, 13TH DIGITS)	10 TO 64

3) ALL VALUES GENERATED ARE LOW IN AT MOST TWO DIGITS (FOR 21! TO 69!)

ALEX REPORT 8/5/80 E. S. ACREE

OPERATION : D.dd → DMS [INV DMS]

TEST #1 - 108 RANDOM VALUES GENERATED BY THE IBM 360 WERE USED AS ARGUMENTS TO THE INV DMS FUNCTION IN PROGRAM MODE. THE ANSWERS WERE COMPARED TO THE RESULTS GENERATED BY THE IBM 360 USING QUADRUPLE PRECISION ARITHMETIC.

RESULTS :

1) 48 VALUES LOW BY 1 IN THE LAST DIGIT

BUG ⇒ 2) 1 VALUE -68.083333 33333 CONVERTS TO -68.046
IN THE DISPLAY INTERNALLY THE VALUE IS
-68.04599999999

ALEX REPORT 8/5/80 E.S. ACREE

OPERATIONS : DMS → D.dd

TEST # 1 - 144 RANDOM VALUES GENERATED BY THE IBM 360 WERE USED AS ARGUMENTS TO THE DMS FUNCTION IN PROGRAM MODE. RESULTS FROM PRODUCT X WERE COMPARED TO RESULTS CALCULATED BY THE IBM 360 USING QUADRUPLE PRECISION ARITHMETIC.

RESULTS

1) OF 144 VALUES TESTED 108 VALUES WERE LOW IN THE LAST DIGIT.
   - 105 VALUES WERE LOW BY 1
   - 3 VALUES WERE LOW BY 2 (THESE 3 VALUES ALL CONTAINED 23 SECONDS AS THE LAST 2 DIGITS OF THE ARGUMENT)

ALEX REPORT 8/7/80 E. S. ACREE

OPERATIONS: OPS 22-27 - CONVERSIONS D↔R↔G

TEST #1 : OPS 22-27 WERE TESTED OVER A WIDE RANGE OF POSITIVE AND NEGATIVE VALUES :

RESULTS : 216 VALUES TESTED TOTAL

		NUMBER OF RESULTS OFF BY ONE OR MORE IN THE LAST DIGIT					
ARGUMENTS IN DEGREES	DEGREE INCREMENT	D→R OP 22	R→D OP 23	R→G OP 24	G→R OP 25	G→D OP 26	D→G OP 27
0° TO 365° 74 VALUES TESTED	+ 5°	66 TOTAL	72 TOTAL	59 TOTAL	27 TOTAL	33 TOTAL	54 TOTAL
0° TO -365° 74 VALUES TESTED	-5°	66 TOTAL	72 TOTAL	58 TOTAL	26 TOTAL	33 TOTAL	54 TOTAL
0° TO 670° 68 VALUES TESTED	+10°	58 TOTAL	65 TOTAL	62 TOTAL	27 TOTAL	32 TOTAL	54 TOTAL
TOTAL NUMBER OFF		190	209	179	80	98	162
NUMBER OF VALUES OFF BY MORE THAN ONE IN THE LAST DIGIT (2 MIN, 5 MAX)		45	22	17	0	1	1

ALEX REPORT 2/5/80 E. S. ACREE

OPERATION : $P \rightarrow R$

TEST : 144 RANDOM VALUE PAIRS WERE GENERATED AS ARGUMENTS BY THE IBM 360. THE RESULTS FROM PRODUCT X WERE COMPARED TO RESULTS GENERATED BY THE IBM 360 USING QUADRUPLE PRECISION COMPUTATIONS.

RESULTS :

	ALL DIGITS CORRECT	LAST DIGIT WRONG	LAST TWO DIGITS WRONG	LAST THREE DIGITS WRONG
X	9	122	13	Ø
Y	28	104	11	1 (AT $\Theta = 180.3128529^{\circ}$, $R = .7653494131$)

ALEX REPORT 8/5/80 E.S. ACREE

OPERATION : R → f

TEST #1 : 144 RANDOM VALUE PAIRS WERE GENERATED AS ARGUMENTS BY THE IBM 360. THE RESULTS FROM PRODUCT X WERE COMPARED TO RESULTS GENERATED BY THE IBM 360 USING QUADRUPLE PRECISION COMPUTATIONS.

RESULTS :

	ALL DIGITS CORRECT	LAST DIGIT OFF	
R X	18	115	
6 4	95	28	
BOTH X, 4	11		

ALEX REPORT 8/21/80 ACREE, E.S.

OPERATIONS: ADD, SUBTRACT, MULTIPLY AND DIVIDE

TEST: 72 PAIRS OF OPERANDS WERE RANDOMLY GENERATED FOR EACH OF THE 4 BASIC FUNCTIONS TO TEST THEIR ACCURACY OVER A WIDE RANGE OF NUMBERS. THE ANSWERS GENERATED BY THE TI-88 WERE COMPARED TO ANSWERS GENERATED BY THE STRING ARITHMETIC PACKAGE WHICH CARRIES UP TO 130 DIGITS INTERNALLY.

I RESULTS: ADDITION AND SUBTRACTION

THE ANSWERS WHICH WERE OFF BY ONE IN THE LAST DIGIT HAD OPERANDS WHICH COULD NOT BE ALIGNED BECAUSE ONE OPERAND WAS SO MUCH BIGGER THAN THE OTHER - THIS RESULT SHOULD BE EXPECTED.

FUNCTION	13 DIGITS CORRECT	HIGH BY 1 IN THE LAST DIG
ADDITION	48	24
SUBTRACTION	48	24

II RESULTS: MULTIPLICATION AND DIVISION

ALL ANSWERS GENERATED BY THE TI 88 WERE CORRECT TO 13 DIGITS TRUNCATED.

FUNCTION	13 DIGITS CORRECT	ANSWERS OFF
MULTIPLICATION	72	NONE
DIVIDE	72	NONE

ALEX REPORT 8/22/80 E. S. ACREE

OPERATIONS: SMALL BASE 10 LOGS

TEST #1: LOG (X) AS X RANGES FROM 1×10^{-5} TO 100×10^{-5} IN INCREMENTS OF 1×10^{-5}

RESULTS

A) 100 VALUES TESTED

B) 40 VALUES ARE OFF

C) DIFFERENCE IS IN LAST DIGIT ONLY

TI VALUES ARE OFF BY NOMORE THAN ONE IN THE LAST DIGIT.

MAXIMUM DIFFERENCE IS 1×10^{-12}

TEST #2: LOG (X) AS X RANGES FROM .999950 TO 1.00005 IN INCREMENTS OF 1×10^{-6}

RESULTS

A) 101 VALUES TESTED

B) 99 VALUES ARE OFF

LOG (X) WHERE X = .99997 AND X = 1.0 } ARE CORRECT

C) ALL DIFFERENCES OCCUR IN FROM 1 TO 4 DIGITS

NUMBER OF DIGITS OFF	NUMBER OF VALUES LOW	HIGH
1	3	2
2	8	22
3	3	17

FOR $X < 1.0$

1	—	4
2	—	16
3	—	28
4	—	2

FOR $X > 1.0$

ALEX REPORT 8/22/80 E.S. ACREE

TEST #3: LOG (X) AX RANGES FROM .01 TO 2.0 (.01 INCREMENTS)

RESULTS

A) 200 VALUES TESTED

B) 100 VALUES ARE OFF

C) DIFFERENCE IS ALWAYS IN THE LAST DIGIT
ALL VALUES ARE LOW BY NO MORE THAN ONE
DIFFERENCE VALUES RANGE FROM $\pm 1 \times 10^{-15}$ TO 1×10^{-12}

ALEX REPORT 5/30/30 E. S. ACKLEY

FUNCTION : BASE 10 LOGS

TEST #1 : LOG(Y) FOR RANDOMLY GENERATED VALUES OF X

RESULTS

A) 146 VALUES TESTED

B) 77 VALUES ARE LOW BY ONE IN THE LAST DIGIT.

ALEX REPORT 8/25/80 E. S. ACREE

FUNCTION: INV LOG (X) [10^X]

TEST: 146 RANDOMLY DISTRIBUTED ARGUMENTS WERE GENERATED AND THE RESULTS WERE COMPARED TO ANSWERS CALCULATED BY THE IBM 360 USING QUADRUPLE PRECISION EXPONENTIATION.

RESULTS:

A) 142 OF 146 VALUES GENERATED ARE OFF IN FROM 1-2 DIGITS.

	1 DIGIT ANSWER OFF IN THE LAST DIGIT (FROM 0-9)	2 DIGITS ANSWER OFF IN THE LAST 2 DIGITS (FROM 10-99)
ANSWERS LOW	30	42
ANSWERS HIGH	19	52

ALEX REPORT 7/30/80 E.S. ACREE

SUBJECT : SMALL NATURAL LOGARITHMS

TEST #1 : LN(X) X RANGES FROM 1×10^{-5} TO 100×10^{-5} (1×10^{-5} INCREMENTS)

RESULTS

A) 100 VALUES TESTED

B) 54 VALUES ARE OFF

C) DIFFERENCE IS IN LAST DIGIT ONLY

TI VALUES ARE ALWAYS LOW BY NO MORE THAN ONE

MAXIMUM DIFFERENCE : 1×10^{-12}

TEST #2 : LN(X) X RANGES FROM .999950 TO 1.00005 (1×10^{-6} INCREMENTS)

RESULTS

A) 101 VALUES TESTED

B) 98 VALUES ARE OFF

.999970
.999999 } ARE CALCULATED CORRECTLY (3 OUT OF 101
1.0 } VALUES TESTED)

C) DIFFERENCES OCCUR IN FROM 1 TO 4 DIGITS

NUMBER OF DIGITS OFF	NUMBER OF VALUES LOW	HIGH
1	3	3
2	9	16
3	1	16

} FOR $X < 1.0$

1	3	—
2	9	—
3	36	—
4	2	—

} FOR $X > 1.0$

DIFFERENCES ARE ON THE ORDER OF $\pm 10^{-15}$ TO 10^{-19}

ALEX REPORT - SMALL NATURAL LOGARITHMS - CONTINUED

TEST #3 : LN (X) AS X RANGES FROM .01 TO 2.0 (.01 INCREMENTS)

RESULTS

A) 200 VALUES TESTED
B) 108 VALUES OFF
C) DIFFERENCE IS ALWAYS IN THE LAST DIGIT
ALL VALUES ARE LOW BY NO MORE THAN ONE
DIFFERENCE VALUES RANGE FROM $\pm 1 \times 10^{-14}$ TO $\pm 1 \times 10^{-12}$

ALEX REPORT 8/22/80 E.S. ACREE

FUNCTION: NATURAL LOGARITHMS

TEST #1: LOG(X) FOR RANDOMLY GENERATED VALUES OF X

RESULTS

A) 146 RANDOMLY GENERATED VALUES WERE TESTED

B) 43 VALUES ARE LOW BY 1 IN THE LAST DIGIT

ALEX REPORT 8/25/40 E.S. ACREE

FUNCTION : INV LN (X) [e^X]

TEST : 146 RANDOMLY DISTRIBUTED ARGUMENTS WERE GENERATED AND THE RESULTS COMPARED TO ANSWERS CALCULATED BY THE IBM 360 FORTRAN QUADRUPLE PRECISION SUBROUTINE

RESULT : 134 OF 146 VALUES GENERATED WERE OFF IN FROM 1 TO 2 DIGITS

	ANSWER OFF IN THE LAST DIGIT (0-9)	ANSWER OFF IN THE LAST 2 DIGITS (10-99)
ANSWERS LOW	33	24
ANSWERS HIGH	54	26

~~ALEX REPORT~~ 8/25/80 E.S. ACREE

FUNCTION: $4 \uparrow X$

TEST: 48 RANDOMLY GENERATED X,Y PAIRS OF ARGUMENTS WERE USED AND THE RESULTS WERE COMPARED TO ANSWERS CALCULATED BY THE IBM 360 USING QUADRUPLE PRECISION ARITHMETIC.

RESULTS.

A) 33 OF 48 VALUES GENERATED WERE OFF IN THE LAST DIGIT

	OFF IN LAST DIGIT (0-9)	OFF IN LAST TWO DGTS (10-99)
ANSWERS LOW	17	3
ANSWERS HIGH	10	3

ALEX REPORT 8/25/80 E. S. ACREE

FUNCTION $( )^N$

TEST : 24 RANDOMLY GENERATED PAIRS OF VALUES WERE USED AS INPUT AND THE ANSWERS WERE COMPARED WITH THE RESULTS CALCULATED USING QUADRUPLE PRECISION ARITHMETIC ON THE IBM 360.

RESULTS : ALL VALUES WERE CALCULATED CORRECTLY TO 13 DIGITS.

ALEX REPORT 8/25/80 E.S. ACREE

FUNCTION: $\sqrt{\ }$

TEST: 48 RANDOMLY GENERATED VALUES WERE USED AS INPUT ARGUMENTS TO THE SQUARE ROOT FUNCTION AND THE RESULTS WERE COMPARED ANSWERS CALCULATED BY THE QUADRUPLE PRECISION SQUARE ROOT ROUTINE ON THE IBM 360.

RESULTS: 23 OF 48 VALUES TESTED WERE LOW BY ONE IN THE LAST DIGIT.

ALEX REPORT 8/25/80 E.S. ACREE

FUNCTION: $(\ )^{-1}$

TEST: 48 RANDOMLY GENERATED ARGUMENTS WERE USED AS INPUT ARGUMENTS AND THE RESULTS GIVEN COMPARED TO ANSWERS GENERATED USING QUADRUPLE PRECISION ARITHMETIC ON AN IBM 360.

RESULTS: A) 29 OF 48 VALUES WERE OFF FROM ±1 TO 5 IN THE LAST DIGIT.

B) 25 VALUES WERE LOW IN THE LAST DIGIT

C) 4 VALUES WERE HIGH IN THE LAST DIGIT.

ALEX REPORT 8/25/80 E.S. ACREE

FUNCTION: SMALL ANGLE SIN

TEST: QUADRUPLE PRECISION SIN VALUES WERE GENERATED ON THE IBM 360 FOR 0 TO 1.0 DEGREE IN INCREMENTS OF .01 DEGREE AND COMPARED TO THE VALUES GENERATED BY PRODUCT X.

RESULTS:

A) 1 ANSWER OUT OF 101 VALUES TESTED WAS CORRECT (0 DEGREES)

B) 100 VALUES WERE LOW IN THE LAST ONE TO THREE DIGITS

	LOW IN THE LAST DIGIT	LOW IN THE LAST 2 DIGITS	LOW IN THE LAST 3 DIGITS
SIN 0 TO 1.0°	43	52	5

ALEX' REPORT 8/25/80 E.S. ACREE

FUNCTION : SIN - QUADRANT BOUNDARIES

TEST : QUADRUPLE PRECISION SIN VALUES WERE GENERATED ON THE IBM 360 FOR ANGLES .5 DEGREE FROM THE QUADRANT BOUNDARY CONTINUING IN .01 DEGREE INCREMENTS UNTIL THE FINAL ANGLE IS .5 DEGREES ON THE OTHER SIDE OF THE QUADRANT BOUNDARY.

RESULTS: (PRESENTED IN TABLE) - 101 VALUES TESTED AT EACH BOUNDARY

ANGLE	LOW ... IN THE LAST DIGIT	LOW IN THE LAST TWO DIGITS	LOW IN THE LAST THREE DIGITS
89.5 - 90.5	44 (LOW BY NO MORE THAN ONE)	—	—
179.5 - 180.5	—	90	10
269.5 - 270.5	44 (LOW BY NO MORE THAN ONE)	—	—
359.5 - 360.5	—	90	10

ALEX REPORT 8/25/80 E. S. ACREE

FUNCTION : SIN (±5° INCREMENTS)

TEST: QUADRUPLE PRECISION SIN VALUES WERE GENERATED ON THE IBM 360 FOR Ø TO 360 DEGREES AND 0 TO -360 IN (±)5 DEGREE INCREMENTS. AND THE RESULTS WERE COMPARED TO THE ANSWERS GENERATED BY PRODUCT X.

RESULTS :

A) 73 VALUES WERE CHECKED FOR POSITIVE 5 DEGREE INCREMENTS AND 73 VALUES WERE CHECKED FOR NEGATIVE 5 DEGREE INCREMENTS

	LOW IN THE LAST DIGIT
0 TO +360 (+5° INCREMENTS)	28
0 TO -360 (-5° INCREMENTS)	28

ALEX REPORT 8/25/80 E.S ACREE

FUNCTION : SMALL ANGLE COS

TEST: SAME AS FOR SMALL ANGLE SIN

RESULTS :

A) 41 ANSWERS OF 100 VALUES TESTED ARE LOW BY 1 IN THE LAST DIGIT.

	LOW IN THE LAST DIGIT
COS 0 TO 1.0	41

ALEX REPORT 8/25/80 E. S. ACREE

FUNCTION: COS - QUADRANT BOUNDARIES

TESTS: SAME AS FOR SIN - QUADRANT BOUNDARIES

RESULTS: (PRESENTED IN TABLE) - 101 VALUES TESTED AT EACH BOUNDARY.

ANGLE	LOW IN THE LAST DIGIT	LOW IN THE LAST TWO DIGITS	LOW IN THE LAST 3 DIGITS
89.5 - 90.5	-	90	10
179.5 - 180.5	44	—	—
269.5 - 270.5	—	90	10
359.5 - 360.5	44	—	—

ALEX REPORTS 8/25/80 E.S. ACREE

FUNCTION : COS (± 5° INCREMENTS)

TESTS : SAME AS FOR SIN ± 5° INCREMENTS

RESULTS :

A) 73 VALUES WERE TESTED FOR POSITIVE 5 DEGREES INCREMENTS AND 73 VALUES WERE CHECKED FOR NEGATIVE 5 DEGREES INCREMENTS.

	LOW IN THE LAST DIGIT
0 TO +360 +5° INCREMENTS	28
0 TO -360 -5° INCREMENTS	28

ALEX REPORT 8/25/80 E.S. ACREE 65

FUNCTION: SMALL ANGLE TAN

TEST: SAME AS FOR SMALL ANGLE SIN

RESULTS: ALL 100 VALUES TESTED GIVE ANSWERS WHICH ARE OFF BY FROM ONE TO 3 DIGITS

	LOW IN THE LAST DIGIT	LOW IN THE LAST 2 DIGITS	LOW IN THE LAST 3 DIGITS
TAN (.01° TO 1.0°)	43	52	5

ALEX REPORT 8/25/80 E.S. ACREE

FUNCTION : TAN (QUADRANT BOUNDARIES)

TEST: SAME AS FOR SIN - QUADRANT BOUNDARIES

RESULTS: 100 VALUES TESTED AROUND THE 90, 270 BOUNDARIES
101 VALUES TESTED AROUND THE 180, 360 BOUNDARIES

ANGLE	OFF IN THE LAST DIGIT.	OFF IN THE LAST 2 DIGITS	OFF IN THE LAST 3 DIGITS	OFF IN THE LAST 4 DIGITS
89.5-90.5	39 (HIGH)	47 (HIGH)	12 (HIGH)	2 (HIGH)
179.5-180.5	—	90 (LOW)	10 (LOW)	—
269.5-270.5	40 (HIGH)	43 (HIGH)	15 (HIGH)	2 (HIGH)
359.5-360.5	18 (LOW)	72 (LOW)	10 (LOW)	—

ALEX REPORT 8/26/80 E. S. ACREE

FUNCTION: TAN (± 5° INCREMENTS)

TESTS: SAME AS FOR SIN ± 5° INCREMENTS

RESULTS: 73 VALUES WERE TESTED FROM 0 TO 360 IN +5° INCREMENTS AND FROM 0 TO -360 IN -5° INCREMENTS.

	LOW IN THE LAST DIGIT
0° TO 360° +5° INCREMENTS	40
0° TO -360° -5° INCREMENTS	40

ALEX REPORT 8/26/80 E. S. ACREE

FUNCTIONS : INV SIN, INV COS, INV TAN

TESTS 1 : 85 RANDOMLY GENERATED VALUES, UNIFORMLY DISTRIBUTED BETWEEN 0. AND 1. WERE USED AS ARGUMENTS TO THE INVERSE TRIG FUNCTIONS. THE RESULTS FROM PRODUCT X WERE COMPARED TO THE RESULTS GENERATED IN QUADRUPLE PRECISION BY THE IBM 360.

RESULTS :

	MAGNITUDE	
FUNCTIONS	LOW IN THE LAST DIGIT	HIGH IN THE LAST DIGIT
INV SIN	20	17
INV COS	44	2
INV TAN	25	19

TEST #2 : INV TAN FOR $X > 1.0$. 144 RANDOMLY GENERATED VALUES OF X (WHERE $X > 1.0$) WERE USED AS ARGUMENTS TO INV TAN WITH THE RESULTS CHECKED AS IN TEST # 1 ABOVE.

	LOW* IN THE LAST DIGIT
INV TAN (X) ($X > 1.0$)	70

* ALL ANSWERS WHICH ARE LOW ARE LOW BY ONE IN THE LAST DIGIT

ALEX REPORT 8/26/80 E.S. ACREE

LIMIT CHECK: SIN

TEST: SIN WAS CHECKED FROM 89.999990000001 TO 89.999970180048 IN 1×10^{-11} DEGREE INCREMENTS. LOOKING FOR VALUES THAT EXCEED SIN'S MAXIMUM VALUE OF 1.0

RESULTS: 18,048 VALUES WERE CHECKED
14,675 ANSWERS WERE = .999999999999
3,373 ANSWERS WERE = 1.0

NO VALUES WERE FOUND WHICH EXCEEDED ONE HOWEVER, ALL POSSIBLE ARGUMENTS WERE NOT TESTED DUE TO THE EXHORBITANT AMOUNT OF TIME REQUIRED TO CHECK ALL OF THE POSSIBLE ARGUMENTS.

ALEX REPORT 8/26/80 E. S. ACREE 70

CONTINUITY TEST : SIN, COS, TAN

TEST*: SIN, COS AND TAN WERE ALL CHECKED AT .5 DEGREE INCREMENTS TO INSURE THAT EACH OF THE FUNCTIONS WAS CONSTANTLY INCREASING OR DECREASING AS IT SHOULD BE IN EACH QUADRANT AND THAT THE SIGN OF THE ANSWER WAS CORRECT.

RESULTS : ALL VALUES CHECKED WERE CONTINUOUS

*NOTE: A BETTER CONTINUITY TEST WOULD HAVE USED A MUCH SMALLER INCREMENT THAN .5°, HOWEVER, IT REQUIRED 2 ½ HOURS TO CHECK ALL 3 FUNCTIONS SO THAT A SMALLER DEGREE INCREMENT WOULD HAVE REQUIRED AN EXHORBITANT AMOUNT OF TIME.

(.1° => 12½ HRS, .01 => 125 HRS, ETC.)

THE LIMIT CHECK FOR SIN AROUND THE 90° QUADRANT SHOWED THAT SIN IS NOT CONTINUOUS AROUND THAT BOUNDARY THE FUNCTION "BOUNCES" BETWEEN .999999999999 AND 1.0

FUNCTION TO BE TESTED: ON/OFF

Definition ON - Display time

ON,ON - Enter prompting sequence } CHANGED

OFF - Turn calculator off

Tests: A) If Keyboard Controller algorithm is locked so that it can not service time, a PUC will occur after approximately 8 seconds. This will be simulated.
B) If the Memory Controller chip does not acknowledge the OFF instruction from the Keyboard Controller chip within 1-2 minutes, the Keyboard Controller chip will remove power from both chips. This will be simulated.

FUNCTION TO BE TESTED: Flags

Definition: Flag Operators:

RESET	Resets all user flags(0-8)
FLIP FLAG	Flip flag (0-F,a-z)
IF FLAG	If flag (0-F,a-z)
INVERSE IF FLAG	If not flag (0-F,a-z)
SET FLAG	Set Flag (0-F,a-z)
INVERSE SET FLAG	Reset flag (0-F,a-z)

Tests: Each flag operator will be tried with every flag. Then, every flag operator will be tried with every other function on the keyboard. i.e. [SIN] [2nd] [STF], [2nd] [STF] [SIN].

By utilizing indirect addressing, the operand of the flag operator can be out of range. This will be tested in all modes.

The results of the above tests will be recorded when the ALEX is actually performed.

Ideally, all calculator functions should be executed after all flags are set to determine if any flag is inadvertantly reset. All functions should also be executed after all flags are reset to determine if any flag is inadvertantly set.

FUNCTION TO BE TESTED: [OP 1]

Definition: Execution of [OP 1] causes the following and only the following operations to occur.

1) Set default partition to 480 program steps.
2) Decimal number mode
3) Deactivates implied multiply except in EQN
4) Set default pause timing to 1.5 seconds
5) Resets partition state to hard
6) Resets the unnormalized number mode
7) DEG, INV EE, INV EGR, FLT
8) Reset flag F
9) Deactivate hierarchy
10) Deactivate indirect hierarchy
11) Clear

5/81

Tests: [OP 1] should be tested in every state as defined by the NOTE on the last page.

FUNCTION TO BE TESTED: [OP 20]

Definition: [OP 20] Save the present value for all user flags (0-B).

Tests: [OP 20] should be tested in every state as defined by the NOTE on the last page.

3/81

FUNCTION TO BE TESTED: [OP 21]

Definition: [OP 21] exchanges the values of all user flags with their values when [OP 20] was executed.

Tests: [OP 21] should be tested in every state as defined by NOTE on the last page. It will also be tested with and without [OP 20] executed previously.

3/81

FUNCTION TO BE TESTED: [OP 29]

Definition: [OP 29] checks which flags are set returns "0-F" in the display to indicate the flags set.

Tests: [OP 29] should be tested in every possible state as defined by the NOTE on the last page.

5/81

FUNCTION TO BE TESTED: Clocks

Definition: The calculator has 3 clocks, a timekeeping clock and 2 processor clocks, one for the Keyboard Controller and one for the Memory Controller

Calculator on and executing - all clocks on.
Calculator on and idle - Timekeeping clock only.
Calculator off - Timekeeping clock only.

Tests: The 3 calculator states defined above will be tested using a scope.

FUNCTION TO BE TESTED: Voltage to display driver (Controlled by R-line)

Definition: Voltage is on when calculator is on
Voltage is off when calculator is off.

Tests: The 2 calculator states defined above will be tested using a scope.

FUNCTION TO BE TESTED: Low battery test

Definition: Every ten minutes, the Low Battery Indicator latch is set and the battery is checked. If low, a message should be returned to the user immediately if idle or as soon as the calculator is turned on if the battery was found to be low while the calculator was off. If the test was performed during execution, the message should be returned when execution stops.

Tests: A low battery will be simulated to determine if the calculator responds correctly in the execution, idle and off states.

NOTE: Each of the tests below should be performed once with all of the variables set and once with all of the variables reset.

1) Manual calculate with HEX on
   Program execution in main memory with HEX on
   Program execution in CRAM with HEX on
   Program execution in CROM with HEX on

2) Manual calculate with HIER on
   Program execution in main memory with HIER on
   Program execution in CRAM with HIER on
   Program execution in CROM with HIER on

3) Manual calculate with implied multiply on
   Program execution in main memory with implied multiply on
   Program execution in CRAM with implied multiply on
   Program execution in CROM with implied multiply on

4) Manual calculate with EQN on
   Program execution in main memory with EQN on
   Program execution in CRAM with EQN on
   Program execution in CROM with EQN on

5) Manual calculate with ALPHA on
   Program execution in main memory with ALPHA on
   Program execution in CRAM with ALPHA on
   Program execution in CROM with ALPHA on

6) Program execution in main memory in CUE mode
   Program execution in CRAM in CUE mode
   Program execution in CROM in CUE mode

7) Program execution in main memory in R/S mode
   Program execution in CRAM in R/S mode
   Program execution in CROM in R/S mode

OP CODES 9, 14, 15

1. USE ALPHA MODE IN LEARN AND EQUATION MODE. THE ALPHA MODE SHOULD BEHAVE EXACTLY AS IT DOES IN MANUAL CALCULATE MODE.

2. FORCE AN ERROR CODE (I.E. 1/0=) EDIT THE ERROR MESSAGE USING CLR OP 09 ALPHA DO THE SAME FOR A USER GENERATED MESSAGE. IN EITHER CASE OP 09 SHOULD RESTORE THE MESSAGE, ALPHA SHOULD RESTORE THE CURSOR AND ANY ALPHA KEYS SHOULD BE ENTERED TO THE DISPLAY. 5/81

3. TRY STORING ALPHA AND MIXED ALPHA-NUMERIC DATA INTO BOTH USER AND HIER REGISTERS. IN ALL CASES ONLY THE NUMERIC PORTION OF THE DATA SHOULD BE STORED. IN THE CASE WHERE ONLY ALPHA DATA IS BEING DISPLAYED, (MORE THAN 5 CHARACTERS ENTERED TO THE DISPLAY), THE NUMERIC PORTION SHOULD STILL BE STORED. REPEAT THE ABOVE PROCEDURE BUT PERFORM ARITHMETIC FUNCTIONS ON THE DATA. 5/81

5. SEE HOW THE ALPHA MODE INTERACTS WITH THE UNNORMALIZED MODE. WITH AN ALPHA DISPLAY, PUT THE UNIT INTO UNNORMALIZED MODE AND TRY TO STORE THE DATA TO AN USER AND A HIER REGISTER. ONLY NUMERIC DATA SHOULD BE STORED. 5/81

6. ENSURE THAT THE ~~DEFINE~~ INV KEY WORKS WITH THE ALPHA OP CODES. 5/81

~~7. ENSURE THAT THE INVERSE OP CODES ARE IGNORED.~~

8. ENSURE THAT OP14 AND OP 15 DO RIGHT AND LEFT CIRCULAR SHIFTS. DO MANY SHIFTS. ENSURE THAT THE MESSAGE CAN BE EDITED AFTER ANY NUMBER OF SHIFTS. 5/81

9. INSURE THAT ALL ALPHA KEY PRESSES PRODUCE CORRECT DISPLAY CHARACTER. INCLUDE CHARACTERS THAT CANNOT BE ACCESED FROM THE KEYBOARD. 3/81

10. INSURE THAT THE SHIFT KEY WORKS. THAT NO OTHER KEY PRESSES OR OPERATIONS CLEAR OUT OR SET THE SHIFT FLAG. THAT KEYS WITH NO SHIFT CHARACTER ARE IGNORED WHEN THE SHIFT FLAG IS SET. 5/81

OP CODES 16,17

1. EXECUTE THESE OP CODES FROM LEARN MODE, ~~EQUATION MODE~~, AND PROGRAM MODE. NOTE RESULTS.

2. IN EACH MODE ATTEMPT TO CHANGE FROM HEX TO DEC IN THE MIDDLE OF A PENDING OPERATION. I.E. SIN OP 16. I WOULD GUESS THIS SHOULD PRODECE AN ERROR CODE AND GIVE THE INVALID SEQUENCE MESSAGE.

3. IN HEX MODE TRY ALL OP CODES AND OPERATIONS. ALL MATHEMATICAL OPERATIONS SHOULD BE IGNORED. ALL OPCODES SHOULD WORK, AS WELL AS ALL MEMORY OPERATIONS. PAY PARTICULAR ATTENTION TO OP CODES THAT CHANGE MODES. I.E. ENSURE THAT UNORMALIZED HEX NUMBERS CAN BE STORED AND RECALLED. SET BITS AND DIGITS FROM UNORMALIZED HIER HEX MODE.

4. NOTE HOW THE ALPHA MODE WORKS IN EACH MODE. IN HEX MODE AN A, B, C, D, E, AND F SHOULD BE ENTERED AS NUMBERS UNLESS THE ALPHA KEY IS DOWN. NOTE HOW MESSAGE EDITING WORKS IN HEX MODE. LETTER/NUMBER KEYS SHOULD AGAIN BE TREATED AS NUMBERS UNLESS THE ALPHA FLAG IS SET. A SMALL A, B, C, D, E, OR F SHOULD BE TREATED AS A LETTER. 3/81

5. ENSURE THAT LABELS ARE FUNCTIONAL IN HEXMODE. WRITE A PROGRAM BEGINNING WITH LABELS A->F. FROM THE HEX MODE CHECK THAT THE SEQUENCES; SBR A, GTO A, ETC. PLACE THE PROGRAM COUNTER AT THE END (R/S OR RETN) OF THE APPROPRIATE SUBROUTINE. 5/81

6. ENSURE THAT GOING FROM HEX MODE TO DEC MODE DOES NOT DESTROY OR MODIFY USER FLAGS, SYSTEM FLAGS ETC. 5/81

7. ENSURE THAT ALL MEMORY OPERATIONS (STO RCL EXC IND HIER) WORK USING HEX NUMBERS. TRY TO RECALL A NUMBER STORED IN HEX, AFTER CHANGING BACK TO DEC MODE, THE NUMBER SHOULD COME BACK AS A HEX NUMBER. THEN TRY PERFORMING ARITHMETIC OPERATIONS ON THIS NUMBER. THE OPERATION SHOULD BE PERFORMED. NOTE THE RESULTS FOR SEVERAL HEX VALUES OF VARYING SIZE (I.E ONE DIGIT THROUGH 16 DIGITS), AND ENSURE THE UNIT DOES NOT LOCK UP. 5/81

PRODUCT X ALEX PROCEDURE — JOSE FERNANDEZ — X-3498
HIERARCHY MODE, BIT AND DIGIT OPERATIONS — 07/08/80

80

INSTRUCTIONS: SBIT, RBIT, FBIT, TBIT, INV TBIT, HIER , INV HIER, INDH, INV INDH, STOH, RCLH.

FOR ALL OPERATIONS NOTE IF INSTRUCTIONS ARE BEING EXCECUTED, OR IF THE INSTRUCTION CAUSES THE ALGORITHM TO GET LOST.

5/81 1. SET BITS 0, 1, 2, AND 3 IN AS MANY DIGITS OF AS MANY REGISTERS AS POSSIBLE . FOLLOW THE SET BIT INSTRUCTION BY A TESTBIT INSTRUCTION. THE BRANCH SHOULD BE EXECUTED.

5/81 2. RESET BITS 0, 1, 2, AND 3 IN AS MANY DIGITS OF AS MANY REGISTERS AS POSSIBLE. FOLLOW THE RESET BIT INSTRUCTION BY AN INVERSE TEST BIT INSTRUCTION. THE BRANCH SHOULD BE EXECUTED.

6/81 3. TOGGLE BITS 0, 1, 2, AND 3 IN AS MANY DIGITS OF AS MANY REGISTERS AS POSSIBLE. IF THE BIT IS FLIPPED TO A ONE, FOLLOW BY AN INVERSE TEST BIT INSTRUCTION; IF THE BIT IS FLIPPED TO A ZERO, FOLLOW BY A TEST BIT INSTRUCTION. THE BRANCH SHOULD BE SKIPPED.

5/81 4. STORE DIGITS IN AS MANY DIGITS OF AS MANY REGISTERS AS POSSIBLE. RECALL THE DIGIT AND ENSURE THAT IT IS THE SAME DIGIT THAT WAS STORED.

5/81 5. ENSURE THAT SETTING THE HIER MODE CAUSES STO AND RCL TO WORK AS STOH AND RCLH, AND THAT RESETING THE HIER MODE RESTORES THESE INSTRUCTIONS.

5/81 6. ENSURE THAT INDIRECT ADDRESSING WORKS IN THE HIER MODE. STORE INDIRECT AND RECALL INDIRECT IN AS MANY HIER REGISTERS AS POSSIBLE.

5/81 7. STORE UNFORMATTED, AND HEX NUMBERS. BOTH SHOULD BE STORED AND RECALLED IN CORRECT FORMAT DEPENDING ON THE MODE. REPEAT WITH HEX DIGITS. TRY TO STORE ALPHA DISPLAY USING STOH.

5/81 8. ENSURE THAT THE EXCHANGE KEY PERFORMS EXCHANGES BETWEEN HIERARCHY REGISTERS IF THE HIER MODE IS SET.

6/81 9. ENSURE THAT THE 2ND CMS KEY DOES NOT CAUSE THE ALGORITHM TO BOMB IF THE HIER MODE IS SET. I AM NOT SURE AS TO HOW MANY HIER REGISTERS, IF ANY, THIS KEY IS SUPPOSED TO CLEAR.

10. PRECEDE EACH OF THE ABOVE OPERATIONS WITH AS MANY DIFFERENT OPERATIONS AS POSSIBLE.

11. FOLLOW EACH OF THE ABOVE OPERATIONS WITH AS MANY DIFFERENT OPERATIONS AS POSSIBLE.

6/81 12. TRANSFER THE CONTENTS OF USER MEMORY TO HIER MEMORY AND BACK AGAIN USING RCL AND STOH. THE CONTENTS OF THE REGISTERS SHOULD BE IDENTICAL AFTER THIS SEQUENCE.

5/81 13. INSURE THAT THE INDIRECT KEY WORKS PROPERLY WITH THE STO AND RECALL WHEN THE HIER KEY IS SET. SEE THAT THE "INCORRECT SEQUENCE" MESAGE RESULTS IF IMPROPER KEY SEQUENCES SUCH AS (EXC 2ND IND 99999) IN THE HIER MODE.

OP CODE 21.

SEE ACCOMPANYING LITERATURE.

1. RUN THE TEST UNDER NORMAL CONDITIONS, INSURE IT WORKS

2. FORCE EACH PART OF THE TEST TO FAIL. FOR THIS I WILL NEED ALICE'S HELP SINCE I DONT KNOW HOW TO OPERATE THE SIMULATOR. IDEALLY, THE FAILURES WILL BE FORCED BY; JAMMING AN UNEXPECTED INSTRUCTION IN THE ARITHMETIC TEST, JAMMING AN UNEXPECTED VALUE TO USER MEMORY, JAMMING AN UNEXPECTED VALUE INTO THE HIERARCHY REGISTER. 3/81

3. SEE IF THE SELF TEST OPERATES FROM LEARN MODE, ~~OR EQUATION MODE.~~ SEE IF SELF TEST IS AFFECTED BY HEX MODE, ALPHA MODE, UNFORMATTED MODE, ETC. ENSURE THAT THE UNIT IS LEFT FULLY OPERATIONAL AFTER SELF TEST BY TRYING ARITHMETIC OPERATIONS, MEMORY OPERATIONS, TRYING TO GET TO LEARN MODE AND EQUATION MODE, ETC. 2/80

OP CODES 41, 43

3/81 1. ENSURE THAT ALL NUMBERS CAN BE STORED. PUT NUMBERS IN EE FORMAT, ENG FORMAT AND FLOATING PT FORMAT IN THE DISPLAY; HIT OP 41; STORE AND RECALL. OP 43 AND RECALL. NOTE RESULTS

2. USING THE HIER MODE, TRANSFER THE CONTENTS OF REGISTERS 45 AND 46 (ALPHA STORE REGISTERS) TO USER REGISTERS, CHANGE THE DISPLAY, THEN 5/81 RESTORE THESE REGISTERS. OP 9 OUGHT TO RESTORE INITIAL DISPLAY. TRANSFER THE CONTENTS OF THE USER REGISTERS TO THE HIER REGISTERS. OP 9 SHOULD NOW SHOW THE DISPLAY AS BEING ALTERED.

3. PRECEDE AND FOLLOW AN UNNORMALIZED DISPLAY WITH ALL KEYS. NOTE RESULTS ALL ARITHMETIC OPERATIONS SHOULD BE IGNORED, ALL MEMORY OPERATIONS SHOULD BE VALID, ALL OPCODES SHOULD BE VALID.

5. SEE WHAT HAPPENS FROM LEARN AND EQN MODES. THEY SHOULD BEHAVE AS MANUAL CALCULATE MODE.

6. SEE IF ALPHA MODE WORKS WITH UNORMALIZED MODE. PUT AN ALPHA MESSAGE IN THE DISPLAY THEN GET TO UNNORMALIZED MODE. THE DISPLAY SHOULD NOW CONTAIN THE 3/81 UNNORMALIZED CONTENTS OF THE LAST VALID NUMERIC ENTRY. IF THE DISPLAY CONTAINS MIXED ALPHA-NUMERIC DATA, ONLY THE NUMERIC PART SHOULD BE UNNORMALIZED. WHEN THE DISPLAY IS NORMALIZED, SEE IF THE ALPHA PART IS RESTORED.

7. TRY UNNORMALIZED MODE AND THEN ALPHA OR ALPHA NUMERIC DATA. AFTER A VALID OPERATION KEY ENSURE THAT ONLY NUMERIC DATA IS UN-NORMALIZED. TRY ALL MEMORY OPERATIONS AND ENSURE ONLY NUMERIC DATA IS STORED.

UNIVERSAL PLANNING FORM

SBR SBL GTO GTL RTN — need to add in from PGM MEMORY SST

HEX, OCT, DEC

VALID TRANSFERS (LBL / ABS)(DIR / IND)

		0	1	2	3	4	5	6	7	8	9
1	0	LBL	A	1	SBR	C	SBL	C	GTO	B	LBL
2											
3	1	C	+	1	=	RTN	LBL	B	+	1	=
4											
5	2	GTL	C	LBL	D	GTO	00	00	LBL	01	00
6											
7	3	STO	A	GTO	LCA	LBL	2	1	STO	A	GTL
8											
9	4	LCA	LBL	3	SBR	00	00	RTN	LBL	4	1
10											
11	5	STO	A	SBL	LCA	RTN	LBL	5	0	STO	A
12											
13	6	SBR	LIA	RTN	LBL	6	0	STO	A	GTO	LCA
14											
15	7	RTN									

		DISP.
16		
17	PRESS SBR A	5
18	GTO A R/S	11
19	SBR D	11
20	SBL 01	11
21	SBL 02	11
22	SBL 03	11
23	" 04	11
24	" 05	11
25	" 06	11

26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41

OPERATING UNIT: TI-3145-A

PREPARED BY:

DATE:

ALEX CHECK OUT OF PRINTER FUNCTIONS FOR PRODUCT X.
SUSAN BAILEY, JULY 14, 1980.
DOCUMENTATION OF THE ALEX IS IN CD1.ALEX.TI88.PRINTER

1. GENERAL PRINTER FUNCTIONS ARE:

   A. TRACE ON, TRACE OFF
   B. 2ND LST
   C. INV 2ND LST
   D. 2ND PRT
   E. 2ND ADV
   F. HEIR INV 2ND LST
   G. OP 00 (LIST OP CODE DEFINITIONS)
   H. OP ~~53~~ (LIST PGM LABELS)
   I. OP ~~47~~ 15, OP ~~43~~ 14 (NORMAL AND UNNORMAL #'S)

OP 02 display cale settings
OP 08 Alph entry table
OP 18 flag definitions

2. CRITICAL MODES TO CHECK FOR THE PRINTER FUNCTIONS ARE:

   A. MANUAL CALCULATE MODE
   B. PROGRAM RUN MODE AND CROM RUN MODE
   C. PROGRAM SINGLE STEP MODE
   D. EE, ENG, FIX POINT AND UNNORMALIZED #'S
   E. EQUATION MODE
   F. USING ANY OF THE ABOVE MODES
      IN A PROGRAM, WHILE TRACING, ETC.

3. THE FOLLOWING THINGS SHOULD BE CHECKED FROM A PROGRAM.

   5/81 A. SETTING AND RESETTING FLAGS C AND D.
   ~~5/8~~ 3/81 B. EXECUTING ALL PRINTER FUNCTIONS FROM
      A PROGRAM.
   5/81 C. TRACING ALL THE KEYS FROM A PROGRAM.
      CHECK THE AUDIT TRAIL.
   ~~5/8~~ D. TURN TRACE ON AND OFF DURING EQN MODE
      WHICH IS CALLED FROM A PROGRAM.

4. THE FOLLOWING THINGS SHOULD BE CHECKED IN MANUAL CALCULATE MODE:

   3/81 A. TRACE, CHECKING THE AUDIT TRAIL FOR ALL KEYS
   5/81 B. 2ND PRT, VERIFYING THAT THE DISPLAY IS ALWAYS
      PRINTED AS IS, INCLUDING ALPHA MODE, AND TIME.
   3/81 C. 2ND LST, INV 2ND LST, ETC.
   3/81 D. INTERACTION OF FLAGS C AND D WITH PRINTER FUNCTIONS.
   3/81 E. INV 2ND LST AND UNNORMALIZED NUMBER MODE.
   3/81 F. ERROR CONDITION PRINTING.
   3/81 G. CHECK LST WHEN POINTING TO INCORRECT REGISTERS AND
      PROGRAM COUNTERS.
   3/81 H. REPARTITION MEMORY AND TRY INV 2ND LST, ETC.

6/81

3/81 5. THE R/S (OR RST) KEY SHOULD STOP A LST IN ALL MODES.

5/81 6. CHECK TRACE ON ALL OP'S AND ~~DFN~~ INV OP'S, INCLUDING USING BAD PARAMETERS.

5/81 7. CHECK PRINTING IN SPECIAL FORMAT MODES SUCH AS EE, ENG AND USING OP ~~47~~ 14.

5/81 8. CHECK PRINTER FUNCTIONS IN MAIN MEMORY, NAMED CRAM, EQN AND CROM.

PROD X PRINTER ALEX.
SUSAN BAILEY

AS OF JULY 18,1980

1. THE FOLLOWING ITEMS HAVE BEEN FIXED:
   RCD, STD, INV STH, STH, INV TBIT, RBIT
   FBIT, RCH, INDH, INV INDH, INV HIER,
   DAT, TIM, INV DAT, INV TIM, SIG+, SIG-
   P->R, INV P->R, FIX, AND FLT.
   OP 40 SEEMS TO WORK.

2. NEW PROBLEMS
   OP 59 TRACES DIFFERENTLY FROM OP 60 -
   DOESN'T GIVE MESSAGE.
   DEG, DRG, ETC NOW TRACE DIFFERENTLY?
   INV LST WILL NOT WORK FROM A PROGRAM
   INV ENG SBL K CONFUSES THE TRACE.
   OP 38 HAS AN S= RATHER THAN s=. OK
   MAY BE POSSIBLE THAT OP 04 GTL 10
   LOCKS UP THE CALCULATOR. OK
   ONCE AN OP 05 CAUSED AN OP 00 INSTEAD.
   DFN OP 50,52,54,56 DIDN'T LIKE BEING
   EXECUTED FROM A PROGRAM. OK
   IN UNNORMALIZED MODE PGM 201 UDA WON'T WORK. OK
   POSSIBLE PROBLEM WITH FIX 6, AS THE
   INT CAUSED 7 DIGITS TO APPEAR.

AS OF JULY 14,1980

1. THE FOLLOWING ITEMS TRACE
   RTN, SBL, PRT, INV STF, OP, YES, UNK, NO, ENT, GFR
   REG, LBL, IF=, IF>=, IF<=, IF<, IF>, IF (NE), CONT
   IFN, IFF, EVAL, STO, RCL, SBR, CMS, a-z, LOG
   LN, (,), LOWER CASE PI, N=, STF, FF, IFF
   INV IFF, INV DSZ, SWAP, FACTORIAL, MINUS 1 POWER,
   ST+, ST-, STx, ST (DIV), EXC, UP ARROW, DOWN
   ARROW, SQUARE ROOT, INV LOG, INV LN, INV EE,
   FIX, FLT, INT, FRC, DEG, RAD, GRAD, SIN,
   ARCSIN, TAN, ARCTAN, COS, ARCCOS, OP 03 (OLD
   NUMBERS, REPARTITION MEMORY), DSZ, GBR,
   INVALID ENTRY, INV LST, GTL,

2. THE FOLLOWING ITEMS DO NOT TRACE:
   INV HIER, HIER, RCLD, STOD, INV STOH, STOH, EQN, CEQ
   SIG+, SIG-, SBIT, RBIT, FBIT, VARH, INDH, INV INDH,
   INS, BST, SST, DEL, INV P->R, P->R, ADV,
   EE, ENG, INV ENG, ARITHMETIC ERRORS WHICH OCCUR
   DURING PROGRAM EXECUTION.

3. PROBLEMS ENCOUNTERRED AS OF JULY 14, 1980

   A. FLAG C WAS NOT FULLY IMPLEMENTED, AND THE OP
      CODE NUMBERS WERE IN THE PROCESS OF BEING
      MODIFIED. THEREFORE, SOME TESTING WAS
      AMBIGUOUS.

   B. IF FLAG D WAS ON, AND FLAG C WAS OFF, THE
      LAST PRT IN A PROGRAM WORKED CORRECTLY. HOWEVER
      IF FLAG C IS ON, THE LAST PRT DOES NOT EXECUTE.

   C. IF FLAG C WAS OFF, ANY PRINTER FUNCTION SUCH AS

INV LST, OR OP 00 DID NOT RETURN CORRECTLY. THE NEXT INSTRUCTION IN THE PROGRAM WAS EITHER A LBL OR SBL , AND THE PROGRAM ABORTED WITH AN "INVALID ENTRY" ERROR. THIS DID NOT HAPPEN OF FLAG C WAS ON.

D. FIX AND FLT TRACE TWICE.

E. AN OP 32 (OLD # FOR LIST PGM LABELS) WHICH WAS EXECUTED FROM A PROGRAM, LOCKS UP THE CALCULATOR IN A NON-RECOVERABLE FASHION, (A PUC IS NECESSARY).

F. INS, BST, AND DEL EXECUTED FROM A PROGRAM ARE IGNORED BUT AN SST WILL GENERATE AN "INVALID SEQUENCE" ERROR.

G. UNEXECUTED BLOCKS OF CODE ARE TRACED IF THEY FOLLOW A CONDITIONAL INSTRUCTION.

I. DURING ARITHMETIC ERRORS, THE DISPLAY FLASHES BUT TRACE IGNORES IT.

4. THE FOLLOWING THINGS HAVE BEEN TESTED.

A. THE AUDIT TRAIL FOR THE INSTRUCTIONS MENTIONED ABOVE.

B. OP 03(REPARTION MEMORY), OP 05,06(CUEING), OP 00 (OP LISTING), OP 41 (NORMALIZED, ETC) HAVE BEEN SUCCESSFULLY EXECUTED FROM A PROGRAM AND SEEM TO TRACE PROPERLY.

C. EQN WAS EXECUTED FROM A PROGRAM AND DOES NOT TRACE BUT EVAL WAS ALSO EXECUTED FROM A PROGRAM AND DOES TRACE.

D. FLAGS D AND C WERE TURNED ON AND OFF FROM A PROGRAM.

D. INV LST, AND HIER INV LST WERE SUCCESSFULLY EXECUTED FROM A PROGRAM. LST ALSO RUNS.

AS OF JULY 16,1980

1. NEW PROBLEMS WHICH WERE FOUND

A. IF CERTAIN OP CODES ARE EXECUTED WHILE IN ENG MODE, THE MESSAGES ARE MESSED UP.
FOR EXAMPLE, OP 02 WILL RESULT IN
9GM STEPS (0000-.47) AND
OP 40 RESULTS IN .002 LBL 41
WHILE OP 47 RESULTS IN 6.C 01 WHEN 6C IS
THE INSTRUCTION AT LOCATION 0. 00.

B. IF A PROGRAM IS LISTED AND THEN R/S IS HIT THE FOLLOWING SEQUENCES WILL NOT WORK.
2ND STF D' WILL RESULT IN EXECUTING AN OP 09 AS WELL AS CMS.

C. IF 9.9999999EE99 IS ENTERRED, AND A OP 13 IS PREFORMED, THE RESULT IS A 1. 00.

D. DFN REQUIRES A CONT KEY TO BE HIT, RATHER THAN

ENT.

E. AN " OP 04 GTL 10 " WHICH IS ILLEGAL CAUSED TH CALCULATOR TO LOCK UP.

F. DEL AND INS TAKE SO LONG THE USER MAY REHIT THE KEY.

G. OP 3 CE DOESN'T REALLY CE.

H. SOME PROBLEMS WITH OP 12 AND "GRAD", AND OP 30. ALSO SOME BAD ALPHA.

I. THE SEQUENCE "FLT 2. 1" CAUSE PGM 101 TO HALT. HAVEN'T TESTED THIS YET.

J. THE PRINTER TURNED ITSELF OFF (69!) AND BOTH FLAG D AND C WERE ON.

2. OLD PROBLEMS WHICH HAVE BEEN FIXED

A. THE FOLLOWING NOW TRACE : TBIT, FBIT, SBIT RBIT, HIER, INDH, RCLD, STOD, RCLH, STOH, AND ALL THE INVERSES. FIX AND FLT ONLY TRACE ONCE NOW.

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: RESET

DEFINITION: 17.88

TEST PROCEDURE:

1. In MANUAL MODE:
   a) When program is running (RAM or CROM) execution stops, resets all user flags 0-5, clears SBR - stack and page register returns control to RAM with no limitations.
   b) Stops listing
2. In RAM
   a) Resets all user flags
3. In CROM (same as 2)

5/81

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: RUN/STOP

DEFINITION: 17.86

TEST PROCEDURE:

1. In MANUAL MODE:
   a) When a program is running (RAM or CROM) to see if execution can be dropped and restarted.
   Check key-limitation if halted in CROM — changed
   Check for no limitation if halted in RAM
2. In RAM
   a) -stops execution; no limitations
3. In CROM
   a) -stops execution; the following keys are not allowed: A' - J', PGM, SBR, GTO, →, EVAL.
4. Other
   a) stops listing from: keyboard
   RAM
   CROM

Herbert Moder July 31, 1980

ALEX CHECKOUT PROCEDURE

FUNCTION: List Main Memory Program Labels

DEFINITION: 18.109 - 111

TEST PROCEDURE:

1. In MANUAL MODE

   a) OP 40 - Lists labels of main memory

      - Starts at PC and ends at end of partitioning

   b) OP 40 - Displays 0 if executed at end of partitioning, or with no program steps, or with no labels, does not list illegal labels. 5/81

   c) OP 40 - Print labels if printer on

      - Pauses labels at pause timing if printer off

      - RST and R/S halt the list.

2. In RAM (same as 1.) 7/80

3. In CROM

FUNCTIONS: ERROR MESSAGES

PRIMARY KEYS:

OP CODES: OP8

TEST PROCEDURE:

A. TEST ALL OF THE FIFTEEN (15) ERROR GROUPS AND EACH ERROR IN COMBINATION WITH ANOTHER ERROR IF POSSIBLE.

B. TEST TO SEE THAT THE ERROR IS PROPER AND THAT THE ERROR AND MESSAGE NUMBERS ARE RECALLED EVEN AFTER OTHER FUNCTIONS ARE DONE THAT DO NOT CAUSE OTHER ERRORS.

C. PRODUCT X ERROR CODES:

1. UNDERFLOW/OVERFLOW -- RESULT OF ANY OPERATION WHICH IS NOT BETWEEN 9.999999999999999 X 10^99 OR -9.999999999999999 X 10^99; OR NUMBERS BETWEEN 1 X 10^-99 AND -1 X 10 -99 EXCEPT ZERO. DIVIDE BY ZERO.
2. EOS OVERFLOW -- WHEN ANY OF THE EOS STACKS ARE OVERFLOWED FOR EXAMPLE: LEFT PARENTHESES EXCEDES 17.
3. INVALID ARGUMENT -- THE ARGUMENT OF A FUNCTION IS OUT OF THE LOGICAL RANGE. EX INV SIN 10 , SQUARE ROOT OF A NEGATIVE NUMBER. LNX OF A NEGATIVE NUMBER. TAN 90 DEG.
4. INVALID SEQUENCE -- ANY FUNCTION THAT SHOULD HAVE AN ARGUMENT FOLLOWING IT BUT IS FOLLOWED BY ANOTHER FUNCTION.
5. INVALID ADDRESS -- ANY PROGRAM STEP PAST 9999 WHICH IS REFERENCED ABSOLUTELY. EXECDING PARTITION LIMIT.
6. LABEL NOT FOUND -- ANY REFERENCE TO AN UNDEFINED LABEL IN MAIN, CROM, OR CRAM PROGRAM.
7. ~~CUE ERROR~~ NO PERIPHERAL -- WHEN WAITING FOR A CUE OR R/S TRYING TO MOVE THE PROGRAM COUNTER RESULTS IN THIS ERROR.
8. MODULE NOT FOUND -- MODULE IS NOT INSTALLED IN CALCULATOR.
9. ~~NOT ENOUGH ROOM~~ INADEQUATE SPACE -- THERE IS NOT ENOUGH ROOM TO UPLOAD OR DOWNLOAD THE DESIRED PROGRAM. PARTITION TO SMALL OR AVAILABLE SPACE IS ALREADY IN USE.
10. INVALID ENTRY -- NAMING A CRAM WITH SOMETHING BESIDES A POSITIVE INTEGER LESS THAN 100. STORING A NUMBER > 15 IN A STOD COMMAND.
11. ~~OUT OF RANGE -- OP TO LARGE (376)~~ INVALID PGM #
12. INVALID MODE -- OP16 FOLLOWED BY OP13 ETC.
13. INADEQUATE DATA -- TRYING TO DO A STATISTICS PROBLEM WITHOUT ANY POINTS ENTERED. TRYING TO DO STATISTICAL PROBLEMS WHICH REQUIRE TWO OR MORE POINTS WITH ONLY A SINGLE DATA POINT (Y GIVEN X ,ETC).
14. SBR STACK FULL -- GOING MORE THAN 10 LEVELS DEEP IN SUBROUTINES.
15. INVALID REGISTER -- ACCESSING A REGISTER OUTSIDE OF THE VALID RANGE OR OUTSIDE THE PARTITION. IN HIER > 62 .

D. REPETE A,B,C FOR PROGRAM MODE.

E. FLAG F OPERATION.

1. TRY EACH ERROR WITH THIS FLAG SET.
2. TRY EACH ERROR WITH THIS FLAG RESET.
3. BE SURE TO DO THIS IN PROGRAMMING MODE AND NOTE THAT ERRORS 1 2 AND 3 ONLY HALT EXECUTION WHEN THE "F" FLAG IS SET. ALL OTHER ERRORS SHOULD CAUSE THE PROGRAM TO HALT UNCONDITIONALY.

F. MAKE SURE FLAG E IS SET AFTER EACH ERROR.

G. LIST ALL THE WAYS IN WHICH THE ERRORS (LISTED IN C) CAN BE GENERATED

FUNCTION TO BE TESTED: Power

Definition: Simulate Battery Insertion
The Keyboard Controller chip expects a "NON A" value in order to perform a battery PUC. The Memory Controller chip however, expects a "NON 95" in order to perform a battery PUC.

Keyboard Controller PUC	Memory Controller PUC
Return to degrees	Cancel beep on error
Pause = 1.5 seconds	Cancel beep on cue
Implied multiply off	Display LEARN PC
Fix off	Cancel soft partition
Clear timekeeping	Reset user flags (0-B)(both sets)
Return to decimal	Partition 480 pgm steps
Return to normalize	Clear statistics
Cancel ALPHA	Clear user program(MM)
Time to numeric display	Clear data memories(MM)
Clear key input buffer	Reset status flags(C-F)(both sets)
INV ENG & INV EE	Clear PC
Time to ALPHA register	Return to decimal
	Cancel HIER & INDH
	Cancel EQN
	Cancel LEARN mode
	Cancel program execution
	Cancel R/S & CUE
	Cancel PGM
	Cancel CROM read
	Cancel CRAM read/write

8/80

Tests: For each of the three tests listed below, all values will be set.

A) Have the Keyboard Controller chip perform a battery PUC.
B) Have the Memory Controller chip perform a battery PUC.
C) A & B will be performed simultaneously.

FUNCTION TO BE TESTED: Key Presses (Not Key Sequences)

Definition: Check Key Rollover
The following key presses: A "2", then pressing "3" while the "2" is still down, and finally letting up on the "2" should cause a "2" and a "3" to be entered in the display (Test 1). However, a "2" then pressing "3" while the "2" is still down, and finally letting up on the "3" should not cause two "2"s to be entered in the display (Test 2).

3/81

Tests:

Key Presses		Expected Result
1) CLR 2 2&3 3		23
2) CLR 2 2&3 2		2
3) CLR 2 2&3 2&3&4 2&3 3		23
4) CLR 2 2&3 2&3&4 2&4 4		24
5) CLR 2 2&3 2&3&4 2&3 2		2
6) CLR 2 2&STO STO	STO	[2]
7) CLR 2 2&5 2&5&STO 5&STO STO	STO	[2]
8) CLR 1 1&5 5		15
9) CLR 1 1&5 1&5&9 1&9 9		19

Definition: Key Buffering (Not display buffering)
Key buffering is the ability of the calculator to store up to 15 key presses while the calculator is performing some function.

Tests: [OP 49] will be used to keep the calculator busy since it takes about 5-10 seconds to execute. The trace function should be used in order to verify that all keys are executed.

3/81

Key rollover will also be checked during PAUSE and trig operations since a different key scan is used.

Key Presses while busy	Expected Result
1) 1234567890123456789	123456789012345
2) INV 18 times	1INV INV INV INV
3) 2+++++++++++++++++	?

Definition: Key Misses.
A missed key is any missed clean key press which does not overflow the display buffer or key buffer.

Tests:
A) 10 rapid presses of various keys while idle. Are any lost?
B) 10 rapid presses of various keys while busy. Are any lost?
C) Be on the look out for any key which is clearly pushed but is ignored.

Definition: Multiple Key Entry.
A multiple key entry is any single key press which is processed as more than one key press.

Tests:
A) Hold a key down for an extended period of time.
B) Push keys slowly, at angles, rocking on key, etc. Determine if any key is entered on release as well on first push.
B) Be on the look out for any key which when pressed clearly a single time, is processed multiple times or causes some other unexpected result.

FUNCTION TO BE TESTED: Continuous Memory

Definition: Continuous Memory refers to those values which are retained, saved or maintained after turning the calculator off and then back on.

SAVED	NOT SAVED OR RESET
DRG	? User & status flags (0-F) (except C & pointers in Rm)
Partition	PC goes to 0 (Main memory)
Pause timing	Return to decimal
Implied multiply	Return to normalize
Fix →	Cancel HIER & INDH
~~Display LEARNING~~	Cancel ALPHA & EQN
Soft partition →	~~Time to numeric display~~
Statistics	Clear key input buffer
Timekeeping	INV ENG & INV EE
User program	~~Time to ALPHA register~~ A&G
Data memories	Cancel LEARN mode
Beep on Error	Cancel program exection
Beep on CUE	Cancel R/S and Cue
	Cancel Pgm
	Cancel CROM read
	Cancel CRAM read/write

3/8,

Tests: Continuous Memory should be tested in every state as defined by the NOTE on the last page.

8/80 1. ENTER NUMBERS CONTAINING FROM ONE TO 16 DIGITS. IF MORE THAN 13 DIGITS ARE ENTERED, THE TRAILING DIGITS SHOULD BE TRUNCATED. REPEAT THIS TEST USING EE AND ENG FORMAT, FOR BOTH INTEGER AND NON INTEGER VALUES.

8/80 2. ENSURE THAT UPON NORMALIZATION, THE LAST DIGIT DIPLAYED IS CORRECTLY ROUNDED. ENTER VARIOUS COMBINATIONS FOR THE 3 GUARD DIGITS AND INSURE THAT THE CARRY IF ANY IS PROPAGATED CORRECTLY.

3/81 3. ENSURE THAT ALL KEY ENTRIES OTHER THAN NUMERIC KEYS, DECIMAL POINT, CHANGE SIGN, LEARN KEY, R/S KEYS, AND 2ND FUNCTION KEYS NORMALIZE THE DISPLAY.

5/81 4. ENSURE THAT TRAILING ZEROES AND LEADING ZEROES ARE ELIMINATED ON NORMALIZATION.

5/81 5. ENSURE THAT LIVE ENTRIES BEHAVE THE SAME IN LEARN AND EQUATION MODE AS THEY DO IN MANUAL CALCULATE MODE.

5/81 6. ENSURE THAT THE EE AND ENG MODES ARE DEAD ENTRIES UNLESS IN THE MIDDLE OF A LIVE ENTRY. I.E. THE SEQUENCE 2=EE39 SHOULD LEAVE A 39 IN THE DISPLAY, WHEREAS THE SEQUENCE 2EE39 SHOULD LEAVE A 2 39 IN THE DISPLAY.

7. TRY ENTERING SEVERAL DPT'S AND CHG SGN KEYS BEFORE OR AFTER THE EE KEY. ONLY ONE DPT SHOULD BE RECOGNIZED BEFORE THE EE KEY, NONE AFTER THE EE KEY. A CHG SGN KEY BEFORE THE EE KEY SHOULD CHANGE THE SIGN OF THE TISSA MANTISSA, AFTER THE EE KEY SHOULD CHANGE THE SIGN OF THE EXPONENT. REPEAT THIS PROCEDURE FOR THE ENG MODE.

5/81 8. TRY ENTERING MULTIPLE EE AND ENG KEYS. THE FIRST SHOULD CAUSE THE UNIT GO INTO GO TO EE OR ENG FORMAT, ALL OTHERS SHOULD BE IGNORED.

5/81 9. ENSURE THAT ENG AND EE MODES CAN BE CLEARED AFTER AN INV EE OR INV ENG KEY SEQUENCE. NOTE THAT THE INV EE SEQUENCE SHOULD NOT CLEAR THE ENG MODE HOWEVER THE INV ENG SEQUENCE SHOULD CLEAR EITHER THE ENG OR THE EE MODE. INSURE THAT INV EE OR INV ENG ARE NOP'S WHEN IN THE FLOATING POINT MODE.

3/81 10. ESTABLISH THE UPPER AND LOWER LIMITS OF THE FLOATING POINT DISPLAY. ANY NUMBER LARGER THAN 10 DIGITS TO THE LEFT OF THE DECIMAL POINT SHOULD CAUSE THE DISPLAY TO GO TO EE MODE. ANY FRACTION SMALLER THAN 4 DIGITS TO THE RIGHT OF THE DECIMAL POINT (I.E. .0000X) SHOULD CAUSE THE DISPLAY TO GO TO EE MODE.

T E S T 1 9

FUNCTIONS:
MEMORY MANIPULATION/EXPANSION

PRIMARY KEYS:
OP, STO, RCL, LRN, R/S, RST, -->, <--, IND

OP CODES:
OP 3, 41, 44, 45, 72, 73, 74

TEST PROCEDURE:

A. WRITE A PROGRAM FOR MAIN MEMORY WHICH WILL STORE THE REGISTER # INTO THAT REG. I.E. REG 3 HAS CONTENTS OF '3'. THIS PROGRAM ALSO IS WRITTEN TO CHECK PGM STEPS, SO IT SHOULD BE WRITTEN TO JUMP IN INCREMENTS OF '8'.

3/81

1.

VERIFY THAT UPON POWER-UP, DEFAULT MODE = REGISTERS = 0-25, PGM STEPS = 0-479 (WITH NO EXTRA MEMORY CRAMS). POWER UP WITH ALL POSSIBLE NUMBER OF CRAMS, AND VERIFY DEFAULT PARTITIONING.

5/81

2

BEGIN BY PARTITIONING PGM STEPS TO 0, USING OP 3. VERIFY THAT THE PARTITION SHOWN IN DFN OP 3, IS VALID BY RUNNING PROGRAM MENTIONED ABOVE.

'INVALID REGISTERS' SHOULD APPEAR WHEN REGISTER BOUNDARY HAS BEEN EXCEEDED.

5/81

TO CHECK PROGRAM STEP BOUNDARIES, LIST THE PROGRAM.

THE LIST WILL STOP WHEN PROGAM PARTITION EXCEEDED.

DOCUMENT RESULTS ON ALL PARTITIONS.

2A:

AT ODD INTERVALS DURING YOUR PARTITION SEQUENCING, EXECUTE A 'CMS' WHICH SHOULD CLEAR ALL DATA REGISTERS AND LEAVE PROGRAM STORAGE ALONE. VERIFY THIS.

2B:

USE THE 'RCL' XXX(A-Z) AS MUCH AS POSSIBLE WITHIN BOUNDARIES AND EXCEEDING BOUNDARIES.

5/81

TRY SOME 'RCL' FUNCTIONS USING SHORT-FORM ADDRESSING. I.E. RCL, 3, =; INSTEAD OF RCL 003.

2C:

5/81

VERIFY THAT CALCULATOR PLUS ONE MODULE HAS MAX STORAGE OF 268 DATA REGISTERS OR 2144 PGM STEPS. CALCULATOR PLUS TWO MODULES HAS 416 DATA REGS OR 3328 PGM STEPS.

2D:

5/81

PARTITION PGM STEPS WITH VALUES THAT ARE MULTIPLES OF '8'; VERIFY THAT WHEN A NUMBER IS ENTERED THAT IS NOT A MULTIPLE OF '8', THE CALCULATOR ROUNDS TO THE NEXT ~~LOWEST~~ HIGHEST MULTIPLE OF '8'.

3

5/81

SET PARTITIONING IN INCREMENTS OF '25' AND (USING OP 45) SET 'SOFT' PARTITIONING. USING LIMITS THAT EXCEED PRESET PARTITIONED VALUE, VERIFY THAT 'SOFT' PARTITIONING IS FUNCTIONAL. UPON VERIFICATION, EXIT 'SOFT' PARTITIONING USING 'INV OP 40' AND THEN DFN OP 3. PARTITION SHOULD BE SET TO SAME VALUE AS BEFORE ENTERING 'SOFT' PARTITIONING.

4

IN CALCULATOR MODE:

TRY 25 INDIRECT ADDRESSING OF REGISTERS.

5/81

TRY 25 INDIRECT ADDRESSING OF PROGRAM STORAGE. SOME OF THESE SHOULD TRY TO ACCESS MEMORY OUTSIDE OF CALUCATOR LIMITS, SOME SHOULD TRY ACCESSING MEMORY OUTSIDE OF PARTITIONED RANGES.

BE SURE AND USE A-Z FOR REGISTERS 0-25 AS MUCH AS POSSIBLE.

5

5/81 REPEAT #4 ABOVE ONLY WRITE IN PROGRAM MODE.

6

5/81 VERIFY THAT REGISTER CONTENTS PARTITIONED INTO PROGRAM STORAGE AND BACK TO REGISTER STORAGE ARE THE SAME.

7

5/81 LOAD ALL RANGES OF CROMS/CRAMS AVAILABLE, AND VERIFY THAT UPON POWER-UP, THE ALGO SEES THE CORRECT NUMBER AND DISPLAYS THE DEFAULT PARTITIONING.

8

5/81 VERIFY THAT THE MAXIMUM ABSOLUTE ADDRESS IS '9999' AND THE MAXIMUM REGISTERS THAT CAN BE ADDRESSED WITHOUT INDIRECT ADDRESSING IS '999'.

9

5/81 DO MEMORY ARITHMETIC USING STO+, STO-, STO*, STO/.

9A:

5/81 TRY A WIDE VARIETY OF VALUES IN MANY COMBINATIONS OF REGISTERS (0-999).

9B:

X VERIFY THE MEMORY ARITHMETIC USING INDIRECT ADDRESSING. THIS MUST BE USED TO VERIFY REGISTER MANIPULATION BEYOND '999'.

9C:

CHECK FOR UNDERFLOWS AND OVERFLOWS USING EACH MEMORY FUNCTION. DOCUMENT WHAT REGISTER IT OCCURED ON, WHAT THE VALUE WAS BEFORE IT UNDER/OVERFLOWED, AND WHAT WAS IN THE REGISTER AFTER THE ERROR OCCURED.

10

5/81 USING DIFFERENT VARIATIONS OF CROMS/CRAMS, COPY INTO MASTER SLOT (OP 74). VERIFY THAT USER MEMORY IS REPARTITIONED TO ELIMINATE CRAM IN MASTER SLOT.

10A:

5/81 USING OP 41, VERIFY THE NUMBER OF THE MODULE IN THE MASTER SLOT.

11

5/81 USING OP 73, NAME CRAMS USING VALID AND INVALID DISPLAY NUMBERS I.E. FRACTIONS, LARGE #'S, NEGATIVE #'S...

12

5/81 USING OP 72, UNNAME CRAMS AND VERIFY THAT ALL NAMED CRAMS ARE CLEARED. TRY CASES THAT WILL CAUSE USER MEMORY TO BE REPARTITIONED.

TI-88 ALEX CHECKOUT PROCEDURE

ENTRIES INTO EOS AND HIERARCHY PRECEDENCE

CHECK EVERY COMBINATION OF OPERATOR PAIRS TO SEE THAT THE CORRECT HIERARCHY PRECEDENCES ARE MAINTAINED BY OBSERVING EOS REGISTERS FOR EACH PAIR AND DISPLAYED RESULT AFTER STRING.

REPEAT THIS IN MANUAL AND SST MODES. STRING RESULTS FOR PROGRAM (BOTH WITH AND WITHOUT LRN PC) AND PROGRAMMED EVAL MODES AND ALSO CHECK INTERMEDIATE RESULTS BY USING PAUSES WITH THE TIMING SET LONG.

WHOLE TEST SHOULD BE DONE ONCE FOR IMPLIED MULTIPLY IN EFFECT AND REPEATED FOR IMPLIED MULTIPLY NOT IN EFFECT.

IN ALL OF ABOVE TESTS, A LARGE VARIETY OF NUMBER TYPES SHOULD BE USED FOR THE OPERANDS: COMBINATIONS OF POSITIVE AND NEGATIVE MANTISSAS AND EXPONENTS, +/-0, +/-99 EXPONENT NUMBERS, LIVE AND DEAD ENTRIES, EE (WITH AND WITHOUT INV EE) AND ENG NUMBERS, VARIOUS FIXES, AND CE AND CLR ZEROS.

LEARN MODE

ALL KEYS, OPERATION PLUS FIELD COMBINATIONS, AND MERGED SEQUENCES SHOULD BE ENTERED TO SEE SEE IF THEY DISPLAY CORRECTLY AND EXECUTED TO SEE THAT THEY REALLY ENTERED THE SYSTEM CORRECTLY.

ALL ALPHA CHARACTERS SHOULD BE ENTERED WITH THE ALPHA STATE BOTH IN AND OUT OF PHASE WITH THE ALPHA STRING STATE AND EXECUTED TO CHECK THEIR OPERATION.

5/81

THE ABOVE TESTS SHOULD BE REPEATED WITH AND WITHOUT THE LEARN MODE PC, WITH DIFFERENT PARTITIONINGS, HARD AND SOFT PARTITION, AND IN VARIOUS PARTS OF THE PROGRAM MEMORY, ESPECIALLY WITH RESPECT TO THE DIFFERENT RAMS (INTERNAL, MASTER SLOT, SLAVE SLOT, AND MEMORY EXPANSION MODULES) AND THE PARTITION.

ALL OF THE COMPOUND SEQUENCES ABOVE SHOULD BE TESTED ACROSS RAM BOUNDARIES AND THE PARTITION.

THE FOUR EDIT KEYS (-->, <--, INS, AND DEL) SHOULD EACH BE TESTED VARIOUSLY WITH AND WITHOUT THE LEARN MODE PC, WITH DIFFERENT PARTITIONINGS, HARD AND SOFT PARTITION, AND IN VARIOUS PARTS OF THE PROGRAM MEMORY, ESPECIALLY WITH RESPECT TO THE DIFFERENT RAMS AND THE PARTITION.

THE FUNCTIONING OF THE VARIOUS CLEAR KEYS (CE, CLR, CMS, INV CMS, AND CEQ SHOULD BE TESTED IN AND OUT OF LEARN MODE AND PROGRAMMED EQN.

EQN SETUP MODE

5/81 ALL KEYS PROPER AND NOT PROPER TO EQN SHOULD BE ENTERED TO SEE TO SEE WHETHER OR NOT THEY ENTER OR ARE IGNORED AS THEY SHOULD AND THEN EVAL DONE TO SEE WHETHER THEY EXECUTE.

THE LEARN MODE ALPHA CHARACTER TEST SHOULD BE REPEATED IN EQN SETUP MODE.

THE TWO EDIT KEYS (--> AND <--) SHOULD EACH BE TESTED.

THE LEARN MODE TESTS OF THE VARIOUS CLEAR KEYS SHOULD BE REPEATED.

DAVE CALDWELL REVISION: 8/1/80 A

JERRY Puckett

ALEX TEST OUTLINE

AREAS TO BE TESTED:
PROGRAMMING FUNCTIONS, CONDITIONALS, OP 13 & 11 & 10

PRIMARY KEYS:

I. PROGRAMMING;

NOP	PGM	GBR	RTN
SBR	GTO	GTL	SST
LBL	GFR	SBL	

II. CONDITIONAL;

IF >	≤	DSZ
IF <	=	INVDSZ
≥	≠	

III. OP 13 - ROUND TO DISPLAY (UNDEFINED INVERSE)

IV. OP 11 - SIGNUM FN

V. OP 10 - 1x1

1. PROGRAMMING & CONDITIONAL FUNCTIONS WILL NEED TESTED IN THE FOLLOWING MODES:
   - PROGRAM RUN
   - PROGRAM SINGLE STEP
   - KEYBOARD OPERATION
   - CROM (PGM.RUN)

   WILL NEED TESTED WITH BOTH VALID & INVALID ARGUMENTS; ABSOLUTE ADDRESSED AND WITH ALL TYPES OF LABELS
   CONCATENATION OF CONDITIONALS (& WITH FLAGS)

PROGRAMMING

NOP -
INSERT INTO PROGRAM
5/81 SHOULD NOT EFFECT PROGRAM OPERATION EXCEPT DIRECTLY FOLLOWING A CONDITIONAL - THEN SHOULD CANCEL THE CONDITIONAL - NEEDS TO BE INSERTED IN FIELD TO TEST.

SBR -
FOLLOWED BY LABEL OR ADDRESS; DIRECT OR INDIRECT
SHOULD TRANSFER TO LABEL OR ADDRESS & CONTINUE EXECUTION-
SHOULD RETURN ERROR CONDITION IF DIRECTED TO NON-EXISTANT LABEL OR ADDRESS OUT OF BOUNDS OF PARTITION.
WILL COMMENCE OPERATION FROM KEYBOARD
TEST FOR 10 LEVELS AND OVERFLOW STACK

GTO -
FOLLOWED BY LABEL OR ADDRESS; DIRECT OR INDIRECT
SHOULD TRANSFER TO LABEL OR ADDRESS - WILL CONTINUE EXECUTION IN RUN MODE
SHOULD RETURN ERROR CONDITION IF DIRECTED TO NON-EXISTANT LABEL OR ADDRESS OUT OF BOUNDS

LBL -
MARKS A LOCATION FOR USE WITH SBR, SBL, GTO, GTL
VALID ARGUMENTS ARE A-Z, a-z, 0-99

SBL -
MERGED SBR LBL - SAME AS SBR FOR ALPHA LABELS - LOOKS FOR LBL 0-99 FOR NUMERIC ARGUMENT
SHOULD COMMENCE EXECUTION

GTL -
SAME AS SBL

IND -
VALID ARGUMENTS A-Z, 0-999 USED WITH SBR, SBL, GTO, GTL, FOR INDIRECT TRANSFER TO LABEL OR ADDRESS

GFR, GBR -
PROGRAM TRANSFER - WILL ATTEMPT TO CONTINUE EXECUTION IF ARGUMENT LEADS OUT OF BOUNDS IN CROM
CAN BE USED WITH IND

SST -
WILL STEP THROUGH PROGRAM - IN MANUAL OR LEARN-LIKE 59
CAN ALSO BE USED TO EXECUTE ONE STEP AT A TIME
5/81

BST -
MOVES PGM POINTER BACK ONE STEP IN LEARN MODE; CURSOR LEFT IN ALPHA;
SHOULD NOT DO ANYTHING OUTSIDE ∝ OR LRN MODE

RTN -
RETURNS PROGRAM CONTROL TO THE STEP AFTER SBR CALL - OR TO KEYBOARD IF CALLED FROM KEYBOARD [SHOULD RETURN TO THE LAST ADDRESS STORED IN SUBROUTINE RETURN ADDRESS STACK OR KYBD. IF NO ADD. IS STORED.]

PGM -
5/81 MMNN - ACCESSES PGM NN OF CROM/CRAM MM - 0 IS MAIN MEMORY

CONDITIONALS

DSZ -
DECREMENTS REGISTER & EXECUTE NEXT NON-CONDITIONAL INSTRUCTION BLOCK IF VALUE IS NOT ZERO, SKIPS ON ZERO.
SHOULD CAUSE INFINITE LOOP FOR NEG. NUMBERS & NON-INTEGERS

3/81

INV DSZ -
SAME AS DSZ WITH VALUE OF ZERO & NOT ZERO REVERSED

IF>-
COMPARES DISP. TO REGISTER SPECIFIED & EXECUTES NEXT NON-CONDITIONAL BLOCK IF TRUE

IF< -
SAME AS > EXCEPT COMPARISON IS LESS THAN

IF = -
SAME AS > EXCEPT COMPARISON EQUALS TO

CONCATENATION -
CAN CHAIN CONDITIONALS TO EACH OTHER & IFF TO FORM "OR" COMBINATION APPLYING TO NEXT NON-CONDITIONAL INSTRUCTION BLOCK

NEED TO TEST $>$, $<$, $\geq$, $\leq$, =, $\neq$, DSZ + EACH COMBINED WITH FLAG TEST
ALL TESTED WITH BOTH TRUE & FALSE CONDITION
ALL TESTED IN PGM.RUN, CROM, SST MODE IN DECIMAL, HEX $\propto$.

TEST REQUIRED

SBR;GTO -
TEST IN PGM RUN, CROM, SST, KYBD MODES
TEST TRANSFER W/LABEL & ABS.ADD (VALID)
TEST NON-EXISTANT LABEL
TEST ADDRESS OUT OF BOUNDS & NEGATIVE ADDRESS

GTL, SBL -
TEST EXISTANT & NON-EXISTANT LABELS - BOTH ∝ & NUMERIC DIRECT AND INDIRECT

GFR, GBR -
TEST VALID ARGUMENTS (0-99) BOTH IN & OUT OF BOUNDS
TEST INVALID ARGUMENTS $<0, >99$; DIRECT & INDIRECT FOR ALL

RTN -
FROM KEYBOARD & PROGRAM

SST -
PGM RUN & LEARN MODE

ALL TESTS NEED DONE WITH HARD & SOFT PARTITIONING.

NIVERSAL PLANNING FORM

INVALID TRANSFERRS

		0	1	2	3	4	5	6	7	8	9		
1		LBL	01	GTL	B	RTN	LBL	2	GTL	50	RTN		
2													
3		LBL	3	SBL	B	RTN	LBL	4	SBR	B	RTN		
4													
5		LBL	5	GTO	B	RTN	LBL	6	SBL	50	RTN		
6													
7		LBL	7	SBR	99	99	RTN	LBL	8	GTO	99		
8													
9		99	RTN										
10													
11	(-1) ; (-99); TOA →	LBL	9	GTO	LCA	LBL	10	GTL	LCA	LBL	11		
12													
13		SBR	LCA	LBL	12	SBL	LCA						
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													

OPERATING UNIT:

PREPARED BY:

DATE:

3145-A

GFR, GBR VALID

UNIVERSAL PLANNING FORM

30 STO A TO BEGIN

		0	1	2	3	4	5	6	7	8	9
0	LBL	A	1	GFR	10						
1		GFR	50		+	1	=	GBR	8		
2	+	1	=	SUMX	A	GFR	LCA				
3		RTN						+	1	=	
4	GFR	5					+	1	=	NOP	
5	NOP	NOP	GFR	LCA							
6			+	1	=	GBR	30				
7							+	1	=	GBR	
8	60			+	1	=	INV	SUM	A	INV	
9	SUM	A	GBR	LCA							
10											
11											
12											
13											
14											
15											
16											
17											
18											

OPERATING UNIT:
TI-3145-A

PREPARED BY:

DATE:

UNIVERSAL PLANNING FORM

INVALID GFR GBR

-10 STOA

1			LBL	A	GFR	9,99	LBL	B	GBR	9	99		
2													
3			LBL	C	GFR	LCA	LBL	D	GBR	LCA			
4													
5													
6													
7			IN ADDITION- PLACE GFR @ End of PARTITION & RUN OUT OF BOUNDS										
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													

OPERATING UNIT:
TI-3145-A

PREPARED BY:

DATE:

97

UNIVERSAL PLANNING FORM

CONDITIONALS

1	STORE VARIABLES IN A Y B		LBL	A	RCL	A	IF >		B	GTL	B	1	
2	A < B												
3	A > B		RTN	LBL	B	2	RTN						
4	A = B						INV IF >						
5	BOTH POS. Y NEG #'s,		LBL	C	RCL	A	(IF <)		B	GTL	D	1	
6	LARGE & SMALL A's												
7			RTN	LBL	D	2	RTN						
8													
9	THEN INSERT IFF X		LBL	E	RCL	A	IF =		B	GTL	F	1	
10	IN EACH TEST & TRY												
11	WITH FLAG SET & NOTSET.		RTN	LBL	F	2	RTN						
12													
13			LBL	G	RCL	A	IF >	IF =	B	GTL	H	1	
14													
15			RTN	LBL	H	2	RTN						
16													
17			LBL	I	RCL	A	IF <	IF =	B	GTL	J	1	
18													
19			RTN	LBL	J	2	RTN						
20													
21			LBL	K	RCL	A	IF ≠		B	GTL	L	1	
22													
23			RTN	LBL	L	2	RTN						
24													
25													
26													
27													
28													
29													
30													
31													
32													
33													
34													
35													
36													
37													
38													
39													
40													
41													

OPERATING UNIT:

TI-3145-A

PREPARED BY:

DATE:

UNIVERSAL PLANNING FORM

DSZ

1	XX STO α STO B		LBL	A	1	SUM	B	DSZ	α	GTO	A	RCL	B	RTN
2														
3	α, β (unlabeled) 0-99													
4														
5	Min SO TRY LC α		LBL	A	1	SUM	B	DSZ	α	GTO	A	RCL	B	RTN
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														

OPERATING UNIT:

PREPARED BY:

DATE:

TI-3145-A