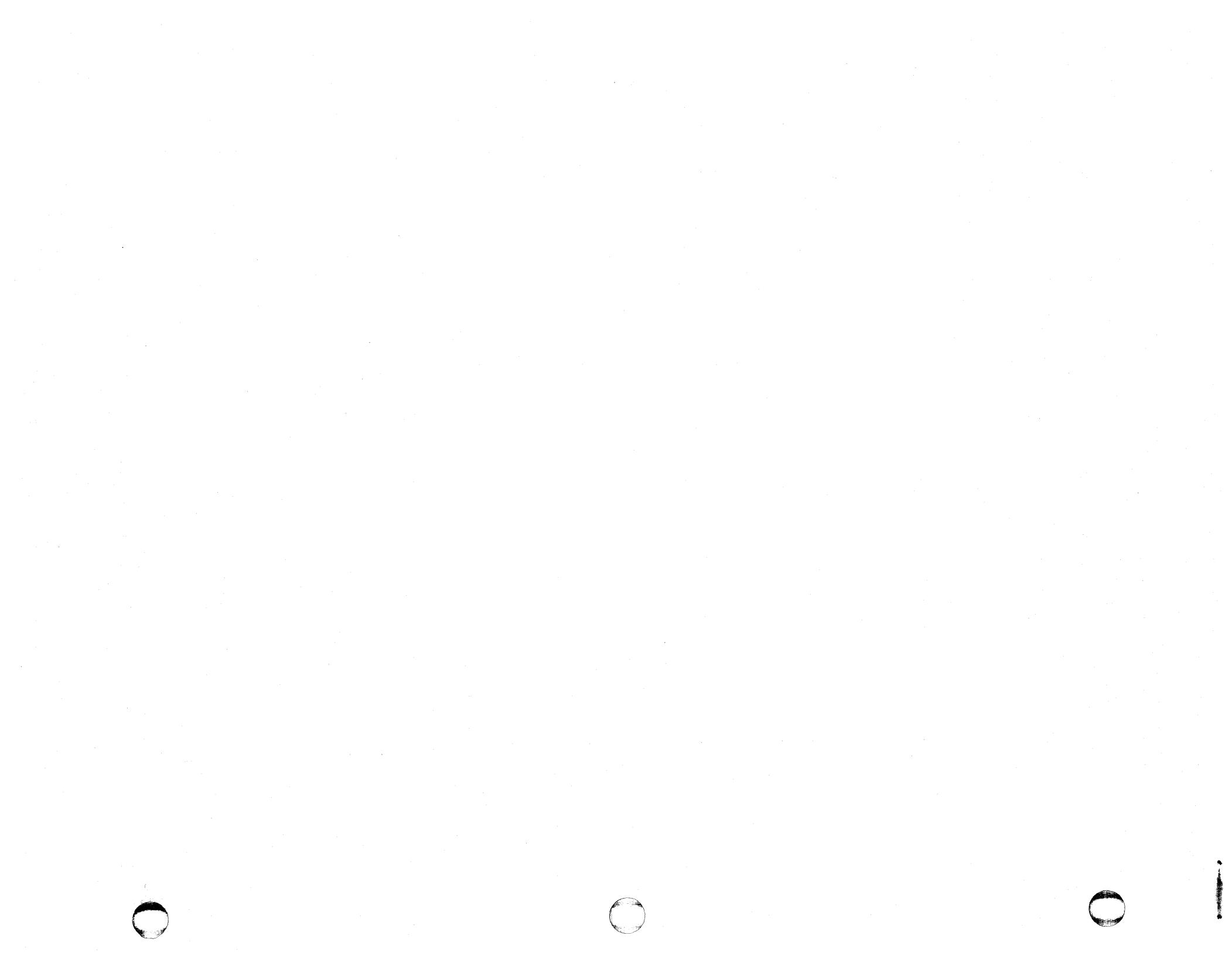


IDENTIFICATION

ad 1440

PRODUCT CODE: MAINDEC-11-DBKEB-A-D
PRODUCT NAME: KE11F (PDP-11 F1S) EXERCISER
DATE CREATED: 1-AUG-72
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: KEN CHAPMAN

COPYRIGHT (C) 1972
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS 01754



CONTENTS

1. ABSTRACT
2. REQUIREMENTS
- 2.1 Equipment
- 2.2 Storage
- 2.3 Preliminary programs
3. LOADING PROCEDURE
4. STARTING PROCEDURE
 - 4.1 Control switch settings
 - 4.2 Starting address
 - 4.3 Program and/or operator action
5. OPERATING PROCEDURE
 - 5.1 Operational switch settings
 - 5.2 Subroutine abstracts
6. ERRORS
 - 6.1 Error printout
 - 6.2 Error recovery
 - 6.3 Error counter
7. RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 Execution time
 - 8.2 Stack pointer
 - 8.3 Pass counter
 - 8.4 Power fail
9. PROGRAM DESCRIPTION

TABLE OF CONTENTS

1. ABSTRACT

This program exercises the KE11F floating point instructions (FADD, FSUB, FMUL, FDIV) with random number patterns. The answers are checked against results obtained using the corresponding FORTRAN software routines. About 200 passes should be run to establish credibility.

2. REQUIREMENTS

2.1 Equipment

PDP-11 (KD11A) standard computer with KE11F option

2.2 Storage

The routines use memory locations 0 - 17500. The map at the end of the listings shows the absolute locations of the FORTRAN math routines which were assembled separately and linked to the main program via LNKX11 on a DECsystem-10.

2.3 Preliminary programs

MAINDEC-11-DBKEA-A KE11F Instruction Tests.

3. LOADING PROCEDURE

Use standard procedure for ABS tapes.

4. STARTING PROCEDURE

4.1 Control switch settings

See 5.1.1 (all down for worst case testing)

4.2 Starting address

The program should always be started at 200.

4.3 Program and/or operator action

- 1) Load program into memory using ABS loader,
- 2) Load address 200,
- 3) Set switches (see 5.1.1) All down for worst case,

- 4) Press start.
- 5) The program will loop and bell will ring once every pass.

5. OPERATING PROCEDURE

5.1 Operational switch settings

SW<15> = 1 HALT ON ERROR
SW<14> = 1 SCOPE LOOP
SW<13> = 1 INHIBIT PRINTOUT
SW<12> = 1 INHIBIT TRACE TRAPPING
SW<11> = 1 INHIBIT ITERATIONS OF SUBTEST
SW<10> = 1 BELL ON ERROR
0 BELL ON PASS COMPLETE
SW<09> = 1 LOOP ON ERROR
SW<08> = 1 LOOP ON TEST IN SW<610>
SW<07> = 1 INPUT DATA FROM THE TELETYPE

Caution: SW<810> are also used for ROM word match with KM11 maintenance card.

5.2 Subroutine Abstracts

5.2.1 TYPIN

If SW<7> is on a 0, the program calculates a pseudo-random number to be used as input data. If SW<7> is on a 1, the program will ask for input data from the teletype at the beginning of each pass. The same data is used with all instructions (FADD, FSUB, FMUL, FDIV) for the entire pass. If SW<7> is put down after entering the data entry routine, that data is used as the starting numbers for the random number generator.

The input format is:

Type Input data:
A1: NNNNNN
A2: NNNNNN
B1: NNNNNN
B2: NNNNNN

Where:

A1 = left word of first argument
A2 = right word of first argument
B1 = left word of second argument
B2 = right word of second argument

Description

i.e., A1,A2(+,-,*,/)B1,B2 = answer

NNNNN = data typed by the operator

A1, A2, B1, and B2 must be 16 bit left justified octal numbers,

E.G.

42 = 000042

200000 = not accepted (17 bits)

4812 = not accepted (8 is not octal)

They are assumed to be in floating point format, i.e., bit 15 of A1 and B1 are the sign bits, bits 7-14 of A1 and B1 are the exponents (excess 128 format) and the rest (bits 0-6 of A1 and B1 and all of A2 and B2) form the mantissa (normalized) less the hidden bit. For more information read the maintenance manual. A1, A2, B1, and B2 are put into RAND,A, RAND,B, RAND,C, and RAND,D respectively.

5.2.2

FORTRAN

This routine make use of "polish mode" to link the FORTRAN MATH PACKAGE ROUTINES TO CALCULATE THE EXPECTED RESULT.

LOCATIONS \$ADD1, \$ADD2 contain addition answer,

Locations \$SUB1, \$SUB2 contain subtract answer,

Locations \$MUL1, \$MUL2 contain multiply answer,

Locations \$DIV1, \$DIV2 contain divide answer,

If a floating error occurs (overflow, underflow, or divide by zero), these answers are meaningless. The locations \$ADDPS, \$SUBPS, \$MULPS, or \$DIVPS contains 340 and \$ADDER, \$SUBER, \$MULER, or \$DIVER, contain the conditions codes of the error.

5.2.3

SCOPE

This subroutine call is placed between each subtest in the test section. It records the starting address of each subtest as it is being entered in location "LADS". If a scope loop is requested, the current subtest will be looped upon. SW<11> on a 1 inhibits iteration of subtests. The contents of LADS may be used to determine the last subtest successfully completed.

Description

5.2.4 HLT

This routine prints out an error message (See 6.1), To inhibit typeouts, put SW<13> on a 1.

5.2.5 TRTRAP

If SW<12> is on a 0, the T-bit will be set on alternate passes. When the T-bit is set, the processor traps after each instruction. The first instruction executed upon trapping is an "RTT" which returns to the interrupted sequence of instructions. This sequence is continued until the end of the program is reached.

5.2.6 TRAPCATCHER

A ",+2" - "HALT" sequence is repeated from 0 = 776 to catch any unexpected traps. Thus any unexpected traps or interrupts will HALT at the vector + 2.

5.2.7 FLOATING POINT TRAP (to 244)

All tests set the floating point trap vector (244) to point to the instruction following the floating point instruction. Thus, whether or not a trap occurs is only detected if the data or the stack pointer(s) are wrong.

6. ERRORS

6.1 Error printout

There are two formats for error typeout; one for normal numbers and one for floating errors (overflow, underflow and divide by zero).

6.1.1 The normal format (when no floating point error is indicated) is as follows:

AAAAAA	MMMMMM	,	MMMMMM	S	MMMMMM	,	MMMMMM
	PSW	SP			ANSWER		
EXPECT:	NNN	NNN	NNNNNN	,	NNNNNN		
GOT:	NNN	NNN	NNNNNN	,	NNNNNN		

Where:

AAAAAA ==> PC of HLT instruction

MMMMMM ==> Input data (RAND,A, RAND,B, RAND,C, RAND,D)

S ==> type of operation being tested (+,-,* , or /)

Description

NNNNNN ==> results

PSW = processor status word

SP = stack pointer (not necessarily R6)

ANSWER = resulting answer off the stack

6.1.2 When a floating point error is indicated (overflow, underflow, or divide by zero) the format is as follows:

AAAAAA MMMMM,MMMM S MMMMM,MMMM

PSW SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6

EXPECT: NNN NNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN

GOT: NNN NNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN

Where:

AAAAAA ==> PC of HLT instruction

MMMMMM ==> Input data (RAND,A, RAND,B, RAND,C, RAND,D)

S ==> type of operation being tested (+,-,* or /)

NNNNNN ==> results

PSW = processor status word

SP = stack pointer (not necessarily R6)

ANS1 = PC of interrupted instruction (should be FIS)

ANS2 = PSW at interrupt time

ANS3 = input data (RAND,C)

ANS4 = " " (RAND,D)

ANS5 = " " (RAND,A)

ANS6 = " " (RAND,B)

To find the failing test, look at the listing above the address typed.

6.2 Error recovery

Restart at 200

6.3 Error count

An error count is kept in "ERRORS" (LOC 1002). It is cleared by restarting at 200.

7. RESTRICTIONS

None

Description

8. MISCELLANEOUS

8.1 Execution time

A bell will ring within 5 seconds with all switches down.
More than 200 passes should be run to insure a wide variety
of number patterns.

8.2 Stack Pointer

Stack is initially set to 604

8.3 Pass counter

A 32 bit (2 words) pass count is kept in "PCNT" (LOC 1004,1006). It is cleared by restarting at 200.

8.4 Power Fail

Each test can be power failed with no errors. To use, start the test as usual and power down then up at any time. The program should type "POWER" and continue to run from where power fail interrupted with no other typeouts.

9. PROGRAM DESCRIPTION

This program tests all the FIS Instructions on the KE11F using all registers except 7 for the "stack pointer". The program has many subtests (the code between 2 SCOPE statements) which are run 256 times before continuing to the next. SW<11> on a 1 causes each subtest to be run only once. The address ICNT (LOC 1000) contains the iteration count in the left byte and the test number in the right byte. All the subtests should be run sequentially by starting at 200 not by starting at the beginning of the subtest. To loop on a particular subtest, put the test number (see listing) in SW<6:0> of the switch register and SW<8> on a 1. This test will be looped upon until SW<8> is put on a 0 or the right byte is changed. If the test is non-existent, the program will be run as usual.

The FORTRAN math routines, which are used to calculate the correct answers, were taken unmodified from the PDP-11 FORTRAN package and assembled as separate modules. They were linked to the main programs via LNKX11 on a DECSYSTEM-10 which produces a binary tape in the normal absolute format. Thus, the program loads and runs just like any other diagnostic program.

1 SWITCH SETTINGS AND ERROR TYPEOUT FORMAT
55 EQUALITIES
104 VECTOR AND ANSWER AREA
162 RANDOM NUMBER GENERATOR, TRACE TSR, AND FIS TSR
164 FOUR WORD RANDOM NUMBER GENERATOR
183 SETUP AREA
277 TEST 1: EXERCISE FADD R0
363 TEST 2: EXERCISE FSUB R1
449 TEST 3: EXERCISE FMUL R2
535 TEST 4: EXERCISE FDIV R3
621 TEST 5: EXERCISE FADD R4
693 TEST 6: EXERCISE FSUB R5
765 TEST 7: EXERCISE FMUL SP
838 TEST 10: EXERCISE FDIV R0
910 TEST 11: EXERCISE FADD R1
982 TEST 12: EXERCISE FSUB R2
1054 TEST 13: EXERCISE FMUL R3
1126 TEST 14: EXERCISE FDIV R4
1198 TEST 15: EXERCISE FADD R5
1270 TEST 16: EXERCISE FSUB SP
1343 TEST 17: EXERCISE FMUL R0
1415 TEST 20: EXERCISE FDIV R1
1487 TEST 21: EXERCISE FADD R2
1559 TEST 22: EXERCISE FSUB R3
1631 TEST 23: EXERCISE FMUL R4
1703 TEST 24: EXERCISE FDIV R5
1775 TEST 25: EXERCISE FADD SP
1848 TEST 26: EXERCISE FSUB R0
1920 TEST 27: EXERCISE FMUL R1
1992 TEST 30: EXERCISE FDIV R2
2064 TEST 31: EXERCISE FADD R3
2136 TEST 32: EXERCISE FSUB R4
2208 TEST 33: EXERCISE FMUL R5
2280 TEST 34: EXERCISE FDIV SP
2353 END ROUTINE
2381 READ OCTAL INPUT ROUTINE (READIN)
2407 POLISH MODE ROUTINES TO ACCESS FORTRAN ROUTINES
2523 PUSH DATA ROUTINE (PUSHR)
2534 SCOPE ROUTINE
2566 HLT ROUTINE (ERROR TYPEOUT)
2721 TTY INPUT ROUTINE
2746 TYPE ROUTINE
2775 OCTAL DUMP OF A WORD & 18 BIT ADDRESS TYPER
2810 POWER DOWN AND UP ROUTINES

MAINDEC-11-DBKEB-A KE11F (PDP-11 F1S) EXERCISER, MAR. 11, 1972 22-AUG-72 11:40 PAGE 1
DBKEBA.P11 SWITCH SETTINGS AND ERROR TYPEOUT FORMAT

1 ,TITLE MAINDEC-11-DBKEB-A KE11F (PDP-11 F1S) EXERCISER,
2 000000 ,ASECT
3 ,GLOBL SADR,SSBR,SMLR,\$DVR,SERR,SERRA

4 ;COPYRIGHT 1972, DIGITAL EQUIPMENT CORP., MAYNARD, MASS
5 ;PROGRAM BY KEN CHAPMAN
6 ;REM

7
8 SWITCH USE
9 *****
10 7 TTY DATA INPUT
11 8 LOOP ON TEST IN SW<610>
12 9 LOOP ON ERROR
13 10 0=BELL ON PASS COMPLETED
14 11 1=BELL ON ERROR
15 12 INHIBIT ITERATIONS
16 13 INHIBIT TRACE TRAP
17 14 INHIBIT ERROR TYPEOUTS
18 15 LOOP ON TEST
19
20 HALT ON ERROR
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

ERROR MESSAGE FORMATS!

1. WHEN NO FLOATING POINT ERROR IS INDICATED

AAAAAA MMMMM,MMMM S MMMMM,MMMM
PSW SP ANSWER
EXPECT: NNN NNN NNNNNN,NNNNNN
GOT: NNN NNN NNNNNN,NNNNNN

WHEREI

AAAAAA => PC OF HLT INSTRUCTION
MMMM,MMMM => INPUT DATA (RAND,A, RAND,B, RAND,C, RAND,D)
S => TYPE OF OPERATION BEING TESTED (+,-,* , OR /)
NNN => RESULTS
PSW = PROCESSOR STATUS WORD
SP = STACK POINTER (NOT NECESSARILY R6)
ANSWER= RESULTING ANSWER OFF THE STACK

2. WHEN A FLOATING POINT ERROR IS INDICATED (OVERFLOW, UNDERFLOW,
OR DIVIDE BY ZERO)

AAAAAA MMMMM,MMMM S MMMMM,MMMM
PSW SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6
EXPECT: NNN NNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN
GOT: NNN NNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN NNNNNN

WHEREI

AAAAAA, MMMMM, S, NNN, PSW, AND SP ARE THE SAME AS ABOVE,
ANS1 = PC OF INTERRUPTED INSTRUCTION (SHOULD BE F1S)
ANS2 = PSW AT INTERRUPT TIME
ANS3 = INPUT DATA (RAND,C)
ANS4 = " " (RAND,D)
ANS5 = " " (RAND,A)
ANS6 = " " (RAND,B);

54	104400	SCOPE#	TRAP
55	104000	HLT#	EMT
56	000004	TYPE#	IOT
57	177776	PS#	177776
58	177570	SWR#	177570
59	177570	DISPLAY#	SWR
60	000007	BELL#	7
61	000000	R0#	X0
62	000001	R1#	X1
63	000002	R2#	X2
64	000003	R3#	X3
65	000004	R4#	X4
66	000005	R5#	X5
67	000005	TTY#	X5
68	000006	SP#	X6
69	000007	PC#	X7
70	100000	SW15#	100000
71	040000	SW14#	40000
72	020000	SW13#	20000
73	010000	SW12#	10000
74	004000	SW11#	4000
75	002000	SW10#	2000
76	001000	SW09#	1000
77	000400	SW08#	400
78	000001	BIT0#	000001
79	000002	BIT1#	000002
80	000004	BIT2#	000004
81	000010	BIT3#	000010
82	000020	BIT4#	000020
83	000040	BIT5#	000040
84	000100	BIT6#	000100
85	000200	BIT7#	000200
86	000400	BIT8#	000400
87	001000	BIT9#	001000
88	002000	BIT10#	002000
89	004000	BIT11#	004000
90	010000	BIT12#	010000
91	020000	BIT13#	020000
92	040000	BIT14#	040000
93	100000	BIT15#	100000
94	000000	LEVEL0#	000
95	000040	LEVEL1#	040
96	000100	LEVEL2#	100
97	000140	LEVEL3#	140
98	000200	LEVEL4#	200
99	000240	LEVEL5#	240
100	000300	LEVEL6#	300
101	000340	LEVEL7#	340

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22=AUG-72 11140 PAGE 3
DBKEBA.P11 VECTOR AND ANSWER AREA

102
103 000000 ,# 0 JTRAP CATCHER FROM 0 = 776
104
105 000200 ,# 200
106
107 200200 000167 000604 JMP BEGIN IJUMP TO STARTING ADDRESS OF PROGRAM
108
109 000204 ,# 204
110 200204 000167 000736 JMP START IRESTART ADDRESS
111
112 000600 ,# 600
113
114 ;THE FOLLOWING LOCATIONS ARE USED FOR THE STACKS; R6 IS INITIALLY SET
115 ;TO 604 (STACK0), AS ARE THE OTHER REGISTERS (R0 THRU R5) WHEN
116 ;THEY ARE TO BE USED AS THE FLOATING POINT STACK POINTER,
117 ;THE DATA IS PUT DIRECTLY ONTO THE STACK, NOT BY PUSHES;
118 ;IF NO ERROR OCCURES THE STACK POINTER (ANY REGISTER) IS POINTING
119 ;TO 610 (ANS1). IF AN ERROR OCCURES, R6 IS POINTING TO 604;
120 ;SO THE TRAP PUTS THE RETURN ADDRESS AND PS IN 600 (STK1)
121 ;AND 602 (STK2) RESPECTIVELY.
122
123 200600 000000 STK1: 0
124 200602 000000 STK2: 0
125 200604 000000 STK3: STACK0! 0
126 200606 000000 STK4: STACK2! 0
127 200610 000000 STK5: STACK4! ANS1! 0
128 200612 000000 STK6: STACK6! ANS2! 0
129 200614 000000 SPSW: 0
130 200616 000000 SSP!: 0
131
132 200620 000000 RAND,A!: 0
133 200622 000000 RAND,B!: 0
134 200624 000000 RAND,C!: 0
135 200626 000000 RAND,D!: 0
136
137 200630 000000 SADDPS: 0
138 200632 000000 SADD1!: 0
139 200634 000000 SADD2!: 0
140 200636 000000 SADDER: 0
141
142 200640 000000 SSUBPS: 0
143 200642 000000 SSUB1!: 0
144 200644 000000 SSUB2!: 0
145 200646 000000 SSUBER: 0
146
147 200650 000000 \$MULPS: 0
148 200652 000000 \$MUL1!: 0
149 200654 000000 \$MUL2!: 0
150 200656 000000 \$MULER: 0
151
152 200660 000000 \$DIVPS: 0
153 200662 000000 \$DIV1!: 0
154 200664 000000 \$DIV2!: 0
155 200666 000000 \$DIVER: 0

156
157 200670 000000 SAVSTK: 0
158 200672 000000 RNDFLG: 0 FOR FLAGS TO KEEP TRACK OF ROUNDING
159
160
161 200674 105367 177726 RAND4\$: DECB RAND,D //INSURE ALL ZEROES WORKS
162 200700 066767 177716 177712 ADD RAND,B, RAND,A
163 200706 005567 177714 ADC RAND,D
164 200712 066767 177706 177702 ADD RAND,C, RAND,B
165 200720 005567 177700 ADC RAND,C
166 200724 066767 177676 177672 ADD RAND,D, RAND,C
167 200732 005567 177664 ADC RAND,B
168 200736 066767 177656 177662 ADD RAND,A, RAND,D
169 200744 005567 177650 ADC RAND,A
170 200750 000207 RTS PC
171
172
173 200752 000006 YESRTI RTT TRACE TRAP SERVICE ROUTINE
174
175 200754 104000 FISTRPI HLT FERRONIOUS FIS TRAP
176 200756 000002 RTI
177

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 5
DBKEBA.P11 SETUP AREA

178
179 001000 ,# 1000
180
181 001000 000000 ICNT: 0 ;ITERATION COUNT (HI BYTE) TEST # (LO BYTE)
182 001002 000000 ERRORS: 0 ;ERROR COUNT LOCATION
183 001004 000000 000000 PCNT: 0,0 ;PASS COUNT LOCATION
184
185 001010 012706 000604 BEGINI MOV #STACK0,SP ;SET UP STACK
186 001014 012737 000752 000014 MOV #YESRT, #14 ;SET UP TRACE TRAP
187 001022 012700 000020 MOV #20,R0
188 001026 012720 015256 MOV #,IOT,(R0)+ ;SET UP IOT VECTOR
189 001032 012720 000340 MOV #340,(R0)+
190 001036 012720 015536 MOV #PDOWNS,(R0)+ ;SET UP POWER FAIL VECTOR
191 001042 012720 000340 MOV #340,(R0)+
192 001046 012720 014020 MOV #HLTS,(R0)+ ;SET EMT VECTOR
193 001052 012720 000340 MOV #340,(R0)+
194 001056 012720 013644 MOV #SCOPES,(R0)+ ;SET TRAP VECTOR
195 001062 012720 000340 MOV #340,(R0)+
196 001066 012737 000754 000244 MOV #FISTRP, #244 ;SET UP FIS VECTOR
197 001074 012737 000340 000246 MOV #340, #246
198 001102 012767 123456 177510 MOV #123456,RAND,A ;PRIME THE RANDOM NUMBER GENERATOR
199 001110 012767 107654 177504 MOV #107654,RAND,B
200 001116 012767 070707 177500 MOV #070707,RAND,C
201 001124 012767 125252 177476 MOV #125252,RAND,D
202 001132 005067 177644 CLR ERRORS ;CLEAR ERROR COUNTER
203 001136 005067 177642 CLR PCNT ;CLEAR PASS COUNTER
204 001142 005067 177640 STARTI CLR PCNT+2
205 001146 012706 000604 177776 MOV #STACK0,SP ;SET UP STACK
206 001152 012737 000140 177776 MOV #140, #PS ;SET UP PROCESSOR STATUS
207 001160 005067 177614 CLR ICNT
208 001164 005067 012622 CLR LADS
209 001170 005067 177476 CLR RNDFLG ;CLEAR THE ROUNDING FLAGS
210 001174 005067 177570 TSTB @NSWR ;CHECK FOR TTY INPUT
211 001200 100403 BM1 TYPIN
212 001202 004767 177466 JSR PC,RAND4\$;BRANCH TO ROUTINE TO CALCULATE ANSWERS
213 001206 000464 BR FORTAN
214
215 ;THE FOLLOWING ROUTINE ACCEPTS DATA FROM THE TELETYPE;
216 ;THE FORMAT IS FIXED! A1,A2 (+,-,*,/) B1,B2;
217 ;THE PROGRAM ASKES FOR ONE ARGUMENT AT A TIME, AND REASKS
218 ;WHEN INVALID DATA IS ENTERED;
219
220 001210 000004 001214 TYPINI TYPE, ,#2
221 001214 005015 054524 042520 ,ASCIZ <15><12>"TYPE INPUT DATA!"<15><12>
222 001222 044440 050116 052125
223 001230 042040 052101 035101
224 001236 005015 000
225 001242
226 000004 001246 1\$: EVEN
227 001246 030501 020072 000040 TYPE, ,#2
228 001254 004567 011502 ,ASCIZ "A1;"
229 001260 000620 JSR R5, READIN ;ACCEPT FIRST ARGUMENT FROM THE TTY
230 001262 103752 BCS TYPIN
231 001264 000004 001270 2\$: TYPE, ,#2

232	001270	031101	020072	000040		,ASCIZ "A2: "		
233	001276	004567	011460			JSR R5, READIN JACCEP SECOND ARGUEMENT FROM THE TTY		
234	001302	000622				RAND,B		
235	001304	103767				BCS 2\$		
236	001306	001340				BNE TYPIN		
237	001310	000004	001314		3\$:	TYPE, ,#2		
238	001314	030502	020072	000040		,ASCIZ "B1: "		
239	001322	004567	011434			JSR R5, READIN JACCEP THIRD ARGUEMENT FROM THE TTY		
240	001326	000624				RAND,C		
241	001330	103767				BCS 3\$		
242	001332	001326				BNE TYPIN		
243	001334	000004	001340		4\$:	TYPE, ,#2		
244	001340	031102	020072	000040		,ASCIZ "B2: "		
245	001346	004567	011410			JSR R5, READIN JACCEP FOURTH ARGUEMENT FROM THE TTY		
246	001352	000626				RAND,D		
247	001354	103767				BCS 4\$		
248	001356	001314				BNE TYPIN		
249								
250	001360	005067	177244			FORTAN: CLR	SADDPS	JCLEAR ALL THE PS SAVE LOCATIONS
251	001364	005067	177250			CLR	SSUBPS	
252	001370	005067	177254			CLR	SMULPS	
253	001374	005067	177260			CLR	SDIVPS	
254								
255	001400	004467	011460			JSR X4, SPOLSH	JENTER POLISH MODE	
256	001404	013044				SPUSH	JPUSH THE DATA ONTO THE STACK	
257	001406	000000G				SADR	JFORTRAN ADD ROUTINE	
258	001410	013066				SPOPAD	JSAVE THE ADD ANSWERS	
259	001412	013044				SPUSH	JPUSH THE DATA ONTO THE STACK	
260	001414	000000G				SSBR	JFORTRAN SUBTRACT ROUTINE	
261	001416	013144				SPOPSB	JSAVE THE SUBTRACT ANSWERS	
262	001420	013044				SPUSH	JPUSH THE DATA ONTO THE STACK	
263	001422	000000G				SMLR	JFORTRAN MULTIPLY ROUTINE	
264	001424	013222				SPOPML	JSAVE THE MULTIPLY ANSWERS	
265	001426	013044				SPUSH	JPUSH THE DATA ONTO THE STACK	
266	001430	000000G				SDVR	JFORTRAN DIVIDE ROUTINE	
267	001432	013300				SPOPOV	JSAVE THE DIVIDE ANSWERS	
268	001434	013412				SEXIT	JEXIT POLISH MODE	
269								
270	001436	104400				SCOPE		

MAINDEC-11-DBKEB-A KE11F (PDP-11 F1S) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 7
DBKEBA.P11 TEST 1: EXERCISE FADD R0

```

271
272
273
274
275
276
277
278 201440 012700 000604 TST1: MOV #STACK0,R0 ;SET UP THE STACK POINTER
279 201444 004767 012130 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
280
281 201450 000240 NOP
282 201452 075000 FADD+ R0 ;FLOATING ADD ON THE R0 STACK
283
284 201454 013767 177776 177132 1$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
285 201462 010067 177130 MOV R0, SSP ;SAVE THE STACK POINTER
286 201466 026767 177136 177120 6$: CMP SADDPS, SPSW ;CHECK THE PROCESSOR STATUS
287 201474 001023 BNE 4$ ;JB CHECK FOR ROUNDING ERROR
288
289 201476 105767 177112 TSTB SPSW ;CHECK FOR ERROR
290 201502 100464 BMJ 2$ ;BRANCH IF ERROR
291
292 201504 012767 000610 177156 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
293 201512 026767 177152 177076 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
294 201520 001401 BEQ ,+4 ;BRANCH IF OK
295 201522 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
296
297 201524 026767 177102 177056 CMP SADD1, ANS1 ;CHECK THE ANSWER
298 201532 001004 BNE 4$ ;JB
299 201534 026767 177074 177050 CMP SADD2, ANS2 ;CHECK THE ANSWER
300 201542 001515 BEQ 3$ ;JB
301 201544 032767 000002 177120 4$: BIT #BIT1, RNDFLG ;CHECK THE ROUNDING FLAG
302 201552 001022 BNE 5$ ;JB
303 201554 052767 000002 177110 BIS #BIT1, RNDFLG ;SET ROUNDING FLAG
304 201562 062767 000001 177044 ADD #1, SADD2 ;INCREMENT FORTRAN ANSWER
305 201570 005567 177036 ADC SADD1 ;ADD CARRY
306 201574 102334 BVC 6$ ;BRANCH IF NO OVERFLOW
307 201576 000257 CCC ;CLEAR ALL CONDITION CODES
308 201600 000262 SEV ;SET VSB1
309 201602 013767 177776 177026 MOV @#PS, SADDER ;SET PSW FOR OVERFLOW
310 201610 012767 000340 177012 MOV #340, SADDPS ;SET "DE" PSW
311 201616 000723 BR 6$ ;TRY AGAIN
312
313 201620 132767 000002 177045 5$: BITB #BIT1, RNDFLG+1 ;CHECK "DEROUNDING" FLAG
314 201626 001010 BNE 7$ ;BRANCH AND SET
315 201630 152767 000002 177035 BISB #BIT1, RNDFLG+1 ;SET "DEROUNDING" FLAG
316 201636 162767 000001 176770 SUB #1, SADD2 ;RESTORE ORIGINAL ANSWER
317 201644 005667 176762 SBC SADD1 ;SUBTRACT CARRY
318 201650 104000 HLT ;WRONG PSW OR ANSWER
319
320 201652 000451 BR 3$ ;JB
321
322 201654 012767 000604 177006 2$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
323 201662 026767 177002 176726 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
324 201670 001401 BEQ ,+4 ;BRANCH IF OK

```

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 8
DBKEBA.P11 TEST 1:

325	001672	104000		HLT		JSTACK POINTER FOULED UP	
326							
327	001674	022767	001454 176676	CMP	#15,	STK1	JCHECK THE RTI ADDRESS ON THE STACK
328	001702	001401		BEQ	,+4		JBRANCH IF OK
329	001704	104000		HLT			JRTI ADDRESS NOT EQUAL TO #15
330							
331	001706	026767	176724 176666	CMP	SADDER,	STK2	JCHECK THE PSW ON THE STACK
332	001714	001401		BEQ	,+4		JBRANCH IF OK
333	001716	104000		HLT			JRTI PSW NOT EQUAL TO 200
334							
335	001720	026767	176700 176656	CMP	RAND,C,	STK3	JCHECK THE DATA ON THE STACK
336	001726	001401		BEQ	,+4		JBRANCH IF OK
337	001730	104000		HLT			JSTK3 NOT EQUAL TO RAND,C
338							
339	001732	026767	176670 176646	CMP	RAND,D,	STK4	JCHECK THE DATA ON THE STACK
340	001740	001401		BEQ	,+4		JBRANCH IF OK
341	001742	104000		HLT			JSTK4 NOT EQUAL TO RAND,D
342							
343	001744	026767	176650 176636	CMP	RAND,A,	STK5	JCHECK THE DATA ON THE STACK
344	001752	001401		BEQ	,+4		JBRANCH IF OK
345	001754	104000		HLT			JSTK5 NOT EQUAL TO RAND,A
346							
347	001756	026767	176640 176626	CMP	RAND,B,	STK6	JCHECK THE DATA ON THE STACK
348	001764	001401		BEQ	,+4		JBRANCH IF OK
349	001766	104000		HLT			JSTK6 NOT EQUAL TO RAND,B
350							
351	001770	012716	001776	MOV	#35,	(SP)	JRESET THE STACK
352	001774	000002		RTI			JRESTORE THE STATUS (T-BIT)
353							
354	001776	104400		3\$: SCOPE			
355							

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 9
DBKEBA.P11 TEST 2: EXERCISE FSUB R1

356
357
358 ;TEST 2: EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
359 ; RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
360 ; STACK POINTER = R1
361 ;
362
363 002000 012701 000604 TST2: MOV #STACK0,R1 ;SET UP THE STACK POINTER
364 002004 004767 011570 JSR PC, PUSH R1 ;PUT THE DATA ON THE STACK
365
366 002010 000240 NOP
367 002012 075011 FSUB+ R1 ;FLOATING SUBTRACT ON THE R1 STACK
368
369 002014 013767 177776 176572 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
370 002022 010167 176570 MOV R1, SSP ;SAVE THE STACK POINTER
371 002026 026767 176606 176560 6\$: CMP SSUBPS, SPSW ;CHECK THE PROCESSOR STATUS
372 002034 001023 BNE 4\$;GO CHECK FOR ROUNDING ERROR
373
374 002036 105767 176552 TSTB SPSW ;CHECK FOR ERROR
375 002042 100464 BMI 2\$;BRANCH IF ERROR
376
377 002044 012767 000610 176616 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
378 002052 026767 176612 176536 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
379 002060 001401 BEQ ,+4 ;BRANCH IF OK
380 002062 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
381
382 002064 026767 176552 176516 CMP SSUB1, ANS1 ;CHECK THE ANSWER
383 002072 001004 BNE 4\$
384 002074 026767 176544 176510 CMP SSUB2, ANS2 ;CHECK THE ANSWER
385 002102 001515 BEQ 3\$
386 002104 032767 000004 176560 4\$: BIT #BIT2, RNDFLG ;CHECK THE ROUNDING FLAG
387 002112 001022 BNE 5\$
388 002114 052767 000004 176550 BIS #BIT2, RNDFLG ;SET ROUNDING FLAG
389 002122 062767 000001 176514 ADD #1, SSUB2 ;INCREMENT FORTRAN ANSWER
390 002130 005567 176506 ADC SSUB1 ;ADD CARRY
391 002134 102334 BVC 6\$;BRANCH IF NO OVERFLOW
392 002136 000257 CCC ;CLEAR ALL CONDITION CODES
393 002140 000262 SEV ;SET V-BIT
394 002142 013767 177776 176476 MOV @#PS, SSUBER ;SET UP PSW FOR OVERFLOW
395 002150 012767 000340 176462 MOV #340, SSUBPS ;SET UP TRAP PSW
396 002156 000723 BR 6\$;TRY IT AGAIN
397
398 002160 132767 000004 176505 5\$: BITB #BIT2, RNDFLG+1 ;CHECK "DEROUNDING" FLAG
399 002166 001010 BNE 7\$;BRANCH IF SET
400 002170 152767 000004 176475 BISB #BIT2, RNDFLG+1 ;SET "DEROUNDING" FLAG
401 002176 162767 000001 176440 SUB #1, SSUB2 ;RESTORE ORIGINAL ANSWER
402 002204 005667 176432 SBC SSUB1 ;SUBTRACT CARRY
403 002210 104000 HLT ;WRONG PSW OR ANSWER
404
405 002212 000451 BR 3\$
406
407 002214 012767 000604 176446 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
408 002222 026767 176442 176366 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
409 002230 001401 BEQ ,+4 ;BRANCH IF OK

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 10
DBKEBA,P11 TEST 2: EXERCISE FSUB R1

410	002232	104000		HLT		ISTACK POINTER FOULED UP
411						
412	002234	022767	002014	176336	CMP #15, STK1	ICHECK THE RTI ADDRESS ON THE STACK
413	002242	001401			BEQ ,+4	IBRANCH IF OK
414	002244	104000			HLT	IRTI ADDRESS NOT EQUAL TO #15
415						
416	002246	026767	176374	176326	CMP SSUBER, STK2	ICHECK THE PSW ON THE STACK
417	002254	001401			BEQ ,+4	IBRANCH IF OK
418	002256	104000			HLT	IRTI PSW NOT EQUAL TO 200
419						
420	002260	026767	176340	176316	CMP RAND.C, STK3	ICHECK THE DATA ON THE STACK
421	002266	001401			BEQ ,+4	IBRANCH IF OK
422	002270	104000			HLT	ISTK3 NOT EQUAL TO RAND.C
423						
424	002272	026767	176330	176306	CMP RAND.D, STK4	ICHECK THE DATA ON THE STACK
425	002300	001401			BEQ ,+4	IBRANCH IF OK
426	002302	104000			HLT	ISTK4 NOT EQUAL TO RAND.D
427						
428	002304	026767	176310	176276	CMP RAND.A, STK5	ICHECK THE DATA ON THE STACK
429	002312	001401			BEQ ,+4	IBRANCH IF OK
430	002314	104000			HLT	ISTK5 NOT EQUAL TO RAND.A
431						
432	002316	026767	176300	176266	CMP RAND.B, STK6	ICHECK THE DATA ON THE STACK
433	002324	001401			BEQ ,+4	IBRANCH IF OK
434	002326	104000			HLT	ISTK6 NOT EQUAL TO RAND.B
435						
436	002330	012716	002336		MOV #38, (SP)	IRESET THE STACK
437	002334	000002			RTI	IRESTORE THE STATUS (T-BIT)
438						
439	002336	104400			SS: SCOPE	
440						

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 11
DBKEBA.P11 TEST 3: EXERCISE FMUL R2

441
442
443 ;*****
444 ;TEST 3I EXERCISE FMUL (PDP-11 FLOATING MULTIPLY INSTRUCTION)
445 ; RAND,A,RAND,B * RAND,C,RAND,D = ANS1,ANS2
446 ; STACK POINTER = R2
447 ;*****
448 002340 012702 000604 TST3: MOV #STACK0,R2 ;SET UP THE STACK POINTER
449 002344 004767 011230 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
450
451 002350 000240 NOP
452 002352 075022 FMUL+, R2 ;FLOATING MULTIPLY ON THE R2 STACK
453
454 002354 013767 177776 176232 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
455 002362 010267 176230 176256 2\$: MOV R2, SSP ;SAVE THE STACK POINTER
456 002366 026767 176220 6\$: CMP SMULPS, SPSW ;CHECK THE PROCESSOR STATUS
457 002374 001023 BNE 4\$;GO CHECK FOR ROUNDING ERROR
458
459 002376 105767 176212 TSTB SPSW ;CHECK FOR ERROR
460 002402 100464 BMI 2\$;BRANCH IF ERROR
461
462 002404 012767 000610 176256 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
463 002412 026767 176252 176176 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
464 002420 001401 BEQ ,+4 ;BRANCH IF OK
465 002422 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
466
467 002424 026767 176222 176156 CMP SMUL1, ANS1 ;CHECK THE ANSWER
468 002432 001004 BNE 4\$
469 002434 026767 176214 176150 CMP SMUL2, ANS2 ;CHECK THE ANSWER
470 002442 001515 BEQ 3\$
471 002444 032767 000010 176220 4\$: BIT #BIT3, RNDFLG ;CHECK THE ROUNDING FLAG
472 002452 001022 BNE 5\$
473 002454 052767 000010 176210 BIS #BIT3, RNDFLG ;SET ROUNDING FLAG
474 002462 062767 000001 176164 ADD #1, SMUL2 ;INCREMENT FORTRAN ANSWER
475 002470 005567 176156 ADC SMUL1 ;ADD CARRY
476 002474 102334 BVC 6\$;BRANCH IF NO OVERFLOW
477 002476 000257 CCC ;CLEAR ALL CONDITION CODES
478 002500 000262 SEV ;SET V-BIT
479 002502 013767 177776 176146 MOV @#PS, SMULER ;SET UP PSW FOR OVERFLOW
480 002510 012767 000340 176132 MOV #340, SMULPS ;SET UP TRAP PSW
481 002516 000723 BR 6\$;TRY IT AGAIN
482
483 002520 132767 000010 176145 5\$: BITB #BIT3, RNDFLG+1 ;CHECK "DEROUNDING" FLAG
484 002526 001010 BNE 7\$;BRANCH IF SET
485 002530 152767 000010 176135 BISB #BIT3, RNDFLG+1 ;SET "DEROUNDING" FLAG
486 002536 162767 000001 176110 SUB #1, SMUL2 ;RESTORE ORIGINAL ANSWER
487 002544 005667 176102 SBC SMUL1 ;SUBTRACT CARRY
488 002550 104000 7\$: HLT ;WRONG PSW OR ANSWER
489
490 002552 000451 BR 3\$
491
492 002554 012767 000604 176106 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
493 002562 026767 176102 176026 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
494 002570 001401 BEQ ,+4 ;BRANCH IF OK

MAINDEC-11-DBKEBA-A KE11F (PDP-11 F1S) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 12
DBKEBA.PSI TEST 3: EXERCISE FMUL R2

495	022572	104000		HLT		;STACK POINTER FOULED UP		
496								
497	022574	022354	175776	CMP	#15,	STK1	;CHECK THE RTI ADDRESS ON THE STACK	
498	022622	001401		BEQ	,+4		;BRANCH IF OK	
499	022624	104000		HLT			;RTI ADDRESS NOT EQUAL TO #15	
500								
501	022626	026767	176044	175766	CMP	SMULER,	STK2	;CHECK THE PSW ON THE STACK
502	022614	001401		BEQ	,+4		;BRANCH IF OK	
503	022616	104000		HLT			;RTI PSW NOT EQUAL TO 200	
504								
505	022620	026767	176000	175756	CMP	RAND,C,	STK3	;CHECK THE DATA ON THE STACK
506	022626	001401		BEQ	,+4		;BRANCH IF OK	
507	022630	104000		HLT			;STK3 NOT EQUAL TO RAND,C	
508								
509	022632	026767	175770	175746	CMP	RAND,D,	STK4	;CHECK THE DATA ON THE STACK
510	022640	001401		BEQ	,+4		;BRANCH IF OK	
511	022642	104000		HLT			;STK4 NOT EQUAL TO RAND,D	
512								
513	022644	026767	175750	175736	CMP	RAND,A,	STK5	;CHECK THE DATA ON THE STACK
514	022652	001401		BEQ	,+4		;BRANCH IF OK	
515	022654	104000		HLT			;STK5 NOT EQUAL TO RAND,A	
516								
517	022656	026767	175740	175726	CMP	RAND,B,	STK6	;CHECK THE DATA ON THE STACK
518	022664	001401		BEQ	,+4		;BRANCH IF OK	
519	022666	104000		HLT			;STK6 NOT EQUAL TO RAND,B	
520								
521	022670	012716	002676	MOV	#35,	(SP)	;RESET THE STACK	
522	022674	000002		RTI			;RESTORE THE STATUS (T-BIT)	
523								
524	022676	104400		3\$: SCOPE				
525								

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 13
DBKEBA.P11 TEST 4: EXERCISE FDIV R3

526
527
528 ;TEST 4: EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
529 ; RAND,A,RAND,B / RAND,C,RAND,D = ANS1,ANS2
530 ; STACK POINTER = R3
531
532
533 202700 012723 000604 TST4: MOV #STACK0,R3 ;SET UP THE STACK POINTER
534 202704 004767 010670 JSR PC, PUSH R3 ;PUT THE DATA ON THE STACK
535
536 202710 000240 NOP
537 202712 075033 FDIV+ R3 ;FLOATING DIVIDE ON THE R3 STACK
538
539 202714 013767 177776 175672 1\$: MOV #PSW, SPSW ;SAVE PROCESSOR STATUS
540 202722 010367 175670 175660 2\$: MOV R3, SSP ;SAVE THE STACK POINTER
541 202726 026767 175726 175660 6\$: CMP SDIVPS, SPSW ;CHECK THE PROCESSOR STATUS
542 202734 001023 BNE 4\$;GO CHECK FOR ROUNDING ERROR
543
544 202736 105767 175652 TSTB SPSW ;CHECK FOR ERROR
545 202742 100464 BMI 2\$;BRANCH IF ERROR
546
547 202744 012767 000610 175716 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
548 202752 026767 175712 175636 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
549 202760 001401 BEQ +4 ;BRANCH IF OK
550 202762 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
551
552 202764 026767 175672 175616 CMP SDIV1, ANS1 ;CHECK THE ANSWER
553 202772 001004 BNE 4\$
554 202774 026767 175664 175610 CMP SDIV2, ANS2 ;CHECK THE ANSWER
555 203002 001515 BEQ 3\$
556 203004 032767 000020 175660 4\$: BIT #BIT4, RNDFLG ;CHECK THE ROUNDING FLAG
557 203012 001022 BNE 5\$
558 203014 052767 000020 175650 BIS #BIT4, RNDFLG ;SET ROUNDING FLAG
559 203022 062767 000001 175634 ADD #1, SDIV2 ;INCREMENT FORTRAN ANSWER
560 203030 005567 175626 ADC SDIV1 ;ADD CARRY
561 203034 102334 BVC 6\$;BRANCH IF NO OVERFLOW
562 203036 000257 CCC ;CLEAR ALL CONDITION CODES
563 203040 000262 SEV ;SET V-BIT
564 203042 013767 177776 175616 MOV #PSW, SDIVER ;SET UP PSW FOR OVERFLOW
565 203050 012767 000340 175602 MOV #340, SDIVPS ;SET UP TRAP PSW
566 203056 000723 BR 6\$;TRY IT AGAIN
567
568 203060 132767 000020 175605 5\$: BITB #BIT4, RNDFLG+1 ;CHECK "DEROUNDING" FLAG
569 203066 001010 BNE 7\$;BRANCH IF SET
570 203070 152767 000020 175575 BISB #BIT4, RNDFLG+1 ;SET "DEROUNDING" FLAG
571 203076 162767 000001 175560 SUB #1, SDIV2 ;RESTORE ORIGINAL ANSWER
572 203104 005667 175552 SBC SDIV1 ;SUBTRACT CARRY
573 203110 104000 HLT ;WRONG PSW OR ANSWER
574
575 203112 000451 BR 3\$
576
577 203114 012767 000604 175546 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
578 203122 026767 175542 175466 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
579 203130 001401 BEQ +4 ;BRANCH IF OK

580	003132	104000		HLT		;STACK POINTER FOULED UP	
581							
582	003134	022767	002714	175436	CMP	#1\$, STK1	;CHECK THE RTI ADDRESS ON THE STACK
583	003142	001401			BEQ	,+4	;BRANCH IF OK
584	003144	104000			HLT		;RTI ADDRESS NOT EQUAL TO #1\$
585							
586	003146	026767	175514	175426	CMP	SDIVER, STK2	;CHECK THE PSW ON THE STACK
587	003154	001401			BEQ	,+4	;BRANCH IF OK
588	003156	104000			HLT		;RTI PSW NOT EQUAL TO 200
589							
590	003160	026767	175440	175416	CMP	RAND,C, STK3	;CHECK THE DATA ON THE STACK
591	003166	001401			BEQ	,+4	;BRANCH IF OK
592	003170	104000			HLT		;STK3 NOT EQUAL TO RAND,C
593							
594	003172	026767	175430	175406	CMP	RAND,D, STK4	;CHECK THE DATA ON THE STACK
595	003200	001401			BEQ	,+4	;BRANCH IF OK
596	003202	104000			HLT		;STK4 NOT EQUAL TO RAND,D
597							
598	003204	026767	175410	175376	CMP	RAND,A, STK5	;CHECK THE DATA ON THE STACK
599	003212	001401			BEQ	,+4	;BRANCH IF OK
600	003214	104000			HLT		;STK5 NOT EQUAL TO RAND,A
601							
602	003216	026767	175400	175366	CMP	RAND,B, STK6	;CHECK THE DATA ON THE STACK
603	003224	001401			BEQ	,+4	;BRANCH IF OK
604	003226	104000			HLT		;STK6 NOT EQUAL TO RAND,B
605							
606	003230	012716	003236		MOV	#3\$, (SP)	;RESET THE STACK
607	003234	000002			RTI		;RESTORE THE STATUS (T-BIT)
608							
609	003236	104400			38:	SCOPE	
610							

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 15
DBKEBA.P11 TEST 5: EXERCISE FADD R4

611
612
613 ;*****
614 ;TEST 5: EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)
615 ; RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2
616 ; STACK POINTER = R4
617 ;*****
618 003240 012704 000604 TST5: MOV #STACK0,R4 ;SET UP THE STACK POINTER
619 003244 004767 010330 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
620
621 003250 000240 NOP
622 003252 075004 FADD+ R4 ;FLOATING ADD ON THE R4 STACK
623
624 003254 013767 177776 175332 1\$: MOV @PS, SPSW ;SAVE PROCESSOR STATUS
625 003262 010467 175330 175330 MOV R4, SSP ;SAVE THE STACK POINTER
626 003266 026767 175336 175320 CMP SADDP\$, SPSW ;CHECK THE PROCESSOR STATUS
627 003274 001401 BEQ ,+4 ;BRANCH IF OK
628 003276 104000 HLT ;PSW NOT EQUAL TO SADDP\$
629
630 003300 105767 175310 TSTB SPSW ;CHECK FOR ERROR
631 003304 100423 BMI 2\$;BRANCH IF ERROR
632
633 003306 012767 000610 175354 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
634 003314 026767 175350 175274 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
635 003322 001401 BEQ ,+4 ;BRANCH IF OK
636 003324 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
637
638 003326 026767 175300 175254 CMP SADD1, ANS1 ;CHECK THE ANSWER
639 003334 001401 BEQ ,+4 ;BRANCH IF OK
640 003336 104000 HLT ;LEFT HALF OF ANSWER WRONG
641
642 003340 026767 175270 175244 CMP SADD2, ANS2 ;CHECK THE ANSWER
643 003346 001401 BEQ ,+4 ;BRANCH IF OK
644 003350 104000 HLT ;RIGHT HALF OF ANSWER WRONG
645
646 003352 000451 BR 3\$
647
648 003354 012767 000604 175306 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
649 003362 026767 175302 175226 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
650 003370 001401 BEQ ,+4 ;BRANCH IF OK
651 003372 104000 HLT ;STACK POINTER FOULED UP
652
653 003374 022767 003254 175176 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
654 003402 001401 BEQ ,+4 ;BRANCH IF OK
655 003404 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1\$
656
657 003406 026767 175224 175166 CMP SADDER, STK2 ;CHECK THE PSW ON THE STACK
658 003414 001401 BEQ ,+4 ;BRANCH IF OK
659 003416 104000 HLT ;RTI PSW NOT EQUAL TO 200
660
661 003420 026767 175200 175156 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
662 003426 001401 BEQ ,+4 ;BRANCH IF OK
663 003430 104000 HLT ;STK3 NOT EQUAL TO RAND,C
664

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11140 PAGE 16
DBKEBA.P11 TEST 5: EXERCISE PADD R4

665 003432 026767 175170 175146 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
666 003440 001401 BEQ ,+4 ;BRANCH IF OK
667 003442 104000 HLT ;STK4 NOT EQUAL TO RAND,D
668
669 003444 026767 175150 175136 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
670 003452 001401 BEQ ,+4 ;BRANCH IF OK
671 003454 104000 HLT ;STK5 NOT EQUAL TO RAND,A
672
673 003456 026767 175140 175126 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
674 003464 001401 BEQ ,+4 ;BRANCH IF OK
675 003466 104000 HLT ;STK6 NOT EQUAL TO RAND,B
676
677 003470 012716 003476 MOV #3\$, (SP) ;RESET THE STACK
678 003474 000002 RTI ;RESTORE THE STATUS (T-BIT)
679
680 003476 104400 3\$: SCOPE
681
682
683 ****
684 ;TEST 6: EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
685 ; RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
686 ; STACK POINTER = R5
687 ****
688
689 003500 012705 000604 TST6: MOV #STACK0,R5 ;SET UP THE STACK POINTER
690 003504 004767 010070 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
691
692 003510 000240 NOP
693 003512 075015 FSUB+ R5 ;FLOATING SUBTRACT ON THE R5 STACK
694
695 003514 013767 177776 175072 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
696 003522 010567 175070 MOV R5, SSP ;SAVE THE STACK POINTER
697 003526 026767 175106 175060 CMP SSUBPS, SPSW ;CHECK THE PROCESSOR STATUS
698 003534 001401 BEQ ,+4 ;BRANCH IF OK
699 003536 104000 HLT ;PSW NOT EQUAL TO SSUBPS
700
701 003540 105767 175050 TSTB SPSW ;CHECK FOR ERROR
702 003544 100423 BMI 2\$;BRANCH IF ERROR
703
704 003546 012767 000610 175114 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
705 003554 026767 175110 175034 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
706 003562 001401 BEQ ,+4 ;BRANCH IF OK
707 003564 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
708
709 003566 026767 175050 175014 CMP SSUB1, ANS1 ;CHECK THE ANSWER
710 003574 001401 BEQ ,+4 ;BRANCH IF OK
711 003576 104000 HLT ;LEFT HALF OF ANSWER WRONG
712
713 003600 026767 175040 175004 CMP SSUB2, ANS2 ;CHECK THE ANSWER
714 003606 001401 BEQ ,+4 ;BRANCH IF OK
715 003610 104000 HLT ;RIGHT HALF OF ANSWER WRONG
716
717 003612 000451 BR 3\$
718

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
 DBKEBA.P11 TEST 6: EXERCISE FSUB R5 MACY11,620 22-AUG-72 11140 PAGE 17

719	003614	012767	000604	175046	2\$: MOV	#STACK0,SAVSTK	SAVE STACK ADDRESS FOR TYPING
720	003622	022767	175042	174766	CMP	SAVSTK, SSP	CHECK THE STACK POINTER
721	003630	001401			BEQ	,+4	BRANCH IF OK
722	003632	104000			HLT		STACK POINTER FOULED UP
723							
724	003634	022767	003514	174736	CMP	#1\$, STK1	CHECK THE RTI ADDRESS ON THE STACK
725	003642	001401			BEQ	,+4	BRANCH IF OK
726	003644	104000			HLT		RTI ADDRESS NOT EQUAL TO #1\$
727							
728	003646	026767	174774	174726	CMP	\$SUBR, STK2	CHECK THE PSW ON THE STACK
729	003654	001401			BEQ	,+4	BRANCH IF OK
730	003656	104000			HLT		RTI PSW NOT EQUAL TO 200
731							
732	003660	026767	174740	174716	CMP	RAND,C, STK3	CHECK THE DATA ON THE STACK
733	003666	001401			BEQ	,+4	BRANCH IF OK
734	003670	104000			HLT		STK3 NOT EQUAL TO RAND,C
735							
736	003672	026767	174730	174706	CMP	RAND,D, STK4	CHECK THE DATA ON THE STACK
737	003700	001401			BEQ	,+4	BRANCH IF OK
738	003702	104000			HLT		STK4 NOT EQUAL TO RAND,D
739							
740	003704	026767	174710	174676	CMP	RAND,A, STK5	CHECK THE DATA ON THE STACK
741	003712	001401			BEQ	,+4	BRANCH IF OK
742	003714	104000			HLT		STK5 NOT EQUAL TO RAND,A
743							
744	003716	026767	174700	174666	CMP	RAND,B, STK6	CHECK THE DATA ON THE STACK
745	003724	001401			BEQ	,+4	BRANCH IF OK
746	003726	104000			HLT		STK6 NOT EQUAL TO RAND,B
747							
748	003730	012716	003736		MOV	#3\$, (SP)	RESET THE STACK
749	003734	000002			RTI		RESTORE THE STATUS (T-BIT)
750							
751	003736	104400			3\$: SCOPE		
752							
753							
754							*****
755							TEST 7: EXERCISE FMUL (PDP-11 FLOATING MULTIPLY INSTRUCTION)
756							RAND,A,RAND,B * RAND,C,RAND,E = ANS1,ANS2
757							STACK POINTER = SP
758							*****
759							
760	003740	012706	000604		TST7: MOV	#STACK0,SP	SET UP THE STACK POINTER
761	003744	004767	007630		JSR	PC, PUSHR	PUT THE DATA ON THE STACK
762							
763	003750	000240			NOP		
764	003752	075026			FMUL+	SP	FLOATING MULTIPLY ON THE SP STACK
765							
766	003754	013767	177776	174632	1\$: MOV	@#PS, SPSW	SAVE PROCESSOR STATUS
767	003762	010667	174630		MOV	SP, SSP	SAVE THE STACK POINTER
768	003766	026767	174656	174620	CMP	\$MULPS, SPSW	CHECK THE PROCESSOR STATUS
769	003774	001401			BEQ	,+4	BRANCH IF OK
770	003776	104000			HLT		PSW NOT EQUAL TO \$MULPS
771							
772	004000	105767	174610		TST8	SPSW	CHECK FOR ERROR

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 7: EXERCISE FMUL SP MACY11,620 22-AUG-72 11140 PAGE 18

773	004004	100424		BMI	2\$	JBRANCH IF ERROR
774						
775	004006	012767	000610	174654	MOV	#STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
776	004014	026767	174650	174574	CMP	SAVSTK, SSP ;CHECK THE STACK POINTER
777	004022	001421			BEQ	,+4 ;BRANCH IF OK
778	004024	104020			HLT	;STACK POINTER NOT EQUAL TO #STACK4
779						
780	004026	026767	174620	174554	CMP	SMUL1, ANS1 ;CHECK THE ANSWER
781	004034	001421			BEQ	,+4 ;BRANCH IF OK
782	004036	104020			HLT	;LEFT HALF OF ANSWER WRONG
783						
784	004040	026767	174610	174544	CMP	SMUL2, ANS2 ;CHECK THE ANSWER
785	004046	001421			BEQ	,+4 ;BRANCH IF OK
786	004050	104020			HLT	;RIGHT HALF OF ANSWER WRONG
787						
788	004052	024646			CMP	= (SP), -(SP) ;RESTORE THE STACK
789	004054	000451			BR	3\$
790						
791	004056	012767	000600	174604 2\$:	MOV	#STK1, SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
792	004064	026767	174600	174524	CMP	SAVSTK, SSP ;CHECK THE STACK POINTER
793	004072	001401			BEQ	,+4 ;BRANCH IF OK
794	004074	104000			HLT	;STACK POINTER FOULED UP
795						
796	004076	022767	003754	174474	CMP	#1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
797	004104	001401			BEQ	,+4 ;BRANCH IF OK
798	004106	104000			HLT	;RTI ADDRESS NOT EQUAL TO #1\$
799						
800	004110	026767	174542	174464	CMP	SMULER, STK2 ;CHECK THE PSW ON THE STACK
801	004116	001401			BEQ	,+4 ;BRANCH IF OK
802	004120	104000			HLT	;RTI PSW NOT EQUAL TO 200
803						
804	004122	026767	174476	174454	CMP	RAND.C, STK3 ;CHECK THE DATA ON THE STACK
805	004130	001401			BEQ	,+4 ;BRANCH IF OK
806	004132	104000			HLT	;STK3 NOT EQUAL TO RAND.C
807						
808	004134	026767	174466	174444	CMP	RAND.D, STK4 ;CHECK THE DATA ON THE STACK
809	004142	001401			BEQ	,+4 ;BRANCH IF OK
810	004144	104000			HLT	;STK4 NOT EQUAL TO RAND.D
811						
812	004146	026767	174446	174434	CMP	RAND.A, STK5 ;CHECK THE DATA ON THE STACK
813	004154	001401			BEQ	,+4 ;BRANCH IF OK
814	004156	104000			HLT	;STK5 NOT EQUAL TO RAND.A
815						
816	004160	026767	174436	174424	CMP	RAND.B, STK6 ;CHECK THE DATA ON THE STACK
817	004166	001401			BEQ	,+4 ;BRANCH IF OK
818	004170	104000			HLT	;STK6 NOT EQUAL TO RAND.B
819						
820	004172	012716	004200		MOV	#3\$, (SP) ;RESET THE STACK
821	004176	000002			RTI	;RESTORE THE STATUS (T-BIT)
822						
823	004200	104400		3\$: SCOPE		
824						

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22AUG672 11140 PAGE 19
DBKEBA.P11 TEST 101 EXERCISE FDIV R0

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11140 PAGE 20
DBKEBA.P11 TEST 101 EXERCISE FDIV R0

879 004374 026767 174226 174204 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
880 004402 001401 BEQ ,+4 ;BRANCH IF OK
881 004404 104000 HLT ;STK4 NOT EQUAL TO RAND,D
882
883 004406 026767 174206 174174 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
884 004414 001401 BEQ ,+4 ;BRANCH IF OK
885 004416 104000 HLT ;STK5 NOT EQUAL TO RAND,A
886
887 004420 026767 174176 174164 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
888 004426 001401 BEQ ,+4 ;BRANCH IF OK
889 004430 104000 HLT ;STK6 NOT EQUAL TO RAND,B
890
891 004432 012716 004440 MOV #3\$, (SP) ;RESET THE STACK
892 004436 000002 RTI ;RESTORE THE STATUS (T-BIT)
893
894 004440 104400 3\$: SCOPE
895
896
897 ;*****
898 ;TEST 11: EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)
899 ;RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2
900 ;STACK POINTER = R1
901 ;*****
902
903 004442 012701 000604 TST11: MOV #STACK0,R1 ;SET UP THE STACK POINTER
904 004446 004767 007126 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
905
906 004452 000240 NOP
907 004454 075001 FADD# R1 ;FLOATING ADD ON THE R1 STACK
908
909 004456 013767 177776 174130 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
910 004464 010167 174126 MOV R1, SSP ;SAVE THE STACK POINTER
911 004470 026767 174134 174116 CMP SADDP\$, SPSW ;CHECK THE PROCESSOR STATUS
912 004476 001401 BEQ ,+4 ;BRANCH IF OK
913 004500 104000 HLT ;IPSW NOT EQUAL TO SADDP\$
914
915 004502 105767 174106 TSTB SPSW ;CHECK FOR ERROR
916 004506 100423 BMI 2\$;BRANCH IF ERROR
917
918 004510 012767 000610 174152 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
919 004516 026767 174146 174072 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
920 004524 001401 BEQ ,+4 ;BRANCH IF OK
921 004526 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
922
923 004530 026767 174076 174052 CMP SADD1, ANS1 ;CHECK THE ANSWER
924 004536 001401 BEQ ,+4 ;BRANCH IF OK
925 004540 104000 HLT ;LEFT HALF OF ANSWER WRONG
926
927 004542 026767 174066 174042 CMP SADD2, ANS2 ;CHECK THE ANSWER
928 004550 001401 BEQ ,+4 ;BRANCH IF OK
929 004552 104000 HLT ;RIGHT HALF OF ANSWER WRONG
930
931 004554 000451 BR 3\$
932

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
 DBKEBA.P11 TEST 111 EXERCISE FADD R1 MACY11,620 22-AUG-72 11:40 PAGE 21

933	004556	012767	000604	174104	2\$: MOV	#STACK0,SAVSTK	;SAVE STACK ADDRESS FOR TYPING
934	004564	026767	174100	174024	CMP	SAVSTK, SSP	;CHECK THE STACK POINTER
935	004572	001401			BEQ	,+4	;BRANCH IF OK
936	004574	104000			HLT		;STACK POINTER FOULED UP
937							
938	004576	022767	004456	173774	CMP	#1\$, STK1	;CHECK THE RTI ADDRESS ON THE STACK
939	004604	001401			BEQ	,+4	;BRANCH IF OK
940	004606	104000			HLT		;RTI ADDRESS NOT EQUAL TO #1\$
941							
942	004610	026767	174022	173764	CMP	\$ADDER, STK2	;CHECK THE PSW ON THE STACK
943	004616	001401			BEQ	,+4	;BRANCH IF OK
944	004620	104000			HLT		;RTI PSW NOT EQUAL TO 200
945							
946	004622	026767	173776	173754	CMP	RAND,C, STK3	;CHECK THE DATA ON THE STACK
947	004630	001401			BEQ	,+4	;BRANCH IF OK
948	004632	104000			HLT		;STK3 NOT EQUAL TO RAND,C
949							
950	004634	026767	173766	173744	CMP	RAND,D, STK4	;CHECK THE DATA ON THE STACK
951	004642	001401			BEQ	,+4	;BRANCH IF OK
952	004644	104000			HLT		;STK4 NOT EQUAL TO RAND,D
953							
954	004646	026767	173746	173734	CMP	RAND,A, STK5	;CHECK THE DATA ON THE STACK
955	004654	001401			BEQ	,+4	;BRANCH IF OK
956	004656	104000			HLT		;STK5 NOT EQUAL TO RAND,A
957							
958	004660	026767	173736	173724	CMP	RAND,B, STK6	;CHECK THE DATA ON THE STACK
959	004666	001401			BEQ	,+4	;BRANCH IF OK
960	004670	104000			HLT		;STK6 NOT EQUAL TO RAND,B
961							
962	004672	012716	004700		MOV	#3\$, (SP)	;RESET THE STACK
963	004676	000002			RTI		;RESTORE THE STATUS (T=BIT)
964							
965	004700	104400			3\$: SCOPE		
966							
967							
968							
969							
970							
971							
972							
973							
974	004702	012702	000604		TST121	MOV #STACK0,R2	;SET UP THE STACK POINTER
975	004706	004767	006666		JSR PC,	PUSHR	;PUT THE DATA ON THE STACK
976							
977	004712	000240			NOP		
978	004714	075012			FSUB+	R2	;FLOATING SUBTRACT ON THE R2 STACK
979							
980	004716	013767	177776	173670	1\$: MOV	@#PS, SPSW	;SAVE PROCESSOR STATUS
981	004724	010267	173666		MOV	R2, SSP	;SAVE THE STACK POINTER
982	004730	026767	173704	173656	CMP	\$SUBPS, SPSW	;CHECK THE PROCESSOR STATUS
983	004736	001401			BEQ	,+4	;BRANCH IF OK
984	004740	104000			HLT		;PSW NOT EQUAL TO \$SUBPS
985							
986	004742	105767	173646		TSTB	SPSW	;CHECK FOR ERROR

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 12: EXERCISE FSUB R2

MACY11,620 22-AUG-72 11140 PAGE 22

987	004746	100423		BMI	2\$	JBRANCH IF ERROR
988				MOV	#STACK4,SAVSTK	JSAVE PROPER STACK ADDRESS FOR TYPING
989	024750	012767	000610	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
990	024756	026767	173706	BEQ	,+4	JBRANCH IF OK
991	024764	001401		HLT		JSTACK POINTER NOT EQUAL TO #STACK4
992	024766	104000		CMP	SSUB1, ANS1	JCHECK THE ANSWER
993				BEQ	,+4	JBRANCH IF OK
994	024770	026767	173646	HLT		JLEFT HALF OF ANSWER WRONG
995	024776	001401		CMP	SSUB2, ANS2	JCHECK THE ANSWER
996	025000	104000		BEQ	,+4	JBRANCH IF OK
997				HLT		JRIGHT HALF OF ANSWER WRONG
998	025002	026767	173636	CMP		
999	025010	001401	173602	BEQ		
1000	025012	104000		HLT		
1001				BR	3\$	
1002	025014	000451		MOV	#STACK0,SAVSTK	JSAVE STACK ADDRESS FOR TYPING
1003				CMP	SAVSTK, SSR	JCHECK THE STACK POINTER
1004	025016	012767	000604	BEQ	,+4	JBRANCH IF OK
1005	025024	026767	173640	HLT		JSTACK POINTER FOULED UP
1006	025032	001401		CMP	#1\$, STK1	JCHECK THE RTI ADDRESS ON THE STACK
1007	025034	104000		BEQ	,+4	JBRANCH IF OK
1008				HLT		JRTI ADDRESS NOT EQUAL TO #1\$
1009	025036	022767	004716	CMP	SSUBER, STK2	JCHECK THE PSW ON THE STACK
1010	025044	001401	173534	BEQ	,+4	JBRANCH IF OK
1011	025046	104000		HLT		JRTI PSW NOT EQUAL TO 200
1012				CMP	RAND,C, STK3	JCHECK THE DATA ON THE STACK
1013	025050	026767	173572	BEQ	,+4	JBRANCH IF OK
1014	025056	001401	173524	HLT		JSTK3 NOT EQUAL TO RAND,C
1015	025060	104000		CMP	RAND,D, STK4	JCHECK THE DATA ON THE STACK
1016				BEQ	,+4	JBRANCH IF OK
1017	025062	026767	173536	HLT		JSTK4 NOT EQUAL TO RAND,D
1018	025070	001401	173514	CMP	RAND,A, STK5	JCHECK THE DATA ON THE STACK
1019	025072	104000		BEQ	,+4	JBRANCH IF OK
1020				HLT		JSTK5 NOT EQUAL TO RAND,A
1021	025074	026767	173526	CMP	RAND,B, STK6	JCHECK THE DATA ON THE STACK
1022	025102	001401	173504	BEQ	,+4	JBRANCH IF OK
1023	025104	104000		HLT		JSTK6 NOT EQUAL TO RAND,B
1024				CMP		
1025	025106	026767	173506	BEQ		
1026	025114	001401	173474	HLT		
1027	025116	104000		CMP		
1028				BEQ		
1029	025120	026767	173476	HLT		
1030	025126	001401	173464	CMP		
1031	025130	104000		BEQ		
1032				HLT		
1033	025132	012716	005140	MOV	#3\$, (SP)	JRESET THE STACK
1034	025136	000002		RTI		JRESTORE THE STATUS (T-BIT)
1035				SCOPE		
1036	025140	104400		3\$:		
1037						

MAINDEC-11-DBKEB-A KE11F (PDP=11 F1S) EXERCISER, MACY11,620 22-AUG-72 11140 PAGE 23
DBKEBA.P11 TEST 131 EXERCISE FMUL R3

```

1038
1039
1040
1041
1042
1043
1044
1045 205142 012703 000604 TST131 MOV #STACK0,R3 ;SET UP THE STACK POINTER
1046 205146 004767 006426 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1047
1048 205152 000240
1049 205154 075023 NOP
1050
1051 205156 013767 177776 173430 1$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
1052 205164 010367 173426 MOV R3, SSP ;SAVE THE STACK POINTER
1053 205170 026767 173454 173416 CMP SMULPS, SPSW ;CHECK THE PROCESSOR STATUS
1054 205176 001401 BEQ ,+4 ;BRANCH IF OK
1055 205200 104000 HLT ;PSW NOT EQUAL TO SMULPS
1056
1057 205202 105767 173406 TSTB SPSW ;CHECK FOR ERROR
1058 205206 100423 BMI 2$ ;BRANCH IF ERROR
1059
1060 205210 012767 000610 173452 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1061 205216 026767 173446 173372 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1062 205224 001401 BEQ ,+4 ;BRANCH IF OK
1063 205226 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1064
1065 205230 026767 173416 173352 CMP SMUL1, ANS1 ;CHECK THE ANSWER
1066 205236 001401 BEQ ,+4 ;BRANCH IF OK
1067 205240 104000 HLT ;LEFT HALF OF ANSWER WRONG
1068
1069 205242 026767 173406 173342 CMP SMUL2, ANS2 ;CHECK THE ANSWER
1070 205250 001401 BEQ ,+4 ;BRANCH IF OK
1071 205252 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1072
1073 205254 000451 BR 3$ ;RTI
1074
1075 205256 012767 000604 173404 2$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
1076 205264 026767 173400 173324 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1077 205272 001401 BEQ ,+4 ;BRANCH IF OK
1078 205274 104000 HLT ;STACK POINTER FOULED UP
1079
1080 205276 022767 005156 173274 CMP #1$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1081 205304 001401 BEQ ,+4 ;BRANCH IF OK
1082 205306 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1$
1083
1084 205310 026767 173342 173264 CMP SMULER, STK2 ;CHECK THE PSW ON THE STACK
1085 205316 001401 BEQ ,+4 ;BRANCH IF OK
1086 205320 104000 HLT ;RTI PSW NOT EQUAL TO 200
1087
1088 205322 026767 173276 173254 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1089 205330 001401 BEQ ,+4 ;BRANCH IF OK
1090 205332 104000 HLT ;STK3 NOT EQUAL TO RAND,C
1091

```

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 24
DBKEBA.P11 TEST 131 EXERCISE FMUL R3

1092 005334 026767 173266 173244 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1093 005342 001401 BEQ ,+4 ;BRANCH IF OK
1094 005344 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1095
1096 005346 026767 173246 173234 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1097 005354 001401 BEQ ,+4 ;BRANCH IF OK
1098 005356 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1099
1100 005360 026767 173236 173224 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1101 005366 001401 BEQ ,+4 ;BRANCH IF OK
1102 005370 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1103
1104 005372 012716 005400 MOV #3\$, (SP) ;RESET THE STACK
1105 005376 000002 RTI ;RESTORE THE STATUS (T-BIT)
1106
1107 005400 104400 3\$: SCOPE
1108
1109
1110 *****
1111 ;TEST 141 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
1112 ;RAND,A,RAND,B / RAND,C,RAND,D = ANS1,ANS2
1113 ;STACK POINTER = R4
1114 *****
1115
1116 005402 012704 000604 TST141 MOV #STACK0,R4 ;SET UP THE STACK POINTER
1117 005406 004767 000166 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1118
1119 005412 000240 NOP
1120 005414 075034 FDIV+ R4 ;FLOATING DIVIDE ON THE R4 STACK
1121
1122 005416 013767 177776 173170 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
1123 005424 010467 173166 MOV R4, SSP ;SAVE THE STACK POINTER
1124 005430 026767 173224 173156 CMP SDIVPS, SPSW ;CHECK THE PROCESSOR STATUS
1125 005436 001401 BEQ ,+4 ;BRANCH IF OK
1126 005440 104000 HLT ;PSW NOT EQUAL TO SDIVPS
1127
1128 005442 005767 173146 TSTB SPSW ;CHECK FOR ERROR
1129 005446 100423 BMI 2\$;BRANCH IF ERROR
1130
1131 005450 012767 000610 173212 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1132 005456 026767 173206 173132 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1133 005464 001401 BEQ ,+4 ;BRANCH IF OK
1134 005466 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1135
1136 005470 026767 173166 173112 CMP SDIV1, ANS1 ;CHECK THE ANSWER
1137 005476 001401 BEQ ,+4 ;BRANCH IF OK
1138 005500 104000 HLT ;LEFT HALF OF ANSWER WRONG
1139
1140 005502 026767 173156 173102 CMP SDIV2, ANS2 ;CHECK THE ANSWER
1141 005510 001401 BEQ ,+4 ;BRANCH IF OK
1142 005512 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1143
1144 005514 000451 BR 3\$
1145

MAINDEC-11-DBKEB-A KE11F (PDP=11 FIS) EXERCISER,
DBKEBA.P11 TEST 141 EXERCISE FDIV R4

MACY11,620 22-AUG-72 11:40 PAGE 25

1146	205516	012767	000604	173144	2\$: MOV	#STACK0,SAVSTK	;SAVE STACK ADDRESS FOR TYPING
1147	205524	026767	173140	173064	CMP	SAVSTK, SSP	;CHECK THE STACK POINTER
1148	205532	001401			BEQ	,+4	;BRANCH IF OK
1149	205534	104000			HLT		;STACK POINTER FOULED UP
1150							
1151	205536	022767	005416	173034	CMP	#1\$, STK1	;CHECK THE RTI ADDRESS ON THE STACK
1152	205544	001401			BEQ	,+4	;BRANCH IF OK
1153	205546	104000			HLT		;RTI ADDRESS NOT EQUAL TO #1\$
1154							
1155	205550	026767	173112	173024	CMP	SDIVER, STK2	;CHECK THE PSW ON THE STACK
1156	205556	001401			BEQ	,+4	;BRANCH IF OK
1157	205560	104000			HLT		;RTI PSW NOT EQUAL TO 200
1158							
1159	205562	026767	173036	173014	CMP	RAND,C, STK3	;CHECK THE DATA ON THE STACK
1160	205570	001401			BEQ	,+4	;BRANCH IF OK
1161	205572	104000			HLT		;STK3 NOT EQUAL TO RAND,C
1162							
1163	205574	026767	173026	173004	CMP	RAND,D, STK4	;CHECK THE DATA ON THE STACK
1164	205602	001401			BEQ	,+4	;BRANCH IF OK
1165	205604	104000			HLT		;STK4 NOT EQUAL TO RAND,D
1166							
1167	205606	026767	173006	172774	CMP	RAND,A, STK5	;CHECK THE DATA ON THE STACK
1168	205614	001401			BEQ	,+4	;BRANCH IF OK
1169	205616	104000			HLT		;STK5 NOT EQUAL TO RAND,A
1170							
1171	205620	026767	172776	172764	CMP	RAND,B, STK6	;CHECK THE DATA ON THE STACK
1172	205626	001401			BEQ	,+4	;BRANCH IF OK
1173	205630	104000			HLT		;STK6 NOT EQUAL TO RAND,B
1174							
1175	205632	012716	005640		MOV	#3\$, (SP)	;RESET THE STACK
1176	205636	000002			RTI		;RESTORE THE STATUS (T=BIT)
1177							
1178	205640	104400			3\$: SCOPE		
1179							
1180							
1181							*****
1182							;TEST 151 EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)
1183							; RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2
1184							; STACK POINTER = R5
1185							*****
1186							
1187	205642	012705	000604		TST151	MOV #STACK0,R5	;SET UP THE STACK POINTER
1188	205646	004767	005726			JSR PC, PUSH R	;PUT THE DATA ON THE STACK
1189							
1190	205652	000240				NOP	
1191	205654	075005				FADD+	R5
1192							;FLOATING ADD ON THE R5 STACK
1193	205656	013767	177776	172730	1\$: MOV	@#PS, SPSW	;SAVE PROCESSOR STATUS
1194	205664	010567	172726		MOV	R5, SSP	;SAVE THE STACK POINTER
1195	205670	026767	172734	172716	CMP	\$ADDPS, SPSW	;CHECK THE PROCESSOR STATUS
1196	205676	001401			BEQ	,+4	;BRANCH IF OK
1197	205700	104000			HLT		;PSW NOT EQUAL TO SADDPS
1198							
1199	205702	105767	172706		TSTB	SPSW	;CHECK FOR ERROR

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 15: EXERCISE FADD R5 MACY11,620 22-AUG-72 11:40 PAGE 26

1200	205706	100423		BMI	2\$	JBRANCH IF ERROR
1201				MOV	#STACK4,SAVSTK	JSAVE PROPER STACK ADDRESS FOR TYPING
1202	205710	012767	000610 172752	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1203	205716	026767	172746 172672	BEQ	,+4	JBRANCH IF OK
1204	205724	001401		HLT		JSTACK POINTER NOT EQUAL TO #STACK4
1205	205726	104000				
1206				CMP	SADD1, ANS1	JCHECK THE ANSWER
1207	205730	026767	172676 172652	BEQ	,+4	JBRANCH IF OK
1208	205736	001401		HLT		JLEFT HALF OF ANSWER WRONG
1209	205740	104000				
1210				CMP	SADD2, ANS2	JCHECK THE ANSWER
1211	205742	026767	172666 172642	BEQ	,+4	JBRANCH IF OK
1212	205750	001401		HLT		JRIGHT HALF OF ANSWER WRONG
1213	205752	104000				
1214				BR	3\$	
1215	205754	000451				
1216				MOV	#STACK0,SAVSTK	JSAVE STACK ADDRESS FOR TYPING
1217	205756	012767	000604 172704 2\$:	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1218	205764	026767	172700 172624	BEQ	,+4	JBRANCH IF OK
1219	205772	001401		HLT		JSTACK POINTER FOULED UP
1220	205774	104000				
1221				CMP	#1\$, STK1	JCHECK THE RTI ADDRESS ON THE STACK
1222	205776	022767	003656 172574	BEQ	,+4	JBRANCH IF OK
1223	206004	001401		HLT		JRTI ADDRESS NOT EQUAL TO #1\$
1224	206006	104000				
1225				CMP	SADDER, STK2	JCHECK THE PSW ON THE STACK
1226	206010	026767	172622 172564	BEQ	,+4	JBRANCH IF OK
1227	206016	001401		HLT		JRTI PSW NOT EQUAL TO 200
1228	206020	104000				
1229				CMP	RAND,C, STK3	JCHECK THE DATA ON THE STACK
1230	206022	026767	172576 172554	BEQ	,+4	JBRANCH IF OK
1231	206030	001401		HLT		JSTK3 NOT EQUAL TO RAND,C
1232	206032	104000				
1233				CMP	RAND,D, STK4	JCHECK THE DATA ON THE STACK
1234	206034	026767	172566 172544	BEQ	,+4	JBRANCH IF OK
1235	206042	001401		HLT		JSTK4 NOT EQUAL TO RAND,D
1236	206044	104000				
1237				CMP	RAND,A, STK5	JCHECK THE DATA ON THE STACK
1238	206046	026767	172546 172534	BEQ	,+4	JBRANCH IF OK
1239	206054	001401		HLT		JSTK5 NOT EQUAL TO RAND,A
1240	206056	104000				
1241				CMP	RAND,B, STK6	JCHECK THE DATA ON THE STACK
1242	206060	026767	172536 172524	BEQ	,+4	JBRANCH IF OK
1243	206066	001401		HLT		JSTK6 NOT EQUAL TO RAND,B
1244	206070	104000				
1245				MOV	#3\$, (SP)	JRESET THE STACK
1246	206072	012716	006100	RTI		JRESTORE THE STATUS (T-BIT)
1247	206076	000002				
1248				3\$:	SCOPE	
1249	206100	104400				
1250						

MAINDEC-11=DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22=AUG=72 11:40 PAGE 27
DBKEBA.P11 TEST 16: EXERCISE FSUB SP

1251
1252
1253 ;*****
1254 ;TEST 16: EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
1255 ; RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
1256 ; STACK POINTER = SP
1257 ;*****
1258 006102 012706 000604 TST16I MOV #STACK0,SP ;SET UP THE STACK POINTER
1259 006106 004767 005466 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1260
1261 006112 000240 NOP
1262 006114 075016 FSUB+ SP ;FLOATING SUBTRACT ON THE SP STACK
1263
1264 006116 013767 177776 172470 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
1265 006124 010667 172466 MOV SP, SSP ;SAVE THE STACK POINTER
1266 006130 026767 172504 172456 CMP \$SUBPS, SPSW ;CHECK THE PROCESSOR STATUS
1267 006136 001401 BEQ ,+4 ;BRANCH IF OK
1268 006140 104000 HLT ;PSW NOT EQUAL TO \$SUBPS
1269
1270 006142 105767 172446 TSTB SPSW ;CHECK FOR ERROR
1271 006146 100424 BMI 2\$;BRANCH IF ERROR
1272
1273 006150 012767 000610 172512 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1274 006156 026767 172506 172432 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1275 006164 001401 BEQ ,+4 ;BRANCH IF OK
1276 006166 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1277
1278 006170 026767 172446 172412 CMP SSUB1, ANS1 ;CHECK THE ANSWER
1279 006176 001401 BEQ ,+4 ;BRANCH IF OK
1280 006200 104000 HLT ;LEFT HALF OF ANSWER WRONG
1281
1282 006202 026767 172436 172402 CMP SSUB2, ANS2 ;CHECK THE ANSWER
1283 006210 001401 BEQ ,+4 ;BRANCH IF OK
1284 006212 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1285
1286 006214 024646 CMP -(SP), -(SP) ;RESTORE THE STACK
1287 006216 000451 BR 3\$
1288
1289 006220 012767 000600 172442 2\$: MOV #STK1, SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1290 006226 026767 172436 172362 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1291 006234 001401 BEQ ,+4 ;BRANCH IF OK
1292 006236 104000 HLT ;STACK POINTER FOULED UP
1293
1294 006240 022767 006116 172332 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1295 006246 001401 BEQ ,+4 ;BRANCH IF OK
1296 006250 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1\$
1297
1298 006252 026767 172370 172322 CMP SSUBER, STK2 ;CHECK THE PSW ON THE STACK
1299 006260 001401 BEQ ,+4 ;BRANCH IF OK
1300 006262 104000 HLT ;RTI PSW NOT EQUAL TO 200
1301
1302 006264 026767 172334 172312 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1303 006272 001401 BEQ ,+4 ;BRANCH IF OK
1304 006274 104000 HLT ;STK3 NOT EQUAL TO RAND,C

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 28
DBKEBA.P11 TEST 16: EXERCISE FSUB SP

1305
1306 006276 026767 172324 172302 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1307 006304 001401 BEQ ,+4 ;BRANCH IF OK
1308 006306 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1309
1310 006310 026767 172304 172272 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1311 006316 001401 BEQ ,+4 ;BRANCH IF OK
1312 006320 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1313
1314 006322 026767 172274 172262 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1315 006330 001401 BEQ ,+4 ;BRANCH IF OK
1316 006332 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1317
1318 006334 012716 006342 MOV #35, (SP) ;RESET THE STACK
1319 006340 000002 RTI ;RESTORE THE STATUS (T-BIT)
1320
1321 006342 104400 3\$: SCOPE
1322
1323
1324 *****
1325 ;TEST 17! EXERCISE FMUL (PDP-11 FLOATING MULTIPLY INSTRUCTION)
1326 ; RAND,A,RAND,B * RAND,C,RAND,D = ANS1,ANS2
1327 ; STACK POINTER = R0
1328 *****
1329
1330 006344 012700 000604 TST171 MOV #STACK0,R0 ;SET UP THE STACK POINTER
1331 006350 004767 005224 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1332
1333 006354 000240 NOP
1334 006356 075020 FMUL+ R0 ;FLOATING MULTIPLY ON THE R0 STACK
1335
1336 006360 013767 177776 172226 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
1337 006366 010067 172224 MOV R0, SSP ;SAVE THE STACK POINTER
1338 006372 026767 172252 172214 CMP \$MULPS, SPSW ;CHECK THE PROCESSOR STATUS
1339 006400 001401 BEQ ,+4 ;BRANCH IF OK
1340 006402 104000 HLT ;PSW NOT EQUAL TO SMULPS
1341
1342 006404 105767 172204 TSTB SPSW ;CHECK FOR ERROR
1343 006410 100423 BMI 2\$;BRANCH IF ERROR
1344
1345 006412 012767 000610 172250 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1346 006420 026767 172244 172170 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1347 006426 001401 BEQ ,+4 ;BRANCH IF OK
1348 006430 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1349
1350 006432 026767 172214 172150 CMP SMUL1, ANS1 ;CHECK THE ANSWER
1351 006440 001401 BEQ ,+4 ;BRANCH IF OK
1352 006442 104000 HLT ;LEFT HALF OF ANSWER WRONG
1353
1354 006444 026767 172204 172140 CMP SMUL2, ANS2 ;CHECK THE ANSWER
1355 006452 001401 BEQ ,+4 ;BRANCH IF OK
1356 006454 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1357
1358 006456 000451 BR 3\$

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 17: EXERCISE FMUL R0

MACY11,620 22-AUG-72 11:40 PAGE 29

1359
1360 006460 012767 000604 172202 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
1361 006466 026767 172176 172122 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1362 006474 001401 BEQ ,+4 ;BRANCH IF OK
1363 006476 104000 HLT ;STACK POINTER FOULED UP
1364
1365 006500 022767 006360 172072 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1366 006506 001401 BEQ ,+4 ;BRANCH IF OK
1367 006510 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1\$
1368
1369 006512 026767 172140 172062 CMP \$MULER, STK2 ;CHECK THE PSW ON THE STACK
1370 006520 001401 BEQ ,+4 ;BRANCH IF OK
1371 006522 104000 HLT ;RTI PSW NOT EQUAL TO 200
1372
1373 006524 026767 172074 172052 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1374 006532 001401 BEQ ,+4 ;BRANCH IF OK
1375 006534 104000 HLT ;STK3 NOT EQUAL TO RAND,C
1376
1377 006536 026767 172064 172042 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1378 006544 001401 BEQ ,+4 ;BRANCH IF OK
1379 006546 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1380
1381 006550 026767 172044 172032 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1382 006556 001401 BEQ ,+4 ;BRANCH IF OK
1383 006560 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1384
1385 006562 026767 172034 172022 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1386 006570 001401 BEQ ,+4 ;BRANCH IF OK
1387 006572 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1388
1389 006574 012716 006602 MOV #3\$, (SP) ;RESET THE STACK
1390 006600 000002 RTI ;RESTORE THE STATUS (T-BIT)
1391
1392 006602 104400 3\$: SCOPE

1393
1394
1395 *****
1396 ;TEST 201 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
1397 ; RAND,A,RAND,B / RAND,C,RAND,D = ANS1,ANS2
1398 ; STACK POINTER = R1
1399 *****
1400
1401 006604 012701 000604 TST201 MOV #STACK0,R1 ;SET JP THE STACK POINTER
1402 006610 004767 004764 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1403
1404 006614 000240 NOP
1405 006616 075031 FDIV+ R1 ;FLOATING DIVIDE ON THE R1 STACK
1406
1407 006620 013767 177776 171766 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
1408 006626 010167 171764 MOV R1, SSP ;SAVE THE STACK POINTER
1409 006632 026767 172022 171754 CMP SDIVPS, SPSW ;CHECK THE PROCESSOR STATUS
1410 006640 001401 BEQ ,+4 ;BRANCH IF OK
1411 006642 104000 HLT ;PSW NOT EQUAL TO SDIVPS
1412

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCiser,
DBKEBA.P11 TEST 20: EXERCISE FDIV R1 MACY11,620 22-AUG-72 11140 PAGE 30

1413	006644	105767	171744	TSTB	SPSW	JCHECK FOR ERROR
1414	006650	100423		BMI	2\$	JBRANCH IF ERROR
1415						
1416	006652	012767	000610 172010	MOV	#STACK4,SAVSTK	JSAVE PROPER STACK ADDRESS FOR TYPING
1417	006660	026767	172004 171730	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1418	006666	001401		BEQ	,+4	JBRANCH IF OK
1419	006670	104000		HLT		JSTACK POINTER NOT EQUAL TO #STACK4
1420						
1421	006672	026767	171764 171710	CMP	\$DIV1, ANS1	JCHECK THE ANSWER
1422	006700	001401		BEQ	,+4	JBRANCH IF OK
1423	006702	104000		HLT		JLEFT HALF OF ANSWER WRONG
1424						
1425	006704	026767	171754 171700	CMP	\$DIV2, ANS2	JCHECK THE ANSWER
1426	006712	001401		BEQ	,+4	JBRANCH IF OK
1427	006714	104000		HLT		JRIGHT HALF OF ANSWER WRONG
1428						
1429	006716	000451		BR	3\$	
1430						
1431	006720	012767	000604 171742 2\$:	MOV	#STACK0,SAVSTK	JSAVE STACK ADDRESS FOR TYPING
1432	006726	026767	171736 171662	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1433	006734	001401		BEQ	,+4	JBRANCH IF OK
1434	006736	104000		HLT		JSTACK POINTER FOULED UP
1435						
1436	006740	022767	006620 171632	CMP	#1\$, STK1	JCHECK THE RTI ADDRESS ON THE STACK
1437	006746	001401		BEQ	,+4	JBRANCH IF OK
1438	006750	104000		HLT		JRTI ADDRESS NOT EQUAL TO #1\$
1439						
1440	006752	026767	171710 171622	CMP	\$DIVER, STK2	JCHECK THE PSW ON THE STACK
1441	006760	001401		BEQ	,+4	JBRANCH IF OK
1442	006762	104000		HLT		JRTI PSW NOT EQUAL TO 200
1443						
1444	006764	026767	171634 171612	CMP	RAND,C, STK3	JCHECK THE DATA ON THE STACK
1445	006772	001401		BEQ	,+4	JBRANCH IF OK
1446	006774	104000		HLT		JSTK3 NOT EQUAL TO RAND,C
1447						
1448	006776	026767	171624 171602	CMP	RAND,D, STK4	JCHECK THE DATA ON THE STACK
1449	007004	001401		BEQ	,+4	JBRANCH IF OK
1450	007006	104000		HLT		JSTK4 NOT EQUAL TO RAND,D
1451						
1452	007010	026767	171604 171572	CMP	RAND,A, STK5	JCHECK THE DATA ON THE STACK
1453	007016	001401		BEQ	,+4	JBRANCH IF OK
1454	007020	104000		HLT		JSTK5 NOT EQUAL TO RAND,A
1455						
1456	007022	026767	171574 171562	CMP	RAND,B, STK6	JCHECK THE DATA ON THE STACK
1457	007030	001401		BEQ	,+4	JBRANCH IF OK
1458	007032	104000		HLT		JSTK6 NOT EQUAL TO RAND,B
1459						
1460	007034	012716	007042	MOV	#3\$, (SP)	JRESET THE STACK
1461	007040	000002		RTI		JRESTORE THE STATUS (T-BIT)
1462						
1463	007042	104400		3\$: SCOPE		
1464						

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11140 PAGE 31
DBKEBA.P11 TEST 21; EXERCISE FADD R2

1465
1466
1467 ;*****
1468 ;TEST 21; EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)
1469 ;RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2
1470 ;STACK POINTER = R2
1471 ;*****
1472 007044 012702 000604 TST211 MOV #STACK0,R2 ;SET UP THE STACK POINTER
1473 007050 004767 004524 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1474
1475 007054 000240 NOP
1476 007056 075002 FADD+ R2 ;FLOATING ADD ON THE R2 STACK
1477
1478 007060 013767 177776 171526 1\$: MOV #PSW, SPSW ;SAVE PROCESSOR STATUS
1479 007066 010267 171524 MOV R2, SSP ;SAVE THE STACK POINTER
1480 007072 026767 171532 171514 CMP SADDPS, SPSW ;CHECK THE PROCESSOR STATUS
1481 007100 001401 BEQ ,+4 ;BRANCH IF OK
1482 007102 104000 HLT ;PSW NOT EQUAL TO SADDPS
1483
1484 007104 105767 171504 TSTB SPSW ;CHECK FOR ERROR
1485 007110 100423 BMI 2\$;BRANCH IF ERROR
1486
1487 007112 012767 000610 171550 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1488 007120 026767 171544 171470 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1489 007126 001401 BEQ ,+4 ;BRANCH IF OK
1490 007130 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1491
1492 007132 026767 171474 171450 CMP SADD1, ANS1 ;CHECK THE ANSWER
1493 007140 001401 BEQ ,+4 ;BRANCH IF OK
1494 007142 104000 HLT ;LEFT HALF OF ANSWER WRONG
1495
1496 007144 026767 171464 171440 CMP SADD2, ANS2 ;CHECK THE ANSWER
1497 007152 001401 BEQ ,+4 ;BRANCH IF OK
1498 007154 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1499
1500 007156 000451 BR 3\$
1501
1502 007160 012767 000604 171502 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
1503 007166 026767 171476 171422 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1504 007174 001401 BEQ ,+4 ;BRANCH IF OK
1505 007176 104000 HLT ;STACK POINTER FOULED UP
1506
1507 007200 022767 007060 171372 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1508 007206 001401 BEQ ,+4 ;BRANCH IF OK
1509 007210 104000 HLT ;RTI PSH NOT EQUAL TO #1\$
1510
1511 007212 026767 171420 171362 CMP SADDER, STK2 ;CHECK THE PSW ON THE STACK
1512 007220 001401 BEQ ,+4 ;BRANCH IF OK
1513 007222 104000 HLT ;RTI PSH NOT EQUAL TO 200
1514
1515 007224 026767 171374 171352 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1516 007232 001401 BEQ ,+4 ;BRANCH IF OK
1517 007234 104000 HLT ;STK3 NOT EQUAL TO RAND,C
1518

MAINDEC-11-DBKEB-A KE11F (PDP-11 F1S) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 32
DBKEBA.P11 TEST 21: EXERCISE FADD R2

1519 007236 026767 171364 171342 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1520 007244 001401 BEQ ,+4 ;BRANCH IF OK
1521 007246 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1522
1523 007250 026767 171344 171332 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1524 007256 001401 BEQ ,+4 ;BRANCH IF OK
1525 007260 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1526
1527 007262 026767 171334 171322 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1528 007270 001401 BEQ ,+4 ;BRANCH IF OK
1529 007272 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1530
1531 007274 012716 007302 MOV #3\$, (SP) ;RESET THE STACK
1532 007300 000002 RTI ;RESTORE THE STATUS (T=BIT)
1533
1534 007302 104400 3\$: SCOPE
1535
1536
1537 ;*****
1538 ;TEST 221 EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
1539 ;RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
1540 ;STACK POINTER = R3
1541 ;*****
1542
1543 007304 012703 000604 TST221 MOV #STACK0,R3 ;SET UP THE STACK POINTER
1544 007310 004767 004264 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1545
1546 007314 000240 NOP
1547 007316 075013 FSUB+ R3 ;FLOATING SUBTRACT ON THE R3 STACK
1548
1549 007320 013767 177776 171266 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
1550 007326 010367 171264 MOV R3, SSP ;SAVE THE STACK POINTER
1551 007332 026767 171302 171254 CMP SSUBPS, SPSW ;CHECK THE PROCESSOR STATUS
1552 007340 001401 BEQ ,+4 ;BRANCH IF OK
1553 007342 104000 HLT ;PSW NOT EQUAL TO SSUBPS
1554
1555 007344 105767 171244 TSTB SPSW ;CHECK FOR ERROR
1556 007350 100423 BMI 2\$;BRANCH IF ERROR
1557
1558 007352 012767 000610 171310 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1559 007360 026767 171304 171230 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1560 007366 001401 BEQ ,+4 ;BRANCH IF OK
1561 007370 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1562
1563 007372 026767 171244 171210 CMP SSUB1, ANS1 ;CHECK THE ANSWER
1564 007400 001401 BEQ ,+4 ;BRANCH IF OK
1565 007402 104000 HLT ;LEFT HALF OF ANSWER WRONG
1566
1567 007404 026767 171234 171200 CMP SSUB2, ANS2 ;CHECK THE ANSWER
1568 007412 001401 BEQ ,+4 ;BRANCH IF OK
1569 007414 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1570
1571 007416 000451 BR 3\$
1572

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER,
 DBKEBA.P11 TEST 22: EXERCISE FSUB R3 MACY11,620 22-AUG-72 11:40 PAGE 33

1573	007420	012767	000604	171242	2\$: MOV	#STACK0,SAVSTK	;SAVE STACK ADDRESS FOR TYPING
1574	007426	026767	171236	171162	CMP	SAVSTK, SSP	;CHECK THE STACK POINTER
1575	007434	001401			BEQ	,+4	;BRANCH IF OK
1576	007436	104000			HLT		;STACK POINTER FOULED UP
1577							
1578	007440	022767	007320	171132	CMP	#1\$, STK1	;CHECK THE RTI ADDRESS ON THE STACK
1579	007446	001401			BEQ	,+4	;BRANCH IF OK
1580	007450	104000			HLT		;RTI ADDRESS NOT EQUAL TO #1\$
1581							
1582	007452	026767	171170	171122	CMP	\$SUBR, STK2	;CHECK THE PSW ON THE STACK
1583	007460	001401			BEQ	,+4	;BRANCH IF OK
1584	007462	104000			HLT		;RTI PSW NOT EQUAL TO 200
1585							
1586	007464	026767	171134	171112	CMP	RAND,C, STK3	;CHECK THE DATA ON THE STACK
1587	007472	001401			BEQ	,+4	;BRANCH IF OK
1588	007474	104000			HLT		;STK3 NOT EQUAL TO RAND,C
1589							
1590	007476	026767	171124	171102	CMP	RAND,D, STK4	;CHECK THE DATA ON THE STACK
1591	007504	001401			BEQ	,+4	;BRANCH IF OK
1592	007506	104000			HLT		;STK4 NOT EQUAL TO RAND,D
1593							
1594	007510	026767	171104	171072	CMP	RAND,A, STK5	;CHECK THE DATA ON THE STACK
1595	007516	001401			BEQ	,+4	;BRANCH IF OK
1596	007520	104000			HLT		;STK5 NOT EQUAL TO RAND,A
1597							
1598	007522	026767	171074	171062	CMP	RAND,B, STK6	;CHECK THE DATA ON THE STACK
1599	007530	001401			BEQ	,+4	;BRANCH IF OK
1600	007532	104000			HLT		;STK6 NOT EQUAL TO RAND,B
1601							
1602	007534	012716	007542		MOV	#3\$, (\$P)	;RESET THE STACK
1603	007540	000002			RTI		;RESTORE THE STATUS (T6BIT)
1604							
1605	007542	104400			3\$: SCOPE		
1606							
1607							
1608							*****
1609							;TEST 23! EXERCISE FMUL (PDP-11 FLOATING MULTIPLY INSTRUCTION)
1610							;RAND,A,RAND,B * RAND,C,RAND,D = ANS1,ANS2
1611							;STACK POINTER = R4
1612							*****
1613							
1614	007544	012704	000604		TST23: JSR	#STACK0,R4 PC, PUSHR	;SET UP THE STACK POINTER ;PUT THE DATA ON THE STACK
1615	007550	004767	004024				
1616							
1617	007554	000240			NOP		
1618	007556	075024			FMUL+	R4	;FLOATING MULTIPLY ON THE R4 STACK
1619							
1620	007560	013767	177776	171026	1\$: MOV	#PS, SPSW	;SAVE PROCESSOR STATUS
1621	007566	010467	171024		MOV	R4, SSP	;SAVE THE STACK POINTER
1622	007572	026767	171052	171014	CMP	\$MULPS, SPSW	;CHECK THE PROCESSOR STATUS
1623	007600	001401			BEQ	,+4	;BRANCH IF OK
1624	007602	104000			HLT		;PSW NOT EQUAL TO SMULPS
1625							
1626	007604	105767	171004		TSTB	SPSW	;CHECK FOR ERROR

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 23; EXERCISE FMUL R4 MACY11,620 22-AUG-72 11140 PAGE 34

1627	007610	100423			BMI	2\$	IBRANCH IF ERROR
1628					MOV	#STACK4,SAVSTK	ISAVE PROPER STACK ADDRESS FOR TYPING
1629	007612	012767	000610	171050	CMP	SAVSTK, SSP	ICHECK THE STACK POINTER
1630	007620	026767	171044	170770	BEQ	,+4	IBRANCH IF OK
1631	007626	001401			HLT		ISTACK POINTER NOT EQUAL TO #STACK4
1632	007630	104000			CMP	SMUL1, ANS1	ICHECK THE ANSWER
1633					BEQ	,+4	IBRANCH IF OK
1634	007632	026767	171014	170750	HLT		ILEFT HALF OF ANSWER WRONG
1635	007640	001401			CMP	SMUL2, ANS2	ICHECK THE ANSWER
1636	007642	104000			BEQ	,+4	IBRANCH IF OK
1637					HLT		IRIGHT HALF OF ANSWER WRONG
1638	007644	026767	171004	170740	BR	3\$	
1639	007652	001401			MOV	#STACK0,SAVSTK	ISAVE STACK ADDRESS FOR TYPING
1640	007654	104000			CMP	SAVSTK, SSP	ICHECK THE STACK POINTER
1641					BEQ	,+4	IBRANCH IF OK
1642	007656	000451			HLT		ISTACK POINTER FOULED UP
1643					CMP	#1\$, STK1	ICHECK THE RTI ADDRESS ON THE STACK
1644	007660	012767	000604	171002	BEQ	,+4	IBRANCH IF OK
1645	007666	026767	170776	170722	HLT		IRTI ADDRESS NOT EQUAL TO #1\$
1646	007674	001401			CMP	SMULER, STK2	ICHECK THE PSW ON THE STACK
1647	007676	104000			BEQ	,+4	IBRANCH IF OK
1648					HLT		IRTI PSW NOT EQUAL TO 200
1649	007700	022767	007560	170672	CMP	RAND,C, STK3	ICHECK THE DATA ON THE STACK
1650	007706	001401			BEQ	,+4	IBRANCH IF OK
1651	007710	104000			HLT		ISTK3 NOT EQUAL TO RAND,C
1652					CMP	RAND,D, STK4	ICHECK THE DATA ON THE STACK
1653	007712	026767	170740	170662	BEQ	,+4	IBRANCH IF OK
1654	007720	001401			HLT		ISTK4 NOT EQUAL TO RAND,D
1655	007722	104000			CMP	RAND,A, STK5	ICHECK THE DATA ON THE STACK
1656					BEQ	,+4	IBRANCH IF OK
1657	007724	026767	170674	170652	HLT		ISTK5 NOT EQUAL TO RAND,A
1658	007732	001401			CMP	RAND,B, STK6	ICHECK THE DATA ON THE STACK
1659	007734	104000			BEQ	,+4	IBRANCH IF OK
1660					HLT		ISTK6 NOT EQUAL TO RAND,B
1661	007736	026767	170664	170642	CMP	#3\$, (SP)	IRESET THE STACK
1662	007744	001401			BEQ		IRESTORE THE STATUS (T-BIT)
1663	007746	104000			HLT		
1664					CMP		
1665	007750	026767	170644	170632	BEQ		
1666	007756	001401			HLT		
1667	007760	104000			CMP		
1668					BEQ		
1669	007762	026767	170634	170622	HLT		
1670	007770	001401			CMP		
1671	007772	104000			BEQ		
1672					HLT		
1673	007774	012716	010002		MOV		
1674	010000	000002			RTI		
1675					SCOPE		
1676	010002	104400			3\$:		
1677							

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 35
 DBKEBA.P11 TEST 24: EXERCISE FDIV R5

```

1678
1679
1680
1681
1682
1683
1684
1685 210004 012705 000604 TST241 MOV #STACK0,R5 ;SET UP THE STACK POINTER
1686 210010 004767 003564 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1687
1688 210014 000240 NOP
1689 210016 075035 FDIV+ R5 ;FLOATING DIVIDE ON THE R5 STACK
1690
1691 210020 013767 177776 170566 1$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
1692 210026 010567 170564 MOV R5, SSP ;SAVE THE STACK POINTER
1693 210032 026767 170622 170554 CMP SDIVPS, SPSW ;CHECK THE PROCESSOR STATUS
1694 210040 001401 BEQ ,+4 ;BRANCH IF OK
1695 210042 104000 HLT ;PSW NOT EQUAL TO SDIVPS
1696
1697 210044 105767 170544 TSTB SPSW ;CHECK FOR ERROR
1698 210050 100423 BMI 2$ ;BRANCH IF ERROR
1699
1700 210052 012767 000610 170610 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1701 210060 026767 170604 170530 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1702 210066 001401 BEQ ,+4 ;BRANCH IF OK
1703 210070 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1704
1705 210072 026767 170564 170510 CMP SDIV1, ANS1 ;CHECK THE ANSWER
1706 210100 001401 BEQ ,+4 ;BRANCH IF OK
1707 210102 104000 HLT ;LEFT HALF OF ANSWER WRONG
1708
1709 210104 026767 170554 170500 CMP SDIV2, ANS2 ;CHECK THE ANSWER
1710 210112 001401 BEQ ,+4 ;BRANCH IF OK
1711 210114 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1712
1713 210116 000451 BR 3$ ;TYPING
1714
1715 210120 012767 000604 170542 2$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
1716 210126 026767 170536 170462 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1717 210134 001401 BEQ ,+4 ;BRANCH IF OK
1718 210136 104000 HLT ;STACK POINTER FOULED UP
1719
1720 210140 022767 010020 170432 CMP #1$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1721 210146 001401 BEQ ,+4 ;BRANCH IF OK
1722 210150 104000 HLT ;RTI PSW NOT EQUAL TO #1$
1723
1724 210152 026767 170510 170422 CMP SDIVER, STK2 ;CHECK THE PSW ON THE STACK
1725 210160 001401 BEQ ,+4 ;BRANCH IF OK
1726 210162 104000 HLT ;RTI PSW NOT EQUAL TO 200
1727
1728 210164 026767 170434 170412 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1729 210172 001401 BEQ ,+4 ;BRANCH IF OK
1730 210174 104000 HLT ;STK3 NOT EQUAL TO RAND,C
1731

```

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 36
DBKEBA.P11 TEST 24: EXERCISE FDIV R5

1732 210176 026767 170424 170402 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1733 210204 001401 BEQ ,+4 ;BRANCH IF OK
1734 210206 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1735
1736 210210 026767 170404 170372 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1737 210216 001401 BEQ ,+4 ;BRANCH IF OK
1738 210220 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1739
1740 210222 026767 170374 170362 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1741 210230 001401 BEQ ,+4 ;BRANCH IF OK
1742 210232 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1743
1744 210234 012716 010242 MOV #3\$, (SP) ;RESET THE STACK
1745 210240 000002 RTI ;RESTORE THE STATUS (T=BIT)
1746
1747 210242 104400 3\$: SCOPE
1748
1749
1750 *****
1751 ;TEST 25: EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)
1752 ; RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2
1753 ; STACK POINTER = SP
1754 *****
1755
1756 210244 012706 000604 TST251 MOV #STACK0,SP ;SET UP THE STACK POINTER
1757 210250 004767 003324 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1758
1759 210254 000240 NOP
1760 210256 075006 FADD+ SP ;FLOATING ADD ON THE SP STACK
1761
1762 210260 013767 177776 170326 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
1763 210266 010667 170324 MOV SP, SSP ;SAVE THE STACK POINTER
1764 210272 026767 170332 170314 CMP SADDPS, SPSW ;CHECK THE PROCESSOR STATUS
1765 210300 001401 BEQ ,+4 ;BRANCH IF OK
1766 210302 104000 HLT ;PSW NOT EQUAL TO SADDPS
1767
1768 210304 105767 170304 TSTB SPSW ;CHECK FOR ERROR
1769 210310 100424 BMI 2\$;BRANCH IF ERROR
1770
1771 210312 012767 000610 170350 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1772 210320 026767 170344 170270 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1773 210326 001401 BEQ ,+4 ;BRANCH IF OK
1774 210330 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
1775
1776 210332 026767 170274 170250 CMP SADD1, ANS1 ;CHECK THE ANSWER
1777 210340 001401 BEQ ,+4 ;BRANCH IF OK
1778 210342 104000 HLT ;LEFT HALF OF ANSWER WRONG
1779
1780 210344 026767 170264 170240 CMP SADD2, ANS2 ;CHECK THE ANSWER
1781 210352 001401 BEQ ,+4 ;BRANCH IF OK
1782 210354 104000 HLT ;RIGHT HALF OF ANSWER WRONG
1783
1784 210356 024646 CMP -(SP), -(SP) ;RESTORE THE STACK
1785 210360 000451 BR 3\$

MAINDEC-11-DBKEB-A
DBKEBA.P11

KE11F (PDP-11 FIS) EXERCISER.
TEST 251

MACY11,620 22-AUG-72 11:40 PAGE 37

1786
1787 010362 012767 000600 170300 2\$: MOV #STK1, SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
1788 010370 026767 170274 170220 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
1789 010376 001401 BEQ ,+4 ;BRANCH IF OK
1790 010400 104000 HLT ;STACK POINTER FOULED UP
1791
1792 010402 022767 010260 170170 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
1793 010410 001401 BEQ ,+4 ;BRANCH IF OK
1794 010412 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1\$
1795
1796 010414 026767 170216 170160 CMP \$ADDER, STK2 ;CHECK THE PSW ON THE STACK
1797 010422 001401 BEQ ,+4 ;BRANCH IF OK
1798 010424 104000 HLT ;RTI PSW NOT EQUAL TO 200
1799
1800 010426 026767 170172 170150 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
1801 010434 001401 BEQ ,+4 ;BRANCH IF OK
1802 010436 104000 HLT ;STK3 NOT EQUAL TO RAND,C
1803
1804 010440 026767 170162 170140 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
1805 010446 001401 BEQ ,+4 ;BRANCH IF OK
1806 010450 104000 HLT ;STK4 NOT EQUAL TO RAND,D
1807
1808 010452 026767 170142 170130 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
1809 010460 001401 BEQ ,+4 ;BRANCH IF OK
1810 010462 104000 HLT ;STK5 NOT EQUAL TO RAND,A
1811
1812 010464 026767 170132 170120 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
1813 010472 001401 BEQ ,+4 ;BRANCH IF OK
1814 010474 104000 HLT ;STK6 NOT EQUAL TO RAND,B
1815
1816 010476 012716 010504 MOV #3\$, (SP) ;RESET THE STACK
1817 010502 000002 RTI ;RESTORE THE STATUS (T-BIT)
1818
1819 010504 104400 3\$: SCOPE
1820
1821
1822
1823 ;***** TEST 26! EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
1824 ; RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
1825 ; STACK POINTER = R0
1826 ;*****
1827
1828 010506 012700 000604 TST261 MOV #STACK0,R0 ;SET UP THE STACK POINTER
1829 010512 004767 003062 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
1830
1831 010516 000240 NOP
1832 010520 075010 FSUB+ RD ;FLOATING SUBTRACT ON THE R0 STACK
1833
1834 010522 013767 177776 170064 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
1835 010530 010067 170062 MOV R0, SSP ;SAVE THE STACK POINTER
1836 010534 026767 170100 170052 CMP SSUBPS, SPSW ;CHECK THE PROCESSOR STATUS
1837 010542 001401 BEQ ,+4 ;BRANCH IF OK
1838 010544 104000 HLT ;PSW NOT EQUAL TO SSUBPS
1839

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11140 PAGE 38
DBKEBA.P\$1 TEST 261 EXERCISE FSUB R0

1840	010546	105767	170042		TSTB	SPSW	JCHECK FOR ERROR
1841	010552	100423			BMI	2\$	JBRANCH IF ERROR
1842							
1843	010554	012767	000610	170106	MOV	#STACK4,SAVSTK	JSAVE PROPER STACK ADDRESS FOR TYPING
1844	010562	026767	170102	170026	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1845	010570	001401			BEQ	,+4	JBRANCH IF OK
1846	010572	104000			HLT		JSTACK POINTER NOT EQUAL TO #STACK4
1847							
1848	010574	026767	170042	170006	CMP	SSUB1, ANS1	JCHECK THE ANSWER
1849	010602	001401			BEQ	,+4	JBRANCH IF OK
1850	010604	104000			HLT		JLEFT HALF OF ANSWER WRONG
1851							
1852	010606	026767	170032	167776	CMP	SSUB2, ANS2	JCHECK THE ANSWER
1853	010614	001401			BEQ	,+4	JBRANCH IF OK
1854	010616	104000			HLT		JRIGHT HALF OF ANSWER WRONG
1855							
1856	010628	000451			BR	3\$	
1857							
1858	010622	012767	000604	170040 2\$:	MOV	#STACK0,SAVSTK	JSAVE STACK ADDRESS FOR TYPING
1859	010630	026767	170034	167760	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1860	010636	001401			BEQ	,+4	JBRANCH IF OK
1861	010640	104000			HLT		JSTACK POINTER FOULED UP
1862							
1863	010642	022767	010522	167730	CMP	#1\$, STK1	JCHECK THE RTI ADDRESS ON THE STACK
1864	010650	001401			BEQ	,+4	JBRANCH IF OK
1865	010652	104000			HLT		JRTI ADDRESS NOT EQUAL TO #1\$
1866							
1867	010654	026767	167766	167720	CMP	SSUBER, STK2	JCHECK THE PSW ON THE STACK
1868	010662	001401			BEQ	,+4	JBRANCH IF OK
1869	010664	104000			HLT		JRTI PSW NOT EQUAL TO 200
1870							
1871	010666	026767	167732	167710	CMP	RAND,C, STK3	JCHECK THE DATA ON THE STACK
1872	010674	001401			BEQ	,+4	JBRANCH IF OK
1873	010676	104000			HLT		JSTK3 NOT EQUAL TO RAND,C
1874							
1875	010700	026767	167722	167700	CMP	RAND,D, STK4	JCHECK THE DATA ON THE STACK
1876	010706	001401			BEQ	,+4	JBRANCH IF OK
1877	010710	104000			HLT		JSTK4 NOT EQUAL TO RAND,D
1878							
1879	010712	026767	167702	167670	CMP	RAND,A, STK5	JCHECK THE DATA ON THE STACK
1880	010720	001401			BEQ	,+4	JBRANCH IF OK
1881	010722	104000			HLT		JSTK5 NOT EQUAL TO RAND,A
1882							
1883	010724	026767	167672	167660	CMP	RAND,B, STK6	JCHECK THE DATA ON THE STACK
1884	010732	001401			BEQ	,+4	JBRANCH IF OK
1885	010734	104000			HLT		JSTK6 NOT EQUAL TO RAND,B
1886							
1887	010736	012716	010744		MOV	#3\$, (SP)	JRESET THE STACK
1888	010742	000002			RTI		JRESTORE THE STATUS (T-BIT)
1889							
1890	010744	104400		3\$: SCOPE			
1891							

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 39
 DBKEBA.P11 TEST 27: EXERCISE FMUL R1

1892							
1893							
1894							
1895							
1896							
1897							
1898							
1899	210746	012701	000604	TST271	MOV #STACK0,R1		ISET UP THE STACK POINTER
1900	210752	004767	002622		JSR PC,	PUSHR	PUT THE DATA ON THE STACK
1901							
1902	210756	000240			NOP		
1903	210760	075021			FMUL+	R1	IFLOATING MULTIPLY ON THE R1 STACK
1904							
1905	210762	013767	177776	167624	1\$: MOV @#PS,	\$PSW	ISAVE PROCESSOR STATUS
1906	210770	010167	167622		MOV R1,	\$SP	ISAVE THE STACK POINTER
1907	210774	026767	167650	167612	CMP SMULPS,	\$PSW	ICHECK THE PROCESSOR STATUS
1908	211002	001401			BEQ ,+4		IBRANCH IF OK
1909	211004	104000			HLT		IPSW NOT EQUAL TO SMULPS
1910							
1911	211006	105767	167602	TSTB	\$PSW		ICHECK FOR ERROR
1912	211012	100423		BM1	2\$		IBRANCH IF ERROR
1913							
1914	211014	012767	000610	167646	MOV #STACK4,SAVSTK		ISAVE PROPER STACK ADDRESS FOR TYPING
1915	211022	026767	167642	167566	CMP SAVSTK,	\$SP	ICHECK THE STACK POINTER
1916	211030	001401			BEQ ,+4		IBRANCH IF OK
1917	211032	104000			HLT		ISTACK POINTER NOT EQUAL TO #STACK4
1918							
1919	211034	026767	167612	167546	CMP SMUL1,	ANS1	ICHECK THE ANSWER
1920	211042	001401			BEQ ,+4		IBRANCH IF OK
1921	211044	104000			HLT		ILEFT HALF OF ANSWER WRONG
1922							
1923	211046	026767	167602	167536	CMP SMUL2,	ANS2	ICHECK THE ANSWER
1924	211054	001401			BEQ ,+4		IBRANCH IF OK
1925	211056	104000			HLT		IRIGHT HALF OF ANSWER WRONG
1926							
1927	211060	000451			BR	3\$	
1928							
1929	211062	012767	000604	167600	2\$: MOV #STACK0,SAVSTK		ISAVE STACK ADDRESS FOR TYPING
1930	211070	026767	167574	167520	CMP SAVSTK,	\$SP	ICHECK THE STACK POINTER
1931	211076	001401			BEQ ,+4		IBRANCH IF OK
1932	211100	104000			HLT		ISTACK POINTER FOULED UP
1933							
1934	211102	022767	010762	167470	CMP #1\$,	STK1	ICHECK THE RTI ADDRESS ON THE STACK
1935	211110	001401			BEQ ,+4		IBRANCH IF OK
1936	211112	104000			HLT		IRTI ADDRESS NOT EQUAL TO #1\$
1937							
1938	211114	026767	167536	167460	CMP SMULER,	STK2	ICHECK THE PSW ON THE STACK
1939	211122	001401			BEQ ,+4		IBRANCH IF OK
1940	211124	104000			HLT		IRTI PSW NOT EQUAL TO 200
1941							
1942	211126	026767	167472	167450	CMP RAND,C,	STK3	ICHECK THE DATA ON THE STACK
1943	211134	001401			BEQ ,+4		IBRANCH IF OK
1944	211136	104000			HLT		ISTK3 NOT EQUAL TO RAND,C
1945							

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
 DBKEBA.P11 TEST 27: EXERCISE FMUL R1 MACY11,620 22-AUG-72 11:40 PAGE 40

1946	211140	026767	167462	167440	CMP	RAND,D, STK4	JCHECK THE DATA ON THE STACK
1947	211146	001401			BEQ	,+4	JBRANCH IF OK
1948	211150	104000			HLT		JSTK4 NOT EQUAL TO RAND,D
1949							
1950	211152	026767	167442	167430	CMP	RAND,A, STK5	JCHECK THE DATA ON THE STACK
1951	211160	001401			BEQ	,+4	JBRANCH IF OK
1952	211162	104000			HLT		JSTK5 NOT EQUAL TO RAND,A
1953							
1954	211164	026767	167432	167420	CMP	RAND,B, STK6	JCHECK THE DATA ON THE STACK
1955	211172	001401			BEQ	,+4	JBRANCH IF OK
1956	211174	104000			HLT		JSTK6 NOT EQUAL TO RAND,B
1957							
1958	211176	012716	011204		MOV	#3\$, (SP)	JRESET THE STACK
1959	211202	000002			RTI		JRESTORE THE STATUS (T-BIT)
1960							
1961	211204	104400			3\$: SCOPE		
1962							
1963							
1964							*****
1965							JTEST 301 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
1966							J RAND,A,RAND,B / RAND,C,RAND,D = ANS1,ANS2
1967							J STACK POINTER \$ R2
1968							*****
1969							
1970	211206	012702	000604		TST301	MOV #STACK0,R2	JSET UP THE STACK POINTER
1971	211212	004767	002362		JSR PC,	PUSHR	JPUT THE DATA ON THE STACK
1972							
1973	211216	000240			NOP		
1974	211220	075032			FDIV+	R2	JFLOATING DIVIDE ON THE R2 STACK
1975							
1976	211222	013767	177776	167364	1\$: MOV	#PS, SPSW	JSAVE PROCESSOR STATUS
1977	211230	010267	167362		MOV	R2, SSP	JSAVE THE STACK POINTER
1978	211234	026767	167420	167352	CMP	SDIVPS, SPSW	JCHECK THE PROCESSOR STATUS
1979	211242	001401			BEQ	,+4	JBRANCH IF OK
1980	211244	104000			HLT		JPSW NOT EQUAL TO SDIVPS
1981							
1982	211246	105767	167342		TSTB	SPSW	JCHECK FOR ERROR
1983	211252	100423			BMI	2\$	JBRANCH IF ERROR
1984							
1985	211254	012767	000610	167406	MOV	#STACK4,SAVSTK	JSAVE PROPER STACK ADDRESS FOR TYPING
1986	211262	026767	167402	167326	CMP	SAVSTK, SSP	JCHECK THE STACK POINTER
1987	211270	001401			BEQ	,+4	JBRANCH IF OK
1988	211272	104000			HLT		JSTACK POINTER NOT EQUAL TO #STACK4
1989							
1990	211274	026767	167362	167306	CMP	\$DIV1, ANS1	JCHECK THE ANSWER
1991	211302	001401			BEQ	,+4	JBRANCH IF OK
1992	211304	104000			HLT		JLEFT HALF OF ANSWER WRONG
1993							
1994	211306	026767	167352	167276	CMP	\$DIV2, ANS2	JCHECK THE ANSWER
1995	211314	001401			BEQ	,+4	JBRANCH IF OK
1996	211316	104000			HLT		JRIGHT HALF OF ANSWER WRONG
1997							
1998	211320	000451			BR	3\$	
1999							

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
 DBKEBA.P11 TEST 301 EXERCISE FDIV R2 MACY11,620 22-AUG-72 11:40 PAGE 41

2000	011322	012767	000604	167340	2\$: MOV #STACK0,SAVSTK	;SAVE STACK ADDRESS FOR TYPING
2001	011330	026767	167334	167260	CMP SAVSTK, SSP	;CHECK THE STACK POINTER
2002	011336	001401			BEQ ,+4	;BRANCH IF OK
2003	011340	104000			HLT	;STACK POINTER FOULED UP
2004						
2005	011342	022767	011222	167230	CMP #1\$, STK1	;CHECK THE RTI ADDRESS ON THE STACK
2006	011350	001401			BEQ ,+4	;BRANCH IF OK
2007	011352	104000			HLT	;RTI ADDRESS NOT EQUAL TO #1\$
2008						
2009	011354	026767	167306	167220	CMP \$DIVER, STK2	;CHECK THE PSW ON THE STACK
2010	011362	001401			BEQ ,+4	;BRANCH IF OK
2011	011364	104000			HLT	;RTI PSW NOT EQUAL TO 200
2012						
2013	011366	026767	167232	167210	CMP RAND,C, STK3	;CHECK THE DATA ON THE STACK
2014	011374	001401			BEQ ,+4	;BRANCH IF OK
2015	011376	104000			HLT	;STK3 NOT EQUAL TO RAND,C
2016						
2017	011400	026767	167222	167200	CMP RAND,D, STK4	;CHECK THE DATA ON THE STACK
2018	011406	001401			BEQ ,+4	;BRANCH IF OK
2019	011410	104000			HLT	;STK4 NOT EQUAL TO RAND,D
2020						
2021	011412	026767	167202	167170	CMP RAND,A, STK5	;CHECK THE DATA ON THE STACK
2022	011420	001401			BEQ ,+4	;BRANCH IF OK
2023	011422	104000			HLT	;STK5 NOT EQUAL TO RAND,A
2024						
2025	011424	026767	167172	167160	CMP RAND,B, STK6	;CHECK THE DATA ON THE STACK
2026	011432	001401			BEQ ,+4	;BRANCH IF OK
2027	011434	104000			HLT	;STK6 NOT EQUAL TO RAND,B
2028						
2029	011436	012716	011444		MOV #3\$, (SP)	;RESET THE STACK
2030	011442	000002			RTI	;RESTORE THE STATUS (T-BIT)
2031						
2032	011444	104400		3\$: SCOPE		
2033						
2034						
2035					*****	
2036					;TEST 31: EXERCISE FADD (PDP-11 FLOATING ADD INSTRUCTION)	
2037					; RAND,A,RAND,B + RAND,C,RAND,D = ANS1,ANS2	
2038					; STACK POINTER = R3	
2039					*****	
2040						
2041	011446	012703	000604	TST31: JSR PC,	MOV #STACK0,R3 PUSHR	;SET UP THE STACK POINTER ;PUT THE DATA ON THE STACK
2042	011452	004767	002122			
2043						
2044	011456	000240			NOP	
2045	011460	075003			FADD+ R3	;FLOATING ADD ON THE R3 STACK
2046						
2047	011462	013767	177776	167124 1\$: MOV @#PS, SPSW		;SAVE PROCESSOR STATUS
2048	011470	010367	167122		MOV R3, SSP	;SAVE THE STACK POINTER
2049	011474	026767	167130	167112	CMP \$ADDPS, SPSW	;CHECK THE PROCESSOR STATUS
2050	011502	001401			BEQ ,+4	;BRANCH IF OK
2051	011504	104000			HLT	;PSW NOT EQUAL TO \$ADDPS
2052						
2053	011506	105767	167102	TSTB	\$SPSW	;CHECK FOR ERROR

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCiser,
 DBKEBA.P11 TEST 31: EXERCISE FADD R3 MACY11,620 22-AUG-72 11:40 PAGE 42

Line	OpCode	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op465	Op466	Op467	Op468	Op469	Op470	Op471	Op472	Op473	Op474	Op475	Op476	Op477	Op478	Op479	Op480	Op481	Op482	Op483	Op484	Op485	Op486	Op487	Op488	Op489	Op490	Op491	Op492	Op493	Op494	Op495	Op496	Op497	Op498	Op499	Op500	Op501	Op502	Op503	Op504	Op505	Op506	Op507	Op508	Op509	Op510	Op511	Op512	Op513	Op514	Op515	Op516	Op517	Op518	Op519	Op520	Op521	Op522	Op523	Op524	Op525	Op526	Op527	Op528	Op529	Op530	Op531	Op532	Op533	Op534	Op535	Op536	Op537	Op538	Op539	Op540	Op541	Op542	Op543	Op544	Op545	Op546	Op547	Op548	Op549	Op550	Op551	Op552	Op553	Op554	Op555	Op556	Op557	Op558	Op559	Op560	Op561	Op562	Op563	Op564	Op565	Op566	Op567	Op568	Op569	Op570	Op571	Op572	Op573	Op574	Op575	Op576	Op577	Op578	Op579	Op580	Op581	Op582	Op583	Op584	Op585	Op586	Op587	Op588	Op589	Op590	Op591	Op592	Op593	Op594	Op595	Op596	Op597	Op598	Op599	Op600	Op601	Op602	Op603	Op604	Op605	Op606	Op607	Op608	Op609	Op610	Op611	Op612	Op613	Op614	Op615	Op616	Op617	Op618	Op619	Op620	Op621	Op622	Op623	Op624	Op625	Op626	Op627	Op628	Op629	Op630	Op631	Op632	Op633	Op634	Op635	Op636	Op637	Op638	Op639	Op640	Op641	Op642	Op643	Op644	Op645	Op646	Op647	Op648	Op649	Op650	Op651	Op652	Op653	Op654	Op655	Op656	Op657	Op658	Op659	Op660	Op661	Op662	Op663	Op664	Op665	Op666	Op667	Op668	Op669	Op670	Op671	Op672	Op673	Op674	Op675	Op676	Op677	Op678	Op679	Op680	Op681	Op682	Op683	Op684	Op685	Op686	Op687	Op688	Op689	Op690	Op691	Op692	Op693	Op694	Op695	Op696	Op697	Op698	Op699	Op700	Op701	Op702	Op703	Op704	Op705	Op706	Op707	Op708	Op709	Op710	Op711	Op712	Op713	Op714	Op715	Op716	Op717	Op718	Op719	Op720	Op721	Op722	Op723	Op724	Op725	Op726	Op727	Op728	Op729	Op730	Op731	Op732	Op733	Op734	Op735	Op736	Op737	Op738	Op739	Op740	Op741	Op742	Op743	Op744	Op745	Op746	Op747	Op748	Op749	Op750	Op751	Op752	Op753	Op754	Op755	Op756	Op757	Op758	Op759	Op760	Op761	Op762	Op763	Op764	Op765	Op766	Op767	Op768	Op769	Op770	Op771	Op772	Op773	Op774	Op775	Op776	Op777	Op778	Op779	Op780	Op781	Op782	Op783	Op784	Op785	Op786	Op787	Op788	Op789	Op790	Op791	Op792	Op793	Op794	Op795	Op796	Op797	Op798	Op799	Op800	Op801	Op802	Op803	Op804	Op805	Op806	Op807	Op808	Op809	Op810	Op811	Op812	Op813	Op814	Op815	Op816	Op817	Op818	Op819	Op820	Op821	Op822	Op823	Op824	Op825	Op826	Op827	Op828	Op829	Op830	Op831	Op832	Op833	Op834	Op835	Op836	Op837	Op838	Op839	Op840	Op841	Op842	Op843	Op844	Op845	Op846	Op847	Op848	Op849	Op850	Op851	Op852	Op853	Op854	Op855	Op856	Op857	Op858	Op859	Op860	Op861	Op862	Op863	Op864	Op865	Op866	Op867	Op868	Op869	Op870	Op871	Op872	Op873	Op874	Op875	Op876	Op877	Op878	Op879	Op880	Op881	Op882	Op883	Op884	Op885	Op886	Op887	Op888	Op889	Op890	Op891	Op892	Op893	Op894	Op895	Op896	Op897	Op898	Op899	Op900	Op901	Op902	Op903	Op904	Op905	Op906	Op907	Op908	Op909	Op910	Op911	Op912	Op913	Op914	Op915	Op916	Op917	Op918	Op919	Op920	Op921	Op922	Op923	Op924	Op925	Op926	Op927	Op928	Op929	Op930	Op931	Op932	Op933	Op934	Op935	Op936	Op937	Op938	Op939	Op940	Op941	Op942	Op943	Op944	Op945	Op946	Op947	Op948	Op949	Op950	Op951	Op952	Op953	Op954	Op955	Op956	Op957	Op958	Op959	Op960	Op961	Op962	Op963	Op964	Op965	Op966	Op967	Op968	Op969	Op970	Op971	Op972	Op973	Op974	Op975	Op976	Op977	Op978	Op979	Op980	Op981	Op982	Op983	Op984	Op985	Op986	Op987	Op988	Op989	Op990	Op991	Op992	Op993	Op994	Op995	Op996	Op997	Op998	Op999	Op1000	Op1001	Op1002	Op1003	Op1004	Op1005	Op1006	Op1007	Op1008	Op1009	Op1010	Op1011	Op1012	Op1013	Op1014	Op1015	Op1016	Op1017	Op1018	Op1019	Op1020	Op1021	Op1022	Op1023	Op1024	Op1025	Op1026	Op1027	Op1028	Op1029	Op1030	Op1031	Op1032	Op1033	Op1034	Op1035	Op1036	Op1037	Op1038	Op1039	Op1040	Op1041	Op1042	Op1043	Op1044	Op1045	Op1046	Op1047	Op1048	Op1049	Op1050	Op1051	Op1052	Op1053	Op1054	Op1055	Op1056	Op1057	Op1058	Op1059	Op1060	Op1061	Op1062	Op1063	Op1064	Op1065	Op1066	Op1067	Op1068	Op1069	Op1070	Op1071	Op1072	Op1073	Op1074	Op1075	Op1076	Op1077	Op1078	Op1079	Op1080	Op1081	Op1082	Op1083	Op1084	Op1085	Op1086	Op1087	Op1088	Op1089	Op1090	Op1091	Op1092	Op1093	Op1094	Op1095	Op1096	Op1097	Op1098	Op1099	Op1100	Op1101	Op1102	Op1103	Op1104	Op1105	Op1106	Op1107	Op1108	Op1109	Op1110	Op1111	Op1112	Op1113	Op1114	Op1115	Op1116	Op1117	Op1118	Op1119	Op1120	Op1121	Op1122	Op1123	Op1124	Op1125	Op1126	Op1127	Op1128	Op1129	Op1130	Op1131	Op1132	Op1133	Op1134	Op1135	Op1136	Op1137	Op1138	Op1139	Op1140	Op1141	Op1142	Op1143	Op1144	Op1145	Op1146	Op1147	Op1148	Op1149	Op1150	Op1151	Op1152	Op1153	Op1154	Op1155	Op1156	Op1157	Op1158	Op1159	Op1160	Op1161	Op1162	Op1163	Op1164	Op1165	Op1166	Op1167	Op1168	Op1169	Op1170	Op1171	Op1172	Op1173	Op1174	Op1175	Op1176	Op1177	Op1178	Op1179	Op1180	Op1181	Op1182	Op1183	Op1184	Op1185	Op1186	Op1187	Op1188	Op1189	Op1190	Op1191	Op1192	Op1193	Op1194	Op1195	Op1196	Op1197	Op1198	Op1199	Op1200	Op1201	Op1202	Op1203	Op1204	Op1205	Op1206	Op1207	Op1208	Op1209	Op1210	Op1211	Op1212	Op1213	Op1214	Op1215	Op1216	Op1217	Op1218	Op1219	Op1220	Op1221	Op1222	Op1223	Op1224	Op1225	Op1226	Op1227	Op1228	Op1229	Op1230	Op1231	Op1232	Op1233	Op1234	Op1235	Op1236	Op1237
------	--------	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 43
DBKEBA.P11 TEST 321 EXERCISE FSUB R4

2105
2106
2107 ;TEST 321 EXERCISE FSUB (PDP-11 FLOATING SUBTRACT INSTRUCTION)
2108 ; RAND,A,RAND,B = RAND,C,RAND,D = ANS1,ANS2
2109 ; STACK POINTER = R4
2110 ;
2111
2112 011706 012704 000604 TST321 MOV #STACK0,R4 ;SET UP THE STACK POINTER
2113 011712 004767 001662 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
2114
2115 011716 000240 NOP
2116 011720 075014 FSUB+ R4 ;FLOATING SUBTRACT ON THE R4 STACK
2117
2118 011722 013767 177776 166664 1\$: MOV #PS, SPSW ;SAVE PROCESSOR STATUS
2119 011730 010467 166662 166662 MOV R4, SSP ;SAVE THE STACK POINTER
2120 011734 026767 166700 166652 CMP \$SUBPS, SPSW ;CHECK THE PROCESSOR STATUS
2121 011742 001401 BEQ ,+4 ;BRANCH IF OK
2122 011744 104000 HLT ;PSW NOT EQUAL TO \$SUBPS
2123
2124 011746 105767 166642 TSTB SPSW ;CHECK FOR ERROR
2125 011752 100423 BMI 2\$;BRANCH IF ERROR
2126
2127 011754 012767 000610 166706 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
2128 011762 026767 166702 166626 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
2129 011770 001401 BEQ ,+4 ;BRANCH IF OK
2130 011772 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
2131
2132 011774 026767 166642 166606 CMP SSUB1, ANS1 ;CHECK THE ANSWER
2133 012002 001401 BEQ ,+4 ;BRANCH IF OK
2134 012004 104000 HLT ;LEFT HALF OF ANSWER WRONG
2135
2136 012006 026767 166632 166576 CMP SSUB2, ANS2 ;CHECK THE ANSWER
2137 012014 001401 BEQ ,+4 ;BRANCH IF OK
2138 012016 104000 HLT ;RIGHT HALF OF ANSWER WRONG
2139
2140 012020 000451 BR 3\$
2141
2142 012022 012767 000604 166640 2\$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
2143 012030 026767 166634 166560 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
2144 012036 001401 BEQ ,+4 ;BRANCH IF OK
2145 012040 104000 HLT ;STACK POINTER FOULED UP
2146
2147 012042 022767 011722 166530 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
2148 012050 001401 BEQ ,+4 ;BRANCH IF OK
2149 012052 104000 HLT ;RTI PSW NOT EQUAL TO #1\$
2150
2151 012054 026767 166566 166520 CMP SSUBER, STK2 ;CHECK THE PSW ON THE STACK
2152 012062 001401 BEQ ,+4 ;BRANCH IF OK
2153 012064 104000 HLT ;RTI PSW NOT EQUAL TO 200
2154
2155 012066 026767 166532 166510 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
2156 012074 001401 BEQ ,+4 ;BRANCH IF OK
2157 012076 104000 HLT ;STK3 NOT EQUAL TO RAND,C
2158

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 TEST 32: EXERCISE FSUB R4 MACY11,620 22-AUG-72 11:40 PAGE 44

2159 012100 026767 166522 166500 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
2160 012106 001401 BEQ ,+4 ;BRANCH IF OK
2161 012110 104000 HLT ;STK4 NOT EQUAL TO RAND,D
2162
2163 012112 026767 166502 166470 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
2164 012120 001401 BEQ ,+4 ;BRANCH IF OK
2165 012122 104000 HLT ;STK5 NOT EQUAL TO RAND,A
2166
2167 012124 026767 166472 166460 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
2168 012132 001401 BEQ ,+4 ;BRANCH IF OK
2169 012134 104000 HLT ;STK6 NOT EQUAL TO RAND,B
2170
2171 012136 012716 012144 MOV #3\$, (SP) ;RESET THE STACK
2172 000002 RTI ;RESTORE THE STATUS (T-BIT)
2173
2174 012144 104400 3\$: SCOPE
2175
2176
2177 *****
2178 ;TEST 33: EXERCISE FMUL (PDP-11 FLOATING MULTIPLY INSTRUCTION)
2179 ; RAND,A,RAND,B * RAND,C,RAND,D = ANS1,ANS2
2180 ; STACK POINTER = R5
2181 *****
2182
2183 012146 012705 000604 TST331 MOV #STACK0,R5 ;SET UP THE STACK POINTER
2184 004767 001422 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
2185
2186 012156 000240 NOP
2187 012160 075025 FMUL+, R5 ;FLOATING MULTIPLY ON THE R5 STACK
2188
2189 012162 013767 177776 166424 1\$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
2190 012170 010567 166422 MOV R5, SSP ;SAVE THE STACK POINTER
2191 012174 026767 166450 166412 CMP SMULPS, SPSW ;CHECK THE PROCESSOR STATUS
2192 012202 001401 BEQ ,+4 ;BRANCH IF OK
2193 012204 104000 HLT ;IPSW NOT EQUAL TO SMULPS
2194
2195 012206 105767 166402 TSTB SPSW ;CHECK FOR ERROR
2196 012212 100423 BMI 2\$;BRANCH IF ERROR
2197
2198 012214 012767 000610 166446 MOV #STACK4,SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
2199 026767 166442 166366 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
2200 012230 001401 BEQ ,+4 ;BRANCH IF OK
2201 012232 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
2202
2203 012234 026767 166412 166346 CMP SMUL1, ANS1 ;CHECK THE ANSWER
2204 001401 BEQ ,+4 ;BRANCH IF OK
2205 012244 104000 HLT ;LEFT HALF OF ANSWER WRONG
2206
2207 012246 026767 166402 166336 CMP SMUL2, ANS2 ;CHECK THE ANSWER
2208 001401 BEQ ,+4 ;BRANCH IF OK
2209 012256 104000 HLT ;RIGHT HALF OF ANSWER WRONG
2210
2211 012260 000451 BR 3\$
2212

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 45
 DBKEBA.P11 TEST 33: EXERCISE FMUL R5

```

    2213 012262 012767 000604 166400 2$: MOV #STACK0,SAVSTK ;SAVE STACK ADDRESS FOR TYPING
    2214 012270 026767 166374 166320 CMP SAVSTK, $SP ;CHECK THE STACK POINTER
    2215 012276 001401 BEQ ,+4 ;BRANCH IF OK
    2216 012300 104000 HLT ;STACK POINTER FOULED UP
    2217
    2218 012302 022767 012162 166270 CMP #1$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
    2219 012310 001401 BEQ ,+4 ;BRANCH IF OK
    2220 012312 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1$
    2221
    2222 012314 026767 166336 166260 CMP $MULER, STK2 ;CHECK THE PSW ON THE STACK
    2223 012322 001401 BEQ ,+4 ;BRANCH IF OK
    2224 012324 104000 HLT ;RTI PSW NOT EQUAL TO 200
    2225
    2226 012326 026767 166272 166250 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
    2227 012334 001401 BEQ ,+4 ;BRANCH IF OK
    2228 012336 104000 HLT ;STK3 NOT EQUAL TO RAND,C
    2229
    2230 012340 026767 166262 166240 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
    2231 012346 001401 BEQ ,+4 ;BRANCH IF OK
    2232 012350 104000 HLT ;STK4 NOT EQUAL TO RAND,D
    2233
    2234 012352 026767 166242 166230 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
    2235 012360 001401 BEQ ,+4 ;BRANCH IF OK
    2236 012362 104000 HLT ;STK5 NOT EQUAL TO RAND,A
    2237
    2238 012364 026767 166232 166220 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
    2239 012372 001401 BEQ ,+4 ;BRANCH IF OK
    2240 012374 104000 HLT ;STK6 NOT EQUAL TO RAND,B
    2241
    2242 012376 012716 012404 MOV #3$, (SP) ;RESET THE STACK
    2243 012402 000002 RTI ;RESTORE THE STATUS (T-BIT)
    2244
    2245 012404 104400 3$: SCOPE
    2246
    2247
    2248 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2249 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2250 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2251 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2252 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2253 **** TEST 341 EXERCISE FDIV (PDP-11 FLOATING DIVIDE INSTRUCTION)
    2254 012406 012706 000604 TST341 MOV #STACK0,SP ;SET UP THE STACK POINTER
    2255 012412 004767 001162 JSR PC, PUSHR ;PUT THE DATA ON THE STACK
    2256
    2257 012416 000240 NOP
    2258 012420 075036 FDIV+ SP ;FLOATING DIVIDE ON THE SP STACK
    2259
    2260 012422 013767 177776 166164 1$: MOV @#PS, SPSW ;SAVE PROCESSOR STATUS
    2261 012430 010667 166162 MOV SP, SSP ;SAVE THE STACK POINTER
    2262 012434 026767 166220 166152 CMP SDIVPS, SPSW ;CHECK THE PROCESSOR STATUS
    2263 012442 001401 BEQ ,+4 ;BRANCH IF OK
    2264 012444 104000 HLT ;PSW NOT EQUAL TO SDIVPS
    2265
    2266 012446 105767 166142 TSTB SPSW ;CHECK FOR ERROR
  
```

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 46
DBKEBA.P11 TEST 348 EXERCISE FDIV SP

2267 012452 100424 BMI 2\$;BRANCH IF ERROR
2268
2269 012454 012767 000610 166226 MOV #STACK4, SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
2270 012462 026767 166202 166126 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
2271 001401 BEQ ,+4 ;BRANCH IF OK
2272 104000 HLT ;STACK POINTER NOT EQUAL TO #STACK4
2273
2274 012474 026767 166162 166106 CMP SDIV1, ANS1 ;CHECK THE ANSWER
2275 001401 BEQ ,+4 ;BRANCH IF OK
2276 104000 HLT ;LEFT HALF OF ANSWER WRONG
2277
2278 012506 026767 166152 166076 CMP SDIV2, ANS2 ;CHECK THE ANSWER
2279 001401 BEQ ,+4 ;BRANCH IF OK
2280 104000 HLT ;RIGHT HALF OF ANSWER WRONG
2281
2282 012520 024646 CMP -(SP), -(SP) ;RESTORE THE STACK
2283 000451 BR 35
2284
2285 012524 012767 000600 166136 2\$: MOV #STK1, SAVSTK ;SAVE PROPER STACK ADDRESS FOR TYPING
2286 026767 166132 166056 CMP SAVSTK, SSP ;CHECK THE STACK POINTER
2287 001401 BEQ ,+4 ;BRANCH IF OK
2288 104000 HLT ;STACK POINTER FOULED UP
2289
2290 012544 022767 012422 166026 CMP #1\$, STK1 ;CHECK THE RTI ADDRESS ON THE STACK
2291 001401 BEQ ,+4 ;BRANCH IF OK
2292 104000 HLT ;RTI ADDRESS NOT EQUAL TO #1\$
2293
2294 012556 026767 166104 166016 CMP SDIVER, STK2 ;CHECK THE PSW ON THE STACK
2295 001401 BEQ ,+4 ;BRANCH IF OK
2296 104000 HLT ;RTI PSW NOT EQUAL TO 200
2297
2298 012570 026767 166030 166006 CMP RAND,C, STK3 ;CHECK THE DATA ON THE STACK
2299 001401 BEQ ,+4 ;BRANCH IF OK
2300 104000 HLT ;STK3 NOT EQUAL TO RAND,C
2301
2302 012602 026767 166020 165776 CMP RAND,D, STK4 ;CHECK THE DATA ON THE STACK
2303 001401 BEQ ,+4 ;BRANCH IF OK
2304 104000 HLT ;STK4 NOT EQUAL TO RAND,D
2305
2306 012614 026767 166000 165766 CMP RAND,A, STK5 ;CHECK THE DATA ON THE STACK
2307 001401 BEQ ,+4 ;BRANCH IF OK
2308 104000 HLT ;STK5 NOT EQUAL TO RAND,A
2309
2310 012626 026767 165770 165756 CMP RAND,B, STK6 ;CHECK THE DATA ON THE STACK
2311 001401 BEQ ,+4 ;BRANCH IF OK
2312 104000 HLT ;STK6 NOT EQUAL TO RAND,B
2313
2314 012640 012716 012646 MOV #3\$, (SP) ;RESET THE STACK
2315 000002 RTI ;RESTORE THE STATUS (T-BIT)
2316
2317 012646 104400 3\$: SCOPE
2318

MAINDEC-11-DBKEBA-A
DBKEBA.P11KE11F (PDP-11 FIS) EXERCISER,
END ROUTINE

MACY11,620 22-AUG-72 11:40 PAGE 47

2319
2320 212650 062767 000001 166130 ADD #1, PCNT+2 JCOUNT PASSES
2321 212656 005567 166122 ADC PCNT
2322
2323 212662 001 DONE:
2324 212662 032737 002000 177570 BIT #SW10,@#SWR JRING THE BELL?
2325 212670 001002 BNE 1\$ JNO:
2326 212672 000004 000007 TYPE ,BELL
2327 212676 005046 1\$: CLR -(6)
2328 212700 032737 010000 177570 BIT #SW12,@#SWR JCLEAR TRACE TRAP
2329 212706 001010 BNE 2\$ JRUN WITH TRT?
2330 212710 005167 000044 COM ,TBIT
2331 212714 100005 BPL 2\$
2332 212716 052716 000020 BIS #20,(6) JSET TRACE TRAP
2333 212722 012746 012754 MOV #3\$,-(6) JJUMP TO START OF TEST
2334 212726 000002 RTI
2335 212730 012746 012736 2\$: MOV #4\$,-(6) JJUMP TO START OF TEST
2336 212734 000002 RTI JRETURN
2337 212736 013700 000042 4\$: MOV @#42,R0 JGET MONITOR ADDRESS
2338 212742 001404 BEQ 3\$ JIF NONE
2339 212744 004710 JSR 7,(0) JGO TO MONITOR
2340 212746 000240 NOP
2341 212750 000240 NOP
2342 212752 000240 NOP
2343 212754 000137 001146 3\$: JMP @#START JRETURN
2344
2345 212760 000000 ,TBITI 0
2346
2347
2348 ISUBROUTINE TO READ TTY INPUT AND SAVE OCTAL NUMBER
2349
2350 212762 004767 002124 READIN: JSR PC,READS
2351 212766 012702 015212 MOV #INPUT,R2
2352 212772 012501 MOV (R5)+,R1
2353 212774 005011 CLR (R1)
2354 212776 112203 1\$: MOVB (R2)+,R3 JSTORE DATA
2355 213000 001420 BEQ 4\$ JBRANCH IF DONE
2356 213002 162703 000060 SUB #60,R3
2357 213006 000241 CLC
2358 213010 032703 177770 BIT #177770,R3
2359 213014 001010 BNE 2\$
2360 213016 006311 ASL (R1)
2361 213020 103407 BCS 3\$
2362 213022 006311 ASL (R1)
2363 213024 103405 BCS 3\$
2364 213026 006311 ASL (R1)
2365 213030 103403 BCS 3\$
2366 213032 050311 BIS R3,(R1)
2367 213034 000760 BR 1\$
2368 213036 000261 2\$: SEC JSET C-BIT IF NOT
2369 213040 000244 3\$: CLZ
2370 213042 000205 4\$: RTS R5

2371							
2372	013044	016746	165552	\$PUSHI	MOV	RAND,B,-(SP)	
2373	013050	016746	165544		MOV	RAND,A,-(SP)	
2374	013054	016746	165546		MOV	RAND,D,-(SP)	
2375	013060	016746	165540		MOV	RAND,C,-(SP)	
2376	013064	000134		\$POLSH:	JMP	@(R4)+	
2377							
2378	013066	005767	165536	\$POPAD:	TST	\$ADDPS	ICHECK FOR ERROR
2379	013072	001145			BNE	\$SKIP	IBRANCH IF PS SET
2380	013074	032716	077600		BIT	#77600, (SP)	ICHECK FOR ZERO
2381	013100	001010			BNE	1\$	IBRANCH IF NOT
2382	013102	013767	177776 165520		MOV	@#PS, \$ADDPS	I2-BIT IN PSW
2383	013110	005067	165516		CLR	SADD1	IZERO ANSWER
2384	013114	005067	165514		CLR	SADD2	
2385	013120	000532			BR	SSKIP	
2386							
2387	013122	005716		1\$:	TST	(SP)	IGET N-BIT, CLEAR C-BIT, V-BIT
2388	013124	013767	177776 165476		MOV	@#PS, \$ADDPS	ISET THE PSW SAVE
2389	013132	012667	165474		MOV	(SP)+, SADD1	
2390	013136	012667	165472		MOV	(SP)+, SADD2	
2391	013142	000134			JMP	@(R4)+	
2392							
2393	013144	005767	165470	\$POPSB:	TST	\$SUBPS	ICHECK FOR ERROR
2394	013150	001116			BNE	\$SKIP	IBRANCH IF PS SET
2395	013152	032716	077600		BIT	#77600, (SP)	ICHECK FOR ZERO
2396	013156	001010			BNE	1\$	IBRANCH IF NOT
2397	013160	013767	177776 165452		MOV	@#PS, \$SUBPS	I2-BIT IN PSW
2398	013166	005067	165450		CLR	SSUB1	IZERO ANSWER
2399	013172	005067	165446		CLR	SSUB2	
2400	013176	000503			BR	SSKIP	
2401							
2402	013200	005716		1\$:	TST	(SP)	IGET N-BIT, CLEAR C-BIT, V-BIT
2403	013202	013767	177776 165430		MOV	@#PS, \$SUBPS	ISET THE PSW SAVE
2404	013210	012667	165426		MOV	(SP)+, SSUB1	
2405	013214	012667	165424		MOV	(SP)+, SSUB2	
2406	013220	000134			JMP	@(R4)+	
2407							
2408	013222	005767	165422	\$POPMI:	TST	\$MULPS	ICHECK FOR ERROR
2409	013226	001067			BNE	\$SKIP	IBRANCH IF PS SET
2410	013230	032716	077600		BIT	#77600, (SP)	ICHECK FOR ZERO
2411	013234	001010			BNE	1\$	IBRANCH IF NOT
2412	013236	013767	177776 165404		MOV	@#PS, \$MULPS	I2-BIT IN PSW
2413	013244	005067	165402		CLR	SMUL1	IZERO ANSWER
2414	013250	005067	165400		CLR	SMUL2	
2415	013254	000454			BR	SSKIP	
2416							
2417	013256	005716		1\$:	TST	(SP)	IGET N-BIT, CLEAR C-BIT, V-BIT
2418	013260	013767	177776 165362		MOV	@#PS, \$MULPS	ISET THE PSW SAVE
2419	013266	012667	165360		MOV	(SP)+, SMUL1	
2420	013272	012667	165356		MOV	(SP)+, SMUL2	
2421	013276	000134			JMP	@(R4)+	

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 49
 DBKEBA.P11 POLISH MODE ROUTINES TO ACCESS FORTRAN ROUTINES

2422								
2423	013300	032767	077600	165316	\$POPODV: BIT	#77600,RAND,C	;CHECK FOR DIVIDED BY ZERO	
2424	013306	001010			RNE	1S		
2425	013310	000277			SCC		;SET ALL CONDITION CODES	
2426	013312	000244			CLZ		;CLEAR THE Z-BIT	
2427	013314	013767	177776	165344	MOV	@#PS, \$DIVVER	;SET UP DIVIDE BY ZERO CC'S	
2428	013322	012767	000340	165330	MOV	#340, \$DIVPS	;SET UP PSW	
2429	013330	005767	165324		1\$: TST	\$DIVPS	;CHECK FOR ERROR	
2430	013334	001024			BNE	\$SKIP	;BRANCH IF PS SET	
2431	013336	032716	077600		BIT	#77600, (SP)	;CHECK FOR ZERO	
2432	013342	001010			BNE	2S	;BRANCH IF NOT	
2433	013344	013767	177776	165306	MOV	@#PS, \$DIVPS	;Z-BIT IN PSW	
2434	013352	005067	165304		CLR	\$DIV1	;ZERO ANSWER	
2435	013356	005067	165302		CLR	\$DIV2		
2436	013362	000411			BR	\$SKIP		
2437								
2438	013364	005716			2\$: TST	(SP)		
2439	013366	013767	177776	165264	MOV	@#PS, \$DIVPS	;GET N-BIT, CLEAR C-BIT, V-BIT	
2440	013374	012667	165262		MOV	(SP)+, \$DIV1	;SET THE PSW SAVE	
2441	013400	012667	165260		MOV	(SP)+, \$DIV2		
2442	013404	000134			JMP	@(R4)+		
2443								
2444	013406	022626			\$SKIP1	CMP	(SP)+, (SP)+	;POP GARBAGE OFF THE STACK
2445	013410	000134			JMP	@(R4)+		
2446								
2447	013412	000204			SEXITI	RTS	R4	;EXIT POLISH MODE
2448								
2449	013414	016500	000002		SERR:	MOV	2(5), R0	;PUT CODE INTO R0
2450	013420	022700	004003		SERRAI	CMP	#4003, R0	;CHECK FOR DIVIDE BY ZERO
2451	013424	001464			BEQ	8\$;SKIP OUT
2452								
2453	013426	122700	000003		CMPB	#3,	R0	;CHECK FOR OVERFLOW
2454	013432	001006			BNE	2S		;BRANCH IF NOT
2455	013434	000257			CCC			;CLEAR ALL CONDITION CODES
2456	013436	000262			SEV			;SET THE V-BIT
2457	013440	013767	177776	165146	MOV	@#PS, SPSW		;SET UP PSW FOR OVERFLOW
2458	013446	000405			BR	3\$		
2459								
2460	013450	000257			2\$: CCC			;CLEAR ALL CONDITION CODES
2461	013452	000272			SNV			;SET N-BIT AND V-BIT
2462	013454	013767	177776	165132	MOV	@#PS, SPSW		;SET UP PSW FOR UNDERFLOW
2463	013462	105000			3\$: CLR B	R0		;CLEAR LOW BYTE
2464	013464	000300			SWAB	R0		;HIGH BYTE INTO LOW
2465	013466	162700	000002		SUB	#2,	R0	;CHECK FOR ADD/SUB
2466	013472	001021			BNE	5\$;BRANCH IF NOT
2467	013474	005767	165130		TST	\$ADDPS		;CHECK FOR ADD
2468	013500	001007			BNE	4\$;BRANCH IF NOT
2469	013502	016767	165106	165126	MOV	SPSW, SADDER		;SET UP ADD ERROR PSW
2470	013510	012767	000340	165112	MOV	#340, \$ADDPS		;SET UP ADD PSW
2471	013516	000427			BR	8\$		
2472								
2473	013520	016767	165070	165120	4\$: MOV	SPSW, \$SUBER		;SET UP SUBTRACT ERROR PSW
2474	013526	012767	000340	165104	MOV	#340, \$SUBPS		;SET UP SUBTRACT PSW
2475	013534	000420			BR	8\$		

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 50
 DBKEBA.P11 POLISH MODE ROUTINES TO ACCESS FORTRAN ROUTINES

```

2476
2477 213536 162700 000004      5$:   SUB    #4,   R0     ;CHECK FOR MUL
2478 213542 003407               BLE    6$     ;BRANCH IF NOT
2479 213544 016767 165044 165104    MOV    SPSW,  SMULER  ;SET UP MULTIPLY ERROR PSW
2480 213552 012767 000340 165070    MOV    #340,  SMULPS  ;SET UP MULTIPLY PSW
2481 213560 000406               BR     8$     ;
2482
2483 213562 016767 165026 165076  6$:   MOV    SPSW,  SDIVER  ;SET UP DIVIDE ERROR PSW
2484 213570 012767 000340 165062  7$:   MOV    #340,  SDIVPS  ;SET UP DIVIDE PSW
2485 213576 000205               8$:   RTS    R5     ;RETURN TO FORTRAN
2486
2487          ;SUBROUTINE TO PUSH DATA ONTO STACK
2488
2489 213600 016767 165016 165024  PUSHRI  MOV    RAND,B, STACK6  ;PUT DATA ON THE STACK
2490 213606 016767 165006 164774  MOVR   RAND,A, STACK4
2491 213614 016767 165006 164764  MOVR   RAND,D, STACK2
2492 213622 016767 164776 164754  MOVR   RAND,C, STACK0
2493 213630 011637 000244               MOV    (SP),  #244   ;SET UP TRAP VECTOR
2494 213634 062737 000004 000244  ADD    #4,   #244
2495 213642 000207               RTS    PC
2496
2497 213644 032737 000400 177570  SCOPES: BIT    #SW08, #SWR  ;KILL LDUB OR LOOP ON SPEC; TEST
2498 213652 001412               BEQ    1$     ;
2499 213654 013767 177570 000134  MOV    #SWR,  SCOTMP  ;SAVE SWR
2500 213662 042767 177600 000126  BIC    #177600,SCOTMP  ;CLR ALL BUT TEST NO,
2501 213670 126767 000122 165102  CMPB   SCOTMP,ICNT  ;ON RIGHT TEST? #SW6=0#
2502 213676 001434               BEQ    OVERS
2503 213700 032737 040000 177570  1$:   BIT    #SW14, #SWR  ;LOOP ON TEST
2504 213706 001026               BNE    KITS
2505 213710 032737 004000 177570  BIT    #SW11, #SWR  ;KILL ITERATIONS
2506 213716 001012               BNE    SVLADS
2507 213720 105767 165055               TSTB   ICNT+1
2508 213724 001404               BEQ    2$     ;BRANCH IF FIRST
2509 213726 126767 000062 165045  CMPB   TIMES,ICNT+1  ;DONE?
2510 213734 001013               BNE    KITS  ;BRANCH IF NOT
2511 213736 112767 000001 165035  2$:   MOVB   #1,ICNT+1  ;FIRST ITERATION
2512 213744 105267 165030               SVLADS: INCB   ICNT  ;COUNT TEST NUMBERS
2513 213750 011667 000036               MOV    (6),LADS  ;SAVE LOOP ADDRESS
2514 213754 016737 165020 177570  MOV    ICNT, #DISPLAY  ;DISPLAY TEST NO, AND ITERATION COUNT
2515 213762 000002               RTI    ICNT+1
2516
2517 213764 105267 165011               KITS:  INCB   ICNT+1
2518 213770 016737 165004 177570  OVERS:  MOV    ICNT, #DISPLAY  ;SET UP DISPLAY
2519 213776 005767 000010               TST    LADS  ;FIRST ONE?
2520 214002 001760               BEQ    SVLADS
2521 214004 016716 000002               MOV    LADS,(6)  ;FUDGE RETURN ADDRESS
2522 214010 000002               RTI    ICNT+1  ;FIXES PS
2523
2524 214012 000000               LADS:  0     ;LOOP ADDRESS
2525 214014 000377               TIMES: 377  ;RUN 377 TIMES
2526 214016 000000               SCOTMP: 0

```

MAINDEC-11-DBKEB-A
DBKEBA.P11KE11F (PDP-11 F1S) EXERCISER,
HLT ROUTINE (ERROR TYPEOUT)

MACY11,620 22-AUG-72 11:40 PAGE 51

2527	014020	032737	002000	177570	HLTS:	BIT	#SW10,@#SWR	;BELL ON ERROR?
2528	014026	001402				BEO	1\$;NO - SKIP
2529	014030	000004	000007			TYPE	:BELL	;RING BELL
2530	014034	005267	164742		1\$:	INC	ERRORS	;COUNT THE NUMBER OF ERRORS
2531	014040	032737	002000	177570		BIT	#SW13,@#SWR	;SKIP TYPEOUT IF SET
2532	014046	001017				BNE	2\$;SKIP TYPEOUTS
2533	014050	000004	015362			TYPE	:RETURN	
2534	014054	011667	000060			MOV	(6),HLTADS	;PUT ADDRESS OF INSTRUCTION ON STACK
2535	014060	162767	000002	000052		SUB	#2,HLTADS	
2536	014066	016705	000046			MOV	HLTADS,TTY	;TYPE HLTADS IN OCTAL
2537	014072	004767	001300			JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2538	014076	000004	015370			TYPE	,SPACE#3	
2539	014102	004767	000034			JSR	PC, ERRORS	;GO TO USER ERROR ROUTINE
2540	014106	005737	177570		2\$:	TST	@#SWR	;HALT ON ERROR
2541	014112	100001				BPL	,+4	;SKIP IF CONTINUE
2542	014114	000000				HALT		;HALT ON ERROR!
2543	014116	032737	001000	177570		BIT	#SW09,@#SWR	;CHECK FOR INHIBIT LOOP ON ERROR
2544	014124	001001				BNE	,+4	;SKIP IF LOOP ON ERROR
2545	014126	000002				RTI		
2546	014130	105067	164645			CLRB	ICNT+1	
2547	014134	000167	177624			JMP	KITS	;LOOP ON TEST UNTIL NO ERRORS
2548								
2549	014140	000000				HLTADS:	0	
2550								
2551	014142	010046				ERRORS:	MOV R0, -(SP)	;SAVE R0
2552	014144	010146					MOV R1, -(SP)	;SAVE R1
2553	014146	000004	015370			TYPE,	SPACE#3	
2554	014152	016705	164442			MOV	RAND,A,TTY	;TYPE RAND,A IN OCTAL
2555	014156	004767	001214			JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2556	014162	000004	014702			TYPE,	COMMA	
2557	014166	016705	164430			MOV	RAND,B,TTY	
2558	014172	004767	001200			JSR	X7,PRINTR	
2559	014176	013700	000244			MOV	@#244, R0	;GET PC+2 OF INSTRUCTION
2560	014202	014001				MOV	-(R0), R1	;GET THE INSTRUCTION
2561	014204	042701	177747			BIC	#177747,R1	;MASK ALL BUT TYPE (+,-,*,/)
2562	014210	006201				ASR	R1	;DIV BY 2
2563	014212	012767	014662	000006		MOV	#\$IGNS, 1\$;SET TO TOP OF SIGN TABLE
2564	014220	060167	000002			ADD	R1, 1\$;ADD OFFSET
2565	014224	000004				TYPE		
2566	014226	014662			1\$:	SIGNS		
2567	014230	016705	164370			MOV	RAND,C,TTY	
2568	014234	004767	001136			JSR	X7,PRINTR	
2569	014240	000004	014702			TYPE,	COMMA	
2570	014244	016705	164356			MOV	RAND,D,TTY	
2571	014250	004767	001122			JSR	X7,PRINTR	
2572	014254	006301				ASL	R1	
2573	014256	062701	000630			ADD	#\$ADDP(S,R1	
2574	014262	105767	164326			TSTB	SPSW	;CHECK FOR ERROR CONDITIONS
2575	014266	100460				BMI	3\$;BRANCH IF ERROR
2576	014270	000004	014704			TYPE,	HEAD1	
2577	014274	000004	015062			TYPE,	EXPECT	
2578	014300	012105				MOV	(R1)+,TTY	;TYPE (R1)+ IN OCTAL
2579	014302	004767	001070			JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2580	014306	000004	015370			TYPE,	SPACE#3	

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER,
DBKEBA.P11 HLT ROUTINE (ERROR TYPEOUT)

MACY11,620 22-AUG-72 11140 PAGE 52

2581	014312	012705	000610	MOV	#STACK4,TTY	;TYPE #STACK4 IN OCTAL
2582	014316	004767	001054	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2583	014322	000004	015370	TYPE,	SPACE+3	
2584	014326	012105		MOV	(R1)+,TTY	;TYPE (R1)* IN OCTAL
2585	014330	004767	001042	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2586	014334	000004	014702	TYPE,	COMMA	
2587	014340	011105		MOV	(R1),TTY	;TYPE (R1) IN OCTAL
2588	014342	004767	001030	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2589	014346	000004	015076	TYPE,	GOT	
2590	014352	016705	164236	MOV	SPSW,TTY	;TYPE SPSW IN OCTAL
2591	014356	004767	001014	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2592	014362	000004	015370	TYPE,	SPACE+3	
2593	014366	016705	164224	MOV	SSP,TTY	;TYPE SSP IN OCTAL
2594	014372	004767	001000	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2595	014376	000004	015370	TYPE,	SPACE+3	
2596	014402	016705	164202	MOV	ANS1,TTY	;TYPE ANS1 IN OCTAL
2597	014406	004767	000764	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2598	014412	000004	014702	TYPE,	COMMA	
2599	014416	016705	164170	MOV	ANS2,TTY	;TYPE ANS2 IN OCTAL
2600	014422	004767	000750	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2601	014426	000510		BR	7S	
2602						
2603	014430	000004	014751	3\$: TYPE,	HEAD2	
2604	014434	000004	015062	TYPE,	EXPECT	
2605	014440	012105		MOV	(R1)+,TTY	;TYPE (R1)* IN OCTAL
2606	014442	004767	000730	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2607	014446	000004	015370	TYPE,	SPACE+3	
2608	014452	016705	164212	MOV	SAVSTK,TTY	;TYPE SAVSTK IN OCTAL
2609	014456	004767	000714	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2610	014462	000004	015370	TYPE,	SPACE+3	
2611	014466	005720		TST	(R0)+	;UPDATE R0 TO RIGHT ADDRESS
2612	014470	010005		MOV	R0,TTY	;TYPE R0 IN OCTAL
2613	014472	004767	000700	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2614	014476	000004	015370	TYPE,	SPACE+3	
2615	014502	022121		CMP	(R1)+, (R1)+	;ADD 4 TO R1
2616	014504	011105		MOV	(R1),TTY	;TYPE (R1) IN OCTAL
2617	014506	004767	000664	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2618	014512	000004	015370	TYPE,	SPACE+3	
2619	014516	016705	164102	MOV	RAND,C,TTY	;TYPE RAND,C IN OCTAL
2620	014522	004767	000650	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2621	014526	000004	015370	TYPE,	SPACE+3	
2622	014532	016705	164070	MOV	RAND,D,TTY	;TYPE RAND,D IN OCTAL
2623	014536	004767	000634	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2624	014542	000004	015370	TYPE,	SPACE+3	
2625	014546	016705	164046	MOV	RAND,A,TTY	;TYPE RAND,A IN OCTAL
2626	014552	004767	000620	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2627	014556	000004	015370	TYPE,	SPACE+3	
2628	014562	016705	164034	MOV	RAND,B,TTY	;TYPE RAND,B IN OCTAL
2629	014566	004767	000604	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2630	014572	000004	015076	TYPE,	GOT	
2631	014576	016705	164012	MOV	SPSW,TTY	;TYPE SPSW IN OCTAL
2632	014602	004767	000570	JSR	X7,PRINTR	;TYPE LEADING ZERO'S
2633	014606	000004	015370	TYPE,	SPACE+3	
2634	014612	016705	164000	MOV	SSP,TTY	;TYPE SSP IN OCTAL

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 53
DBKEBA.P11 HLT ROUTINE (ERROR TIMEOUT)

2680							
2681	015112	010346	READ\$1	MOV	R3,-(6)	;SAVE R3	
2682	015114	012703	015212	1\$: MOV	#INPUT,R3	;GET ADDRESS	
2683	015120	022703	015252	2\$: CMP	#,QUES, R3	;CHECK FOR BUFFER OVERFLOW	
2684	015124	001412		BEQ	4\$;ABORT	
2685	015126	105737	177560	TSTB	##177560	;WAIT FOR	
2686	015132	100375		BPL	,=4	;A CHARACTER	
2687	015134	113713	177562	MOV B	##177562,(3)	;GET CHARACTER	
2688	015140	142713	000200	BICB	#200,(3)	;GET RID OF JUNK	
2689	015144	122713	000177	CMPB	#177,(3)	;IS IT A RUBOUT	
2690	015150	001003		BNE	3\$;SKIP IF NOT	
2691	015152	000004	015252	4\$: TYPE	,,QUES	;TYPE A '?'	
2692	015156	000756		BR	1\$;ZAP THE BUFFER AND LOOP	
2693	015160	111367	000210	3\$: MOVB	(3),,TYPE	;SET UP FOR TYPING	
2694	015164	000004	015374	TYPE	,,TYPE	;ECHO IT	
2695	015170	122723	000015	CMPB	#15,(3)+	;CHECK FOR RETURN	
2696	015174	001351		BNE	2\$;LOOP IF NOT RETURN	
2697	015176	105063	177777	CLRB	=1(3)	;ZAP RETURN (THE 15)	
2698	015202	000004	000012	TYPE	,12	;TYPE A LINE FEED	
2699	015206	012603		MOV	(6)+,R3	;RESTORE R3	
2700	015210	000207		RTS	PC	;RETURN	
2701							
2702	015212	000020		INPUT1	,BLKW	20	
2703	015252	006477	000012	,QUEST1	,ASCII2	"?"<15><12>	
2704							
2705	015256	010546		,IOT:	MOV	TTY,-(6)	;SAVE TTY
2706	015260	017605	000002		MOV	#2(6),TTY	;GET ADDRESS TO BE TYPED
2707	015264	032705	177400		BIT	#177400,TTY	;IS IT A TYPEM?
2708	015270	001004			BNE	1\$;NO
2709	015272	010567	000076		MOV	TTY,,TYPE	;GET THE CHARACTER
2710	015276	012705	015374		MOV	,#TYPE,TTY	;FUDGE THE ADDRESS
2711	015302	105715		1\$: TSTB	(TTY)	;TERMINATOR?	
2712	015304	001406		BEQ	2\$;GET OUT IF SO	
2713	015306	112537	177566		MOVB	(TTY)+,#177566	;LOAD AND TYPE THE CHARACTER
2714	015312	105737	177564		TSTB	##177564	;IS THE PRINTER READY
2715	015316	100375			BPL	,=4	;WAIT UNTIL IT IS
2716	015320	000770			BR	1\$;GET THE NEXT CHARACTER
2717	015322	017646	000002	2\$: MOV	#2(6),-(6)	;GET ADDRESS TO BE TYPED	
2718	015326	062766	000002	ADD	#2,4(6)	;ADD 2 TO THE ADDRESS	
2719	015334	022666	000002	CMP	(6)+,2(6)	;IS IT ,+2?	
2720	015340	001006		BNE	3\$;NO	
2721	015342	062705	000002	ADD	#2,TTY	;ADD 2 TO THE ADDRESS	
2722	015346	042705	000001	BIC	#1,TTY	;BACK UP TO AN EVEN BYTE	
2723	015352	010566	000002	MOV	TTY,2(6)	;RESTORE ADDRESS	
2724	015356	012605		3\$: MOV	(6)+,TTY	;RESTORE TTY	
2725	015360	000002		RTI		;RETURN	
2726							
2727	015362	005015	000	RETURN1	,ASCII2	<15><12>	;RETURN AND LINEFEED
2728	015365	015	020012	SPACE1	,ASCII2	<15><12>" "	;RETURN AND 3 SPACES
2729	015372	000		,EVEN			
2730	015374	015374		,TYPE1	0		;CHARACTER TYPE LOCATION
2731	015374	000000					

MAINDEC-11-DBKEB-A KE11F (PDP-11 FIS) EXERCISER. MACY11,620 22-AUG-72 11:40 PAGE 55
DBKEBA.P11 TYPE ROUTINE

2732								
2733	015376	112767	000001	000130	PRINTR:	MOV B #1,,PR		SET ZERO FILL SWITCH
2734	015404	000402				BR ,#6		SKIP
2735	015406	005067	000122		PRINTS:	CLR ,PR		SUPPRESS LEADING ZERO'S
2736	015412	112767	177772	000115		MOV B #=6,,PR+1		SET COUNT
2737	015420	010446				MOV R4,-(6)		SAVE R4
2738	015422	012704	015524			MOV #,PRBUF,R4		SET POINTER TO FIRST ASCII CHAR,
2739	015426	105014				CLRB (4)		CLEAR FIRST BYTE
2740	015430	000405			,PRL:	BR ,PRF		ROTATE FIRST BIT
2741	015432	105014				CLRB (4)		CLEAR BYTE OF CHARACTER
2742	015434	006105				ROL TTY		ROTATE BIT INTO C
2743	015436	106114				ROLB (4)		PACK IT
2744	015440	006105				ROL TTY		ROTATE BIT INTO C
2745	015442	106114				ROLB (4)		PACK IT
2746	015444	006105			,PRF:	ROL TTY		ROTATE BIT INTO C
2747	015446	106114				ROLB (4)		PACK IT
2748	015450	105714				TSTB (4)		IS IT ZERO?
2749	015452	001402				BEQ ,#6		SKIP INC
2750	015454	105267	000054			INC B ,PR		SET FILL SWITCH
2751	015460	105767	000050			TSTB ,PR		CHECK FILL SWITCH
2752	015464	001402				BEO ,#6		SKIP BITSET
2753	015466	152724	000060			BISB #10,(4)+		MAKE INTO ASCII CHAR
2754	015472	105267	000037			INC B ,PR+1		INC COUNT
2755	015476	001355				BNE ,PRL		REPEAT
2756	015500	022704	015524			CMP #,PRBUF,R4		EMPTY BUFFER?
2757	015504	001002				BNE ,#6		SKIP IF NOT
2758	015506	112724	000060			MOV B #10,(4)+		LOAD 1 ZERO
2759	015512	105014				CLRB (4)		NULL TERMINATOR
2760	015514	000004	015524			TYPE ,,PRBUF		TYPE IT
2761	015520	012604				MOV (6)+,R4		RESTORE R4
2762	015522	000207				RTS PC		RETURN
2763								
2764	015524	000004			,PRBUF: ,BLKW	4		OUTPUT BUFFER
2765	015534	000000			,PR: 0			COUNT AND SWITCH

MAINDEC-11-DBKEBA-A KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 56
DBKEBA.P11 OCTAL DUMP OF A WORD & 18 BIT ADDRESS TYPER

2766
2767 215536 012777 015652 000120 PDOWN\$: MOV #ILLUP, @PUVECS ;SET FOR FAST UP
2768 215544 012777 000340 000114 MOV #340, @PUVECS+2 ;PRI017
2769 215552 010046 MOV R0,-(6) ;PUSH R0 ON STACK
2770 215554 010146 MOV R1,-(6) ;PUSH R1 ON STACK
2771 215556 010246 MOV R2,-(6) ;PUSH R2 ON STACK
2772 215560 010346 MOV R3,-(6) ;PUSH R3 ON STACK
2773 215562 010446 MOV R4,-(6) ;PUSH R4 ON STACK
2774 215564 010546 MOV R5,-(6) ;PUSH R5 ON STACK
2775 215566 010667 200064 MOV SP, ,SAVR6 ;SAVE SP
2776 215572 012777 015602 000064 MOV #PUPS, @PUVECS ;SET UP VECTOR
2777 215600 000000 HALT
2778
2779 215602 016706 000050 PUPS: MOV ,SAVR6,SP ;GET SP
2780 215606 005001 CLR R1 ;WAIT LOOP FOR THE TTY
2781 215610 005201 IS: INC R1 ;WAIT FOR THE INC
2782 215612 001376 BNE IS ;JOF WORD
2783 215614 012605 MOV (6)+,R5 ;IPOP STACK INTO R5
2784 215616 012604 MOV (6)+,R4 ;IPOP STACK INTO R4
2785 215620 012603 MOV (6)+,R3 ;IPOP STACK INTO R3
2786 215622 012602 MOV (6)+,R2 ;IPOP STACK INTO R2
2787 215624 012601 MOV (6)+,R1 ;IPOP STACK INTO R1
2788 215626 012600 MOV (6)+,R0 ;IPOP STACK INTO R0
2789 215630 012777 015536 000022 MOV #PDOWN\$, @PDVECS ;SET UP THE POWER DOWN VECTOR
2790 215636 012777 000340 000016 MOV #340, @PDVECS+2 ;PRI017
2791 215644 000004 015670 TYPE ,POWERS
2792 215650 000002 RTI
2793
2794 215652 000000 ILLUPI HALT ;THE POWER UP SEQUENCE WAS STARTED
2795 215654 000776 BR ,#2 ;BEFORE THE POWER DOWN WAS COMPLETE
2796
2797 215656 000000 ,SAVR6: 0 ;PUT THE SP HERE
2798 215660 000024 000026 PDVECS: 24,26 ;POWER DOWN VECTOR
2799 215664 000024 000026 PUVECS: 24,26 ;POWER UP VECTOR
2800 215670 005015 047520 042527 POWERS: ,ASCIZ <15><12>"POWER"
2801 215676 000122 ,EVEN
2802
2803
2804 000001 ,END

MAINDEC-11-DBKEB-A KE11
DBKEBA.P11 SYMBOL TABLE

KE11F (PDP-11 FIS) EXERCISER, MACY11,620 22-AUG-72 11:40 PAGE 57

ANS1	000610	ANS2	000612	BEGIN	001010	BELL	= 000007
BIT0	= 000001	BIT1	= 000002	BIT10	= 002000	BIT11	= 004000
BIT12	= 010000	BIT13	= 020000	BIT14	= 040000	BIT15	= 100000
BIT2	= 020004	BIT3	= 000010	BIT4	= 000020	BIT5	= 000040
BIT6	= 000100	BIT7	= 000200	BIT8	= 000400	BIT9	= 001000
CCC	= 00257	COMMA	014702	DISPLA	177570	DONE	012662
ERRORS	001002	ERROR\$	014142	EXPECT	015062	FADD	075000
FDIV	= 075030	FISTRP	000754	FMUL	= 075020	FORTAN	001360
FSUB	= 075010	GOT	015076	HEAD1	014704	HEAD2	014751
HLT	= 104000	HLTADS	014140	HTLS	014020	ICNT	001000
ILLUP	015652	INPUT	015212	KITS	013764	LADS	014012
LEVEL0	= 000000	LEVEL1	= 000040	LEVEL2	= 000100	LEVEL3	= 000140
LEVEL4	= 000200	LEVEL5	= 000240	LEVEL6	= 000300	LEVEL7	= 000340
N	= 000035	OVERS	013770	PC	=X000007	PCNT	001004
PDOWNS	015536	PDVECS	015660	POWERS	015670	PRINTR	015376
PRINTS	015406	PS	= 177776	PUPS	015602	PUSHR	013600
PUVECS	015664	RAND.A	000620	RAND.B	000622	RAND.C	000624
RAND.D	000626	RAND4\$	000674	READIN	012762	READS	015112
RETURN	015362	RNDFLG	000672	R0	=X000000	R1	=%0000001
R2	=%000002	R3	=%000003	R4	=%000004	R5	=%000005
SAVSTK	000670	SCC	= 000277	SCOPE	= 104400	SCOPES	013644
SCOTMP	014016	SIGNS	014662	SNV	= 000272	SP	=%000006
SPACE	015365	STACK0	000604	STACK2	000606	STACK4	000610
STACK6	000612	START	001146	STK1	000600	STK2	000602
STK3	000604	STK4	000606	STK5	000610	STK6	000612
SVLADS	013744	SWR	= 177570	SW08	= 000400	SW09	= 001000
SW10	= 002000	SW11	= 004000	SW12	= 010000	SW13	= 020000
SW14	= 040000	SW15	= 100000	TIMES	014014	TST1	001440
TST10	004202	TST11	004442	TST12	004702	TST13	005142
TST14	005402	TST15	005642	TST16	006102	TST17	006344
TST2	002000	TST20	006604	TST21	007044	TST22	007304
TST23	007544	TST24	010004	TST25	010244	TST26	010506
TST27	010746	TST3	002340	TST30	011206	TST31	011446
TST32	011706	TST33	012146	TST34	012406	TST4	002700
TST5	003240	TST6	003500	TST7	003740	TTY	=X000005
TYPE	= 000004	TYPIN	001210	YESRT	000752	SADDER	000630
SADDPS	000630	SADD1	000632	SADD2	000634	SADR	= ****#
SDIVER	000666	SDIVPS	000660	SDIV1	000662	SDIV2	000664
\$DVR	= ***** G	SERR	013414 G	SERRA	013420 G	SEXIT	013412
\$MLR	= ***** G	SMULER	000656	\$MULPS	000650	SMUL1	000652
\$MUL2	000654	SPOLSH	013064	\$POPAD	013066	SPOPDV	013300
\$POMPML	013222	SPOPSB	013144	\$PSW	000614	SPUSH	013044
\$\$BR	= ***** G	SSKIP	013406	\$SP	000616	SSUBER	000646
\$\$SUBPS	000640	SSUB1	000642	SSUB2	000644	BIT	= 177777
.IOT	015256	,PR	015534	,PRBUF	015524	PRF	015444
.PRL	015432	,QUES	015252	,SAVR6	015656	TBIT	012760
.TYPE	015374		= 015700				

0203320

ERRORS DETECTED: 0

```

1      ;TITLE SADR
2      ;CSECT
3      ;GLOBL SADR,SSBR,$ERR
4      ;SADR ---- THE REAL ADD ROUTINE
5      ;COPYRIGHT 1971, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
6      ;REPLACE THE TWO ITEMS ON TOP OF THE STACK
7      ;WITH THEIR SUM,
8      ;SSBR ---- THE REAL SUBTRACT ROUTINE
9      ;SUBTRACT THE TOP STACK ITEM FROM THE SECOND ITEM
10     ;REPLACE THEM BOTH WITH THE DIFFERENCE,
11     000000
12     000001
13     000002
14     000003
15     000004
16     000005
17     000006
18     000007
19     000000
20     000004
21     000006
22     000010
23     000012
24     177302
25     177304
26     177312
27     177316
28     000000
29     000000' 062716 100000      SSBR: ADD    #100000, @SP      ;CHANGE THE SIGN OF TOP ITEM
30           001          SADRI: IFDF FPU
31           SADR: WORD 170001  I18ETP
32           WORD 172426  I1LDF  (SP)+,F0      ;GET OPERAND
33           WORD 172026  I1ADD  (SP)+,F0      ;ADD
34           WORD 174046  I1STP  F0,-(SP)      ;SUM TO STACK
35           JMP   @({R4})+
36           ENDC
37           000
38           001
39           SADR: IFNDF FPU
40           MOV   R4,-(SP)
41           CLR   -(SP)      ;CLEAR SIGNS
42           000004' 010446
43           000006' 005046
44           00010' 005002
45           00012' 005003
46           00014' 006366 000006
47           00020' 006166 000004
48           00024' 156603 000005
49           00030' 001574
50           00032' 106116
51           00034' 006366 000012
52           00040' 006166 000010
53           00044' 156602 000011
54           00050' 001014
55           00052' 106016
56           00054' 006066 000004
57           00060' 006066 000006
58           00064' 016666 000004 000010      SSBR: R0=X0
59           R1=X1
60           R2=X2
61           R3=X3
62           R4=X4
63           R5=X5
64           R6=X6
65           PC=X7
66           SIGNS@0
67           A1#4
68           B1#6
69           A2#8
70           B2#10
71           AC#177302
72           MQ#177304
73           NOR#177312
74           ASH#177316
75           F0=X0
76           R1=X1
77           R2=X2
78           R3=X3
79           R4=X4
80           R5=X5
81           R6=X6
82           PC=X7
83           SIGNS@0
84           A1#4
85           B1#6
86           A2#8
87           B2#10
88           AC#177302
89           MQ#177304
90           NOR#177312
91           ASH#177316
92           F0=X0
93           R1=X1
94           R2=X2
95           R3=X3
96           R4=X4
97           R5=X5
98           R6=X6
99           PC=X7
100          SIGNS@0
101          A1#4
102          B1#6
103          A2#8
104          B2#10
105          AC#177302
106          MQ#177304
107          NOR#177312
108          ASH#177316
109          F0=X0
110          R1=X1
111          R2=X2
112          R3=X3
113          R4=X4
114          R5=X5
115          R6=X6
116          PC=X7
117          SIGNS@0
118          A1#4
119          B1#6
120          A2#8
121          B2#10
122          AC#177302
123          MQ#177304
124          NOR#177312
125          ASH#177316
126          F0=X0
127          R1=X1
128          R2=X2
129          R3=X3
130          R4=X4
131          R5=X5
132          R6=X6
133          PC=X7
134          SIGNS@0
135          A1#4
136          B1#6
137          A2#8
138          B2#10
139          AC#177302
140          MQ#177304
141          NOR#177312
142          ASH#177316
143          F0=X0
144          R1=X1
145          R2=X2
146          R3=X3
147          R4=X4
148          R5=X5
149          R6=X6
150          PC=X7
151          SIGNS@0
152          A1#4
153          B1#6
154          A2#8
155          B2#10
156          AC#177302
157          MQ#177304
158          NOR#177312
159          ASH#177316
160          F0=X0
161          R1=X1
162          R2=X2
163          R3=X3
164          R4=X4
165          R5=X5
166          R6=X6
167          PC=X7
168          SIGNS@0
169          A1#4
170          B1#6
171          A2#8
172          B2#10
173          AC#177302
174          MQ#177304
175          NOR#177312
176          ASH#177316
177          F0=X0
178          R1=X1
179          R2=X2
180          R3=X3
181          R4=X4
182          R5=X5
183          R6=X6
184          PC=X7
185          SIGNS@0
186          A1#4
187          B1#6
188          A2#8
189          B2#10
190          AC#177302
191          MQ#177304
192          NOR#177312
193          ASH#177316
194          F0=X0
195          R1=X1
196          R2=X2
197          R3=X3
198          R4=X4
199          R5=X5
200          R6=X6
201          PC=X7
202          SIGNS@0
203          A1#4
204          B1#6
205          A2#8
206          B2#10
207          AC#177302
208          MQ#177304
209          NOR#177312
210          ASH#177316
211          F0=X0
212          R1=X1
213          R2=X2
214          R3=X3
215          R4=X4
216          R5=X5
217          R6=X6
218          PC=X7
219          SIGNS@0
220          A1#4
221          B1#6
222          A2#8
223          B2#10
224          AC#177302
225          MQ#177304
226          NOR#177312
227          ASH#177316
228          F0=X0
229          R1=X1
230          R2=X2
231          R3=X3
232          R4=X4
233          R5=X5
234          R6=X6
235          PC=X7
236          SIGNS@0
237          A1#4
238          B1#6
239          A2#8
240          B2#10
241          AC#177302
242          MQ#177304
243          NOR#177312
244          ASH#177316
245          F0=X0
246          R1=X1
247          R2=X2
248          R3=X3
249          R4=X4
250          R5=X5
251          R6=X6
252          PC=X7
253          SIGNS@0
254          A1#4
255          B1#6
256          A2#8
257          B2#10
258          AC#177302
259          MQ#177304
260          NOR#177312
261          ASH#177316
262          F0=X0
263          R1=X1
264          R2=X2
265          R3=X3
266          R4=X4
267          R5=X5
268          R6=X6
269          PC=X7
270          SIGNS@0
271          A1#4
272          B1#6
273          A2#8
274          B2#10
275          AC#177302
276          MQ#177304
277          NOR#177312
278          ASH#177316
279          F0=X0
280          R1=X1
281          R2=X2
282          R3=X3
283          R4=X4
284          R5=X5
285          R6=X6
286          PC=X7
287          SIGNS@0
288          A1#4
289          B1#6
290          A2#8
291          B2#10
292          AC#177302
293          MQ#177304
294          NOR#177312
295          ASH#177316
296          F0=X0
297          R1=X1
298          R2=X2
299          R3=X3
300          R4=X4
301          R5=X5
302          R6=X6
303          PC=X7
304          SIGNS@0
305          A1#4
306          B1#6
307          A2#8
308          B2#10
309          AC#177302
310          MQ#177304
311          NOR#177312
312          ASH#177316
313          F0=X0
314          R1=X1
315          R2=X2
316          R3=X3
317          R4=X4
318          R5=X5
319          R6=X6
320          PC=X7
321          SIGNS@0
322          A1#4
323          B1#6
324          A2#8
325          B2#10
326          AC#177302
327          MQ#177304
328          NOR#177312
329          ASH#177316
330          F0=X0
331          R1=X1
332          R2=X2
333          R3=X3
334          R4=X4
335          R5=X5
336          R6=X6
337          PC=X7
338          SIGNS@0
339          A1#4
340          B1#6
341          A2#8
342          B2#10
343          AC#177302
344          MQ#177304
345          NOR#177312
346          ASH#177316
347          F0=X0
348          R1=X1
349          R2=X2
350          R3=X3
351          R4=X4
352          R5=X5
353          R6=X6
354          PC=X7
355          SIGNS@0
356          A1#4
357          B1#6
358          A2#8
359          B2#10
360          AC#177302
361          MQ#177304
362          NOR#177312
363          ASH#177316
364          F0=X0
365          R1=X1
366          R2=X2
367          R3=X3
368          R4=X4
369          R5=X5
370          R6=X6
371          PC=X7
372          SIGNS@0
373          A1#4
374          B1#6
375          A2#8
376          B2#10
377          AC#177302
378          MQ#177304
379          NOR#177312
380          ASH#177316
381          F0=X0
382          R1=X1
383          R2=X2
384          R3=X3
385          R4=X4
386          R5=X5
387          R6=X6
388          PC=X7
389          SIGNS@0
390          A1#4
391          B1#6
392          A2#8
393          B2#10
394          AC#177302
395          MQ#177304
396          NOR#177312
397          ASH#177316
398          F0=X0
399          R1=X1
400          R2=X2
401          R3=X3
402          R4=X4
403          R5=X5
404          R6=X6
405          PC=X7
406          SIGNS@0
407          A1#4
408          B1#6
409          A2#8
410          B2#10
411          AC#177302
412          MQ#177304
413          NOR#177312
414          ASH#177316
415          F0=X0
416          R1=X1
417          R2=X2
418          R3=X3
419          R4=X4
420          R5=X5
421          R6=X6
422          PC=X7
423          SIGNS@0
424          A1#4
425          B1#6
426          A2#8
427          B2#10
428          AC#177302
429          MQ#177304
430          NOR#177312
431          ASH#177316
432          F0=X0
433          R1=X1
434          R2=X2
435          R3=X3
436          R4=X4
437          R5=X5
438          R6=X6
439          PC=X7
440          SIGNS@0
441          A1#4
442          B1#6
443          A2#8
444          B2#10
445          AC#177302
446          MQ#177304
447          NOR#177312
448          ASH#177316
449          F0=X0
450          R1=X1
451          R2=X2
452          R3=X3
453          R4=X4
454          R5=X5
455          R6=X6
456          PC=X7
457          SIGNS@0
458          A1#4
459          B1#6
460          A2#8
461          B2#10
462          AC#177302
463          MQ#177304
464          NOR#177312
465          ASH#177316
466          F0=X0
467          R1=X1
468          R2=X2
469          R3=X3
470          R4=X4
471          R5=X5
472          R6=X6
473          PC=X7
474          SIGNS@0
475          A1#4
476          B1#6
477          A2#8
478          B2#10
479          AC#177302
480          MQ#177304
481          NOR#177312
482          ASH#177316
483          F0=X0
484          R1=X1
485          R2=X2
486          R3=X3
487          R4=X4
488          R5=X5
489          R6=X6
490          PC=X7
491          SIGNS@0
492          A1#4
493          B1#6
494          A2#8
495          B2#10
496          AC#177302
497          MQ#177304
498          NOR#177312
499          ASH#177316
500          F0=X0
501          R1=X1
502          R2=X2
503          R3=X3
504          R4=X4
505          R5=X5
506          R6=X6
507          PC=X7
508          SIGNS@0
509          A1#4
510          B1#6
511          A2#8
512          B2#10
513          AC#177302
514          MQ#177304
515          NOR#177312
516          ASH#177316
517          F0=X0
518          R1=X1
519          R2=X2
520          R3=X3
521          R4=X4
522          R5=X5
523          R6=X6
524          PC=X7
525          SIGNS@0
526          A1#4
527          B1#6
528          A2#8
529          B2#10
530          AC#177302
531          MQ#177304
532          NOR#177312
533          ASH#177316
534          F0=X0
535          R1=X1
536          R2=X2
537          R3=X3
538          R4=X4
539          R5=X5
540          R6=X6
541          PC=X7
542          SIGNS@0
543          A1#4
544          B1#6
545          A2#8
546          B2#10
547          AC#177302
548          MQ#177304
549          NOR#177312
550          ASH#177316
551          F0=X0
552          R1=X1
553          R2=X2
554          R3=X3
555          R4=X4
556          R5=X5
557          R6=X6
558          PC=X7
559          SIGNS@0
560          A1#4
561          B1#6
562          A2#8
563          B2#10
564          AC#177302
565          MQ#177304
566          NOR#177312
567          ASH#177316
568          F0=X0
569          R1=X1
570          R2=X2
571          R3=X3
572          R4=X4
573          R5=X5
574          R6=X6
575          PC=X7
576          SIGNS@0
577          A1#4
578          B1#6
579          A2#8
580          B2#10
581          AC#177302
582          MQ#177304
583          NOR#177312
584          ASH#177316
585          F0=X0
586          R1=X1
587          R2=X2
588          R3=X3
589          R4=X4
590          R5=X5
591          R6=X6
592          PC=X7
593          SIGNS@0
594          A1#4
595          B1#6
596          A2#8
597          B2#10
598          AC#177302
599          MQ#177304
600          NOR#177312
601          ASH#177316
602          F0=X0
603          R1=X1
604          R2=X2
605          R3=X3
606          R4=X4
607          R5=X5
608          R6=X6
609          PC=X7
610          SIGNS@0
611          A1#4
612          B1#6
613          A2#8
614          B2#10
615          AC#177302
616          MQ#177304
617          NOR#177312
618          ASH#177316
619          F0=X0
620          R1=X1
621          R2=X2
622          R3=X3
623          R4=X4
624          R5=X5
625          R6=X6
626          PC=X7
627          SIGNS@0
628          A1#4
629          B1#6
630          A2#8
631          B2#10
632          AC#177302
633          MQ#177304
634          NOR#177312
635          ASH#177316
636          F0=X0
637          R1=X1
638          R2=X2
639          R3=X3
640          R4=X4
641          R5=X5
642          R6=X6
643          PC=X7
644          SIGNS@0
645          A1#4
646          B1#6
647          A2#8
648          B2#10
649          AC#177302
650          MQ#177304
651          NOR#177312
652          ASH#177316
653          F0=X0
654          R1=X1
655          R2=X2
656          R3=X3
657          R4=X4
658          R5=X5
659          R6=X6
660          PC=X7
661          SIGNS@0
662          A1#4
663          B1#6
664          A2#8
665          B2#10
666          AC#177302
667          MQ#177304
668          NOR#177312
669          ASH#177316
670          F0=X0
671          R1=X1
672          R2=X2
673          R3=X3
674          R4=X4
675          R5=X5
676          R6=X6
677          PC=X7
678          SIGNS@0
679          A1#4
680          B1#6
681          A2#8
682          B2#10
683          AC#177302
684          MQ#177304
685          NOR#177312
686          ASH#177316
687          F0=X0
688          R1=X1
689          R2=X2
690          R3=X3
691          R4=X4
692          R5=X5
693          R6=X6
694          PC=X7
695          SIGNS@0
696          A1#4
697          B1#6
698          A2#8
699          B2#10
700          AC#177302
701          MQ#177304
702          NOR#177312
703          ASH#177316
704          F0=X0
705          R1=X1
706          R2=X2
707          R3=X3
708          R4=X4
709          R5=X5
710          R6=X6
711          PC=X7
712          SIGNS@0
713          A1#4
714          B1#6
715          A2#8
716          B2#10
717          AC#177302
718          MQ#177304
719          NOR#177312
720          ASH#177316
721          F0=X0
722          R1=X1
723          R2=X2
724          R3=X3
725          R4=X4
726          R5=X5
727          R6=X6
728          PC=X7
729          SIGNS@0
730          A1#4
731          B1#6
732          A2#8
733          B2#10
734          AC#177302
735          MQ#177304
736          NOR#177312
737          ASH#177316
738          F0=X0
739          R1=X1
740          R2=X2
741          R3=X3
742          R4=X4
743          R5=X5
744          R6=X6
745          PC=X7
746          SIGNS@0
747          A1#4
748          B1#6
749          A2#8
750          B2#10
751          AC#177302
752          MQ#177304
753          NOR#177312
754          ASH#177316
755          F0=X0
756          R1=X1
757          R2=X2
758          R3=X3
759          R4=X4
760          R5=X5
761          R6=X6
762          PC=X7
763          SIGNS@0
764          A1#4
765          B1#6
766          A2#8
767          B2#10
768          AC#177302
769          MQ#177304
770          NOR#177312
771          ASH#177316
772          F0=X0
773          R1=X1
774          R2=X2
775          R3=X3
776          R4=X4
777          R5=X5
778          R6=X6
779          PC=X7
780          SIGNS@0
781          A1#4
782          B1#6
783          A2#8
784          B2#10
785          AC#177302
786          MQ#177304
787          NOR#177312
788          ASH#177316
789          F0=X0
790          R1=X1
791          R2=X2
792          R3=X3
793          R4=X4
794          R5=X5
795          R6=X6
796          PC=X7
797          SIGNS@0
798          A1#4
799          B1#6
800          A2#8
801          B2#10
802          AC#177302
803          MQ#177304
804          NOR#177312
805          ASH#177316
806          F0=X0
807          R1=X1
808          R2=X2
809          R3=X3
810          R4=X4
811          R5=X5
812          R6=X6
813          PC=X7
814          SIGNS@0
815          A1#4
816          B1#6
817          A2#8
818          B2#10
819          AC#177302
820          MQ#177304
821          NOR#177312
822          ASH#177316
823          F0=X0
824          R1=X1
825          R2=X2
826          R3=X3
827          R4=X4
828          R5=X5
829          R6=X6
830          PC=X7
831          SIGNS@0
832          A1#4
833          B1#6
834          A2#8
835          B2#10
836          AC#177302
837          MQ#177304
838          NOR#177312
839          ASH#177316
840          F0=X0
841          R1=X1
842          R2=X2
843          R3=X3
844          R4=X4
845          R5=X5
846          R6=X6
847          PC=X7
848          SIGNS@0
849          A1#4
850          B1#6
851          A2#8
852          B2#10
853          AC#177302
854          MQ#177304
855          NOR#177312
856          ASH#177316
857          F0=X0
858          R1=X1
859          R2=X2
860          R3=X3
861          R4=X4
862          R5=X5
863          R6=X6
864          PC=X7
865          SIGNS@0
866          A1#4
867          B1#6
868          A2#8
869          B2#10
870          AC#177302
871          MQ#177304
872          NOR#177312
873          ASH#177316
874          F0=X0
875          R1=X1
876          R2=X2
877          R3=X3
878          R4=X4
879          R5=X5
880          R6=X6
881          PC=X7
882          SIGNS@0
883          A1#4
884          B1#6
885          A2#8
886          B2#10
887          AC#177302
888          MQ#177304
889          NOR#177312
890          ASH#177316
891          F0=X0
892          R1=X1
893          R2=X2
894          R3=X3
895          R4=X4
896          R5=X5
897          R6=X6
898          PC=X7
899          SIGNS@0
900          A1#4
901          B1#6
902          A2#8
903          B2#10
904          AC#177302
905          MQ#177304
906          NOR#177312
907          ASH#177316
908          F0=X0
909          R1=X1
910          R2=X2
911          R3=X3
912          R4=X4
913          R5=X5
914          R6=X6
915          PC=X7
916          SIGNS@0
917          A1#4
918          B1#6
919          A2#8
920          B2#10
921          AC#177302
922          MQ#177304
923          NOR#177312
924          ASH#177316
925          F0=X0
926          R1=X1
927          R2=X2
928          R3=X3
929          R4=X4
930          R5=X5
931          R6=X6
932          PC=X7
933          SIGNS@0
934          A1#4
935          B1#6
936          A2#8
937          B2#10
938          AC#177302
939          MQ#177304
940          NOR#177312
941          ASH#177316
942          F0=X0
943          R1=X1
944          R2=X2
945          R3=X3
946          R4=X4
947          R5=X5
948          R6=X6
949          PC=X7
950          SIGNS@0
951          A1#4
952          B1#6
953          A2#8
954          B2#10
955          AC#177302
956          MQ#177304
957          NOR#177312
958          ASH#177316
959          F0=X0
960          R1=X1
961          R2=X2
962          R3=X3
963          R4=X4
964          R5=X5
965          R6=X6
966          PC=X7
967          SIGNS@0
968          A1#4
969          B1#6
970          A2#8
971          B2#10
972          AC#177302
973          MQ#177304
974          NOR#177312
975          ASH#177316
976          F0=X0
977          R1=X1
978          R2=X2
979          R3=X3
980          R4=X4
981          R5=X5
982          R6=X6
983          PC=X7
984          SIGNS@0
985          A1#4
986          B1#6
987          A2#8
988          B2#10
989          AC#177302
990          MQ#177304
991          NOR#177312
992          ASH#177316
993          F0=X0
994          R1=X1
995          R2=X2
996          R3=X3
997          R4=X4
998          R5=X5
999          R6=X6
1000         PC=X7

```

SADR MACY11,620 22-AUG-72 11:41 PAGE 2
ADR04.PAL

55 000072' 016666 000006 000012 MOV B1(SP),B2(SP)
56 000100' 000550 BR OUT ;DONE
57 000102' 106166 000001 A2NZ: ROLB SIGNS+1(SP) ;GET S2
58 000106' 112766 000001 000011 MOVB #1,A2+1(SP) ;INSERT NORMAL BIT
59 000114' 112766 000001 000005 MOVB #1,A1+1(SP) ;INSERT NORMAL BIT
60 000122' 160302 SUB R3,R2 ;R2=E2-E1, R3=E1
61 000124' 003005 BGT EXPA ;JUMP IF E2>=E1
62 000126' 016600 000010 MOV A2(SP),R0 ;R0=A2
63 000132' 016601 000012 MOV B2(SP),R1 ;R1=B2
64 000136' 000415 BR SCHK ;CHECK SIGNS
65 000140' 060203 EXPAA: ADD R2,R3 ;R2=E2-E1,R3=E2,E2>E1
66 000142' 016600 000004 MOV A1(SP),R0 ;R0=A1
67 000146' 016601 000006 MOV B1(SP),R1 ;R1=B1
68 000152' 016666 000010 000004 MOV A2(SP),A1(SP)
69 000160' 016666 000012 000006 MOV B2(SP),B1(SP)
70 000166' 000316 SWAB @SP ;EXCHANGE SIGNS
71 000170' 005402 NEG R2 ;E1=E2
72 000172' 126616 000001 SCHK: CMPB SIGNS+1(SP),@SP ;SEE IF SIGNS ARE THE SAME
73 000176' 001403 BEQ ECHK ;YES, CHECK EXPONENTS
74 000200' 005401 NEG R1 ;NEGATE FRACTION
75 000202' 005500 ADC R0
76 000204' 005400 NEG R0
77 000206' 005702 ECHK: TST R2
78 000210' 001450 BEQ SHFTD ;JUMP IF E1=E2
79 000212' 022702 177747 SHFT: CMP #=25,,R2 ;IS THERE ANY POINT IN SHIFTING?
80 000216' 003405 BLE SHFTR ;YES
81 000220' 016600 000004 MOV A1(SP),R0 ;NO, ANSWER IS OPERAND
82 000224' 016601 000006 MOV B1(SP),R1 ;WITH THE LARGER EXPONENT
83 000230' 000456 BR NORMD
84 0002 002 ,IFDF EAE
85 SHFTRI: MOV R1,@#MQ ;MOVE FRACTION TO AC,MQ
86 MOV R0,@#AC
87 MOV R2,@#ASH ;SHIFT RIGHT TO EQUALIZE EXPONENTS
88 MOV @#MQ,R1 ;RECOVER SHIFTED FRACTION
89 MOV @#AC,R0
90 001 ,ENDC
91 002 ,IFDF
92 SHFTRI: WORD MULDIV
93 001 ,ENDC 073002 ;IASHC R2,R0
94 002 ,IFNDF ,EAE&MULDIV
95 000232' 022702 177770 SHFTRI: CMP #=8,,R2 ;CHECK # OF BITS TO SHIFT
96 000236' 003431 BLE SHFTR0 ;JUMP IF NOT MORE THAN 1/2 WORD
97 000240' 005004 CLR R4 ;SET UP EXTENSION BITS
98 000242' 005700 TST R0 ;BASED ON HIGH ORDER FRACTION
99 000244' 100001 BPL NCMP ;JUMP IF *
100 000246' 005104 COM R4 ; OTHERWISE
101 000250' 022702 177760 NCOMP: CMP #=16,,R2
102 000254' 002405 BLT SHFTRL ;JUMP IF LESS THAN ONE WORD TO SHIFT
103 000256' 010001 MOV R0,R1 ;SHIFT RIGHT A WHOLE WORD
104 000260' 010400 MOV R4,R0 ;USE EXTENSION BITS
105 000262' 062702 000020 ADD #16,,R2 ;ACCOUNT FOR SHIFT
106 000266' 001421 BEQ SHFTD
107 000270' 022702 177770 SHFTRL: CMP #=8,,R2
108 000274' 003412 BLE SHFTR0 ;JUMP IF NOT MORE THAN 1/2 WORD

SADR MACY11.620 22-AUG-72 11:41 PAGE 3
ADR04.PAL

109 000276' 062702 000020 ADD #16,,R2 !SHIFT LEFT 16-X
110 000302' 006301 SHFTL1 ASL R1
111 000304' 006100 ROL R0
112 000306' 006104 ROL R4
113 000310' 005302 DEC R2 !COUNT LOOP
114 000312' 003373 BGT SHFTL
115 000314' 010001 MOV R0,R1 !PUT RESULT IN R0, R1
116 000316' 010400 MOV R4,R0
117 000320' 000404 BR SHFTD
118 000322' 006200 SHFTR0: ASR R0 !SHIFT A MIN AND B MIN
119 000324' 006001 ROR R1
120 000326' 005202 INC R2 !REDUCE EXPONENT DIFFERENCE
121 000330' 002774 BLT SHFTR0
122 001 ENDC
123 000332' 066600 000004 SHFTD1 ADD A1(SP),R0 !A1+A2
124 000336' 066601 000006 ADD B1(SP),R1 !B1+B2
125 000342' 005500 ADC R0
126 000344' 126616 000001 CMPB SIGNS+1(SP),@SP
127 000350' 001034 BNE SUB !GO CLEAN UP SUBTRACT
128 000352' 030027 001000 BIT R0,#1000
129 000356' 001403 BEQ NORMD !JUMP IF NO NORMAL BIT OVERFLOW
130 000360' 006200 ASR R0
131 000362' 006001 ROR R1
132 000364' 005203 INC R3 !INCREASE EXPONENT
133 000366' 000303 SWAB R3 !MOVE EXPONENT LEFT
134 000370' 001020 BNE OVER !JUMP IF OVERFLOW
135 000372' 150003 BISB R0,R3
136 000374' 006016 ROR @SP !INSERT SIGN
137 000376' 006003 ROR R3
138 000400' 006001 ROR R1 !ROUND SUM
139 000402' 005501 ADC R1
140 000404' 005503 ADC R3
141 000406' 102411 BVS OVER !JUMP IF OVERFLOW ON ROUND
142 000410' 103410 BCS OVER
143 000412' 010366 000010 STOREI MOV R3,A2(SP) !STORE EXPONENT AND SIGN
144 000416' 010166 000012 MOV R1,B2(SP) !INSERT LOW ORDER FRACTION
145 000422' 005726 OUT: TST (SP)+ !POP SIGNS
146 000424' 012604 MOV (SP)+,R4
147 000426' 022626 CMP (SP)+,(SP)+ !POP FIRST ARGUMENT
148 000430' 000134 JMP @R4+ !DONE, RETURN
149
150 000432' 004567 000000G OVER: JSR R5,SERR !ERROR 3,2
151 000436' 000771 BR OUT
152 000440' 003 ,BYTE 3
153 000441' 002 ,BYTE 2
154
155 000442' 005700 SUB: TST R0 !CHECK HIGH ORDER RESULT FRACTION
156 000444' 003005 BGT BIT9 !IF POSITIVE SIGN IS OK
157 000446' 001413 BEQ ZTEST !CHECK FOR ZERO RESULT
158 000450' 005400 NEG R0 !GET ABSOLUTE VALUE
159 000452' 005401 NEG R1
160 000454' 005600 SBC R0
161 000456' 000316 SWAB @SP !EXCHANGE SIGNS
162 000460'

163 002 ,IFDF EAE
164 BIT R0,#700
165 BNE BIT9A ;JUMP IF NOT MORE THAN 2 TO SHIFT
166 MOV R1,@#MQ ;RESULT FRACTION TO AC,MQ
167 MOV R0,@#AC
168 CLR @#NOR ;NORMALIZE
169 SUB @#NOR,R3 ;ADJUST EXPONENT
170 MOV #=6,@ASH ;SHIFT TO CORRECT POSITION
171 ADD #6,R3 ;COMPENSATE EXPONENT
172 BLE UNDERF ;JUMP IF UNDERFLOW
173 MOV @#AC,R0
174 MOV @#MQ,R1 ;GET FRACTION BACK
175 BR NORMD
176 001 ,ENDC
177 000460' 030027 000400 BIT9A1 BIT RD,#400
178 000464' 001014 BNE UTEST ;JUMP IF NORMAL BIT FOUND
179 000466' 005303 DEC R3 ;DECREASE EXPONENT
180 000470' 006301 ASL R1 ;DOUBLE FRACTION
181 000472' 006100 ROL R0
182 000474' 000771 BR BIT9A ;TRY AGAIN
183 000476' 005701 ZTESTI TST R1 ;CHECK LOW ORDER PART
184 002 ,IFDF EAE
185 BNE BIT9
186 BR ZERO
187 001 ,ENDC
188 002 ,IFNDF EAE
189 000500' 001415 BEQ ZERO
190 000502' 000301 SWAB R1 ;SAVE NORMALIZE SOME TIME
191 000504' 150100 BISB R1,R0 ;MOVE BITS LEFT
192 000506' 105001 CLRB R1
193 000510' 162703 000010 SUB #8,,R3 ;TELL EXPONENT ABOUT IT
194 000514' 000761 BR BIT9
195 001 ,ENDC
196 000516' 005703 UTESTI TST R3 ;CHECK FOR UNDERFLOW
197 000520' 003322 BGT NORMD ;JUMP IF NONE
198 000522' 004567 000000G UNDERF: JSR R5,SERR ;ERROR 5,2
199 000526' 000401 BR UNDER
200 000530' 005 ,BYTE 5
201 000531' 002 ,BYTE 2
202 000532' 005001 UNDERI CLR R1 ;UNDERFLOW, TREAT AS 0
203 000534' 005003 ZERO: CLR R3 ;CLEAR EXPONENT
204 000536' 000725 BR STORE
205 000 000 ,ENDC
206 000001 ,END

SADR MACY11.620 22-AUG-72 11:41 PAGE 5

ADR04.PAL SYMBOL TABLE

AC	= 177302	ASH	= 177316	A1	= 000004	A2	= 000010
A2NZ	= 000102R	BIT9	= 000460R	BIT9A	= 000460R	B1	= 000006
B2	= 000012	ECHK	= 000206R	EXPA	= 000140R	F0	=X000000
MQ	= 177304	NCOMP	= 000250R	NQR	= 177312	NORMD	= 000366R
OUT	= 000422R	OVER	= 000432R	PC	=X000007	R0	=X000000
R1	=%000001	R2	=%000002	R3	=%000003	R4	=%000004
R5	=%000005	SCHK	= 000172R	SHFT	= 000212R	SHFTD	= 000332R
SHFTL	= 000302R	SHFTR	= 000232R	SHFTRL	= 000270R	SHFTRO	= 000322R
SIGNS	= 000000	SP	=%000006	STORE	= 000412R	SUB	= 000442R
UNDER	= 000532R	UNDERF	= 000522R	UTEST	= 000516R	ZERO	= 000534R
ZTEST	= 000476R	SADR	= 000004RG	SERR	= ***** G	SSBR	= 000000RG

000540

ERRORS DETECTED: 0

```

1      000000'          ,TITLE SMLR05
2      000000'          ,CSECT
3      000000'          ,GLOBL SMLR,SERRA
4      000000'          SMLR   THE REAL MULTIPLY ROUTINE
5
6
7      SMLR   V005A
8
9      COPYRIGHT 1971, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
10
11
12      CALLED IN POLISH MODE.
13      REPLACES THE TOP TWO REALS ON THE STACK
14      WITH THEIR PRODUCT.
15      R0=X0
16      R1=X1
17      R2=X2
18      R3=X3
19      R4=X4
20      R5=X5
21      SP=X6
22      PC=X7
23      MQ=177304
24      SR=177311
25      LSH=177314
26      F0=X0
27      A#8,
28      B#12,
29      RESLT#8,
30      SIGN=2
31      SMLR:    .IFDF  FPU
32              ,WORD  170001  ||SETF
33              ,WORD  172426  ||LDFF  (SP)+,F0  |GET MULTIPLICAND
34              ,WORD  171026  ||MULF  (SP)+,F0  |MULTIPLY
35              ,WORD  174046  ||STF   F0,=(SP)  |PRODUCT TO STACK
36              JMP   0(R4)+ 
37              ENDC
38      SMLR:    ,IFNDF FPU
39              MOV   R4,=(SP)
40              MOV   R5,=(SP)
41              ,IFNDF EAE&MULDIV
42              MOV   A+0-4(SP),R2
43              ASL   R2    |SHIFT MULTIPLICAND
44              ROL   =(SP)  |KEEP SIGN
45              CLR   =(SP)  |CLEAR EXPONENT
46              SWAB  R2
47              MOVB R2,0SP  |KEEP MULTIPLICAND EXPONENT
48              BEQ   ZERO1 |JUMP IF ANSWER IS ZERO
49              SEC   R2    |INSERT NORMAL BIT
50              ROR   R2
51              CLRB  R2
52              BISB  A+3(SP),R2
53              CLR   R3
54              BISB  A+2(SP),R3
55              SWAB  R3

```

55 000046' 006366 000014 ASL B(SP) ;SHIFT HIGH MULTIPLIER
56 000052' 005566 000002 ADC SIGN(SP) ;GET PRODUCT SIGN
57 000056' 105766 000015 TSTB B+1(SP)
58 000062' 001467 BEQ ZERO1 ;JUMP IF ZERO
59 000064' 006066 000014 ROR B(SP) ;SIGN IS NOW ZERO
60 000070' 005000 CLR R0 ;CLEAR PRODUCT
61 000072' 005001 CLR R1
62 000074' 016604 000016 MOV B+2(SP),R4 ;GET LOW ORDER MULTIPLIER
63 000100' 001406 BEQ B2Z
64 000102' 012705 000017 B2NZ: MOV #15,,R5
65 000106' 004767 000220 JSR PC,MULT0
66 000112' 004767 000160 JSR PC,MULT1 DO LAST LOW BIT FULL PRECISION
67 000116' 016604 000014 B2Z: MOV B(SP),R4 ;GET HIGH ORDER BITS
68 000122' 012705 000007 MOV #7,R5 ;THERE ARE ONLY SEVEN OF THEM
69 000126' 004767 000144 JSR PC,MULT
70 000132' 004767 000144 JSR PC,MULT1 ;GO DO THE NORMAL BIT
71 000136' 062604 ADD (SP)+,R4 ;ADD EXPONENTS
72 001 ENDC
73 002 | EAE CODE
74 | ,IFDF EAE
75 | (A1+A2*2**=16)*(B1+B2*2**=16)
76 | MOV #MQ,R4 ;POINT TO MQ
77 | MOV #100000,R5 ;GET LEADING BIT
78 | MOV B+2-4(SP),@R4 ;LOW ORDER B TO MQ
79 | MOV B+0-4(SP),@(R4) ;HIGH TO AC
80 | BEQ ZERO1 ;JUMP IF 0
81 | INC @#LSH ;GET SIGN
82 | RORB @#SR
83 | ROL -(SP) ;SAVE IT
84 | MOV (R4)+,-(SP) ;SAVE EXPONENT
85 | CLR B ;RIGHT JUSTIFY IT
86 | SWAB @#SP
87 | MOV #7,@#LSH ;MOVE FRACTION LEFT
88 | MOV @R4,-(SP) ;SAVE B2
89 | BIS R5,-(R4) ;INSERT NORMAL BIT
90 | MOV (R4)+,-(SP) ;SAVE B1
91 | MOV A+2-4(SP),@R4 ;LOW ORDER A TO MQ
92 | MOV A+0-4(SP),@(R4) ;HIGH TO AC
93 | BEQ ZERO2 ;JUMP IF 0
94 | INC @#LSH ;GET SIGN
95 | RORB @#SR
96 | ADC 6(SP) ;GET RESULT SIGN
97 | MOV @R4,R3 ;GET EXPONENT
98 | CLR B ;RIGHT JUSTIFY IT
99 | SWAB R3
100 | ADD R3,4(SP) ;GET SUM OF EXPONENTS
101 | MOV #7,@#LSH ;LEFT JUSTIFY FRACTION
102 | MOV (R4)+,R2 ;SAVE A1
103 | BIS R5,R2 ;INSERT NORMAL BIT
104 | CLR R0 ;CLEAR PRODUCT
105 | CLR R1
106 | MOV (R4)+,R3 ;SAVE A2
107 | BNE A2NZ
108 | TST -(R4) ;POINT TO MQ

MLR05, PAL

```

109          BR      A2Z    ;SHORT CUT
110          MOV     @SP, @R4  ;GET B1*A2
111          CMP     = (R4), -(R4) ;POINT TO AC
112          ADD     R3, @R4  ;A2, 2/S COMP CORRECTION
113          TST     R3
114          BPL     A2P
115          ADD     @SP, @R4  ;B1, CORRECTION
116          MOV     (R4)+, R1  ;HIGH PRODUCT TO R1
117          A2Z     2(SP), (R4)+  ;B2 TO MQ
118          BNE     B2NZ
119          TST     = (R4) ;POINT TO MQ
120          BR      B2Z    ;SHORT CUT
121          MOV     R2, @R4  ;GET B2*A1
122          CMP     = (R4), -(R4) ;POINT TO AC
123          ADD     2(SP), @R4  ;B2, CORRECTION
124          TST     2(SP)
125          BPL     B2P    ;JUMP IF B2 +
126          ADD     R2, @R4  ;A1, CORRECTION
127          B2P     ADD     (R4)+, R1  ;HIGH PRODUCT TO R1
128          ADC     R0
129          B2Z     MOV     R2, (R4)+  ;A1 TO MQ
130          ADD     R2, R0
131          MOV     @SP, @R4  ;GET A1*B1
132          ADD     (SP)+, R0
133          ADD     = (R4), R1
134          ADC     R0
135          ADD     = (R4), R0  ;AC+R0
136          TST     (SP)+  ;POP B2
137          MOV     (SP)+, R4  ;GET SUM OF EXPONENTS
138          ENDC
139          001
140          002
141          MUL/DIV CODE
142          :IFDF  MULDIV
143          (A1+A2*2**16)*(B1+B2*2**16)
144          MOV     B**2+4(SP), R5  ;LOW ORDER B
145          MOV     B**0+4(SP), R4  ;HIGH ORDER
146          BEQ     ZERO
147          ,WORD   073427,1  ; ASHC #1,R4  ;GET SIGN BIT
148          ROL     -(SP)  ;SAVE IT
149          MOV     R4, -(SP)  ;SAVE EXPONENT
150          CLRB   @SP
151          SWAB   @SP  ;RIGHT JUSTIFY
152          ,WORD   073427,7  ; ASHC #7,R4  ;LEFT JUSTIFY FRACTION
153          MOV     R5, -(SP)  ;SAVE B2
154          BJS     #1000000, R4  ;INSERT NORMAL BIT
155          MOV     R4, -(SP)  ;SAVE B1
156          MOV     A**2+4(SP), R3  ;GET A2
157          MOV     A**0+4(SP), R2  ;GET A1
158          BEQ     ZERO2  ;JUMP IF RESULT TO BE 0
159          ,WORD   073227,1  ; ASHC #1,R2  ;GET SIGN
160          ADC     6(SP)  ;GET RESULT SIGN
161          MOV     R2, R0  ;GET EXPONENT
162          CLRB   R0
163          SWAB   R0
164          ADD     R0, 4(SP)  ;GET SUM OF EXPONENTS

```

163 ,WORD 073227,7 ; ASHC #7,R2 IGET A1
164 BIS #100000,R2 ;INSERT NORMAL BIT
165 CLR R0 ;CLEAR ACCUMULATOR
166 CLR R1
167 TST R3 ;CHECK A2
168 BEQ A2Z ;IJUMP IF 0
169 ,WORD 070403 ; MUL R3,R4 IGET A2*B1
170 ADD R3,R4
171 TST R3
172 BPL A2P ;IJUMP IF A2 +
173 ADD @SP,R4 ;B1 CORRECTION
174 MOV R4,R1 ;A2*B1*2**16
175 A2P: MOV 2(SP),R4 ;B2 TO MULTIPLIER
176 BEQ B2Z ;IJUMP IF 0
177 ,WORD 070402 ; MUL R2,R4 IGET A1*B2
178 ADD 2(SP),R4
179 TST 2(SP)
180 BPL B2P ;IJUMP IF B2 +
181 ADD R2,R4 ;A1 CORRECTION
182 B2P: ADD R4,R1 ;A1*B2*2**16
183 ADC R0
184 MOV R2,R4 ;A1 TO MULTIPLIER
185 ADD R2,R0
186 ,WORD 070416 ; MUL @SP,R4 IGET A1*B1
187 ADD (SP)+,R0
188 ADD R5,R1 ;LOW ORDER A1*B1
189 ADC R0
190 ADD R4,R0 ;HIGH ORDER A1*B1
191 TST (SP)+
192 MOV (SP)+,R4 ;IPOP B2
193 ENDC ;IGET SUM OF EXPONENTS
194 000140' 006101
195 000142' 006100
196 000144' 103403
197 000146' 006101
198 000150' 006100
199 000152' 005304
200 000154' 162704 000200
201 000160' 003436
202 000162' 022704 000377
203 000166' 002427
204 000170' 105001
205 000172' 150001
206 000174' 000301
207 000176' 105000
208 000200' 150400
209 000202' 000300
210 000204' 006026
211 000206' 006000
212 000210' 006001
213 000212' 005501
214 000214' 005500
215 000216' 103414
216 000220' 102413
NORM: ROL R1 ;SHIFT OUT NORMAL BIT
ROL R0
BCS NORM ;IJUMP IF IT WAS FOUND
ROL R1
ROL R0 ;I MUST HAVE GOT IT NOW
DEC R4 ;ADJUST EXPONENT
SUB #200,R4 ;TAKE OUT ONE OF THE EXCESS 128'S
BLE UNDER ;IJUMP IF UNDERFLOW
CMP #377,R4
BLT OVER ;IJUMP IF OVERFLOW
CLRB R1
BISB R0,R1
SWAB R1
CLRB R0
BJSB R4,R0
SWAB R0
ROR (SP)+ ;IGET PRODUCT SIGN
ROR R0 ;INSERT IT IN RESULT
ROR R1
ADC R1
ADC R0
BCS OVER1 ;IJUMP IF OVERFLOW ON ROUND
BVS OVER1

217 000222' 010066 000010 OUT: MOV R0,RESLT(SP) ;PUT OUT ANSWER
218 000226' 010166 000012 MOV R1,RESLT+2(SP)
219 000232' 012605 MOV (SP)+,R5
220 000234' 012604 MOV (SP)+,R4
221 000236' 022626 CMP (SP)+,(SP)+ ;FLUSH TOP ARGUMENT
222 000240' 000134 JMP @R4+ ;RETURN
223 002 ZERO21 ,IFDF EAE!MULDIV
224 001 ZERO21 CMP (SP)+,(SP)+ ;POP B1,B2
225 000242' 022626 ,ENDC
226 000244' 000411 ZERO11 CMP (SP)+,(SP)+ ;POP SIGN AND EXPONENT
227 000246' 005726 BR ZERO
228 000250' 012700 006003 OVER1 TST (SP)+ ;FLUSH SIGN
229 000254' 000403 OVER1 MOV #6003,R0 ;ERROR 3,12
230 000256' 012700 003405 UNDER1 MOV #3405,R0 ;ERROR 5,7
231 000262' 005726 TST (SP)+ ;FLUSH SIGN
232 000264' 004567 000000G ECALLI JSR R5,SERRA ;CALL ERROR
233 000270' 005000 ZERO1 CLR R0 ;CLEAR RESULT
234 000272' 005001 CLR R1
235 000274' 000752 BR OUT
236 000276' 0002
237 000276' 006204 :IFNDF EAE&MULDIV
238 000300' 103004 MULT: ASR R4 ;TEST NEXT MULTIPLIER BIT
239 000302' 060301 BCC X0 ;IJUMP IF IT IS 0
240 000304' 005500 MULT11 ADD R3,R1
241 000306' 103406 ADC R0
242 000310' 060200 BCS COVER
243 000312' 006000 ADD R2,R0
244 000314' 006001 X0: ROR R0 ;NOW SHIFT PRODUCT
245 000316' 005305 ROR R1
246 000318' 003366 DEC R5 ;COUNT LOOP
247 000320' 003366 BGT MULT ;AGAIN PLEASE
248 000322' 000207 RTS PC ;RETURN TO CALLER
249 000324' 060200 COVER1 ADD R2,R0 ;FIRST ADD OVERFLOWED R0
250 000326' 000261 SEC X0 ;SHOW THIS OVERFLOW TO SHIFT
251 000330' 000770
252 000332' 006204 MULT01 ASR R4 ;REDUCED PRECISION MULTIPLY
253 000334' 103001 BCC X00
254 000336' 060200 ADD R2,R0 ;USE ONLY HIGH ORDER MULTIPLICAND
255 000340' 006000 X00: ROR R0
256 000342' 006001 ROR R1
257 000344' 005305 DEC R5
258 000346' 003371 BGT MULT0
259 000350' 000207 RTS PC
260 001 ,ENDC
261 000 ,ENDC
262 000001 ,END

SMLR05 MACY11,620 22-AUG-72 11:41 PAGE 6
MLR05.PAL SYMBOL TABLE

A	= 000010	B	= 000014	B2NZ	= 000102R	B2E	= 000116R
COVER	= 000324R	ECALL	= 000264R	F0	= %000000	LSH	= 177314
MQ	= 177304	MULT	= 000276R	MULT0	= 000332R	MULT1	= 000302R
NORM	= 000154R	OUT	= 000222R	OVER	= 000246R	OVER1	= 000250R
PC	= %000007	RESLT	= 000017	R0	= %000000	R1	= %000001
R2	= %000002	R3	= %000003	R4	= %000004	R5	= %000005
SIGN	= 000002	SP	= %000006	SR	= 177311	UNDER	= 000256R
X0	= 000312R	X00	= 000340R	ZERO	= 000270R	ZERO1	= 000242R
SERRA	= **** G	SMLR	= 000000RG		= 000352R		

000352

ERRORS DETECTED: 0

\$DVR05 MACY11.620 22-AUG-72 11:41 PAGE 1
DVR05.PAL

```

1      SDVR05
2      000000'
3      ,CSECT
4      ,GLOBL SDVR,SERRA
5      SDVR --- THE REAL DIVIDE ROUTINE
6
7      SDVR    V005A
8
9      COPYRIGHT 1971, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
10
11     CALLED IN THE POLISH MODE
12     THE NUMERATOR IS THE SECOND ITEM ON THE STACK
13     AND THE DENOMINATOR IS ON TOP
14     TAKES THE QUOTIENT AND PUTS IT ON TOP
15     OF THE STACK IN THEIR PLACE
16     R0=X0
17     R1=X1
18     R2=X2
19     R3=X3
20     R4=X4
21     R5=X5
22     SP=X6
23     PC=X7
24     MQ=177304
25     NOR=177312
26     LSH=177314
27     ASH=177316
28     F0=X0
29     Fi=X1
30     Ds,
31     Ne12,
32     Os12,
33     001
34     SDVR:   ,IFDF  FPU
35             ,WORD  170001  ||SETF
36             ,WORD  172526  ||LDF   (SP)+,F1      IGET DIVISOR
37             ,WORD  172426  ||LDF   (SP)+,F0      IGET DIVIDEND
38             ,WORD  174401  ||DIVF  F1,F0  IDIVIDE
39             ,WORD  174046  ||STP   F0,-(SP)   IQUOTIENT TO STCK
40             JMP   @R4+
41
42     000
43     001
44     SDVR:   ,ENDC  FPU
45             ,IFNDF  FPU
46             MOV   R4,-(SP)
47             MOV   R5,-(SP)
48             CLR   R0
49             CLR   R1
50             CLR   -(SP)
51             ASL   N+0=2(SP)    ;SHIFT NUMERATOR
52             ROL   @SP    IGET NUMERATOR SIGN
53             CLR   -(SP)
54             BISB  N+1(SP),@SP  IGET NUMERATOR EXPONENT
55             BEQ   ZERO  IJUMP IF NUMERATOR IS ZERO
56             BISB  N(SP),R0
57             SWAB  R0    ILEFT JUSTIFY NUMERATOR FRACTION
58             SEC   R0    IINSERT NORMAL BIT

```

55	000040'	006020	ROR	R0		
56	000042'	156600	002017	BISB	N+3(SP),R0	
57	000046'	156601	002016	BISB	N+2(SP),R1	
58	000052'	000301	SWAB	R1		
59	000054'	005002	CLR	R2		
60	000056'	005003	CLR	R3		
61	000060'	006366	ASL	D(SP) ISHIFT DENOMINATOR		
62	000064'	005566	ADC	2(SP) IGET RESULT SIGN		
63	000070'	156602	000011	BISB	D+1(SP),R2 IGET DIVISOR EXPONENT	
64	000074'	001431	BEQ	DCHK IJUMP IF DIVISOR IS ZERO		
65	000076'	160216	SUB	R2,ESP ISUBTRACT EXPONENTS		
66	000100'	005002	CLR	R2		
67	000102'	156602	000010	BISB	D(SP),R2 IGET HIGH ORDER FRACTION	
68	000106'	000302	SWAB	R2		
69	000110'	000261	SEC	IINSERT NORMAL BIT		
70	000112'	006002	ROR	R2		
71	000114'	156602	000013	BISB	D+3(SP),R2	
72	000120'	156603	000012	BISB	D+2(SP),R3	
73	000124'	000303	SWAB	R3		
74	000126'	020002	CMP	R0,R2 ICMPARE HIGH NUMERATOR AND DENOMINATOR		
75	000130'	103440	BLO	DHI IJUMP IF DENOMINATOR HIGH		
76		002	,IFNDF	EAE&MULDIV		
77	000132'	101034	BHI	DLOW IJUMP IF DENOMINATOR LOW		
78	000134'	020103	CMP	R1,R3 ICMPARE LOW ORDER PARTS		
79	000136'	101032	BHI	DLOW		
80	000140'	001034	BNE	DHI		
81	000142'	005066	000014	CLR	Q(SP) IQUOTIENT FRACTION IS 1	
82	000146'	005216	INC	ESP IBUMP EXPONENT		
83	000150'	005005	CLR	R5		
84	000152'	000445	BR	FLOAT		
85		001	,ENDC	EAE!MULDIV		
86		002	,IFDF	DLOW IJUMP IF DENOMINATOR LOW OR SAME		
87		001	,ENDC	ZERO:		
88	000154'	022626	CMP	(SP)+,(SP)+ IFLUSH EXP AND SIGN		
89	000156'	000415	BR	ECALL1		
90	000160'	005726	DCHK:	TST		
91	000162'	012700	004003	MOV	(SP)+ IFLUSH EXP	
92				#4003,R0 JERROR 3,8		
93	000166'	000406	BR	ECALL		
94	000170'	005746	OVER1:	TST		
95	000172'	012700	003003	MOV	#3003,R0 JERROR 3,6	
96	000176'	000402	BR	ECALL		
97	000200'	012700	002405	UNDER1	MOV	
98	000204'	005726	ECALL1	TST	#2405,R0 JERROR 5,3	
99	000206'	004567	000000G	JSR	(SP)+ IFLUSH SIGN	
100	000212'	005066	000010	ECALL1:	CLR	R5,SERRA
101	000216'	005066	000012	CLR	Q#0-4(SP) JRETURN 0	
102	000222'	000445	BR	Q#2-4(SP)		
103	000224'	006000	DLOW1:	ROR	RTN	
104	000226'	006001	ROR	R0 IHALVE NUMERATOR (C=0)		
105	000230'	005216	INC	R1 ITO ENSURE THAT N<D		
106		002	,IFNDF	ESP ICOMPENSATE EXPONENT		
107	000232'	012704	000011	EAE&MULDIV		
108	00036'	004767	000104	MOV	#9,,R4 JGO DO FIRST 9 QUOTIENT BITS	
			JSR	PC,DIV1		

```

109 000242' 110566 000014      MOVB   R5,Q(SP)      ;SAVE ALL HIGH ORDER Q FRACTION
110                                         ;EXCEPT NORMAL BIT
111 000246' 005704      TST    R4      ;SEE IF DONE
112 000250' 001402      BEQ    NOT0   ;NO, NUMERATOR NOT 0
113 000252' 005005      CLR    R5      ;ALL THE REST OF THE QUOTIENT IS ZERO
114 000254' 000404      BR     FLOAT
115 000256' 012704 000020      NOT0: MOV    #16,,R4 ;GO DO 16 MORE BITS
116 000262' 004767 000060      JSR    PC,DIV1
117          001      ,ENDC
118          002      ,IFDF EAE;MULDIV
119          DHI: CLC
120          ROR   R0      ;ENSURE NUM. AND DENOM. +
121          ROR   R1
122          ROR   R2      ;LOW ORDER R1 AND R3 ARE 0
123          ROR   R3
124          ROR   R0
125          ROR   R1
126
127          001      ,ENDC
128          002      ,IFDF EAE
129          MOV    #MQ,R5 ;POINT TO MQ
130          MOV    R1,@R5 ;NUMERATOR TO AC,MQ
131          MOV    R0,-(R5)
132          MOV    R2,-(R5)      ;(A+S*B)/C
133          TST    (R5)+ ;POINT TO AC
134          MOV    (R5)+,R1      ;KEEP REMAINDER
135          MOV    (R5)+,R4      ;KEEP QUOTIENT
136          MOV    R3,@R5      ;GET Q*D
137          TST    -(R5)      ;POINT TO MQ
138          ASR    R1      ;SCALE R
139          SUB    R1,-(R5)      ;Q*D=R
140          DEC    @#ASH
141          MOV    R2,-(R5)      ;(Q*D=R)/C
142          CMP    (R5)+,(R5)+      ;MQ
143          NEG    @R5
144          MOV    #2,@#ASH      ;MULT BY 4
145          ADD    R4,-(R5)      ;Q+(Q*D=R)*S/C
146          CLR    @#NOR      ;NORMALIZE
147          SUB    @#NOR,@SP      ;APPLY TO EXPONENT
148          MOV    #-6,@#LSH      ;POSITION NORMAL BIT
149          MOV    (R5)+,Q(SP)      ;STORE QUOTIENT
150          MOV    @R5,R5
151          001      ,ENDC
152          002      ,IFDF MULDIV
153          MOV    R0,R4      ;INUMERATOR TO DIVIDEND
154          MOV    R1,R5
155          ,WORD  071402  ;DIV.    R2,R4      ;(A+S*B)/C
156          MOV    R5,R1      ;SAVE REMAINDER
157          MOV    R4,R0      ;SAVE QUOTIENT
158          ,WORD  070403  ;MUL     R3,R4      ;GET Q*D
159          ASR    R1      ;SCALE R
160          SUB    R1,R4      ;Q*D=R
161          ,WORD  073427,-1 ;ASHC   #=1,R4      ;SCALE
162          ,WORD  071402  ;DIV     R2,R4      ;GET (Q*D=R)/C

```

163 NEG R4 ;(R=Q*D)/C
 164 ,WORD 073427,-14, ; ASHC #14,,R4 IUNSCALE
 165 ADD R0,R4 ;Q+(R=Q*D)*S/C
 166 NBTSTI ,WORD 073427,1 ; ASHC #1,R4 ISHIFT
 167 BMI NBIT ;CHECK FOR NORMAL BIT
 168 DEC @SP ;COMPENSATE EXPONENT
 169 BR NBTST ;GO AGAIN
 170 NBIT: ,WORD 073427,-7 ;ASHC #8,R4 ALIGN FRACTION
 171 MOV R4,Q(SP) ;STORE HIGH ORDER
 172 ,ENDC
 173 000266' 012604 001
 174 000270' 062704 000200
 175 000274' 003741
 176 000276' 022704 000377
 177 000302' 002733
 178 000304' 110466 000013
 179 000310' 006026
 180 000312' 006066 000010
 181 000316' 006005
 182 000320' 005505
 183 000322' 005566 000010
 184 000326' 010566 000012
 185 000332' 103716
 186 000334' 102715
 187 000336' 012605
 188 000340' 012604
 189 000342' 022626
 190 000344' 000134
 191 002
 192 000346' 006305
 193 000350' 006301
 194 000352' 006100
 195 000354' 103406
 196 000356' 020200
 197 000360' 101010
 198 000362' 103403
 199 000364' 020301
 200 000366' 101005
 201 000370' 001407
 202 000372' 160301
 203 000374' 005600
 204 000376' 160200
 205 000400' 005205
 206 000402' 005304
 207 000404' 003360
 208 000406' 000207
 209 000410' 005205
 210 000412' 000401
 211 000414' 006305
 212 000416' 005304
 213 000420' 003375
 214 000422' 005204
 215 000424' 000207
 216 001

NBTSTI ,WORD 073427,1 ; ASHC #1,R4 ISHIFT
 BMI NBIT ;CHECK FOR NORMAL BIT
 DEC @SP ;COMPENSATE EXPONENT
 BR NBTST ;GO AGAIN
 NBIT: ,WORD 073427,-7 ;ASHC #8,R4 ALIGN FRACTION
 MOV R4,Q(SP) ;STORE HIGH ORDER
 ,ENDC
 FLOATI MOV (SP)+,R4 ;PUSH UP EXPONENT
 ADD #200,R4 ;ADD IN EXCESS 200
 BLE UNDER ;UNDERFLOW
 CMP #377,R4
 BLT OVER ;OVERFLOW
 MOVB R4,Q+1=2(SP) ;INSERT EXPONENT IN RESULT
 SIGN: ROR (SP)+ ;INSERT QUOTIENT SIGN
 ROR Q+0=4(SP)
 ROR R5
 ADC R5 ;ROUND
 ADC Q+0=4(SP)
 MOV R5,Q+2=4(SP) ;INSERT LOW ORDER FRACTION
 BCS OVER1
 BYS OVER1
 RTN: MOV (SP)+,R5
 MOV (SP)+,R4
 CMP (SP)+,(SP)+ ;IFLUSH FIRST ARGUMENT
 JMP @R4)+
 ,IFNDIV EAE&MULDIV
 DIV1: ASL R5 ;ISHIFT QUOTIENT
 ASL R1 ;ISHIFT NUMERATOR
 ROL R0
 BCS GO ;IGUARANTEED TO GO
 CMP R2,R0 ;ICOMPARE HIGH DIVISOR AND DIVIDEND
 BHI NOGO ;IJUMP IF DIVISOR BIGGER
 BLO GO ;IJUMP IF DIVISOR SMALLER
 CMP R3,R1 ;ICHECK THE LOW ORDERS
 BHI NOGO ;IJUMP IF NUMERATOR =DENOMINATOR
 BEQ NEQD ;IJUMP IF NUMERATOR =DENOMINATOR
 GO: SUB R3,R1 ;INEN=D
 SBC R0
 SUB R2,R0
 INC R5 ;INSERT QUOTIENT BIT
 DEC R4 ;COUNT LOOP
 BGT DIV1
 RTS PC
 NEQD: INC R5 ;INSERT LAST 1 BIT IN QUOTIENT
 BR EQ1
 EQ2: ASL R5 ;IFINISH OUT QUOTIENT WITH 0'S
 EQ1: DEC R4
 BGT EQ2
 INC R4 ;IFLAG NO MORE NUMERATOR
 RTS PC ;IRETURN TO CALLER
 ,ENDC

SDVR05 ACY11.620 22-AUG-72 11:41 PAGE 5

DVR05.PAL

217 000
218 000001

,ENDC
,END

\$DVR05 MACY11,620 22-AUG-72 11:41 PAGE 6
DVR05.PAL SYMBOL TABLE

ASH	= 177316	D	= 000010	DCHK	000160R	DH!	000232R
DIV1	000346R	DLOW	000224R	ECALL	000204R	ECALL1	000212R
EQ1	000416R	EQ2	000414R	FLOAT	000266R	F0	=%000000
F1	=%000001	GO	000372R	LSH	= 177314	MQ	= 177304
N	= 000014	NEQD	000410R	NOGO	000402R	NOR	= 177312
NOT0	000256R	OVER	000172R	OVER1	000170R	PC	=%000007
Q	= 000014	RTN	000336R	RTS	000424R	R0	=%000000
R1	=%000001	R2	=%000002	R3	=%000003	R4	=%000004
R5	=%000005	SIGN	000310R	SP	=%000006	UNDER	000200R
ZERO	000154R	\$DVR	000000RG	SERRA	= ***** G		= 000426R

000426

ERRORS DETECTED: 0

\$DVR05.PAL ACY11.620 22-AUG-72 11:41 PAGE 5

DVR05.PAL

217 020
218 000001

,ENDC
,END

SDVR05 MACY11,620 22-AUG-72 11:41 PAGE 6

DVR05.PAL SYMBOL TABLE

ASH	= 177316	D	= 000010	DCHK	000160R	DH!	000232R
DIV1	000346R	DLOW	000224R	E CALL	000204R	E CALL1	000212R
EQ1	000416R	EQ2	000414R	FLOAT	000266R	F0	=%000000
F1	=%000001	GO	000372R	LSH	= 177314	HQ	= 177304
N	= 200014	NEQD	000410R	NOGO	000402R	NOR	= 177312
NOT0	000256R	OVER	000172R	OVER1	000170R	PC	=%000007
Q	= 000014	RTN	000336R	RTS	000424R	R0	=%000000
R1	=%000001	R2	=%000002	R3	=%000003	R4	=%000004
R5	=%000005	SIGN	000310R	SP	=%000006	UNDER	000200R
ZERO	000154R	SDVR	000000RG	SERRA	= ***** G	:	= 000426R

000426

ERRORS DETECTED: 0

LNKX11 V021 22-AUG-72 11:41

#DBKEBA/T:17440,DBKEBA+DBKEBA,ADR04,MLR05,DVR05./E

LOAD MAP

TRANSFER ADDRESS: 002001

LOW LIMIT: 015700

HIGH LIMIT: 017440

MODULE MAINDE

SECTION ENTRY	ADDRESS	SIZE
<, ABS.>	000000	000000
\$ERR	013414	
\$ERRA	013420	
< >	015700	000000

MODULE SADR

SECTION ENTRY	ADDRESS	SIZE
< >	015700	000540
\$ADR	015704	
\$SBR	015700	

MODULE SMLR05

SECTION ENTRY	ADDRESS	SIZE
< >	016440	000352
\$MLR	016440	

MODULE SDVR05

SECTION ENTRY	ADDRESS	SIZE
< >	017012	000426
\$DVR	017012	

RUN-TIME! 2 SECONDS

2K CORE USED

