

- 1. IDENTIFICATION
- 1.1 Digital-8-10-S
- 1.2 CALCULATOR
- 1.3 June 28, 1965



2. ABSTRACT

The Calculator program has been written to demonstrate the arithmetic capabilities of the PDP-8, as well as to serve as a useful computational tool. Input is in a form similar to the FORTRAN language. The Calculator consists of a compiler section and an operating section and uses the PDP-8 Floating-Point System (Digital-8-5D-S).

3. REQUIREMENTS

3.1 Storage

Calculator occupies memory from 5-3374 (octal) and from 4557-7577 (Digital-8-5D-S).

3.2 Subprograms and/or Subroutines

Digital-8-5D-S is used and is supplied with the binary tape.

3.3 Equipment

4K PDP-8, 33ASR Teletype.

4. USAGE

4.1 Loading

Calculator is loaded via the Binary Loader (Digital-8-2-U) with the Floating-Point Package (Digital-8-5D-S) in memory. The binary tape supplied consists of two parts: the Floating-Point Package and the Calculator program.

4.4 Startup and/or Entry

After the program is loaded, set 0200 in the switch register, depress LOAD ADDRESS, then START. Calculator will type a carriage return-line feed combination and wait for a command.

4.5 Errors in Usage

There are three error messages:

4.5.1 SYNTAX?

Calculator will type this when it is unable to recognize the commands it has been given.

4.5.2 IO

Input overflow. More than 400(8) valid characters have been typed without giving a go command. Calculator will restart and ignore all previous input.

4.5.3 STACK ERROR

The operating system has been unable to execute the code generated by the compiler section. This will be caused by illegal input that the compiler was unable to diagnose. Calculator will restart (see 8.1.1).

4.6 Recovery

Calculator restarts after all errors.

5. RESTRICTIONS (Not Applicable)

6. DESCRIPTION

6.1 Discussion

The compiler will reduce the input commands to a "reverse polish" form.

For example:

$1 + 6/4 = ;$

Will compile as:

LOAD	STACK	(1)
LOAD	STACK	(6)
LOAD	STACK	(4)
DIVIDE		
ADD		
OUTPUT		
STOP		

When compilation is complete, the operating system is entered, and the compiled instructions are executed.

7. METHODS (Not applicable)

8. FORMAT

8.1 Input Data

8.1.1 Arithmetic Expressions

Each arithmetic operation must be explicitly indicated by the keyboard character representing the operation. These characters are called operators.

Extraneous spaces, tabs, carriage returns, and line-feeds are ignored by the program. The character semicolon (;) is used to terminate input and to start compilation and execution.

An arithmetic expression is normally evaluated from left to right; however, certain operations are always performed before others, regardless of their order in the expression. The operators and their priority of evaluation within expressions are listed below:

- | | |
|-----------------------------------|------|
| 1. Expressions within parentheses | () |
| 2. Exponentiation | ↑ |
| 3. Multiplication, division | *, / |
| 4. Unary minus | - |
| 5. Addition, subtraction | +, - |
| 6. Output | = |

For example:

- | | | |
|----|--------------------------|----------|
| a. | $4 + 6 = ;$ | produces |
| | $+ 0.1000000E + 02$ | or 10 |
| b. | $4 + 2*3 = ;$ | produces |
| | $+ 0.1000000E + 02$ | or 10 |
| c. | $(4 + 2)*3 = ;$ | produces |
| | $+ 0.1800000E + 02$ | or 18 |
| d. | $(4 + 2=) * 3 = ;$ | produces |
| | $+ 0.6000000E + 01$ | or 6 |
| | $+ 0.1800000E + 02$ | or 18 |
| e. | $6 \uparrow 2 = ;$ | produces |
| | $+ 0.3600000E + 02$ | or 36 |
| f. | $9 \uparrow \cdot 5 = ;$ | produces |
| | $+ 0.3000000E + 01$ | or 3 |
| g. | $(4 + (3*2=)) * 2 = ;$ | produces |
| | $+ 0.6000000E + 01$ | or 6 |
| | $+ 0.1000000E + 02$ | or 10 |
| | $+ 0.2000000E + 02$ | or 20 |

The following functional commands may be incorporated in expressions to be evaluated.

- | | |
|--------|--|
| ABS() | Take the absolute value of the expression within the parentheses. |
| SQT() | Take the square root of the absolute value of the expression within the parentheses. |
| SIN() | Take the sine of the value of the expression within the parentheses (considered to be in radians). |
| COS() | Take the cosine of the value of the expression within the parentheses (considered to be in radians). |
| ATN() | Take the arc-tangent of the value of the expression within the parentheses (answer in radians). |
| EXP() | Take the exponential (base e) of the value within the parentheses. |

LOG() Take the natural logarithm of the expression within the parentheses.

These functions have a priority that is between exponentiation (\uparrow) and multiplication and division (*, /).

For example:

ATN((SIN(.1) =)/(COS(.1) =) =) = ; produces

+ 0.9983341E + 01	SIN(.1)
+ 0.9950040E + 00	COS(.1)
+ 0.1003347E + 00	SIN(.1)/COS(.1) = TANGENT(.1)
+ 0.9999999E + 00	ATN(TAN(.1))

(SIN(.1)) \uparrow 2 + (COS(.1)) \uparrow 2 = ; produces

+ 0.1000000E + 01

The RUBOUT key causes the previous character that was typed (as input) to be erased. The character erased is then retyped by the input part of the program.

For example:

(A $\begin{matrix} \text{RUBOUT} \\ \text{RUBOUT} \end{matrix}$ A (6 = ; produces

+ 0.6000000E + 01

In the above case, the second A and the second (were typed by the input program after it processed the rubouts.

If RUBOUT is used to erase more characters than were typed, the input program will type STACK ERROR, and Calculator will restart itself.

If the compiler detects a source language error, it will type SYNTAX? and restart itself. For example:

1*/6 = ; SYNTAX?

8.1.2 Loop Controlling

Calculator has two loop-controlling commands.

8.1.2.1 Repeat

The repeat command is indicated by R followed by an integer. It will cause Calculator to evaluate the expression from beginning to end a specified number of times.

4 \div 2 = R2; produces

+ 0.6000000E + 01
+ 0.6000000E + 01

8.1.2.2 Modification

The modification command is specified as follows:

expression 1 [operator expression 2]

Expression 1 is modified once on each pass through the loop. For example:

0 [+ 1] = R3; produces
+ 0.1000000E + 01
+ 0.2000000E + 01
+ 0.3000000E + 01

For example: To produce a table of the first ten integers and their square roots, Calculator would be instructed:

SQT(0 [+ 1] =) = R10; and it would respond with:

```
SQT(0[+1]=)=R10;  
+0.1000000E+01  
+0.1000000E+01  
  
+0.2000000E+01  
+0.1414213E+01  
  
+0.3000000E+01  
+0.1732050E+01  
  
+0.4000000E+01  
+0.2000000E+01  
  
+0.5000000E+01  
+0.2236067E+01  
  
+0.6000000E+01  
+0.2449489E+01  
  
+0.7000000E+01  
+0.2645751E+01  
  
+0.8000000E+01  
+0.2828426E+01  
  
+0.8999999E+01  
+0.3000000E+01  
  
+0.1000000E+02  
+0.3162277E+01
```

8.2 Output Format

Calculator's normal output mode is floating-point decimal (E format):

$\pm 0.XXXXXXXXXE \pm XX$

There is a command to change the output format:

FOR(X, Y)

where X and Y are positive integers less than or equal to 31. X is equal to the total number of digits to be outputted and Y is equal to the number of digits to the right of the decimal point. On output, leading 0's are suppressed. If the number is larger than the field width shows, X's will be typed. E format is specified by FOR(E). The current output format is maintained until explicitly changed.

The previous example could be rewritten as:

SQRT(0[+1]FOR(6, 4)=)FOR(9, 7)=R10; which produces:

```
+ 1.0000
+ 1.0000000

+ 2.0000
+ 1.4142130

+ 3.0000
+ 1.7320500

+ 4.0000
+ 2.0000000

+ 5.0000
+ 2.2360670

+ 6.0000
+ 2.4494890

+ 7.0000
+ 2.6457510

+ 8.0000
+ 2.8284260

+ 8.9999
+ 3.0000000

+10.0000
+ 3.1622770
```

9. EXECUTION TIME (Not applicable)

10. PROGRAM

10.4 Program Listing

```

*5
IN=JMS I .           /DEFINITIONS
0005 7400             7400
OUT=JMS I .
0006 7200             7200
EIM=JMS I .
0007 5600             5600
EXIT=1400
GETSGN=TAD 45
GETSWT=TAD 60

*20
0020 7776 M2,        -2
0021 7775 M3,        -3
0022 7774 M4,        -4
0023 7770 M7,       -10
0024 0002 P2,         2
0025 0003 P3,         3
0026 0000 COUNTR,    0
0027 0000 COUNT1,   0
0030 0000 STKVAL,    0
0031 1523 SCON1,     PUSH1
0032 2125 SCON2,     PUSH2
0033 2247 SCON3,     PUSH3
0034 0000 AD1,        0
0035 0000 AD2,        0
0036 0000 POINT,     0
0037 0000 TEMP,      0

*63
PUSH=JMS .
0063 0000             0           /PUSH DOWN ROUTINE
0064 3037             DCA TEMP    /C(CALL+1)=ADDRESS OF POINTER
0065 1463             TAD I .-2   /C(POINTER+1)=COUNT
0066 2063             ISZ .-3
0067 3034             DCA AD1
0070 2434             ISZ I AD1
0071 1434             TAD I AD1
0072 3035             DCA AD2
0073 2034             ISZ AD1
0074 1434             TAD I AD1
0075 1145             TAD M60
0076 7700             SMA CLA

```



```

0077 5551      ERRORI
0100 2434      ISZ I ADI
0101 1037      TAD TEMP
0102 3435      DCA I AD2
0103 5463      EXIT PUSH

```

```

/PUSH ALGORITHM
/C(POINTER):=C(POINTER)+1
/C(C(POINTER)):=C(AC)
/C(POINTER+1):=C(POINTER+1)+1
/IF C(POINTER+1)>40, THEN OVERFLOW
POP=JMS .

```

```

0104 0000      0 /POP UP ROUTINE
0105 1504      TAD I .-1
0106 2104      ISZ .-2
0107 3034      DCA AD1
0110 1434      TAD I AD1
0111 3035      DCA AD2
0112 7240      CLA CMA
0113 1035      TAD AD2
0114 3434      DCA I AD1
0115 2034      ISZ AD1
0116 7240      CLA CMA
0117 1434      TAD I AD1
0120 3434      DCA I AD1
0121 1434      TAD I AD1
0122 7710      SPA CLA
0123 5551      ERRORI
0124 1435      TAD I AD2
0125 5504      EXIT POP

```

```

/POP ALGORITHM
/C(AC):=C(C(POINTER))
/C(POINTER):=C(POINTER)-1
/C(POINTER+1):=C(POINTER+1)-1
/IF C(POINTER+1)<0, THEN UNDERFLOW

```

```

0126 0000      STACK1, 0 /STACK POINTER
0127 0000      0 /COUNT FOR OVERFLOW
0130 0000      STACK2, 0
0131 0000      0
0132 0000      STACK3, 0
0133 0000      0
0134 7700      M100, -100
0135 2773      ACON, INTAB /POINTER TO INPUT BUFFER
0136 0077      MASKR, 0077
0137 7700      MASKL, 7700
0140 0000      TEMP, 0
0141 0000      SAC1, 0
0142 0240      PA0, 0240

```

```

0143 0037 MASK5, 0037
0144 7773 M5, -5
0145 7660 M60, -120

0146 0531 CRLF=JMS I .
PCRLF
TEST=JMS I .
TSTCSE
0147 0600 ERROR=JMP I .
ERR
0150 0564 ERROR1=JMP I .
ERR1
0151 0563 POLISH=JMS I .
POL5
0152 0400 EXEC=JMP I .
EXCTE
0153 1000 INPUT=JMS I .
INGO
0154 0444 G01, GO
0155 0357 G02, POL1
0156 0275 G03, POL2
0157 0311 LEFT, CLEFT
0160 0432 RIGHT, CRIGHT
0161 1200 RGO, RCOMP
0162 1217 OCOUNT, 0
0163 0000 DECR=JMS . /DECREMENT POINTER
0164 0000 0
0165 7240 CLA CMA
0166 1036 TAD POINT
0167 3036 DCA POINT
0170 5564 EXIT DECR

```

/TRUTH TABLE STRUCTURE OF THE LANGUAGE

*200

```

0200 6032 BEGIN, KCC
0201 6046 TLS
0202 3062 DCA 62 /RESET FORMAT
0203 7040 CMA
0204 3163 DCA OCOUNT /RESET LOOP COUNT
0205 4546 CRLF /TYPE CR, LF
0206 1031 TAD SCON1 /RESET ALL STACK POINTERS
0207 3126 DCA STACK1
0210 1134 TAD M100
0211 3127 DCA STACK1+1
0212 1032 TAD SCON2
0213 3130 DCA STACK2
0214 3131 DCA STACK2+1
0215 1033 TAD SCON3
0216 3132 DCA STACK3

```

0217	3133		DCA STACK3+1	
0220	1135		TAD ACON	/SET INPUT POINTER
0221	3036		DCA POINT	
0222	4554		INPUT	/GET INPUT
0223	1135		TAD ACON	/RESET POINTER
0224	3036		DCA POINT	
0225	4063		PUSH	/PUT 0 ONTO STACK
0226	0130		STACK2	
0227	4547	START,	TEST	/TEST INPUT STRING
0230	5240		JMP NEG1	/IF -, IT IS NEGATIVE
0231	5550		ERROR	
0232	5323		JMP POL3	
0233	5550		ERROR	
0234	5243		JMP CONV1	
0235	2026		ISZ COUNTR	
0236	5550		ERROR	
0237	5243		JMP CONV1	
0240	2026	NEG1,	ISZ COUNTR	
0241	5227		JMP START	/+ IGNORE
0242	5273		JMP NEGATE	
0243	4405	CONV1,	IN	/CONVERT INPUT TO F.P.
0244	1060		GETSWT	
0245	7650		SNA CLA	/INPUT?
0246	5550		ERROR	/SOURCE LANGUAGE ERROR
0247	4407		EIM	/YES - PUSH INTO STACK
0250	6526		FPUT I STACK1	
0251	0000		FEXT	
0252	1025		TAD P3	
0253	1126		TAD STACK1	
0254	3126		DCA STACK1	
0255	2127		ISZ STACK1+1	/OVERFLOW?
0256	7410		SKP	/NO
0257	5550		ERROR	
0260	1366		TAD INCON	
0261	4063		PUSH	
0262	0132		STACK3	/PUT LOAD STACK ON OPERATE STAB
0263	4164		DECR	
0264	4547		TEST	/WHAT NEXT?
0265	5275		JMP POL1	/+,-
0266	5311		JMP POL2	/ /,*,+,=
0267	5550		ERROR	/ (, OR FNC
0270	5343		JMP POL4	/), OR ;
0271	5550		ERROR	
0272	5550		ERROR	
0273	1367	NEGATE,	TAD NEG	
0274	3030		DCA STKVAL	
0275	4552	POL1,	POLISH	/COMPILE THIS
0276	4547		TEST	/EXAMINE NEXT
0277	5307		JMP IN2	
0300	5550		ERROR	

0301	5323		JMP POL3	
0302	5550		ERROR	
0303	5243		JMP CONVRT	
0304	2026		ISZ COUNTR	
0305	5550		ERROR	
0306	5243		JMP CONVRT	
0307	4164	IN2,	DECR	
0310	5243		JMP CONVRT	
0311	4552	POL2,	POLISH	/COMPILE THIS
0312	1026		TAD COUNTR	
0313	3140		DCA TEM5	
0314	4547		TEST	
0315	5307		JMP IN2	/CONTINUE INPUT
0316	5550		ERROR	
0317	5323		JMP POL3	
0320	5341		JMP POL4T	
0321	5243		JMP CONVRT	
0322	5304		JMP IN2-3	
0323	1030	POL3,	TAD STKVAL	/IS IT (?
0324	7640		SZA CLA	
0325	5331		JMP .+4	
0326	4063		PUSH	/YES
0327	0130		STACK2	
0330	7410		SKP	
0331	4552		POLISH	/NO
0332	4547		TEST	
0333	5240		JMP NEG1	
0334	5550		ERROR	
0335	5323		JMP POL3	
0336	5550		ERROR	
0337	5243		JMP CONVRT	
0340	5304		JMP IN2-3	/INPUT?
0341	2140	POL4T,	ISZ TEM5	
0342	5550		ERROR	
0343	1030	POL4,	TAD STKVAL	/;?
0344	7001		IAC	
0345	7650		SNA CLA	
0346	5553		EXEC	/YES
0347	4104		POP	/UNSTACK TO (
0350	0130		STACK2	
0351	7450		SNA	
0352	5357		JMP GO	
0353	0136		AND MASKR	
0354	4063		PUSH	
0355	0132		STACK3	

0356	5347		JMP .-7	
0357	4547	GO,	TEST	
0360	5275		JMP POL1	
0361	5311		JMP POL2	
0362	5550		ERROR	
0363	5343		JMP POL4	
0364	5550		ERROR	
0365	5550		ERROR	
0366	0005	INCON,	0005	/CREATES LOAD-STACK
0367	0320	NEG, *400	0320	
0400	0000	POLS,	0	/COMPARE STACK PRIORITIES
0401	4104		POP	
0402	0130		STACK2	/IF THIS < STACK THEN UNSTACK
0403	3140		DCA TEM5	
0404	1140		TAD TEM5	
0405	0137		AND MASKL	
0406	3231		DCA TEM2	
0407	1030		TAD STKVAL	
0410	0137		AND MASKL	
0411	7161		CLL CML CMA IAC	
0412	1231		TAD TEM2	
0413	7630		SZL CLA	
0414	5222		JMP POLGO	
0415	1140		TAD TEM5	
0416	0136		AND MASKR	
0417	4063		PUSH	
0420	0132		STACK3	
0421	5201		JMP POLS+1	
0422	1140	POLGO,	TAD TEM5	/RESTORE STACKS
0423	4063		PUSH	
0424	0130		STACK2	
0425	1030		TAD STKVAL	
0426	4063		PUSH	
0427	0130		STACK2	
0430	5600		JMP I POLS	
0431	0000	TEM2,	0	
0432	4063	CLEFT,	PUSH	/HANDLE
0433	0130		STACK2	
0434	2036		ISZ POINT	
0435	4547		TEST	
0436	5556		JMP I GO2	

0437	5557		JMP I G03	
0440	5550		ERROR	
0441	5550		ERROR	
0442	5550		ERROR	
0443	5550		ERROR	
0444	0000	INGO,	0	/INPUT ROUTINE
0445	6031		KSF	
0446	5245		JMP .-1	
0447	6036		KRB	
0450	7450		SNA	/IGNORE BLANKS
0451	5245		JMP INGO+1	
0452	3140		DCA TEM5	
0453	1140		TAD TEM5	
0454	4337		TYPE	
0455	1140		TAD TEM5	/IGNORE PARITY BIT
0456	0352		AND BIT7	
0457	3140		DCA TEM5	
0460	1022		TAD M4	/TABLE FOR IGNORING
0461	3026		DCA COUNTR	
0462	1353		TAD TAB1	
0463	3010		DCA 10	
0464	1140		TAD TEM5	
0465	1410		TAD I 10	
0466	7450		SNA	/ONE OF THESE?
0467	5245		JMP INGO+1	/YES: FORGET IT
0470	2026		ISZ COUNTR	
0471	5265		JMP .-4	
0472	1410		TAD I 10	
0473	7650		SNA CLA	/RUBOUT?
0474	5320		JMP RUB	/YES
0475	1140		TAD TEM5	
0476	1362		TAD P200	
0477	3436		DCA I POINT	
0500	1436		TAD I POINT	
0501	1351		TAD MINN;	/;?
0502	7650		SNA CLA	
0503	5644		JMP I INGO	/YES: EXIT INPUT ROUTINE
0504	2036		ISZ POINT	
0505	1036		TAD POINT	
0506	1361		TAD TOHIGH	/OVERFLOW?
0507	7640		SZA CLA	
0510	5245		JMP INGO+1	/NO: CONTINUE
0511	1347		TAD CHI	/YES: TYPE "IO"
0512	4337		TYPE	
0513	1350		TAD CHO	
0514	4337		TYPE	
0515	4546		CRLF	
0516	5717		JMP I .+1	/START OVER AGAIN
0517	0200		BEGIN	
0520	4164	RUB,	DECR	/RUBOUT FOUND

0521	1036		TAD POINT	/DECREMENT POINTER
0522	7040		CMA	
0523	1135		TAD ACON	/UNDER FLOW?
0524	7650		SNA CLA	
0525	5551		ERRORI	/YES
0526	1436		TAD I POINT	/NO-TYPE ERASED
0527	4337		TYPE	/CHARACTER
0530	5245		JMP INGO+1	/CONTINUE
0531	0000	PCRLF,	Ø	/TYPE CR-LF
0532	1345		TAD CR	
0533	4337		TYPE	
0534	1346		TAD LF	
0535	4337		TYPE	
0536	5731		JMP I PCRLF	
		TYPE=JMS .		/TYPE SUBROUTINE
0537	0000		Ø	
0540	6041		TSF	
0541	5340		JMP .-1	
0542	6046		TLS	
0543	7200		CLA	
0544	5737		EXIT TYPE	
0545	0215	CR,	0215	
0546	0212	LF,	0212	
0547	0311	CHI,	311	
0550	0317	CHO,	317	
0551	7505	MINN;,	-273	
0552	0177	BIT7,	0177	
0553	0553	TAB1,	.	/IGNORE TABLE
0554	7740		-40	/SPACE
0555	0026		40-12	/LINE FEED
0556	7775		12-15	/CARRIAGE RETURN
0557	0004		15-11	/TAB
0560	7612		11-177	/RUBOUT
0561	4405	TOHIGH,	-400-INTAB	
0562	0200	P200,	0200	
0563	1375	ERR1,	TAD TCON2	
0564	1374	ERR,	TAD TCON1	
0565	3010		DCA 10	
0566	1410		TAD I 10	
0567	7450		SNA	
0570	5773		JMP I XGO	
0571	4337		TYPE	
0572	5366		JMP .-4	
0573	1150	XGO,	OPEND+3	
0574	1476	TCON1,	PTAB1-1	
0575	0010	TCON2,	PTAB2-PTAB1	

/PART II

```

/TYPE TEST ROUTINE
/TEST SYMBOLS OR OPERATORS
/RETURN TO          CALL+1  IF  +,-
/                   CALL+2  IF  /,* ,†,=
/                   CALL+3  IF  (, OR FNC
/                   CALL+4  IF  ), ;
/                   CALL+5  IF  DIGIT
/                   CALL+6  IF  . OR E

```

*600

```

0600  0000  TSTCSE,  0
0601  1366          TAD  SADTAB
0602  3010          DCA  10
0603  1020          TAD  M2
0604  4320          JMS  COMPAR
0605  5220          JMP  DCDE1
0606  2200          ISZ  TSTCSE
0607  7200          CLA
0610  1022          TAD  M4
0611  4320          JMS  COMPAR
0612  5223          JMP  DCDE2
0613  2200          ISZ  TSTCSE
0614  1410          TAD  I 10
0615  7640          SZA  CLA
0616  5226          JMP  TRYSYM
0617  5334          JMP  EXIT1+2
0620  1026  DCDE1,  TAD  COUNTR
0621  1343          TAD  TABL1
0622  5332          JMP  EXIT1
0623  1026  DCDE2,  TAD  COUNTR
0624  1346          TAD  TABL2
0625  5332          JMP  EXIT1

0626  1023  TRYSYM,  TAD  M7
0627  3026          DCA  COUNTR
0630  1367          TAD  FNTAB
0631  3011          DCA  11
0632  7240  TRYAGN,  CLA  CMA
0633  1036          TAD  POINT
0634  3012          DCA  12
0635  1021          TAD  M3
0636  3027          DCA  COUNT1
0637  1411          TAD  I 11
0640  1412          TAD  I 12
0641  7640          SZA  CLA
0642  5256          JMP  NOGO
0643  2027          ISZ  COUNT1

```


0644	5231		JMP .-5	
0645	1024		TAD P2	
0646	1036		TAD POINT	
0647	3036		DCA POINT	
0650	2026		ISZ COUNTR	
0651	7410		SKP	
0652	5770		JMP I FORMAT	
0653	1026		TAD COUNTR	
0654	1353		TAD TABL3	
0655	5332		JMP EXIT1	
0656	1027	NOGO,	TAD COUNT1	
0657	7040		CMA	
0660	1011		TAD I1	
0661	3011		DCA I1	
0662	2026		ISZ COUNTR	
0663	5232		JMP TRYAGN	
0664	2200		ISZ TSTCSE	
0665	1020		TAD M2	
0666	4320		JMS COMPAR	
0667	5337		JMP DCDE3	
0670	2200		ISZ TSTCSE	
0671	1410		TAD I 10	
0672	7500		SMA	
0673	5300		JMP .+5	
0674	1410		TAD I 10	
0675	7710		SPA CLA	
0676	5301		JMP .+3	
0677	5600		JMP I TSTCSE	
0700	2010		ISZ I0	
0701	2200		ISZ TSTCSE	
0702	7200		CLA	
0703	1021		TAD M3	
0704	4320		JMS COMPAR	
0705	5600		JMP I TSTCSE	
0706	1410		TAD I 10	
0707	7450		SNA	
0710	5560		JMP I LEFT	
0711	1410		TAD I 10	
0712	7450		SNA	
0713	5561		JMP I RIGHT	
0714	1410		TAD I 10	
0715	7640		SZA CLA	
0716	5550		ERROR	
0717	5562		JMP I RGO	
0720	0000	COMPAR,	Ø	/COMPARE SUBROUTINE
0721	3026		DCA COUNTR	
0722	1436		TAD I POINT	
0723	1410		TAD I 10	

0724	7450		SNA	
0725	5720		JMP I COMPAR	
0726	2026		ISZ COUNTR	
0727	5323		JMP .-4	
0730	2320		ISZ COMPAR	
0731	5720		JMP I COMPAR	
0732	3342	EXIT1,	DCA TEM4	/PUT PRIORITY
0733	1742		TAD I TEM4	/IN STACK VALUE
0734	3030		DCA STKVAL	
0735	2036		ISZ POINT	/UPDATE CHARACTER POINTER
0736	5600		JMP I TSTCSE	
0737	1026	DCDE3,	TAD COUNTR	
0740	1363		TAD TABL4	
0741	5332		JMP EXIT1	
0742	0000	TEM4,	0	
0743	0746	TABL1,	+.3	
0744	0401		0401	/+
0745	0402		0402	/-
0746	0753	TABL2,	+.5	
0747	0504		0504	/ /
0750	0503		0503	/ *
0751	0722		0722	/ †
0752	0106		0106	/ =
0753	0763	TABL3,	+.10	
0754	0611		0611	/ABS
0755	0612		0612	/SQT
0756	0613		0613	/SIN
0757	0614		0614	/COS
0760	0615		0615	/ATN
0761	0617		0617	/LOG
0762	0616		0616	/EXP
0763	0766	TABL4,	+.3	
0764	0100		0100	/)
0765	7777		7777	/ ;
0766	1425	SADTAB,	DCTAB-1	
0767	1446	FNTAB,	TABFN-1	
0770	1246	FORMAT,	FORMIT	
0771	0000	ABSF,	0	
0772	1045		TAD 45	
0773	7700		SMA CLA	
0774	5771		JMP I ABSF	
0775	4777		JMS I .+2	
0776	5771		JMP I ABSF	
0777	6000		6000	

		*6545		
6545	0771		ABSF	
		*6554		
6554	6000		6000	/SET UP NEGATE
		*1200		
		/EXECUTION		
1000	1032	EXCTE,	TAD SCON2	/POP UP REST OF
1001	7040		CMA	/STACK AND PUT
1002	1130		TAD STACK2	
1003	7650		SNA CLA	/ON OPERATE STACK
1004	5215		JMP OPGO	
1005	4104		POP	
1006	0130		STACK2	
1007	7450		SNA	
1010	5550		ERROR	
1011	0136		AND MASKR	
1012	4063		PUSH	
1013	0132		STACK3	
1014	5200		JMP EXCTE	
1015	7040	OPGO,	CMA	
1016	4063		PUSH	/PUT TERMINATOR ON
1017	0132		STACK3	/OPERATE STACK
1020	1033		TAD SCON3	
1021	3132		DCA STACK3	
1022	1344		TAD SCON4	
1023	3343		DCA STACK4	
1024	1031		TAD SCON1	
1025	3126		DCA STACK1	
1026	4545		CRLF	
1027	2132	OPGO1,	ISZ STACK3	
1030	1532		TAD I STACK3	
1031	7510		SPA	/TERMINATOR?
1032	5345		JMP OPEND	/YES
1033	3140		DCA TEM5	
1034	1140		TAD TEM5	
1035	1144		TAD M5	
1036	7510		SPA	
1037	5264		JMP OPR1	
1040	7450		SNA	
1041	5305		JMP LOAD	
1042	1020		TAD M2	
1043	7510		SPA	
1044	5321		JMP OUTPUT	
1045	7450		SNA	
1046	5756		JMP I FORM	
1047	1353		TAD M12	
1050	7500		SMA	

1051	5357		JMP EXP	
1052	1354		TAD P11	
1053	7450		SNA	
1054	5331		JMP STORE	
1055	3260		DCA OP2PT	
1056	4407		EIM	
1057	5743		FGET I STACK4	
1060	0000	OP2PT,	0	/SINGLE OPERAND
1061	6743		FPUT I STACK4	
1062	0000		FEXT	
1063	5227		JMP OPG01	
			/DOUBLE OPERAND COMMANDS	
1064	7200	OPR1,	CLA	
1065	1140		TAD TEM5	
1066	7112		CLL RTR	
1067	7012		RTR	
1070	1355		TAD CON	
1071	3277		DCA OPIPT	
1072	1343		TAD STACK4	
1073	1021		TAD M3	
1074	3342		DCA STACK	
1075	4407		EIM	
1076	5742		FGET I STACK	
1077	0000	OPIPT,	0	
1100	6742		FPUT I STACK	
1101	0000		FEXT	
1102	1342		TAD STACK	
1103	3343		DCA STACK4	
1104	5227		JMP OPG01	
			/LOAD STACK	
1105	7200	LOAD,	CLA	
1106	1025		TAD P3	
1107	1343		TAD STACK4	
1110	3343		DCA STACK4	
1111	4407		EIM	
1112	5526		FGET I STACK1	
1113	6743		FPUT I STACK4	
1114	0000		FEXT	
1115	1025		TAD P3	
1116	1126		TAD STACK1	
1117	3126		DCA STACK1	
1120	5227		JMP OPG01	
			/OUTPUT TOP OF STACK	
1121	7200	OUTPUT,	CLA	
1122	4407		EIM	
1123	5743		FGET I STACK4	
1124	0000		FEXT	

1125	1141		TAD SAC1
1126	4406		OUT
1127	4546		CRLF
1130	5227		JMP OPG01
1131	1341	STORE,	TAD M6
1132	1126		TAD STACK1
1133	3342		DCA STACK
1134	4407		EIM
1135	5743		FGET I STACK4
1136	6742		FPUT I STACK
1137	0000		FEXT
1140	5227		JMP OPG01
1141	7772	M6,	-6
1142	0000	STACK,	0
1143	0000	STACK4,	0
1144	2366	SCON4,	PUSH4-3
1145	7200	OPEND,	CLA
1146	2163		ISZ OCOUNT
1147	5220		JMP OPG0+3
1150	4546		CRLF
1151	5752		JMP I .+1
1152	0203		BEGIN+3
1153	7766	M12,	-12
1154	0011	P11,	11
1155	0743	CON,	AND I STACK4
1156	1400	FORM,	FORMOP
		/EXPONENTIATE	
1157	7200	EXP,	CLA
1160	1343		TAD STACK4
1161	1021		TAD M3
1162	3342		DCA STACK
1163	4407		EIM
1164	5742		FGET I STACK
1165	0007		0007
1166	3743		FMPY I STACK4
1167	0006		0006
1170	6742		FPUT I STACK
1171	0000		FEXT
1172	1342		TAD STACK
1173	3343		DCA STACK4
1174	5227		JMP OPG01
		*1200	
		/HANDLE]	
1200	1144	CRIGHT,	TAD M5
1201	4364		JMS SAVE
1202	4104		POP
1203	0130		STACK2

1204	7450	SNA
1205	5212	JMP .+5
1206	0136	AND MASKR
1207	4063	PUSH
1210	0132	STACK3
1211	5200	JMP CRIGHT
1212	1244	TAD STORE1
1213	4063	PUSH
1214	0132	STACK3
1215	2036	ISZ POINT
1216	5370	JMP EXIT3

/HANDLE RN

1217	1144	RCOMP,	TAD M5
1220	4364		JMS SAVE
1221	3060		DCA 60
1222	2036		ISZ POINT
1223	4547		TEST
1224	5550		ERROR
1225	5550		ERROR
1226	5550		ERROR
1227	5550		ERROR
1230	7410		SKP
1231	5550		ERROR
1232	4645		JMS I INDIG
1233	7200		CLA
1234	1060		GETSWT
1235	7650		SNA CLA
1236	5550		ERROR
1237	1046		TAD 46
1240	7041		CMA IAC
1241	3163		DCA OCOUNT
1242	4164		DECR
1243	5370		JMP EXIT3
1244	0010	STORE1,	0010
1245	7000	INDIG,	7000

/INPUT INTEGER

/HANDLE FOR(X,Y)

1246	1020	FORMIT,	TAD M2
1247	4364		JMS SAVE
1250	2036		ISZ POINT
1251	4547		TEST
1252	5550		ERROR
1253	5550		ERROR
1254	5260		JMP .+4
1255	5550		ERROR
1256	5550		ERROR
1257	5550		ERROR
1260	1030		TAD STKVAL
1261	7640		SZA CLA
1262	5550		ERROR

1263	4547		TEST	
1264	5550		ERROR	
1265	5550		ERROR	
1266	5550		ERROR	
1267	5550		ERROR	
1270	5277		JMP INIT	
1271	2026		ISZ COUNTR	
1272	7410		SKP	
1273	5550		ERROR	
1274	2026		ISZ COUNTR	
1275	5353		JMP FGO	
1276	5550		ERROR	
1277	3060	INIT,	DCA 60	
1300	4645		JMS I INDIG	/INPUT INTEGER
1301	7200		CLA	
1302	1060		GETSWT	
1303	7650		SNA CLA	
1304	5550		ERROR	
1305	1046		TAD 46	
1306	0143		AND MASK5	/5 BIT
1307	1142		TAD P40	
1310	4063		PUSH	
1311	0130		STACK2	
1312	4164		DECR	
1313	4547		TEST	
1314	5550		ERROR	
1315	5550		ERROR	
1316	5550		ERROR	
1317	5550		ERROR	
1320	5550		ERROR	
1321	2036		ISZ POINT	
1322	2026		ISZ COUNTR	
1323	7410		SKP	
1324	5550		ERROR	
1325	2026		ISZ COUNTR	
1326	5550		ERROR	
1327	4645		JMS I INDIG	/INPUT INTEGER
1330	7200		CLA	
1331	1046		TAD 46	
1332	0143		AND MASK5	
1333	1142		TAD P40	
1334	4063		PUSH	
1335	0130		STACK2	
1336	4164		DECR	
1337	1363	FEND,	TAD FCON	
1340	4063		PUSH	
1341	0130		STACK2	
1342	4547		TEST	
1343	5550		ERROR	
1344	5550		ERROR	

1423	3062		DCA 62	
1424	5625		JMP I .+1	
1425	1027		OPG01	
		/DECODING TABLE		
1426	7525	DCTAB,	-253	
1427	7776		53-55	
1430	7521		-257	
1431	0005		57-52	
1432	7714		52-136	
1433	0041		136-75	
1434	0025		75-50	
1435	7527		-251	
1436	7756		51-73	
1437	0001		73-72	
1440	0012		72-60	
1441	7473		-305	
1442	0031		105-54	
1443	7776		54-56	
1444	7723		56-133	
1445	7776		133-135	
1446	0013		135-122	
1447	7477	TABFN,	-301	/ABS
1450	7476		-302	
1451	7455		-323	
1452	7455		-323	/SIN
1453	7457		-321	
1454	7454		-324	
1455	7455		-323	
1456	7467		-311	
1457	7462		-316	
1460	7475		-303	
1461	7461		-317	
1462	7455		-323	
1463	7477		-301	
1464	7454		-324	
1465	7462		-316	
1466	7464		-314	
1467	7461		-317	
1470	7471		-307	
1471	7473		-305	
1472	7450		-330	
1473	7460		-320	
1474	7472		-306	
1475	7461		-317	
1476	7456		-322	
1477	0323	PIAB1,	323	/PRINT OUT TABLE
1500	0331		331	
1501	0316		316	
1502	0324		324	

1503	0301		301
1504	0330		330
1505	0277		277
1506	0000		000
1507	0323	PTAB2,	323
1510	0324		324
1511	0301		301
1512	0303		303
1513	0313		313
1514	0240		240
1515	0305		305
1516	0322		322
1517	0322		322
1520	0317		317
1521	0322		322
1522	0000		000
1523	0000	PUSH1,	0
		*.+401	
2125	0000	PUSH2,	0
		*.+121	
2247	0000	PUSH3,	0
		*.+121	
2371	0000	PUSH4,	0
		*.+401	
2773	0000	INTAB,	0
		*.+400	
		XXXXXX,	
		/INPUT SETUP	
		*7144	
7144	1436		TAD I POINT
7145	2036		ISZ POINT
7146	7000		NOP
		*7150	
7150	7000		NOP
7151	7000		NOP
ABSF	0771		
ACON	0135		
ADI	0034		

AD2	0035
BEGIN	0200
BIT7	0552
CHI	0547
CHO	0550
CLEFT	0432
COMPAR	0720
CON	1155
CONVRT	0243
COUNTR	0026
COUNT1	0027
CR	0545
CRIGHT	1200
CRLF	4546
DCDE1	0620
DCDE2	0623
DCDE3	0737
DCTAB	1426
DECR	4164
EIM	4407
ERR	0564
ERROR	5550
ERROR1	5551
ERR1	0563
EXCTE	1000
EXEC	5553
EXIT	1400
EXIT1	0732
EXIT3	1370
EXP	1157
FCON	1363
FEND	1337
FGO	1353
FNTAB	0767
FORM	1156
FORMAT	0770
FORMIT	1246
FORMOP	1400
GETSGN	1045
GETSWT	1060
GO	0357
G01	0155
G02	0156
G03	0157
IN	4405
INCON	0366
INDIG	1245
INGO	0444
INIT	1277
INPUT	4554

INTAB	2773
IN2	0307
LEFT	0160
LF	0546
LOAD	1105
MASKL	0137
MASKR	0136
MASK5	0143
MINN:	0551
M100	0134
M12	1153
M2	0020
M3	0021
M4	0022
M5	0144
M6	1141
M60	0145
M7	0023
NEG	0367
NEGATE	0273
NEGT	0240
NOGO	0656
OCOUNT	0163
OPEND	1145
OPGO	1015
OPG01	1027
OPR1	1064
OP1PT	1077
OP2PT	1060
OUT	4406
OUTPUT	1121
PCRLF	0531
POINT	0036
POLGO	0422
POLISH	4552
POLS	0400
POL1	0275
POL2	0311
POL3	0323
POL4	0343
POL4T	0341
POP	4104
PTAB1	1477
PTAB2	1507
PT1	1374
PUSH	4063
PUSH1	1523
PUSH2	2125
PUSH3	2247
PUSH4	2371
PI1	1154

F2	000
P200	0562
P3	0025
P40	0142
RCOMP	1217
RG0	0162
RIGHT	0161
RUR	0520
SAC1	0141
SADTAB	0766
SAVE	1364
SCON1	0031
SCON2	0032
SCON3	0033
SCON4	1144
SPC	1375
STACK	1142
STACK1	0126
STACK2	0130
STACK3	0132
STACK4	1143
START	0227
STKVAL	0030
STORE	1131
STORE1	1244
TABFN	1447
TABL1	0743
TABL2	0746
TABL3	0753
TABL4	0763
TAB1	0553
TCON1	0574
TCON2	0575
TEMP	0037
TEM2	0431
TEM4	0742
TEM5	0140
TEST	4547
TOHIGH	0561
TRYAGN	0632
TRYSYM	0626
TSTCSE	0600
TYPE	4337
XGO	0573
XXXXXX	3374

- 12. REFERENCES
- 12.1 Other Library Programs
See Digital-8-5-S.