

The Complete
SPECTRUM
ROM
DISASSEMBLY

BY

Dr Ian Logan & Dr Frank O'Hara

Transcribed by the following readers of
the *comp.sys.sinclair* newsgroup:-

J.R. Biesma
Biggo
Dr. J. Bland
Paul E .Collins
Chris Cowley
Dr. Rupert Goodwins
Jonathan G Harston
Marcus Lund
Joe Mackay
Russell Marks
Eduardo Yañez Parareda
Adam Stonehewer
Mark Street
Gerard Sweeney
Geoff Wearmouth
Matthew Westcott
Matthew Wilson
Witchy

SIN X

```

10 REM DEMONSTRATION FOR SIN X
20 REM USING THE 'SERIES GENERATOR'.
30 DIM A(6)
40 LET A(1)=-.0000000003
50 LET A(2)=0.000000592
60 LET A(3)=-.000068294
70 LET A(4)=0.004559008
80 LET A(5)=-.142630785
90 LET A(6)=1.276278962
100 PRINT
110 PRINT "ENTER START VALUE IN DEGREES"
120 INPUT C
130 CLS
140 LET C=C-10
150 PRINT "BASIC PROGRAM", "ROM PROGRAM"
160 PRINT "-----", "-----"
170 PRINT
180 FOR J=1 TO 4
190 LET C=C+10
200 LET Y=C/360-INT (C/360+.5)
210 LET W=4*Y
220 IF W > 1 THEN LET W=2-W
230 IF W < -1 THEN LET W=-W-2
240 LET Z=2*W*W-1
250 LET BREG=6
260 REM USE 'SERIES GENERATOR'
270 GO SUB 550
280 PRINT TAB 6; "SIN ";C;" DEGREES"
290 PRINT
300 PRINT T*W,SIN (PI*C/180)
310 PRINT
320 NEXT J
330 GO TO 100

```

NOTES:

- I. When C is entered this program calculates and prints SIN C degrees, SIN (C+10) degrees, SIN (C+20) degrees and SIN (C+30) degrees. It also prints the values obtained by using the ROM program. For a specimen of results, try entering these values in degrees: 0; 5; 100; -80; -260; 3600; -7200.
- II. The constants A(1) to A(6) in lines 40 to 90 are given (apart from a factor of 1/2) in Abramowitz and Stegun Handbook of Mathematical Functions (Dover 1965) page 76. They can be checked by integrating $(\sin(\pi^2 X^2))/X$ over the interval U=0 to π , after first multiplying by $\cos(N^*U)$ for each constant (i.e. N=1,2,...,6) and substituting $\cos U=2^*X^2-1$. Each result should then be divided by π . (This integration can be performed by approximate methods e.g. using Simpson's Rule if there is a reasonable computer or programmable calculator to hand.)

EXP X

```

10 REM DEMONSTRATION FOR EXP X
20 REM USING THE 'SERIES GENERATOR'
30 LET T=0 (This makes T the first variable.)
40 DIM A(8)
50 LET A(1)=0.000000001
60 LET A(2)=0.000000053
70 LET A(3)=0.000001851
80 LET A(4)=0.000053453
90 LET A(5)=0.001235714
100 LET A(6)=0.021446556
110 LET A(7)=0.248762434
120 LET A(8)=1.456999875
130 PRINT
140 PRINT "ENTER START VALUE"
150 INPUT C
160 CLS
170 LET C=C-10
180 PRINT "BASIC PROGRAM", "ROM PROGRAM"
190 PRINT "-----", "-----"
200 PRINT
210 FOR J=1 TO 4
220 LET C=C+10
230 LET D=C*1.442695041 (D=C*(1/LN 2);EXP C=2**D).
240 LET N=INT D
250 LET Z=D-N (2**(N+Z) is now required).
260 LET Z=2*Z-1
270 LET BREG=8
280 REM USE "SERIES GENERATOR"
290 GO SUB 550
300 LET V=PEEK 23627+256*PEEK 23628+1 (V=(VARS)+1)
310 LET N=N+PEEK V
320 IF N > 255 THEN STOP (STOP with arithmetic overflow).
330 IF N < 0 THEN GO TO 360
340 POKE V,N
350 GO TO 370
360 LET T=0
370 PRINT TAB 11;"EXP ";C
380 PRINT
390 PRINT T,EXP C
400 PRINT
410 NEXT J
420 GO TO 130

```

NOTES:

- I. When C is entered this program calculates and prints EXP C, EXP (C+10), EXP (C+20) and EXP (C+30). It also prints the values obtained by using the ROM program. For a specimen of results, try entering these values: 0; 15; 65 (with overflow at the end); -100; -40.
- II. The exponent is tested for overflow and for a zero result in lines 320 and 330. These tests are simpler in BASIC than in machine code, since the variable N, unlike the A register, is not confined to one byte.
- III. The constants A(1) to A(8) in lines 50 to 120 can be obtained by integrating $2^{**}X$ over the interval U=0 to PI, after first multiplying the COS (N*U) for each constant (i.e. for N=1,2,...,8) and substituting COS U = $2^{**}X-1$. Each result should then be divided by PI.

LN X:

```

10 REM DEMONSTRATION FOR LN X
20 REM USING THE 'SERIES GENERATOR'
30 LET D=0 (This makes D the first variable).
40 DIM A(12)
50 LET A(1)= -.0000000003
60 LET A(2)=0.0000000020
70 LET A(3)= -.0000000127
80 LET A(4)=-0.0000000823
90 LET A(5)= -.0000005389
100 LET A(6)=0.0000035828
110 LET A(7)= -.0000243013
120 LET A(8)=0.0001693953
130 LET A(9)= -.0012282837
140 LET A(10)=0.0094766116
150 LET A(11)= -.0818414567
160 LET A(12)=0.9302292213
170 PRINT
180 PRINT "ENTER START VALUE"
190 INPUT C
200 CLS
210 PRINT "BASIC PROGRAM", "ROM PROGRAM"
220 PRINT "-----", "-----"
230 PRINT
240 LET C=SQR C
250 FOR J=1 TO 4
260 LET C=C*C
270 IF C=0 THEN STOP (STOP with 'invalid argument'.)
280 LET D=C
290 LET V=PEEK 23627+256*PEEK 23628+1
300 LET N=PEEK V-128 (N holds e').
310 POKE V,128
320 IF D<=0.8 THEN GO TO 360 (D holds X').
330 LET S=D-1
340 LET Z=2.5*D-3
350 GO TO 390
360 LET N=N-1
370 LET S=2*D-1
380 LET Z=5*D-3
390 LET R=N*0.6931471806 (R holds N*LN 2).
400 LET BREG=12
410 REM USE 'SERIES GENERATOR'
420 GO SUB 550
430 PRINT TAB 8;"LN ";C
440 PRINT
450 PRINT S*T+R, LN C
460 PRINT
470 NEXT J
480 GO TO 170

```

NOTES:

- I. When C is entered this program calculates and prints LN C, LN (C**2), LN (C**4) and LN (C**8). It also prints the values obtained by using the ROM program. For a specimen of results, try entering these values: 1.1; 0.9; 300; 0.004; 1E5 (for overflow) and 1E-5 (STOP as 'invalid argument').
- II. The constants A(1) to A(12) in lines 50 to 160 can be obtained by integrating $5 \cdot \text{LN} (4 \cdot (X+1)/5) / (4 \cdot X - 1)$ over the interval U=0 to PI, after first multiplying by COS (N*U) for each constant (i.e. for N=1,2,...,12) and substituting COS U=2*X-1. Each result should then be divided by PI.

ATN X:

```

10 REM DEMONSTRATION FOR ATN X
20 REM USING THE 'SERIES GENERATOR'
30 DIM A(12)
40 LET A(1)= -.0000000002
50 LET A(2)=0.0000000010
60 LET A(3)= -.00000000066
70 LET A(4)=0.00000000432
80 LET A(5)= -.0000002850
90 LET A(6)=0.0000019105
100 LET A(7)= -.0000131076
110 LET A(8)=0.0000928715
120 LET A(9)= -.0006905975
130 LET A(10)=0.0055679210
140 LET A(11)= -.0529464623
150 LET A(12)=0.8813735870
160 PRINT
170 PRINT "ENTER START VALUE"
180 INPUT C
190 CLS
200 PRINT "BASIC PROGRAM", "ROM PROGRAM"
210 PRINT "-----", "-----"
220 PRINT
230 FOR J=1 TO 4
240 LET B=J*C
250 LET D=B
260 IF ABS B>=1 THEN LET D= -1/B
270 LET Z=2*D*D-1
280 LET BREG=12
290 REM USE 'SERIES GENERATOR'
300 GO SUB 550
310 LET T=D*T
320 IF B > =1 THEN LET T=T+PI/2
330 IF B < =-1 THEN LET T=T-PI/2
340 PRINT TAB 8;"ATN ";B
350 PRINT
360 PRINT T,ATN B (or PRINT T*180/PI,ATN B*180/PI
370 PRINT to obtain the answers in degrees)
380 NEXT J
390 GO TO 160

```

NOTES:

- I. When C is entered this program calculates and prints ATN C, ATN (C²), ATN (C³) and ATN (C⁴). For a specimen of results, try entering these values: 0.2; -1; 10 and -100. The results may be found more interesting if converted to yield degrees by multiplying the answers in line 360 by 180/PI.
- II. The constants A(1) to A(12) in lines 40 to 150 are given (apart from a factor of 1/2) in Abramowitz and Stegun Handbook of Mathematical Functions (Dover 1965) page 82. They can be checked by integrating ATN X/X over the interval U=0 to PI, after first multiplying by COS (N*U) for each parameter (i.e. for n=1,2,...,12) and substituting COS U=2*X*X-1. Each result should then be divided by PI.

An alternative subroutine for SIN X:

It is straightforward to produce the full expansion of the Chebyshev polynomials and this can be written in BASIC as follows:

```
550 LET T =(32*Z*Z*Z*Z*Z-40*Z*Z*Z+10*Z)*A(1)
      +(16*Z*Z*Z*Z-16*Z*Z+2)*A(2)
      +(8*Z*Z*Z-6*Z)*A(3)
      +(4*Z*Z-2)*A(4)
      +2*Z*A(5)
      +A(6)
560 RETURN
```

This subroutine is called instead of the SERIES GENERATOR and can be seen to be of a similar accuracy.

An alternative subroutine for EXP X:

The full expansion for EXP X is:

```
550 LET T =(128*Z*Z*Z*Z*Z*Z-224*Z*Z*Z*Z*Z+112*Z*Z*Z-14*Z)*A(1)
      +(64*Z*Z*Z*Z*Z*Z-96*Z*Z*Z*Z+36*Z*Z-2)*A(2)
      +(32*Z*Z*Z*Z*Z-40*Z*Z*Z+10*Z)*A(3)
      +(16*Z*Z*Z*Z-16*Z*Z+2)*A(4)
      +(8*Z*Z*Z-6*Z)*A(5)
      +(4*Z*Z-2)*A(6)
      +2*Z*A(7)
      +A(8)
560 RETURN
```

The expansion for LN X and A TN X, given algebraically, will be:

```
(2048z11-5632z9+5632z7-2464z5+440z3-22z) * A (1)
+
(1024z10-2560z8+2240z6-800z4+100z2-2) * A(2)
+
(512z9-1152z7+864z5-240z3+18z) * A(3)
+
(256z8-512z6+320z4-64z2+2) * A(4)
+
(128z7-224z5+112z3-14z) * A(5)
+
(64z6-96z4+36z2-2) * A(6)
+
(32z5-40z3+10z) * A(7)
+
(16z4-16z2+2) * A(8)
+
(8z3-6z) * A(9)
+
(4z2-2) * A(10)
+
(2z) * A(11)
+
A(12)
```

The following BASIC program illustrates the essential parts of the DRAW operation when being used to produce a straight line. The program in its present form only allows for lines where $X > Y$.

A complete algorithm is to found in the following program, as a subroutine that will 'DRAW A LINE' from the last position to X,Y.

The arcs are then drawn by repeated calls to the line drawing subroutine that on each call draws a single line from the 'last position' to the position 'X,Y'.

```

10 REM A CIRCLE PROGRAM
20 LET X=127: LET Y=87: LET Z=87
30 REM How many arcs?
40 LET Arcs=4*INT (INT (ABS (PI*SQR Z)+0.5)/4)+4
50 REM Set up memory area; M0-M5
60 LET M0=X+Z
70 LET M1=0
80 LET M2=2*Z*SIN (PI/Arcs)
90 LET M3=1-2*(SIN (PI/Arcs)) ^ 2
100 LET M4=SIN (2*PI/Arcs)
110 LET M5=2*PI
120 REM Set up stack; Sa-Sd
130 LET Sa=X+Z
140 LET Sb=Y-Z*SIN (PI/Arcs)
150 LET Sc=Sa
160 LET Sd=Sb
170 REM Initialise COORDS
180 POKE 23677,Sa: POKE 23678,Sb
190 LET M0=Sd
200 REM 'DRAW THE ARCS'
210 LET M0=M0+M2
220 LET Sc=Sc+M1
230 LET X=Sc-PEEK 23677
240 LET Y=M0-PEEK 23678

```



```

250 GO SUB 510
260 LET Arcs=Arcs-1: IF Arcs=0 THEN STOP
270 LET MM1=M1
280 LET M1=M1*M3-M2*M4
290 LET M2=MM1*M4+M2*M3
300 GO TO 210

500 REM 'DRAW A LINE' from last position to X,Y
510 LET PLOTx=PEEK 23677: LET PLOTy=PEEK 23678
520 LET dx=SGN X: LET dy=SGN Y
530 LET X=ABS X: LET Y=ABS Y
540 IF X>=Y THEN GO TO 580
550 LET L=X: LET B=Y
560 LET ddx=0: LET ddy=dy
570 GO TO 610
580 IF X+Y=0 THEN STOP
590 LET L=Y: LET B=X
600 LET ddx=dx: LET ddy=0
610 LET H=B
620 LET i=INT (B/2)
630 FOR N=B TO 1 STEP -1
640 LET i=i+L
650 IF i < H THEN GO TO 690
660 LET i=i-H
670 LET ix=dx: LET iy=dy
680 GO TO 700
690 LET ix=ddx: LET iy=ddy
700 LET PLOTy=PLOTy+iy
710 IF PLOTy <0 OR PLOTy > 175 THEN STOP
720 LET PLOTx=PLOTx+ix
730 IF PLOTx <0 OR PLOTx > 255 THEN STOP
740 PLOT PLOTx,PLOTy
750 NEXT N
760 RETURN

```

NOTE ON SMALL INTEGERS AND -65536.

1. Small integers n are those for which -65535 is less than or equal to n which is less than or equal to 65535. The form in which they are held is described in 'STACK-BC'. Note that the manual is inaccurate when it says that the third and fourth bytes hold n plus 131072 if n is negative. Since the range of n is then -1 to -65535, the two bytes can only hold n plus 131072 if it is taken mod 65536; i.e. they hold n plus 65536. The manual is fudging the issue. The fact is that this is not a true twos complement form (as the form n plus 131072, in other circumstances, could be). Here the same number can stand for two different numbers according to the sign byte: e.g. 00 01 stands for 1 if the sign byte is 00 and for -65535 if the sign byte is FF; similarly FF FF stands for 65535 if the sign byte is 00 and for -1 if the sign byte is FF.

2. Accepting that negative numbers are given a special 'twos complement' form, the main feature about this method of holding numbers is that they are ready for 'short addition' without any further twos complementing. They are simply fetched and stored direct by the addition subroutine. But for multiplication they need to be fetched by INT-FETCH and stored afterwards by INT-STORE. These subroutines twos complement the number when fetching or storing it. The calls to INT-STORE are from 'multiply' (after 'short multiplication'), from 'truncate' (after forming a 'small integer' between -65535 and 65535 inclusive), from 'negate'/abs' for the 'integer case' and from 'sgn' to store 1 or -1. The calls to INT-FETCH are from PRINT-FP to fetch the integer part of the number when it is 'small', from 'multiply' twice to fetch two 'small integers', from 'RE-STACK' to fetch a 'small integer' for re-stacking, from 'negate'/abs' to fetch a 'small integer' for manipulation and from FP-TO-BC to fetch the integer for transfer to BC.