STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM

REKENAFDELING

STANDAARDFUNCTIES
in
X8 Assembler-code ELAN

november 1965

[redacted]

In het navolgende wordt een samenvatting gegeven van enkele door de Rekenafdeling van het Mathematisch Centrum in ELAN opgestelde standaardprogramma's t.b.v. de EL-X8. Het betreft achtereenvolgens programma's voor de berekening van:

polynoom (POL)
entier (ENTIER)
integerdeling (IDI)
machtsverheffing (TTP)
worteltrekking (SQRT)
natuurlijke logarithme (LN)
exponentiële functie (EXP)
sinus, cosinus (SIN, COS)
arctangens (ARCTAN).

De programma's zijn nog niet zo grondig getest, dat ten volle voor de deugdelijkheid ervan kan worden ingestaan.

Amsterdam, 24 november 1965

F.J.M. Barning.

```
" standard-functions nr. 1

                    'BEGIN'  START, CYCLE

POL IN FF:          F ×    F
POL IN F:           M[B] = F
                 U, GOTO (:START)          " clear OF
START:              F =  MA                " take last coefficient
CYCLE:              F × M[B]               " × argument
                    F + MA[-2]             " + coefficient
                 U, GOTOR (MC[-1])         " if OF then exit
                    A -  2
                    GOTO (:CYCLE)

                    'END'  POL IN FF       " 9 instructions


                    'BEGIN'  HALF, SCHOLTEN

ENTIER:             F +   0, P
ENTIER1:         U, S =   F, E             " argument already integer ?
                 Y, GOTOR (MC[-1])
                    F - HALF, E            " +0 < F < .5 ?
                 Y, JUMP (4)               " then entier = + 0
                 U, S =   F, E             " F - .5 integer ?
                 Y, GOTOR (MC[-1])
                    F + SCHOLTEN
                    F - SCHOLTEN, Z
                 Y, F =   0
                    GOTOR (MC[-1])
HALF:               -     65535
                    +         1            " .5
SCHOLTEN:           +     12288
                    +         0            " 3 × d38 = 824 633 720 832

                    'END'  ENTIER          " 15 instructions


IDI:                F +   0
                    MC =  F, P
                 U, S = - F, E             " divisor not integer ?
                 N, F =   0
                 N, F + MC[-5], P
                 U, S = - F, E             " V dividend not integer ?
                 Y, SUBC (:ERRORTABLE[11])
                    F /   MC, P
                 N, F = - F                " take absolute value of quotient
                    SUBC (:ENTIER)
                 N, F = - F                " entier with correct sign
                    GOTOR (MC[+1])         " 12 instructions
```

```
" standard-functions nr.  2

TTP:              'BEGIN'  EVEN TEST, LN MUL EXP, END,
                           MUL, CYCLE, END1

                     F +   0, Z           " exponent = 0 ?
                  Y, F =   1              " then result = 1
                  Y, GOTO (:END)
                     MC =  F
                     F = M[B-5]           " take base
                     F +   0, Z           " = 0 ?
                  Y, S = M[B-1], P        " and exponent > 0 ?
                  Y, F =   0              " then result = 0
                  Y, GOTO (:END1)
                     S = M[B-1], P
                     A = - M[B-2], E      " exponent not integer ?
                  Y, GOTO (:LN MUL EXP)
                     A +   0, Z           " head of exponent = 0 ?
                  N, GOTO (:EVEN TEST)
                     S +   0, P           " exponent positive ?
                  N, S = - S              " take absolute value
                  U, S -   31, P          " abs (exponent) > 31 ?
                  N, GOTO (:MUL)
EVEN TEST:           S +   0, P
                     RCS (1), E           " exponent even ?
                  N, S =   G, P           " if odd, then base positive ?
LN MUL EXP:          SUBC (:LN)           " ln (abs (base))
                     F × MC[-2]           " × exponent
                     SUBC (:EXP1)         " and exponential of that
                  N, F = - F              " and inversed if necessary
END:                 B -   2
                     GOTOR (MC[1])
MUL:                 F =   1, P
                     M[B-2] = F           " start cycle with 1 en condition YES
CYCLE:               F = M[B-5]           " base ⋀ (2 ⋀ i)
                  N, F ×   F              " becomes (except for the first time)
                  N, M[B-5] = F           " base ⋀ (2 ⋀ (i+1))
                  U, S '×' 1, Z           " this power of base of interest ?
                  N, F × M[B-2]
                  N, M[B-2] = F           " then incorporate it in the result
                     RUS (1), Z           " ready ?
                  N, GOTO (:CYCLE)
                     A +   0, P           " was exponent originally negative ?
                  Y, F =   1
                  Y, F / M[B-2]           " then invert the result
END1:                B -   4
                     GOTOR (MC[1])        " 42 instructions

                  'END' TTP
```

```
" standard-functions nr. 3

                 'BEGIN'  D26, CO, C1, C2, HALF

SQRT:              F +   0, Z           " argument = 0 ?
                Y, F =   0              " then result = + 0
                Y, GOTOR (MC[-1])
                U, A =   1, E
                N, F = - F              " replace x by abs (x)
                   MC =  F              " preserve x
                   A =   F
                   RUA (15)             " isolate
                   MC =  A              " and preserve binary exponent
                   A =   F
                   S =   G
                   F = :MC              " preserve stack pointer
                   A 'x' 32767          " isolate mantissa in AS
                   NORA, Z              " take
                Y, A =   S
                Y, NORA                 " the 26 most significant bits
                Y, B +  54
                   B = - B              " of mantissa in A
                N, RUS  (B+26)
                N, A +   S              " and the corresponding
                N, B -  28
                   S =   B              " binary exponent in S
                   B = :MG              " restore stack pointer
                   S + MC[-1], P        " form complete binary exponent
                   RCS  (1), E          " is it even ?
                N, S + D26
                   MC =  S
                N, RUA (1)
                   MC =  A
                U, A -  C2, P           " estimate sqrt (a):
                   RUA (3)
                Y, A +  CO              " if a > .7124
                   MC =  A              " then x0 = (1 + a) / 2
                N, RUA (1)
                N, A +  C1
                N, M[B-1] + A           " else x0 = .3219 + (3/4) x a
                   A = M[B-2]
                   RUA (4)
                   DIVA (M[B-1])        " first Newton step:
                   M[B-1] + S           " x1 = (a/x0 + x0) / 2
                   A = MC[-2]
                   RUA  (2)
                   DIVA (MC)            " second Newton step:
                   S + M[B+1]           " x2 = (a/x1 + x1) / 2
                   A = MC[-1]           " take binary exponent
                   LUA (1)
                   A 'x' - 1
                   LUAS (14)            " transform x2 into floating point
                   F = MC[-2]           " restore x
                   F /   A              " third Newton step:
                   F +   A
                   F x HALF             " x3 = (F/x2 + x2) / 2
                   GOTOR (MC[-1])
```

```
" standard-functions nr. 4

" SQRT continued

D26:                    '400 000 000'
C0:                     +  83 88608         " .125
C1:                     +  54 00586         " .080475
C2:                     + 478 08355         " .7124
HALF:                   -      65535
                        +          1        " .5

                        'END'  SQRT         " 59 instructions
```

" standard-functions nr. 5

```
                 'BEGIN'  CYCLE, LIST, LN X, LN Y, C1, C3, C5,
                          LN2, ONE, EPSILON, BIN EXP 40

LN:                 F +   0, Z
                 Y, F = EPSILON, P          " replace argument 0 by 2 ∧ (-2047)
                    A =   1, E
                 N, F = - F                 " replace x by abs (x)
                    A =   F
                    RUA (15)                " isolate
                    MC =  A                 " and preserve binary exponent
                    A =   F
                    A 'x' 32767             " replace binary exponent
                    A + BIN EXP 40          " by 40
                    S =   G
                    F =   A
                    F +   O                 " standarize
                    A =   F                 " and isolate anew
                    RUA (15)                " binary exponent
                    M[B-1] + A              " add it to former binary exponent
                    A =   F                 " replace
                    A 'x' 32767             " binary exponent by 0
                    S =   G
                    F =   A                 " and consider 40 bits of F as fraction
                    LUAS (12)
                    S =   0                 " analyse first 27 bits of fraction
                    M[B] = A
CYCLE:              RUA (3)                 " in order to
                    PLUSA (M[B]), P
                 Y, S -   1                 " bring fraction
                 Y, GOTO (:CYCLE)
                    A = :LIST               " in range sqrt (8/9) < f < sqrt (9/8)
                    A -   S
                    G x  MA                 " multiply fraction by appropriate power of 9/8
                    MC =  F                 " and preserve resulting fraction
                    F + ONE
                    MC =  F                 " compute
                    F = ONE                 " f1 =
                    F - MC[-4]              " (1 - f) /
                    F /  MC                 " (1 + f)
                    MC =  F                 " preserve f1
                    F x   F
                    MC =  F
                    F x  C5                 " compute
                    F +  C3                 " polynomial
                    F x MC[-2]              " in
                    F +  C1                 " f1 square
                    F x MC[-2]              " result x f1
                    MC =  F
                    G =   S, Z
                 N, F x LN Y                " appropriate multiple of ln (8/9)
                    F + MC[-2]              " added
                    F + LN X                " and (1/2) x ln (9/8) added
```

```
" standard-functions nr. 6

" LN continued

                MC = - F
                G = MC[-3], Z           " take binary exponent
             N, F × LN2                 " × ln (2)
                F + MC[-1]
                GOTOR (MC[-1])
LIST:           +      32768            "          2↑15
                +      36864            " (9/8)   × 2↑15
                +      41472            " (9/8)↑2 × 2↑15
                +      46656            " (9/8)↑3 × 2↑15
                +      52488            " (9/8)↑4 × 2↑15
                +      59049            " (9/8)↑5 × 2↑15
LN X:           -   14 59121
                +   38 95640            " ln (9/8) / 2 = +.588915178282₁₀-1
LN Y:           +   14 26353
                -   38 95640            " ln (8/9) = -.117783035656
C1 :            -   12 69759
                +            6          " +.200000000002₁₀+1
C3:             -   13 32565
                +  445 32834            " +.666666478939
C5:             -   13 63134
                +  266 81432            " +.400433275889
LN2:            -   13 32131
                +  351 25202            " ln (2) = +.693147180560
ONE:            +    5 06966
                +  659 90648            " sqrt (8/9) × 2 ↑ 55 = +.339682755868₁₀+17
EPSILON:        -  671 08863
                +            1          " 2 ↑ (-2047) = +.618869209477₁₀-616
BIN EXP 40:     +   13 10720            " 40 × d15

                'END'  LN               " 77 instructions
```

" standard-functions nr. 7

```
                    'BEGIN'  ENTIRE, C0, C1, C2, C3, C4, C5, C6, C7,
                             LOG E, OMEGA, HALF

EXP:                    F +   0
EXP1:                   M[B] = F, P
                     N, F = - F                 " abs (x)
                        F - 1447, P             " > 1447 ?
                     N, F = M[B]
                     Y, F = 1447                " then replace
                     Y, S = - M[B+1], P         " x by 1447
                     Y, F = - 1447              " with correct sign
                        F × LOG E               " x log (e) => computation of 2/x
                        MC =  F, P
                        SUBC (:ENTIER1)
                        A =   G                 " integer part of x
                        F - MC[-2], Z           " fractional part of x = zero ?
                     Y, F =     1
                     Y, GOTO (:ENTIRE)
                        MC =  A
                        F = - F                 " compute log (fractional part)
                        F - HALF, P
                        F - HALF
                     N, F × HALF                " bring argument in range -1/2 < x < 0
                        A = :C7
                        SUBC (:POL IN F)        " and compute polynomial
                     N, F ×    F                " square, if argument was halved
                        A = MC[-1]              " integer part
                        A +    1                " corrected for subtraction of 2 × HALF
ENTIRE:              U, A + 2048, P
                     U, A - 2047, E             " outside range -2047, +2047 ?
                     N, JUMP (5)
                        A =    A, P             " > 2047 ?
                     N, F / OMEGA
                     N, GOTOR (MC[-1])
                        F × OMEGA
                        A - 2047
                        LUA (15)
                        A 'x' - 32767
                        S =    1
                        F ×    A
                        GOTOR (MC[-1])
C0:                     -  13 27104
                        -          0            " +.999999999999
C1:                     -  13 32131
                        + 351 25162             " +.693147180524
C2:                     -  13 93280
                        + 324 98472             " +.240226505508
C3:                     -  14 60009
                        +  42 24865             " +.555040853706₁₀-1
C4:                     -  15 30010
                        +  98 20595             " +.961794504001₁₀-2
C5:                     -  16 27221
                        + 234 73550             " +.133256313600₁₀-2
```

" standard-functions nr. 8

" EXP continued

```
C6:                 -  17 26494
                    + 299 64123       " +.152132608000⏨-3
C7:                 -  17 35842
                    + 348 22787       " +.128376319999⏨-4
LOG E:              -  12 98901
                    + 374 31644       " log (e) = +.144269504089⏨+1
OMEGA:              + 670 76096
                    +           1     " 2 ↑ 2047 = +.161585030357⏨+617
HALF:               -     65535
                    +           1     " 1/2

                    'END' EXP         " 60 instructions
```

" standard-functions nr. 9

```
                 'BEGIN'  JOINT, PICO, TWO OVER PI,
                          C0, C1, C2, C3, C4, C5, C6

COS:                S =    0                 " cos wanted
                    GOTO (:JOINT)
SIN:                MC =   F, P
                 N, F = - F                  " abs (x)
                    F - PICO, P              " > 2 ∧ (-20) ?
                    F = MC[-2]
                 N, GOTOR (MC[-1])           " otherwise sin (x) = x
                    S = - 0                  " sin wanted
                    A = - 2                  " and sin is an odd function
JOINT:              F × TWO OVER PI, P       " argument positive ?
                 N, F = - F                  " take absolute value of argument
                 N, S +    0, P              " cos wanted ?
                 Y, A =    0                 " cos is an even function
                    MC =   F, P
                    SUBC (:ENTIER1)
                    S +    0, P              " cos wanted ?
                    S =    G                 " take the quadrant number
                 Y, S +    1                 " cos (x) = sin (x + pi/2)
                 U, S 'x'  1, Z              " 2nd or 4th quadrant ?
                 N, F +    1                 " otherwise take complement
                    F - MC[-2]
                 Y, F = - F
                    S 'x'  2
                    A +    S, Z              " 3rd and 4th quadrant give extra - sign
                 Y, MC =   F
                 N, MC = - F                 " store reduced argument with sign of result
                    A = :C6
                    SUBC (:POL IN FF)        " compute polynomial in F square
                    F × MC[-2]
                    GOTOR (MC[-1])
PICO:               -     6 88127
                    +          1             " 2 ∧ (-20) = +.953674316406ₙ-6
TWO OVER PI:        -    13 33057
                    + 253 90670              " 2/pi = +.636619772367
C0:                 -    12 97852
                    -    24 48862            " +.157079632679ₙ+1
C1:                 +    13 32904
                    - 319 28613              " -.645964097498
C2:                 -    14 31346
                    + 316 68077              " +.796926261224ₙ-1
C3:                 +    15 63045
                    - 233 86613              " -.468175339112ₙ-2
C4:                 -    16 98552
                    + 296 17533              " +.160439054240ₙ-3
C5:                 +    17 03815
                    - 437 50969              " -.359570801978ₙ-5
C6:                 -    17 03934
                    + 558 98867              " +.546264060652ₙ-7

                    'END'  COS               " 48 instructions
```

" standard-functions nr. 10

```
                    'BEGIN'  TOGETHER, C1 , C2, C3, C4, C5, C6,
                             TG15, TG30, PI OVER 6

ARCTAN:                F +   0
                       MC =  G, P           " preserve original sign of x
                    N, F = - F              " and replace x by absx = abs (x)
                       MC =  F              " preserve absx
                       S =   3              " and start analysis:
                       F -   1, P           " absx > 1 ?
                    Y, F =   1
                    Y, F / MC[-2]
                    Y, MC =  F              " then replace absx by 1/absx
                    N, F = M[B-2]           " else restore absx in F
                    N, S -   2              " and change indication appropriately
                       F - TG15, E          " abs (x) > 1  ≡  absx > tg (pi/12) ?
                    Y, S -   1              " then change indication appropriately
                    U, A =   1, E           " absx > tg (pi/12) ?
                       F = M[B-2]
                    N, GOTO (:TOGETHER)
                       F × TG30             " then
                       F +   1              " replace
                       MC =  F              " absx
                       F = M[B-4]           " by
                       F - TG30             " (absx - tg (pi/6)) /
                       F / MC[-2]           " (1 + absx × tg (pi/6))
                       M[B-2] = F
TOGETHER:              MC =  S              " preserve indication
                       A = :C6
                       SUBC (:POL IN FF)    " compute polynomial in F square
                       F × M[B+1]           " × absx square
                       F +   1
                       F × MC[-3]           " × absx
                       S =MC[1]             " indication
                    U, S -   1, P           " > 1 ?
                    Y, MC = - F
                    N, MC =  F
                       G =   S, Z           " indication = 0 ?
                    N, F × PI OVER 6
                       F + MC[-2]
                       S = MC[-1], P        " original sign of x > 0 ?
                    N, F = - F
                       GOTOR (MC[-1])
C1:                    +  13 65333
                       + 223 69812          " -.333333333246
C2:                    -  13 69702
                       + 402 22387          " +.199999980477
C3:                    +  13 99661
                       - 119 33742          " -.142855496622
C4:                    -  14 27237
                       + 571 72235          " +.111044707738
```

```
" standard-functions nr. 11

" ARCTAN continued

C5:            +  14 03157
               - 595 65289       " -.895216002193₁₀-1
C6:            -  14 58249
               + 433 90644       " +.622201788749₁₀-1
TG15:          -  13 08524
               +  26 69885       " tg (pi/12) = +.267949192430
TG30:          -  13 05990
               + 438 49281       " tg (pi/6) = 1/sqrt(3) = +.577350269190
PI OVER 6:     -  13 34909
               + 431 06668       " pi/6 = +.523598775599

               'END' ARCTAN      " 57 instructions
```