```
        section .data
SA:     dd      500.312
DA:     dq      500.312
SPI:    dd      3.141592653589793238462
DPI     dq      3.141592653589793238462
SB:     dd      1.456e6
DB:     dq      1.456e6

HEX:    db      '0123456789ABCDEF'                                      ;hex table
L1:     db      'SA :'
L2:     db      'DA :'
L3:     db      'SPI:'
L4:     db      'DPI:'
L5:     db      'SB :'
L6:     db      'DB :'
L7:     db      'SC :'
L8:     db      'EX1:'
L9:     db      'EX2:'
L10:    db      'EXP:'
L11:    db      '2s :'
OUTPUT: db      '                                                   '   ;output buffer
SC:     dd      0.0
SC1:    dd      0.0
LEN:    db      0


        section .text
        global  main                                                    ;Tell linker about main
        extern  write, exit
main:
        mov rbp, rsp                                                    ; for correct debugging
        push    rbp
        mov     rbp, rsp

        lea     rsi, [L1]                                               ; Write Out SA
        call    MYLABEL
        mov     rbx,[SA]
        mov     eax,8
        call    TOHEX
        mov     eax,9
        call    MYWRITE

        lea     rsi, [L2]                                               ; Write Out DA
        call    MYLABEL
        mov     rbx,[DA]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE

        lea     rsi, [L3]                                               ; Write Out SPI
        call    MYLABEL
        mov     rbx,[SPI]
        mov     eax,8
        call    TOHEX
        mov     eax,9
        call    MYWRITE

        lea     rsi, [L4]                                               ; Write Out DPI
        call    MYLABEL
        mov     rbx,[DPI]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE

        lea     rsi, [L5]                                               ; Write Out SB
        call    MYLABEL
        mov     rbx,[SB]
        mov     eax,8
        call    TOHEX
        mov     eax,9
        call    MYWRITE

        lea     rsi, [L6]                                           ; Write Out DB
        call    MYLABEL
        mov     rbx,[DB]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE

        fld     dword [SA]                                          ; load SA Value
        fmul    dword [SB]                                          ; Mul by SB, ST(0) has value
        fstp    dword [SC]                                          ; store it

        lea     rsi, [L7]                                           ; Write Out SC
        call    MYLABEL
        mov     rbx,[SC]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE


        fld     dword [SB]                                          ; load SB Value
        FXTRACT                                                     ; split into Exponent and Mantissa
        fstp    dword [SC]                                          ; Pop Mantissa
        fstp    dword [SC]                                          ; store Exponent

        lea     rsi, [L8]                                           ; Write Out SB Exponent
        call    MYLABEL
        mov     rbx,[SC]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE

        fld     qword [DB]                                          ; load SA Value
        FXTRACT                                                     ; split into Exponent and Mantissa
        fstp    dword [SC]                                          ; Pop Mantissa
        fstp    dword [SC]                                          ; store exponent

        lea     rsi, [L9]                                           ; Write Out DB Exponent
        call    MYLABEL
        mov     rbx,[SC]
        mov     eax,16
        call    TOHEX
        mov     eax,17
        call    MYWRITE

        lea     rsi, [L11]                                          ; Write Out 2s complement
        call    MYLABEL
        mov     rax,[SC]                                            ; get exponent value
        neg     rax                                                 ; Negate it
        call    TOBIN
        mov     eax,33
        call    MYWRITE                                             ; write in binary



        fld     dword [SC]                                          ; load exponent Value again..
        frndint
        fist    dword [SC1]                                         ; store back as integer...
        lea     rsi, [L10]                                              ; Write Out exponent as decimal
        call    MYLABEL
        mov     eax,[SC1]
        call    TODEC
        mov     eax,[LEN]
        call    MYWRITE


        xor     edi, edi                                                ; 0 return = success
        call    exit


; Usage: Load eax with length
MYWRITE:
        mov     edx, eax                                                ; Parameter 3 for write
        lea     rsi, [OUTPUT]                                           ; Parameter 2 for write
        mov     edi, 1                                                  ; Parameter 1 (fd)
        call    write
        ret

MYLABEL:
        mov     edx, 4                                                  ; Parameter 3 for write
        mov     edi, 1                                                  ; Parameter 1 (fd)
        call    write
        ret



; Usage: Load RBX with value
TOHEX:
        mov ecx,OUTPUT                                                  ;point to end of output string
needed
        add ecx,eax                                                     ; add in the length
        mov [ecx], byte 0xa                                             ;New Line at end
        dec ecx

                                                                        ;mov rbx,
[SC]                ;Load EBX with value
TH1:                                                                    ;loop start
        mov rax,rbx                                                     ;move working value to EAX
        and rax,0xF                                                     ;and to get lowest byte value...

        mov al,[HEX+eax]
        mov [ecx],al                                                    ;store number...

        shr rbx,4                                                       ;shift working value right for
next byte

        dec ecx                                                         ;dec str pointer
        cmp ecx,OUTPUT                                                  ;at beggining?
        jge TH1                                                         ;loop if not
        ret

;Usage : Load value into EAX
TOBIN:
        mov ebx,0x80000000                                              ;load divisor...
        mov ecx,OUTPUT                                                  ;point to output string
TB1:
        xor edx,edx                                                     ;clear things
        div ebx                                                         ;eax = quotient, edx = remainder
        add eax,48                                                      ;ascii adjust
        mov [ecx],al                                                    ;store number...
        inc ecx                                                         ;inc String Pointer
        mov eax,edx                                                     ;get remainer
        ror ebx,1                                                       ;rotate divisor
        cmp ebx,0x80000000                                              ;have we gone thru whole thing?
        jne TB1                                                       ;loop if not
        mov [ecx], byte 0xa                                           ;Line Feed
        ret

;Usage : Load value into EAX
TODEC:
        mov ebx,0
        mov [LEN],ebx                                                 ;clear length
        mov ebx,1000000000                                            ;load divisor...
        mov ecx,OUTPUT                                                ;point to output string
TD1:
        xor edx,edx                                                   ;clear things
        div ebx                                                       ;eax = quotient, edx = remainder
        add eax,48                                                    ;ascii adjust
        mov [ecx],al                                                  ;store number...
        mov eax,[LEN]                                                 ;Inc Length
        inc eax
        mov [LEN],eax
        inc ecx                                                       ;inc string ptr
        mov edi,edx                                                   ;store remainer
        mov eax,ebx                                                   ;mov divisor for divide
        mov ebx,10                                                    ;setup divide
        xor edx,edx                                                   ;clear things
        div ebx                                                       ;reduce divisor
        mov ebx,eax                                                   ;get divsor
        mov eax,edi                                                   ;restore remainer
        cmp ebx,0                                                     ;have we gone thru whole thing?
        jg TD1                                                        ;loop if not

        mov [ecx], byte 0xa                                           ;null term
        mov eax,[LEN]                                                 ;Inc Length
        inc eax
        mov [LEN],eax
        ret
```