```
* ninth.asm ( 6809 OS-9 assembler module )

* Y: Instruction Pointer
* X: temporary W register
* U: Param Stack
* S: Return Stack

  nam ninth
  ttl Ninth Forth

  use   defsfile

  org 0

tylg     set   Prgrm+Objct
atrv     set   ReEnt+rev
rev      set   $00
edition  set   1

  mod   eom,name,tylg,atrv,start,$800

name
  fcs /ninth/
  fcb edition

HelloNinth
  fcc /Hello Ninth!/
  fcb 10
  fcb 13
  fcb 0
endHelloNinth


start
  lda #1  ; stdout
  leax HelloNinth,pcr
  ldy #endHelloNinth-HelloNinth
  os9 I$WritLn

  leaU $-200,s  ; U is Parameter Stack.
  clrD          ;
  tfr d,y       ; Y is IP
  tfr d,x       ; X is W or Temp
  pshs d,x,y    ; push some zeroes for fun.
  pshu d,x,y    ; push some zeroes for fun.
  jsr Init,pcr
  leax c_main,pcr
  pshu x
  jmp Execute,pcr
```

```
* Print D (currently in %04x) and a space.
PrintDsp
  pshS D
  bsr PrintD
  ldb #32
  bsr putchar
  pulS D,PC

* Print D (currently in %04x).
PrintD
  pshS A,B
  pshS B
  tfr A,B
  bsr PrintB
  pulS b
  bsr PrintB
  puls a,b,pc

* Print B (as %02x)
PrintB
  pshS B
  lsrb
  lsrb
  lsrb
  lsrb
  bsr PrintNyb
  pulS B
  pshS B
  bsr PrintNyb
  pulS B,PC

* print low nyb of B.
PrintNyb
  pshS B
  andB #$0f  ; just low nybble
  addB #$30  ; add '0'

  cmpB #$3a  ; is it beyond '9'?
  blt Lpn001
  addB #('A-$3a)  ; covert $3a -> 'A'

Lpn001
  jsr putchar,pcr
  pulS B,PC


* putchar(b)
putchar
  pshS A,B,X,Y,U
  leaX 1,S     ; where B was stored
  ldy #1       ; y = just 1 char
  lda #1       ; a = path 1
  os9 I$WritLn ; putchar, trust it works.
  pulS A,B,X,Y,U,PC
```

```
Execute
  pulU x       ; arg -> W
  ldd 0,x      ; goto W+[W]
  jmp D,X

Enter
  pshS y       ; push old IP onto Return Stack.
  leay 2,x     ; load new IP after W.
  bra Next     ; start executing.

Next
  ldd 0,y
  leax d,y

  IFNE 0
  *** BEGIN printing IP.
  pshs d,x,y
  ldb #$28       "("
  bsr putchar

  tfr y,d
  leax 0,pcr        ; absolute addr of module
  pshs x            ; put it in mem (on S stack)
  subd 0,s          ; subtract begin of module
  leas 2,s          ; drop it from S stack
  jsr PrintD,pcr

  ldb #$29       ")"
  bsr putchar
  ldb #$20       " "
  bsr putchar
  puls d,x,y
  *** END printing IP.
  ENDC

  leay 2,y
  ldd 0,x
  jmp d,x


Exit
  pulU y       ; pop previous IP.
  bra Next     ; and keep going.

  use prelude.asm

  emod
eom equ *
```