

Intersystems FDC II bootstrap Program

```

; Intersystems FDC II bootstrap program
;   Written by Robert Bedichek February 1980

```

0000

if 0

This program resides in a 1K nonvolatile memory that starts at address zero. It reads a variable number of sectors from the first two cylinders of drive zero, surface zero. The first eight bytes of the first sector of the outermost cylinder gives information about how to load the data. The diskette can be recorded in either FM (single density) or MFM (double density). Sector sizes ranging from 128 bytes/sector to 8196 bytes/sector are accommodated. After the data is loaded, control is passed to an address specified in the first eight bytes of the first sector. The memory in which this program resides is switched off before control is passed.

Note that this program uses the end of read/write memory for stack space. So, the load parameters should not be set so that the transfer over-writes this part of memory.

An area WorkLen bytes from the end of memory is used to read in the first sector for the load parameters. This may be over-written by the data loaded.
Bootstrap register allocation and conventions

Register : Contents

```

-----
A   :   Scratch register
B   :   Starting sector number for next read operation
C   :   Ending   "   "   "   "   "   "
D   :   Instruction length (looked at by StartIO)
E   :   Instruction opcode ( " " " " )
H   :   First argument for instruction (looked at by startio)
L   :   Second  "   "   "   ( " " " " )

FLAGS :   Scratch

(alternate set)
A'  :   N - bytes per sector (0:128 1:256 2:512 3:1024 4:2048 .. )
B'  :   Starting sector number for cylinder 1
C'  :   Ending   "   "   "   "   "
D'  :   Scratch
E'  :   "
HL' :   Points to list of load parameters.

FLAGS' :   Carry = 1 (==) MFM   Carry = 0 (==) FM

IX  :   Starting address for booted program
IY  :   Not used

```

The load parameters are on drive zero, side zero, cylinder zero in the first eight bytes of the first sector. Their format:

```

      |-----|-----|-----|-----| unused ...
Word  0       1       2       3

```

Word 0: Address to start loading at.
 Word 1: Address to pass control to after data is loaded.
 Word 2: Start and end sectors of segment to be loaded from cylinder 0.
 (0, 0) means don't load anything from cylinder 0.
 Word 3: Start and end sectors of segment to be loaded from cylinder 1.
 (0, 0) means don't load anything from cylinder 1.

Example:

00 E4 03 FA 02 26 1 26

This will load sectors 2 through 26 from cylinder 0 and then sectors 1 through 26 from cylinder 1. The data will be load continuously starting at location E400 and execution will begin at FA03.

endif

; Port addresses

```

00B0      DBase  equ  0B0H      ; FDC II occupies block 16 bytes long.
00B0      DStat  equ  DBase+0   ; FDC II status register.
00B1      DData  equ  DBase+1   ; Command port address.

00B2      DRead  equ  DBase+2   ; out DRead sets controller for read operation.
00B6      ENROM  equ  DBase+6   ; Writing to this port will enable the ROM.
00B8      HiDMA  equ  DBase+8   ; High byte of DMA address register.
00BA      MiDMA  equ  DBase+10  ; Middle byte (A15 - A8) of DMA address res.
00BC      LoDMA  equ  DBase+12  ; Low byte of DMA address.

00BE      DsROM  equ  DBase+14  ; Writing to this port turns off the ROM.

00FF      Lights equ  0FFH     ; Front panel programmed output lights.

000A      Step  equ  10        ; Milliseconds per step.

2000      MStart equ  8192     ; Address to start testing memory at.
2100      WorkLen equ 8192+256 ; Must be a multiple of 256

0000*0000 Start  org  0
0000 F3      di          ; No funny business.

0001 D3 B2      out      DRead  ; Set up the DMA controller for readings.

; Non-destructively test memory to find the end of it (to find a scratch
; area to use for reading the load parameters.
0003 21 2000    lxi      h,MStart  ; Address to start testing at.
0006 7E        MemLP  mov      a,m
0007 2F        cmc
0008 77        mov      m,a      ; Store complement.
0009 BE        cmf      m      ; Did the location actually get complemented?
000A 20 07     jrnz     EndHit  ; Branch if we just ran into a non-RAM area.
000C 2F        cmc
000D 77        mov      m,a      ; Restore the byte to its original value.
000E 23        inx      h
000F 7C        mov      a,h
0010 B5        ora      l      ; Have we just overflowed?

```

```

0011 20 F3          jrnz   MemLf

; HL points to one past the last good RAM location.
0013 F9          EndHit  sphl          ; So we can do procedure calls.
0014 7C          mov     ash
0015 D6 21       sui     WorkLen/256
0017 67          mov     h:a          ; HL points to an area WorkLen long.
0018 D9          exx

0019 08          exaf
001A AF          xra     a
001B 08          exaf

; Initialize NEC 765
001C 11 0303     lxi     d:303H      ; 3 byte instruction, opcode = specify
001F 21 6EFE     lxi     h:16-Step*16+0EH*256+0FEH      ; Steps rate and other junk.
0022 CD 00C1     call    StartIO    ; Do it.

; Sense Drive zero status, wait until drive is ready
0025 11 0204     RdyWait lxi     d:200H+4      ; 2 byte instruction, opcode=sense drive status
0028 21 0000     lxi     h:0          ; Drive zero, head zero.
002B CD 00C1     call    StartIO    ; Do it.
002E CD 0181     call    WaitStat
0031 DB B1       in     DData          ; retrieve Status Register 3
0033 E6 20       ani     20h          ; check ready bit
0035 28 EE       jrz    RdyWait     ; repeat if drive not ready

; Recalibrate Drive zero.
0037 11 0207     lxi     d:200H+7      ; 2 byte instruction, opcode = recalibrate
003A 26 00       mvi     H:0          ; Drive zero, head zero.
003C CD 00C1     call    StartIO

003F CD 0131     call    SenInt      ; Sense interrupt status to make NEC happy.

; Alternately try reading the header at single and double density.
0042 11 020A     RHLF   lxi     d:200H+0AH      ; 2 byte instruction, opcode = read header
0045 21 0000     lxi     h:0          ; Drive zero.
0048 CD 00C1     call    StartIO

004B CD 014C     call    Result
004E 28 05       jrz    Success     ; Branch if the read was good.

; Complement the C' to switch the density settings.
0050 0B          exaf
0051 3F          cmc
0052 0B          exaf
0053 18 ED       jmpr   RHLF

; A' now has the right N and C' has the right density.
; Now we can load the data

; Load the DMA registers with the address of the work area.
0055 D9          Success exx
0056 AF          xra     a
0057 D3 B8       out    HiDMA        ; A23 - A16 are zero.

```

```

0059 7C          mov     a,h      ; High byte of work area.
005A D3 BA      out     HiDMA
005C 7D          mov     a,l      ; Low byte of work area address.
005D D3 BC      out     LoDMA
005F D9          exx

; Read sector 1 on side zero, drive zero, head zero, cylinder zero
; to set the load information.
0060 11 0406     lxi     d,400H+6 ; 9 byte instruction, opcode = read.
0063 21 0000     lxi     h,0      ; head 0, drive 0, cylinder 0
0066 01 0101     lxi     b,101h   ; Read from sector 1 to sector 1.
0069 CD 00C1     call    StartIO

006C CD 014C     call    Result
006F C2 0000     jnz     Start    ; restart process if error

; Read the data

; Load the DMA registers with the load address.
0072 D9          exx
0073 7E          mov     a,m      ; Low byte of load address.
0074 D3 BC      out     LoDMA
0076 23          inx     h
0077 7E          mov     a,m
0078 D3 BA      out     HiDMA

; Put next word (the start address) in IX.
007A 23          inx     h
007B 5E          mov     e,m
007C 23          inx     h
007D 56          mov     d,m
007E D5          push   d        ; Get start address into IX by pushing it
007F DD E1      pop     ix       ; onto the stack from DE and popping from IX.

; Get starting and ending sector numbers
0081 23          inx     h
0082 7E          mov     a,m      ; Starting sector number for track zero.
0083 D9          exx
0084 47          mov     b,a

0085 D9          exx
0086 23          inx     h
0087 7E          mov     a,m
0088 D9          exx
0089 4F          mov     c,a
008A D9          exx

; Now set starting and ending sector numbers for cylinder one.
008B 23          inx     h
008C 46          mov     b,m
008D 23          inx     h
008E 4E          mov     c,m
008F D9          exx
0090 21 0000     lxi     h,0      ; Start on drive zero, cylinder zero.

0093 11 0406     RdNext lxi     d,400H+6 ; 9 byte instruction, opcode = read.

```

```

0096 CD 00C1      call   StartIO      ; Read a cylinder.

0099 CD 014C      call   Result
009C C2 0000      jnz    Start        ; restart process if error
009F 2C          inr    i            ; cylinder := cylinder + 1
00A0 78          mov    a,r1
00A1 FE 02       cpi    2            ; Are we done ?
00A3 CA 0194      jz     ByeBye

; Seek to cylinder one.
00A6 11 030F     lxi    d,300H+0FH   ; 2 byte instruction, opcode = seek
00A9 CD 00C1     call   StartIO
00AC CD 0131     call   SenInt       ; Make NEC happy.
; Put cylinder 1's start and end sectors in B and C.
00AF D9         exx
00B0 C5         push  b
00B1 D9         exx
00B2 C1         pop   b
; Check if either sector specification for track two is 0
; If so, we are done
00B3 78         mov    a,b
00B4 FE 00      cpi    0
00B6 CA 0194    jz     ByeBye
00B9 79         mov    a,c
00BA FE 00      cpi    0
00BC CA 0194    jz     ByeBye

00BF 18 D2      jmp    RdNext

; StartIO loads up the NEC 765's registers with bytes from the Z-80's
; registers. If the carry is set, the operation will be done in MFM.
; Registers A and D are trashed.

00C1 CD 011C     StartIO call   WaitCom
00C4 0E         exaf
00C5 30 02      Jrnc   FM        ; Branch if we are doing instruction in FM.
00C7 CB F3      bset   6,r6      ; Turn on MFM bit.

00C9 08         FM    exaf
00CA 7B         mov    a,e
00CB D3 B1      out   DData     ; Open wide 765.
00CD D3 FF      out   Lights    ; Display opcode on front panel lights.

; Check to see if this is a one byte instruction.
00CF 15         dcr   d
00D0 C8         rz

; Load first argument
00D1 CD 011C     call   WaitCom
00D4 7C         mov    a,h
00D5 D3 B1      out   DData

00D7 15         dcr   d
00D8 C8         rz

; Load second argument.

```

```

00D9 CD 011C      call   WaitCom
00DC 7D          mov    a,r1
00DD D3 B1       out    DData

00DF 15          dcr    d
00E0 CB          rz

```

```

; Load head number - always zero.

```

```

00E1 CD 011C      call   WaitCom
00E4 AF          xra    a
00E5 D3 B1       out    DData

```

```

; Load starting sector number.

```

```

00E7 CD 011C      call   WaitCom
00EA 78          mov    a,b
00EB D3 B1       out    DData

```

```

; Load N

```

```

00ED CD 011C      call   WaitCom
00F0 C8          exaf
00F1 D3 B1       out    DData
00F3 08          exaf

```

```

; Load ending sector number.

```

```

00F4 CD 011C      call   WaitCom
00F7 79          mov    a,c
00F8 D3 B1       out    DData

```

```

; Load saf length

```

```

00FA CD 011C      call   WaitCom
00FD 0E          exaf
00FE F5          push   PSW
00FF 11 018C     lxi    d,GPLTAB
0102 83          add    e
0103 5F          MOV    E,A
0104 F1          POP    PSW
0105 08          exaf
0106 1A          ldax  d
0107 D3 B1       out    DData

```

```

; I know that GPLTAB does not span a 256 byte boundry.

```

```

; A := GPLTAB( N )

```

```

; Load data transfer length.

```

```

0109 CD 011C      call   WaitCom
010C 3E FF       mvi    a,0FFh
010E 0E          exaf
010F F5          push   PSW
0110 B7          ora    a
0111 20 04       jrnz   NZero
0113 08          exaf
0114 3E 80       mvi    A,80H
0116 0E          exaf

```

```

; Is N = 0

```

```

0117 F1          NZero POP    PSW
0118 08          exaf
0119 D3 B1       out    DData
011B C9          ret

```

; WaitCom waits for the data register in the NEC 765 to be ready for
; loading. Affects no registers.

```

0110 F5      WaitCom push   PSW
011B C5              push   b
011E 06 14      TF      mvi    b,20      ; Retry count for loading command register.
0120 DB B0      WaitLP in     DStat
0122 E6 C0              ani    0C0H      ; Just look at the top two bit of the status register.
0124 FE 80              cpi    080H
0126 2B 06              jrz    OKNow
0128 10 F6              djnz  WaitLP

```

; Time out - read the data port to try to remedy the problem.

```

012A DB B1              in     DData
012C 18 F0              jmpf  TF

```

```

012E C1      OKNow pop    b
012F F1              pop    PSW
0130 C9              ret

```

; Read the interrupt status by issuing a sense interrupt status instruction.
; This is mandatory after a seek or recalibrate.

```

0131 11 0108      SenInt lxi    d,100H+8      ; 1 byte instruction; opcode = sense int.
0134 CD 00C1              call  StartIO
0137 06 14              mvi    b,20      ; A fine number.
0139 DB B0      RD      in     DStat
013B E6 C0              ani    0c0h
013D FE C0              cpi    0c0h
013F 20 02              jrnz  NotYet
0141 DB B1              in     DData      ; DUMP it.
0143 10 F4      NotYet djnz  RD
0145 DB B0              in     DStat
0147 E6 01              ani    1      ; Look at the FDC busy bits.
0149 20 E6              jrnz  SenInt
014B C9              ret

```

; Result is called to perform the Result Phase of the NEC 765's operation
; sequence. Seven status bytes are read from the 765. A' is set to
; N (the last byte read), which gives the number of bytes/sector

```

014C E5      Result push  h
; ST0 reads the first status byte in the result phase. A () 0 (==) error

014D CD 0181              call  WaitStat
0150 DB B1              in     DData

0152 E6 18              ani    18h      ; Look at fault and not ready bits.
0154 67              mov    h,a

```

; Read second status byte (ST1).

```

0155 CD 0181              call  WaitStat
0158 DB B1              in     DData

```



```

015A E6 35      ani    35H      ; Look at CRC, Over Run, Missing Header & sector bits.
015C B4         ora    h
015D 67         mov    h,a

```

```

; Read the third status byte (ST2)

```

```

015E CD 0181    call   WaitStat
0161 DB B1      in     DData

```

```

0163 E6 31      ani    31h      ; CRC error, wrong cylinder, missing address mark.
0165 B4         ora    h
0166 67         mov    h,a

```

```

; Read all those dumb bytes that you get at the end.

```

```

0167 CD 0181    call   WaitStat
016A DB B1      in     DData    ; Cylinder
016C CD 0181    call   WaitStat
016F DB B1      in     DData    ; Head number (0..1)
0171 CD 0181    call   WaitStat
0174 DB B1      in     DData    ; Sector number.
0176 CD 0181    call   WaitStat
0179 08         exaf
017A DB B1      in     DData    ; N - bytes/sector
017C 08         exaf

```

```

017D 7C         mov    a,h      ; Get error flag.
017E B7         ora    a        ; Set Z to 1 if error occurred.
017F E1         pop    h
0180 C9         ret

```

```

; WaitStat returns when the status register is ready to be read.
; Register A and the flags are burned.

```

```

WaitStat

```

```

0181 C5         push   b
0182 DB B0      wtlf   in     DStat
0184 E6 C0      ani    0C0H    ; Look at top two bits of the status register.
0186 FE C0      cpi    0C0H
0188 20 FB      jrnz   WtLf

```

```

018A C1         pop    b
018B C9         ret

```

```

; Gap length table - indexed by N

```

```

018C 07 0E 1B 35 GPLTAB DB      7,14,27,53

```

```

; The followins code is executed after the data has been loaded
; from disk. It turns off the ROM and then executes a pcix
; instruction.
; ram by code at the SKIP label.

```

```

0190 D3 BE      instr  out     DsROM
0192 DB E9      pcix
0004          instr1 equ    $-instr

```

```

ByeBye

```

```

0194 21 FFF6    lxi   h,-10    ; Space for instructions below the stack.

```

```
0197 39          dad    sf
0198 E5          push   h
0199 EB          xchs
019A 31 0190     lxi    h,instr    ; First byte of instructions to move to RAM.
019D 01 0004     lxi    b,instr1   ; Length of instructions.
01A0 ED B0      ldir
01A2 E1          pop    h
01A3 E9          pchl          ; Branch to instructions in RAM.

          0000          end    Start
```

0 errors. 38 symbols generated. Space for 5515 more symbols.

\$ 01A4
RYEBYE 0194
DBASE B0
DDATA B1
DREAD B2
DSROM BE
DSTAT B0
ENDHIT 13
ENROM B6
FM C9
GPLTAB 018C
HIDMA B8
INSTR 0190
INSTRL 04
LIGHTS FF
LODMA BC
MEMLP 06
MIDMA BA
MSTART 2000
NDTYET 0143
NZERO 0117
OKNOW 012E
RDNEXT 93
RDYWAI 25
RESULT 014C
RHLP 42
RO 0139
SENINT 0131
START 00
STARTI C1
STEP 0A
SUCCES 55
TP 011E
WAITCO 011C
WAITLP 0120
WAITST 0181
WORKLE 2100
WTLF 0182