**Code:**

```
;       Reading the harddisk using ports!
;       +------------------------------+   by qark
;
;
;  This took me months to get working but I finally managed it.
;
;  This code only works for the 286+ so you must detect for 8088's somewhere
;  in your code.
;
;  Technical Information on the ports:
;      Port    Read/Write   Misc
;     ------  ------------ ------------------------------------------------
;       1f0       r/w       data register, the bytes are written/read here
;       1f1       r         error register  (look these values up yourself)
;       1f2       r/w       sector count, how many sectors to read/write
;       1f3       r/w       sector number, the actual sector wanted
;       1f4       r/w       cylinder low, cylinders is 0-1024
;       1f5       r/w       cylinder high, this makes up the rest of the 1024
;       1f6       r/w       drive/head
;                             bit 7 = 1
;                             bit 6 = 0
;                             bit 5 = 1
;                             bit 4 = 0  drive 0 select
;                                   = 1  drive 1 select
;                             bit 3-0    head select bits
;       1f7       r         status register
;                             bit 7 = 1  controller is executing a command
;                             bit 6 = 1  drive is ready
;                             bit 5 = 1  write fault
;                             bit 4 = 1  seek complete
;                             bit 3 = 1  sector buffer requires servicing
;                             bit 2 = 1  disk data read corrected
;                             bit 1 = 1  index - set to 1 each revolution
;                             bit 0 = 1  previous command ended in an error
;       1f7       w         command register
;                            commands:
;                             50h format track
;                             20h read sectors with retry
;                             21h read sectors without retry
;                             22h read long with retry
;                             23h read long without retry
;                             30h write sectors with retry
;                             31h write sectors without retry
;                             32h write long with retry
;                             33h write long without retry
;
```

```
;  Most of these should work on even non-IDE hard disks.
;  This code is for reading, the code for writing is the next article.

   mov     dx,1f6h         ;Drive and head port
   mov     al,0a0h         ;Drive 0, head 0
   out     dx,al

   mov     dx,1f2h         ;Sector count port
   mov     al,1            ;Read one sector
   out     dx,al

   mov     dx,1f3h         ;Sector number port
   mov     al,1            ;Read sector one
   out     dx,al

   mov     dx,1f4h         ;Cylinder low port
   mov     al,0            ;Cylinder 0
   out     dx,al

   mov     dx,1f5h         ;Cylinder high port
   mov     al,0            ;The rest of the cylinder 0
   out     dx,al

   mov     dx,1f7h         ;Command port
   mov     al,20h          ;Read with retry.
   out     dx,al
still_going:
   in      al,dx
   test    al,8            ;This means the sector buffer requires
            ;servicing.
   jz      still_going     ;Don't continue until the sector buffer
            ;is ready.

   mov     cx,512/2        ;One sector /2
   mov     di,offset buffer
   mov     dx,1f0h         ;Data port - data comes in and out of here.
   rep     insw

;   ------
   mov     ax,201h         ;Read using int13h then compare buffers.
   mov     dx,80h
   mov     cx,1
   mov     bx,offset buffer2
   int     13h

   mov     cx,512
   mov     si,offset buffer
   mov     di,offset buffer2
   repe    cmpsb
   jne     failure
```

```
   mov     ah,9
   mov     dx,offset readmsg
   int     21h
   jmp     good_exit
failure:
   mov     ah,9
   mov     dx,offset failmsg
   int     21h
good_exit:
   mov     ax,4c00h        ;Exit the program
   int     21h

   readmsg db      'The buffers match.  Hard disk read using ports.$'
   failmsg db      'The buffers do not match.$'

buffer  db      512 dup ('V')
buffer2 db      512 dup ('L')

;
;       Writing to the hard disk using the ports!     by qark
;       +-------------------------------------+
;
;  The only differences between reading and writing using the ports is
;  that 30h is sent to the command register, and instead of INSW you
;  OUTSW.
;

   mov     dx,1f6h         ;Drive and head port
   mov     al,0a0h         ;Drive 0, head 0
   out     dx,al

   mov     dx,1f2h         ;Sector count port
   mov     al,1            ;Write one sector
   out     dx,al

   mov     dx,1f3h         ;Sector number port
   mov     al,1           ;Wrote to sector one
   out     dx,al

   mov     dx,1f4h         ;Cylinder low port
   mov     al,0            ;Cylinder 0
   out     dx,al

   mov     dx,1f5h         ;Cylinder high port
   mov     al,0            ;The rest of the cylinder 0
   out     dx,al

   mov     dx,1f7h         ;Command port
   mov     al,30h          ;Write with retry.
   out     dx,al
```

```
oogle:
   in      al,dx
   test    al,8            ;Wait for sector buffer ready.
   jz      oogle

   mov     cx,512/2        ;One sector /2
   mov     si,offset buffer
   mov     dx,1f0h         ;Data port - data comes in and out of here.
   rep     outsw           ;Send it.

;    ------------

   mov     ax,201h                 ;We'll read in sector 1 using
   mov     bx,offset buffer2       ;int13h and see if we are successful.
   mov     cx,1
   mov     dx,80h
   int     13h

   mov     cx,512
   mov     si,offset buffer
   mov     di,offset buffer2
   repe    cmpsb                   ;Compare the buffers.
   jne     failure

   mov     ah,9
   mov     dx,offset write_msg
   int     21h
   jmp     w_exit
failure:
   mov     ah,9
   mov     dx,offset fail
   int     21h

w_exit:
   mov     ax,4c00h        ;Exit the program
   int     21h

   write_msg       db      'Sector one written to using the ports, OH NO! there goes
XP.$'
   fail            db      'Writing using ports failed.$'

buffer  db      512 dup ('A')
buffer2 db      512 dup ('D')
```