

```

1  OSI Week10: I/O and device driver
2
3  1. An example of I/O instructions:
4      Setting the cursor position
5
6  The code below is excerpted from WeensyOS's k-hardware.c. It
7  uses I/O instructions to set a blinking cursor in the upper left of
8  the screen.
9
10 // console_show_cursor(cpos)
11 // Move the console cursor to position 'cpos',
12 // which should be between 0 and 80 * 25.
13
14 void console_show_cursor(int cpos) {
15     if (cpos < 0 || cpos > CONSOLE_ROWS * CONSOLE_COLUMNS)
16         cpos = 0;
17
18     outb(0x3D4, 14); // Command 14 = upper byte of position
19     outb(0x3D5, cpos / 256); // row
20     outb(0x3D4, 15); // Command 15 = lower byte of position
21     outb(0x3D5, cpos % 256); // column
22
23 }
24
25
26

```

Cheng Tan, OSI

```

27
28 2. Memory-mapped I/O
29
30 a. Here is a 32-bit PC's physical memory map:
31
32 +-----+ <- 0xFFFFFFFF (4GB)
33 |       |
34 |       32-bit
35 |       memory mapped
36 |       devices
37 |       |
38 |       \/\_/\_/\_/\_/\_/\_/\_
39 |       \/\_/\_/\_/\_/\_/\_/\_
40 |       |
41 |       Unused
42 |       |
43 +-----+ <- depends on amount of RAM
44 |       |
45 |       |
46 |       Extended Memory
47 |       |
48 |       |
49 +-----+ <- 0x00100000 (1MB)
50 |       |
51 +-----+ <- 0x000F0000 (960KB)
52 |       |
53 |       16-bit devices,
54 |       expansion ROMs
55 |       |
56 +-----+ <- 0x000C0000 (768KB)
57 |       |
58 |       VGA Display
59 |       |
60 +-----+ <- 0x000A0000 (640KB)
61 |       |
62 |       Low Memory
63 |       |
64 +-----+ <- 0x00000000

```

[Credit to Frans Kaashoek, Robert Morris, and  
Nickolai Zeldovich for this picture]

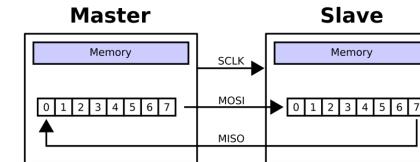
```

65   b. Loads and stores to the device memory "go to hardware".
66
67   An example is in the console printing code from WeensyOS.
68   Here is an excerpt from link/shared.ld:
69
70   /* Compare the address below to the map above. */
71   PROVIDE(console = 0xB8000);
72
73   This is an excerpt from lib.c; notice how it uses the address
74   "console":
75
76
77   /*
78    * prints a character to the console at the specified
79    * cursor position in the specified color.
80    * Question: what is going on in the check
81    * if (c == '\n')
82    * ?
83    * Hint: '\n' is "C" for "newline" (the user pressed enter).
84    */
85   static void console_putc(console_printer* p, unsigned char c, int color) {
86       console_printer* cp = (console_printer*) p;
87       if (cp->cursor >= console + CONSOLE_ROWS * CONSOLE_COLUMNS) {
88           cp->cursor = console;
89       }
90       if (c == '\n') {
91           int pos = (cp->cursor - console) % 80;
92           for (; pos != 80; pos++) {
93               *cp->cursor++ = ' ' | color;
94           }
95       } else {
96           *cp->cursor++ = c | color;
97       }
98   }

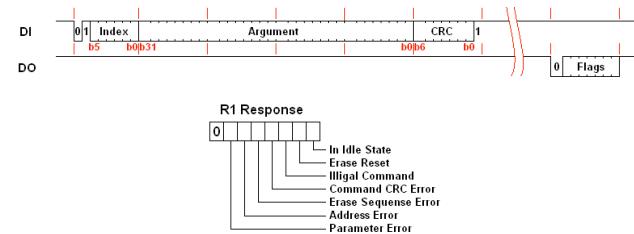
```

## 3. SPI basics

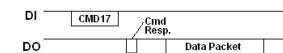
- SPI architecture



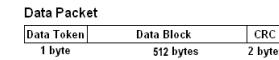
- Command and response



- Single-block read



- Data packet



- Single-block write



- Data response

