Software and Hardware Interaction
The Division Between Hardware and Software

Most modern computer systems are built as a series of layers, or levels. The lowest level is usually considered the physical "hardware" layer and the topmost layer is usually a user application or source code.
Computers Are Really Pretty Stupid!

Fact: A computer really only knows two things: 1s and 0s. That's it. Period.

Question 1:

Then how does a computer system "know" about things like:
- Programs
- Operating Systems (i.e., MS-DOS)
- BIOS
- Modems
- Joysticks, etc.

Question 2:

"where" is a user's program?
"where" is the operating system?
"where" is the BIOS?

Question 3:

How do we get all these pieces to play nicely together?
The Answer Is Simple, But Pretty Detailed

Remember, a computer is a state machine. That means that if the processor is forced into it's initial condition, then it’s next state is completely predictable. We can use this simple fact to see what happens when a computer system boots.

![Diagram of the boot process](image-url)
Addresses Provide A Critical Link

All of these operations depend on particular pieces of information being in predictable locations. Every piece of information stored in a computer has a unique address in the system’s ROM, RWM, or bulk storage.
Memory Addresses

Different processors have different numbers of address bits, which means that they can access different amounts of memory. For example:

<table>
<thead>
<tr>
<th>Processor</th>
<th>Addx Bits</th>
<th>Number of Addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>8085</td>
<td>16</td>
<td>64K</td>
</tr>
<tr>
<td>8088/8086</td>
<td>20</td>
<td>1M</td>
</tr>
<tr>
<td>80286</td>
<td>24</td>
<td>16M</td>
</tr>
<tr>
<td>80386</td>
<td>32</td>
<td>4G</td>
</tr>
</tbody>
</table>

All programs reside somewhere in the memory space of the processor when they execute.

Some hardware devices may reside in the memory space of the processor as well!
A Processor's Port Address Space

Now for some real confusion.

Many processors, like the 80x86, actually have two address spaces they can access. The primary one is the memory space we've been discussing. The other one is called the Port, or I/O, Address Space.

To add even more confusion:

The Port Address Space uses exactly the same address pins on the processor as the Memory address space! This means that the processor must provide a signal that tells the memory system if the access is a memory access or a port access. In an ISA bus computer, a total of four signals are provided:

**MEMR-** goes low when the processor initiates a read from the memory address space.

**MEMW-** goes low when the processor initiates a write to the memory address space.

**IOR-** goes low when the processor initiates a read from the port address space.

**IOW-** goes low when the processor initiates a write to the port address space.

From software, there are only two instructions that can access the port address space:

```
in reg,[addx] ;load reg from port addx.
out addx, reg ;write reg to the port addx.
```