Lecture 24: I/O

Synchronous I/O – read/write access is blocked until the data buffer is written/read and cleared
Asynchronous I/O – read/write operations are done in parallel along with other operations

Asynchronous I/O is obviously more desirable (like the difference between sockets or pipes), but keep in mind that the CPU (on a single processor system) still must be time-sliced between the read/write operations and hence are not typically done in parallel. On unix, a hidden disk cache is used for disk output. All systems need an AIO control block, which is a structure that contains the file descriptor, file offset, pointer, byte size, request priority, signal information, and I/O type of operation of the buffer to be received/transmitted.

On 68K uprocessors, a DUART is used (Dual Universal Asynchronous Receiver and Transmitter) as a serial port. It contains 7 distinct interrupt level lines, a clock, input/output pins from/to external interface registers, and a transmit and receive line to a RS232 serial port. However, serial ports are for slow data communications (10 to 20 kbaud), but can extend to 50 feet before data attenuation.

Peripheral computer devices (printers, disks, tape drives) use parallel ports for fast data transfer (10 MB/sec), but can only be up to 6 feet long due to risk of data loss.

For different I/O devices, you have to ask which is best for you: data loggers, databases, web servers, keyboards, terminal display devices, printers, etc.

I/O registers and other hardware interface drivers are stored in the lowest layer of any software design system. At the second layer you have translation routines that allow the upper layer analysis routines to communicate with the hardware. Third and fourth layers are used for the data manipulation and analysis. And the fifth layer is the user interface or GUI.

Every layer is discrete, independent, hierarchical, using its own state machine. They communicate with each other using service provisional points (SPP) and service access points (SAP). Keep in mind the upper layers always **drive** the lower layers, and the lower layers always **service** the upper layers of any software design system. Note that nothing keeps you from using multi-dimensional layers in parallel.