1.2 Textbook readings

Design For Moore's Law

Moore's law stated that intefrated curcuit resources double every 18-24 months. Moore (a founder of intel), found that because chip designs take years the resources may double, or even triple because of the lengthy process of designing a processor. Architects therefore must anticipate where tecnology will be in future years in order to compinsate for this discrepency.

Using abstaction

Programmers and architects use abstraction in order to increase productivity. This is a way that architects get past the problem posed by moores law. Abstraction is used to represent a design at different levels of its representation.

The Common Case

The common case must be made fast in order to enhance performance. This is because the common case is simply the most common function / use case. For example in building a game. It would make more sense to optimize the games performance while in the bounds of the play space, rather than some area that should not be visited.

Performance Via Parallelism

By performing parallel operations we may see better performance because of this. An example being the turbines of a jet. A jet will perform better with two turbines, while they are working parellel to one another.

Performance via Pipelining

A pipeline allows the user to breakdown the steps in order to make them more manageable. It is a response to the previous "pipe." \

Performance via Prediction

Prediction allows the designer to improve the performance of a design as long as the action is certain. However the designer must be careful in making sure that an incorrect performance is not too expensive of an operation.

Hierarchy of Memorys

Programmers may assign importance to memory in order to increase its speed. The fastest, smallest, and most expensive per bit is at the top of importance because the data will take the longest to load. Say for example a social media account. An account that you may not have used in 5 years may take longer to load because within the hierarchy of data it is not often accessed.

Dependability via Redundancy

We make systems have redundant data because of the possibility that some may fail. This is known as **redundancy**. We use this in order to ensure data correctness no matter the circumstance.

For more about these topics read What is a Computer?