Parallelism

Parallel Systems perform multiple actions at the same time

Multitasking systems execute programs in parallel

- Example: surfing the web while playing music files
- or editing a document while updating a spreadsheet

Switching frequently between sequentially executing tasks

- Gives the illusion of continuous execution
- Tasks execute concurrently rather than simultaneously
- The tasks may run on the same processor

Forms of Parallelism

Pipelining overlaps multiple instructions
- Example of fine grained instruction level parallelism (ILP)
- Overlaps instruction steps (fetch, decode, execute, etc.)
- Scalar pipeline contains just one execution unit

Superscalar processors allow true parallel processing
- Takes place within the CPU
- Executes separate instructions simultaneously
- Invisible to the user
- Compiler assists in supplying more instructions
- Separate execute units may each be pipelined

Forms of Parallelism

Hyperthreading is another form of parallelism

- Single execute unit switches between threads
- Registers & PC are replicated (one set per thread)
- Thread switches cause a different register set to be used
- Hides memory latency caused by cache misses

True parallel processing uses multiple execution units

- The units run at the same time
- Each executing a separate thread
- These are called multiprocessor systems

Forms of Parallelism

Multi-core systems have 2 or more processors on a single chip

- required resources are shared:
  - Memory interface
  - Cache control
  - Interconnect systems

More cost effective than single sophisticated processors

- Increase performance without greater complexity
- Use lower clock rates, thus less power

Forms of Parallelism

In parallel systems, groups of processors work together

- *topology* defines the way the processors are interconnected

The collaborating processors must communicate

- There are two options:
  - Shared bus
  - Shared interconnection network

- Bus-based multiprocessors share memory
  - Processors can access another's memory directly

Forms of Parallelism

Message-passing multiprocessors use communication links such as ethernet or other proprietary high speed links

The degree of coupling differs for bus-based vs. message-passing
- Coupling is far higher for bus-based systems
- Bus-based systems have high bandwidth but are expensive
- Message-passing systems are less complicated

Bus-based systems are harder to scale
- access to the shared memory becomes harder to manage
- These are referred to a "tightly coupled"
- Versus the "loosely coupled" message-passing systems

Forms of Parallelism

Memory systems greatly affect multiprocessor performance

- Uniform memory access (UMA)
  - Equally accessible by all processors
  - Used with smaller tightly coupled bus-based systems

- NUMA (non-uniform memory access)
  - memory is not homogeneous
  - not all memory is equally accessible to all processors
  - Used in large message passing multiprocessors
  - Referred to a loosely coupled