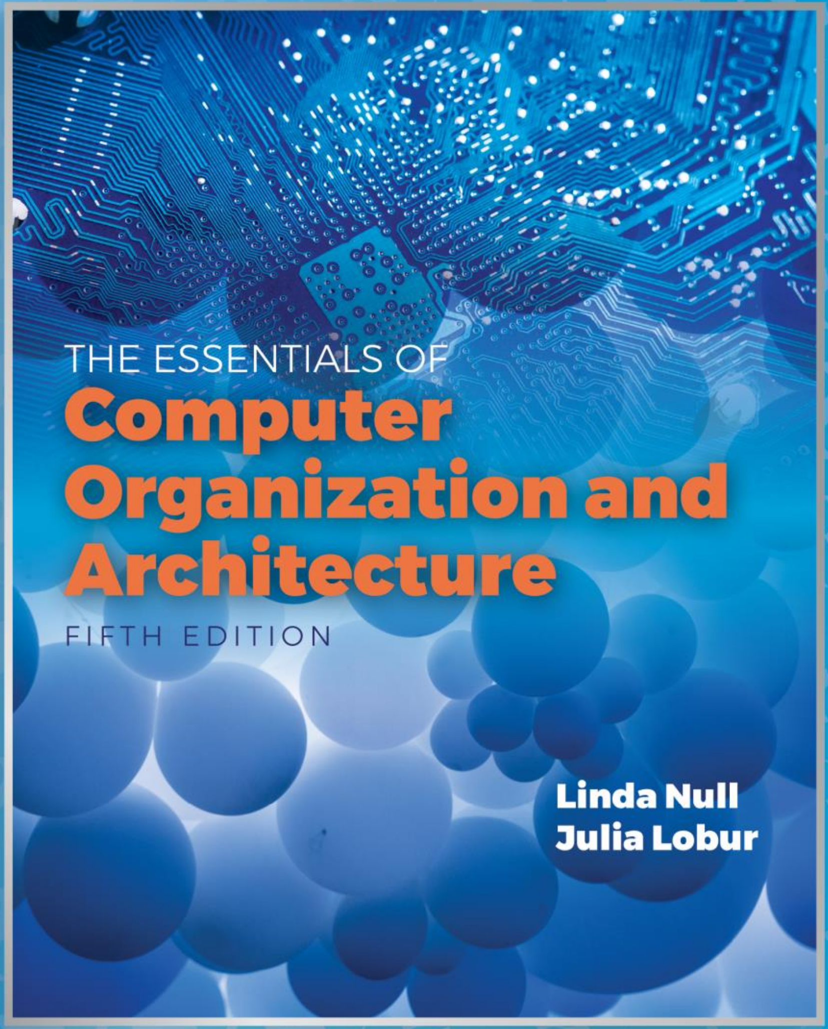
This is the third lecture of Chapter 9

Chapter 9

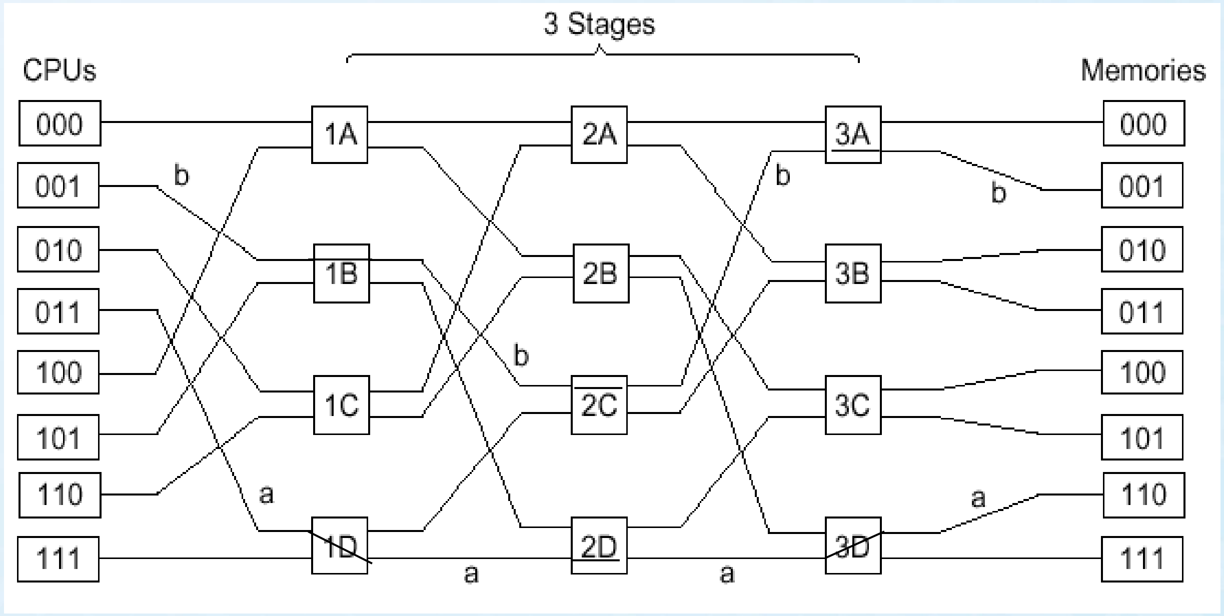
Alternative Architectures (**C**)



Quick review of last lecture

- Parallel and Multiprocessor Architectures
  - Superscalar
    - *Superpipelining,*
    - *Dynamically issue multiple instructions per clock cycle*
  - VLIW
  - Vector Processors
  - Interconnection Networks
    - Several different topologies: Complete, star, ring, mesh, …
    - Switching networks with 2×2 switches: Omega Network
    - Switching networks with crossbar switches
    - Blocking vs non-blocking

An 8×8 Omega Network



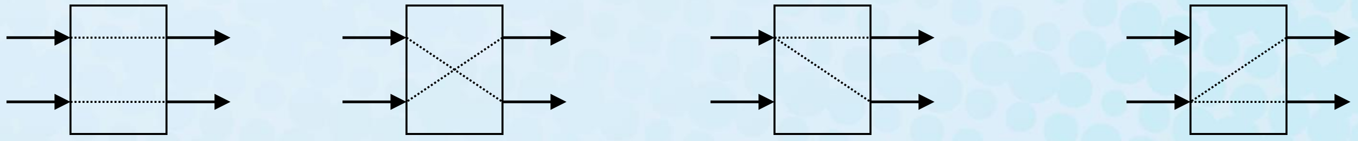
In n×n Omega Network

- $\log_2 n$ stages
- n/2 2×2 switches per stage
- Perfect shuffle ISC

Routing

- a: 011→110
- b: 001→001

The use of 2×2 switches



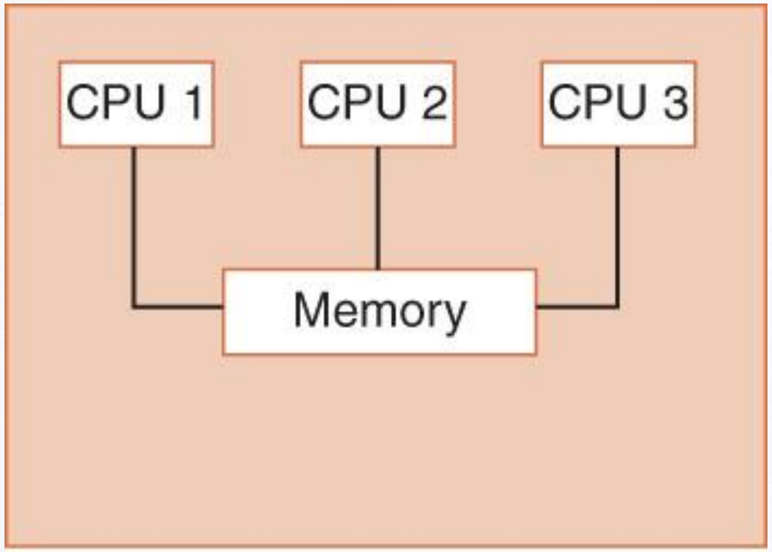
Straight Crossover Upper broadcast Lower broadcast

9.4 Parallel and Multiprocessor Architectures (13 of 21)

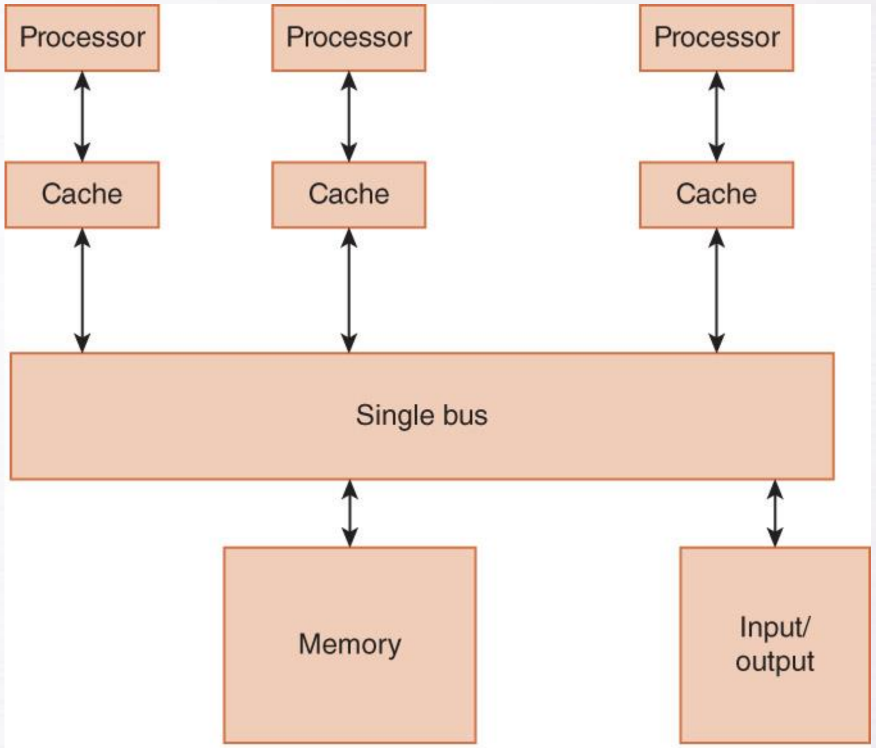
9.4.4 Shared Memory Multiprocessors

- Tightly-coupled multiprocessor systems use the same memory. They are also referred to as shared memory multiprocessors.
- The processors do not necessarily have to share the same block of physical memory.
- Each processor can have its own memory, but it must share it with the other processors.
- Configurations such as these are called *distributed shared memory multiprocessors*.

- Tightly-coupled multiprocessor

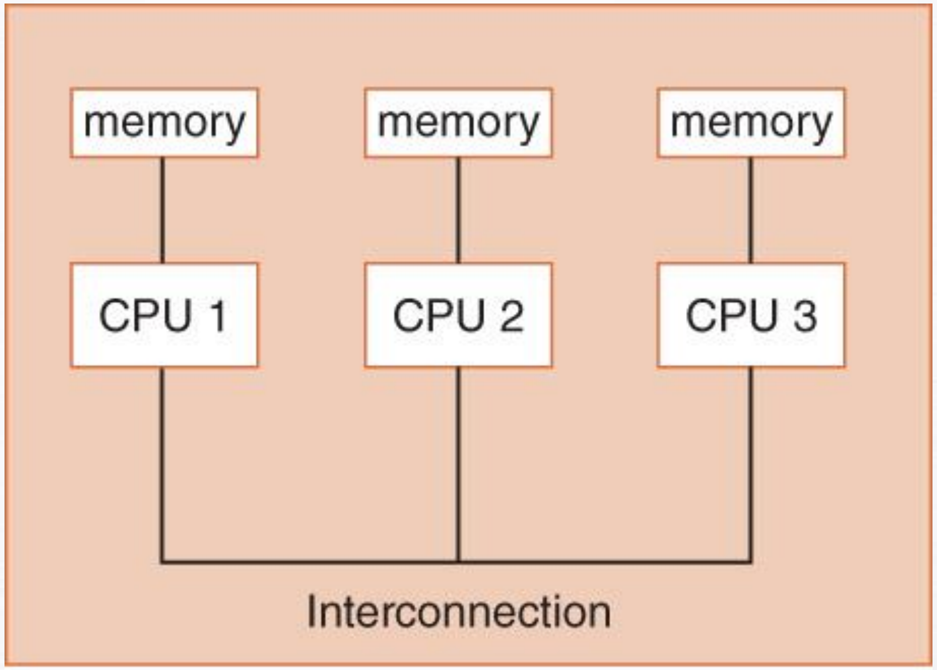


(a) Global shared memory



(c) Global shared memory with separate cache at processors

- Distributed shared memory multiprocessor



(b) Distributed shared memory

9.4 Parallel and Multiprocessor Architectures (14 of 21)

- Shared memory MIMD machines can be divided into two categories based upon how they access memory.
- In *uniform memory access* (UMA) systems, all memory accesses take the same amount of time.
- To realize the advantages of a multiprocessor system, the interconnection network must be fast enough to support multiple concurrent accesses to memory, or it will slow down the whole system.
- Thus, the interconnection network limits the number of processors in a UMA system.

9.4 Parallel and Multiprocessor Architectures (15 of 21)

- The other category of MIMD machines are the *nonuniform memory access* (NUMA) systems.
- While NUMA machines see memory as one contiguous addressable space, each processor gets its own piece of it.
- Thus, a processor can access its own memory much more quickly than it can access memory that is elsewhere.
- Not only does each processor have its own memory, it also has its own cache, a configuration that can lead to *cache coherence* problems.

9.4 Parallel and Multiprocessor Architectures (17 of 21)

- When a processor’s cached value is updated concurrently with the update to memory, we say that the system uses a *write-through* cache update protocol.
- If the *write-through with update* protocol is used, a message containing the update is broadcast to all processors so that they may update their caches.
- If the *write-through with invalidate* protocol is used, a broadcast asks all processors to invalidate the stale cached value.

9.4 Parallel and Multiprocessor Architectures (18 of 21)

- Write-invalidate uses less bandwidth because it uses the network only the first time the data is updated, but retrieval of the fresh data takes longer.
- Write-update creates more message traffic, but all caches are kept current.
- Another approach is the *write-back* protocol that delays an update to memory until the modified cache block must be replaced.
- At replacement time, the processor writing the cached value must obtain exclusive rights to the data. When rights are granted, all other cached copies are invalidated.

9.4 Parallel and Multiprocessor Architectures (19 of 21)

9.4.5 Distributed computing

- Distributed computing is another form of multiprocessing. However, the term *distributed computing* means different things to different people.
- In a sense, all multiprocessor systems are distributed systems because the processing load is distributed among processors that work collaboratively.
- The common understanding is that a distributed system consists of very loosely-coupled processing units.
- Recently, NOWs have been used as distributed systems to solve large, intractable problems.

9.4 Parallel and Multiprocessor Architectures (20 of 21)

- For general-use computing, the details of the network and the nature of the multiplatform computing should be transparent to the users of the system.
- Remote procedure calls (RPCs) enable this transparency. RPCs use resources on remote machines by invoking procedures that reside and are executed on the remote machines.
- RPCs are employed by numerous vendors of distributed computing architectures including the Common Object Request Broker Architecture (CORBA) and Java’s Remote Method Invocation (RMI).

9.4 Parallel and Multiprocessor Architectures (21 of 21)

- Cloud computing is distributed computing to the extreme.
- It provides *services* over the Internet through a collection of loosely-coupled systems.
- In theory, the service consumer has no awareness of the hardware, or even its location.
  - Your services and data may even be located on the same physical system as that of your business competitor.
  - The hardware might even be located in another country.
- Security concerns are a major inhibiting factor for cloud computing.

9.5 Alternative Parallel Processing Approaches (1 of 15)

- Some people argue that real breakthroughs in computational power—breakthroughs that will enable us to solve today’s intractable problems— will occur only by abandoning the von Neumann model.
- Numerous efforts are now underway to devise systems that could change the way that we think about computers and computation.
- In this section, we will look at three of these: dataflow computing, neural networks, and systolic processing.

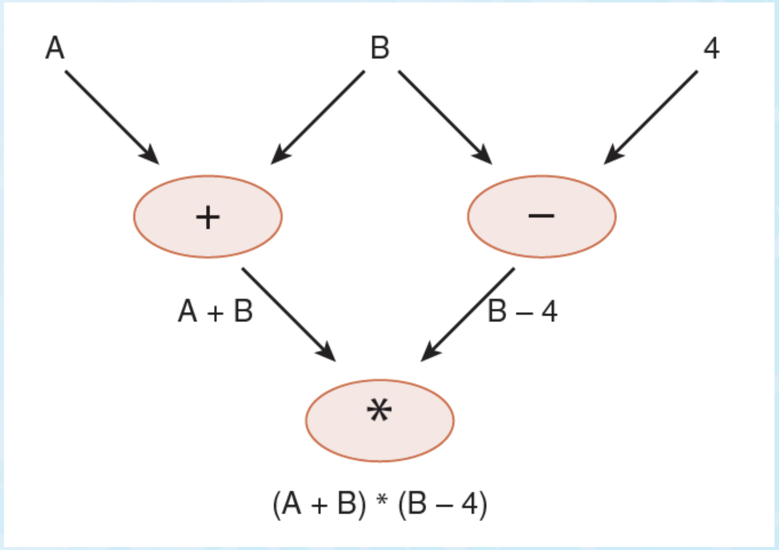
9.5 Alternative Parallel Processing Approaches (2 of 15)

9.5.1 Dataflow Computing

- Von Neumann machines exhibit sequential control flow: A linear stream of instructions is fetched from memory, and they act upon data.
- Program flow changes under the direction of branching instructions.
- In *dataflow* computing, program control is directly controlled by data dependencies.
- There is no program counter or shared storage.
- Data flows continuously and is available to multiple instructions simultaneously.

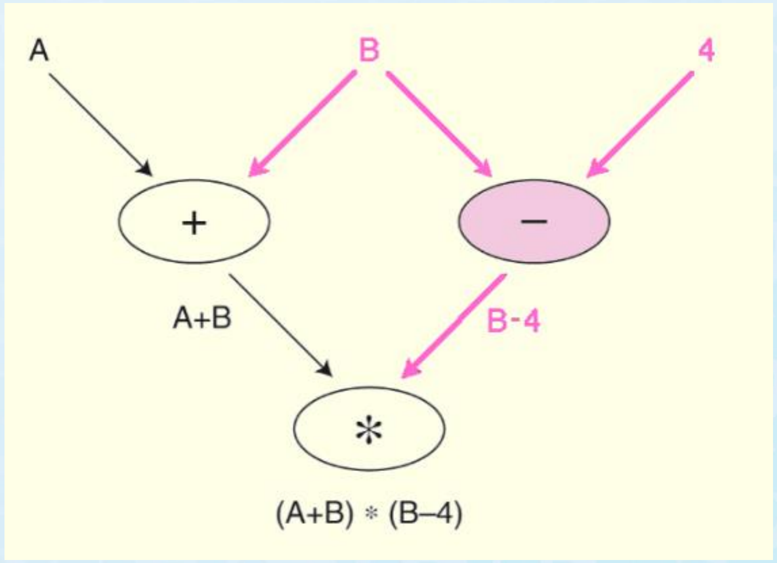
9.5 Alternative Parallel Processing Approaches (3 of 15)

- A *data flow* graph represents the computation flow in a dataflow computer.
- Its nodes contain the instructions and its arcs indicate the data dependencies.



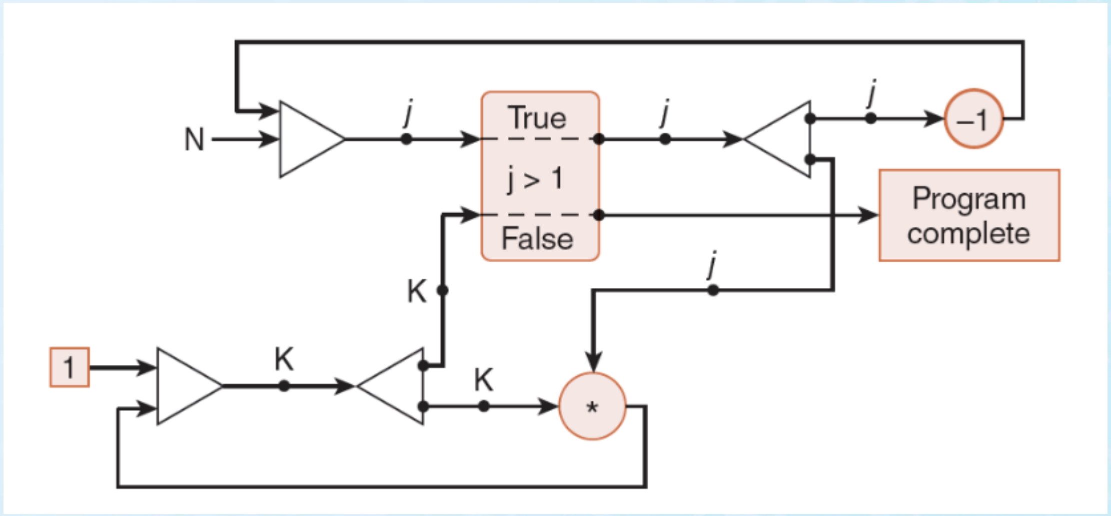
9.5 Alternative Parallel Processing Approaches (4 of 15)

- When a node has all of the data tokens it needs, it fires, performing the required operation, and consuming the token.
- The result is placed on an output arc.



9.5 Alternative Parallel Processing Approaches (5 of 15)

- A dataflow program to calculate N! and its corresponding graph are shown on the right.



```
(initial j <- n; k <- 1
  while j > 1 do
    new k <- * j;
    new j <- j - 1;
return k)
```

9.5 Alternative Parallel Processing Approaches (6 of 15)

- The architecture of a dataflow computer consists of processing elements that communicate with one another.
- Each processing element has an *enabling unit* that sequentially accepts tokens and stores them in memory.
- If the node to which this token is addressed fires, the input tokens are extracted from memory and are combined with the node itself to form an executable packet.

9.5 Alternative Parallel Processing Approaches (7 of 15)

- Using the executable packet, the processing element's *functional unit* computes any output values and combines them with destination addresses to form more tokens.
- The tokens are then sent back to the enabling unit, optionally enabling other nodes.
- Because dataflow machines are data driven, multiprocessor dataflow architectures are not subject to the cache coherency and contention problems that plague other multiprocessor systems.

9.5 Alternative Parallel Processing Approaches (8 of 15)

9.5.2 Neural networks

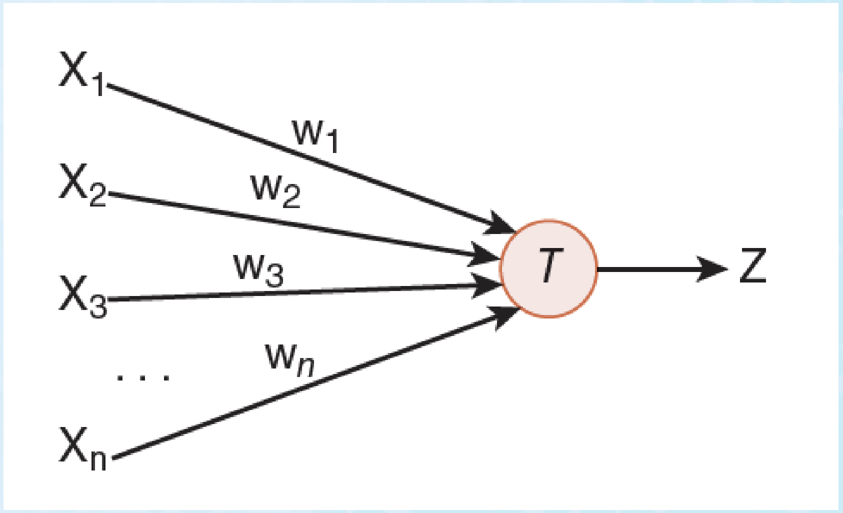
- *Neural network* computers consist of a large number of simple processing elements that individually solve a small piece of a much larger problem.
- They are particularly useful in dynamic situations that are an accumulation of previous behavior, and where an exact algorithmic solution cannot be formulated.
- Like their biological analogues, neural networks can deal with imprecise, probabilistic information, and allow for adaptive interactions.

9.5 Alternative Parallel Processing Approaches (9 of 15)

- Neural network processing elements (PEs) multiply a set of input values by an adaptable set of weights to yield a single output value.
- The computation carried out by each PE is simplistic—almost trivial—when compared to a traditional microprocessor. Their power lies in their massively parallel architecture and their ability to adapt to the dynamics of the problem space.
- Neural networks learn from their environments. A built-in *learning algorithm* directs this process.

9.5 Alternative Parallel Processing Approaches (10 of 15)

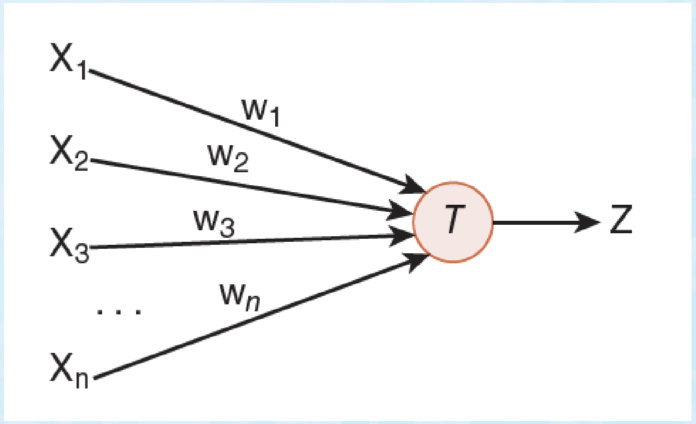
- The simplest neural net PE is the *perceptron*.
- Perceptrons are trainable neurons. A perceptron produces a Boolean output based upon the values that it receives from several inputs.



9.5 Alternative Parallel Processing Approaches (11 of 15)

- Perceptrons are trainable because the threshold and input weights are modifiable.
- In this example,

$$Z = \begin{cases} true(1), & if\ w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq T \\ false(0), & otherwise \end{cases}$$



Examples of Perceptrons

$$Z = \begin{cases} true(1), & if\ w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq T \\ false(0), & otherwise \end{cases}$$

$\boxed{w_1 = 1, w_2 = 1,\ and\ T = 2} \Rightarrow \boxed{Z = x_1\ and\ x_2}$

$\boxed{w_1 = 1, w_2 = 1,\ and\ T = 1} \Rightarrow \boxed{Z = x_1\ or\ x_2}$

9.5 Alternative Parallel Processing Approaches (12 of 15)

- Perceptrons are trained by use of supervised or unsupervised learning.
- Supervised learning assumes prior knowledge of correct results, which are fed to the neural net during the training phase. If the output is incorrect, the network modifies the input weights to produce correct results.
- Unsupervised learning does not provide correct results during training. The network adapts solely in response to inputs, learning to recognize patterns and structure in the input sets.

9.5 Alternative Parallel Processing Approaches (13 of 15)

- The biggest problem with neural nets is that when they consist of more than 10 or 20 neurons, it is impossible to understand how the net is arriving at its results. They can derive meaning from data that are too complex to be analyzed by people.
  - The U.S. military once used a neural net to try to locate camouflaged tanks in a series of photographs. It turned out that the nets were basing their decisions on the cloud cover instead of the presence or absence of the tanks.
- Despite early setbacks, neural nets are gaining credibility in sales forecasting, data validation, and facial recognition.