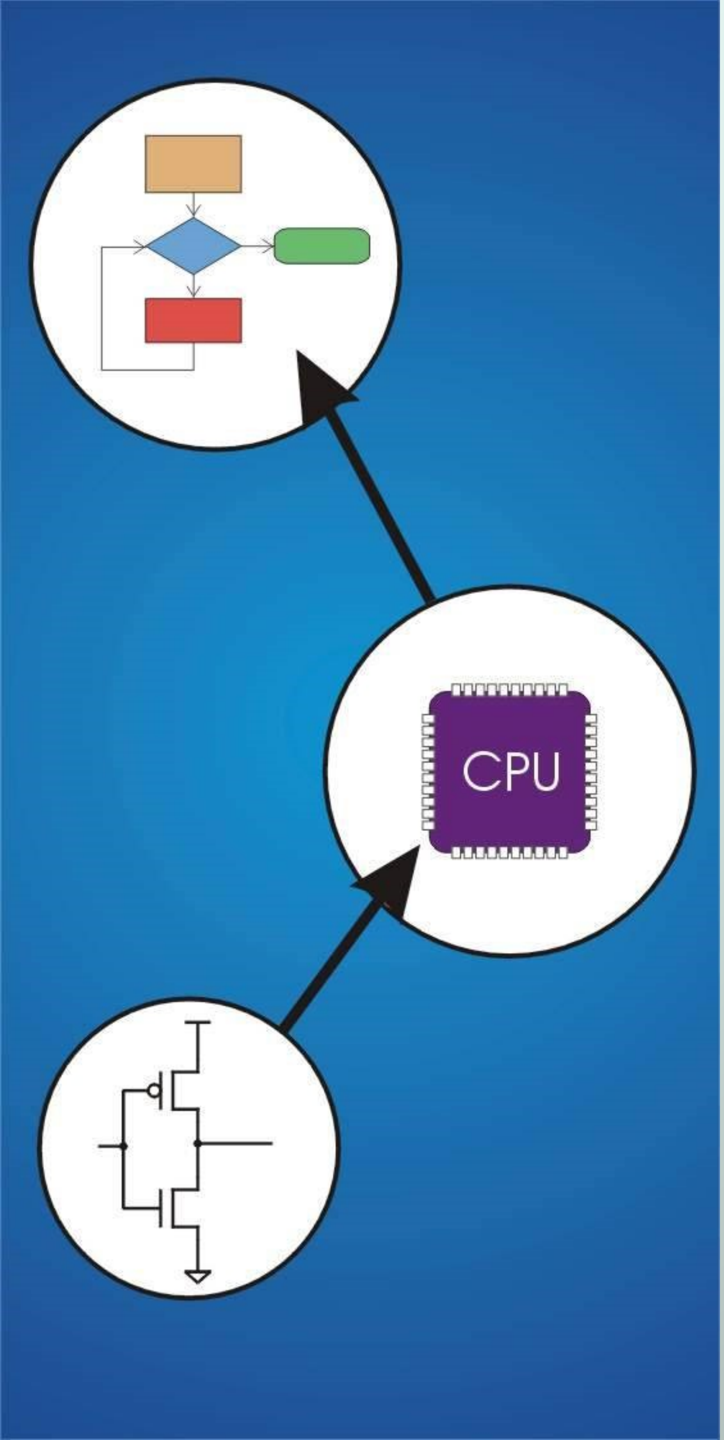


Final Exam Review

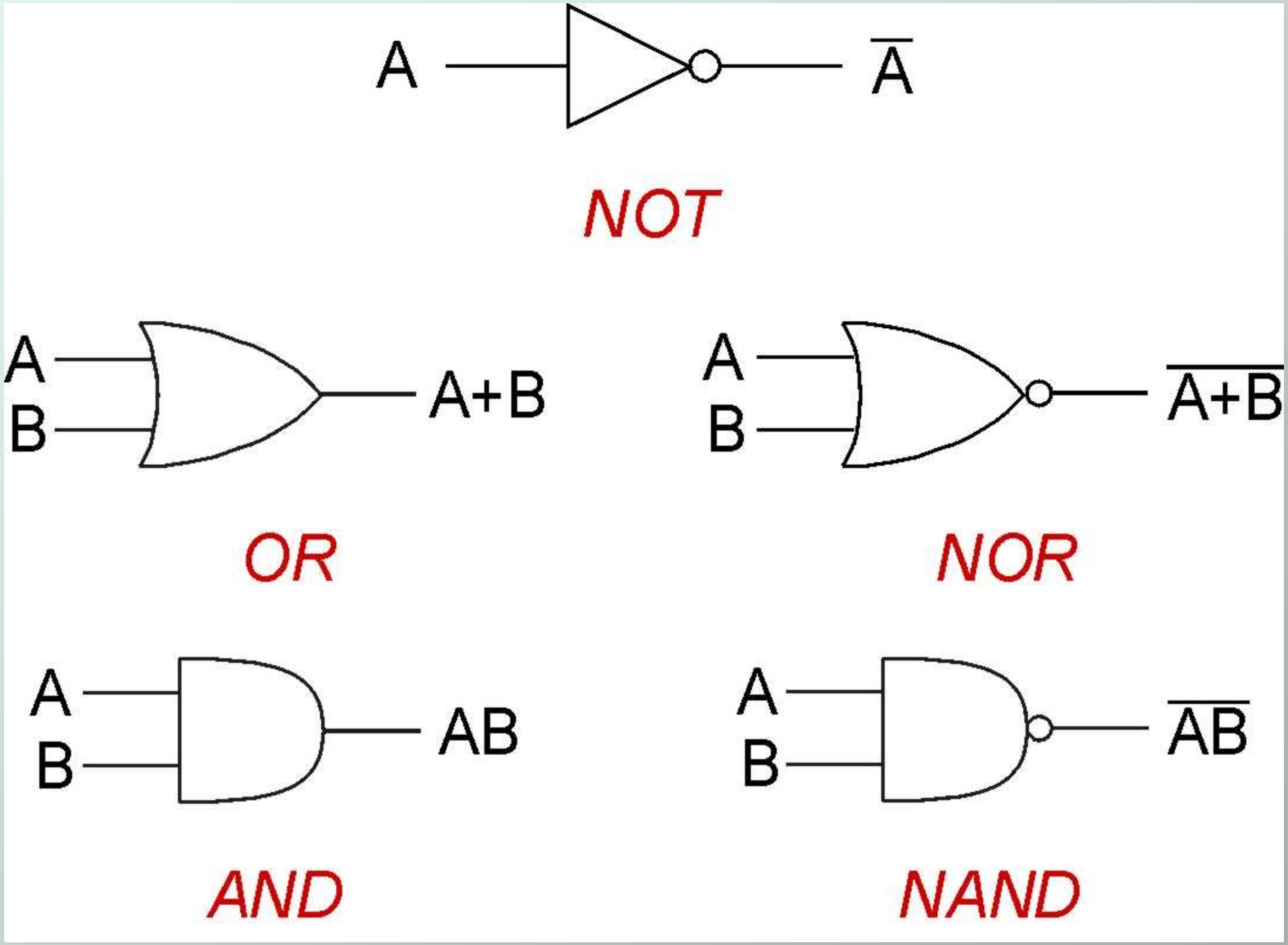
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Transistor: Digital Building Blocks

- Logically, each transistor acts as a switch
- Combined to implement logic functions (gates)
  - AND, OR, NOT
- Combined to build higher-level structures
  - Multiplexer, decoder, register, memory …
  - Adder, multiplier …
- Combined to build simple processor
  - LC-3

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Basic Logic Gates



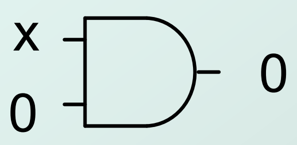
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

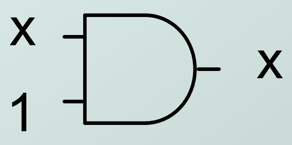
Propagation Delay

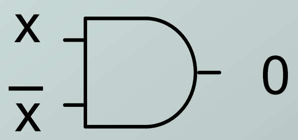
- Each gate has a propagation delay, typically fraction of a nanosecond (10^{-9} sec).
- Delays accumulate depending on the chain of gates the signals have to go through.
- Clock frequency of a processor is determined by the delay of the longest combinational path between storage elements, i.e. cycle time.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Boolean Algebra



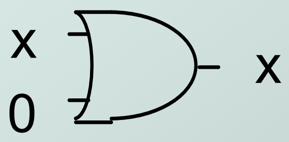


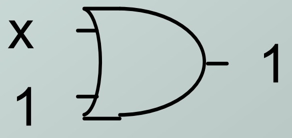


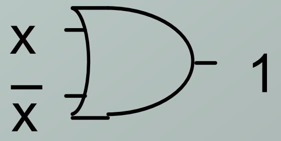
$x \bullet 0 = 0$

$x \bullet 1 = x$

$x \bullet \overline{x} = 0$







$x+0 = x$

$x+1 =$

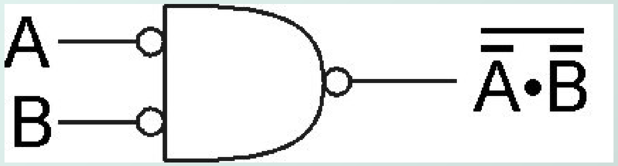
$x+\overline{x} =$

Remember Identify, Domination, Negation Laws from Logic!

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

DeMorgan's Law

- Converting AND to OR (with some help from NOT)
- Consider the following gate:



A	B	$\overline{A}$	$\overline{B}$	$\overline{A} \cdot \overline{B}$	$\overline{\overline{A} \cdot \overline{B}}$
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1

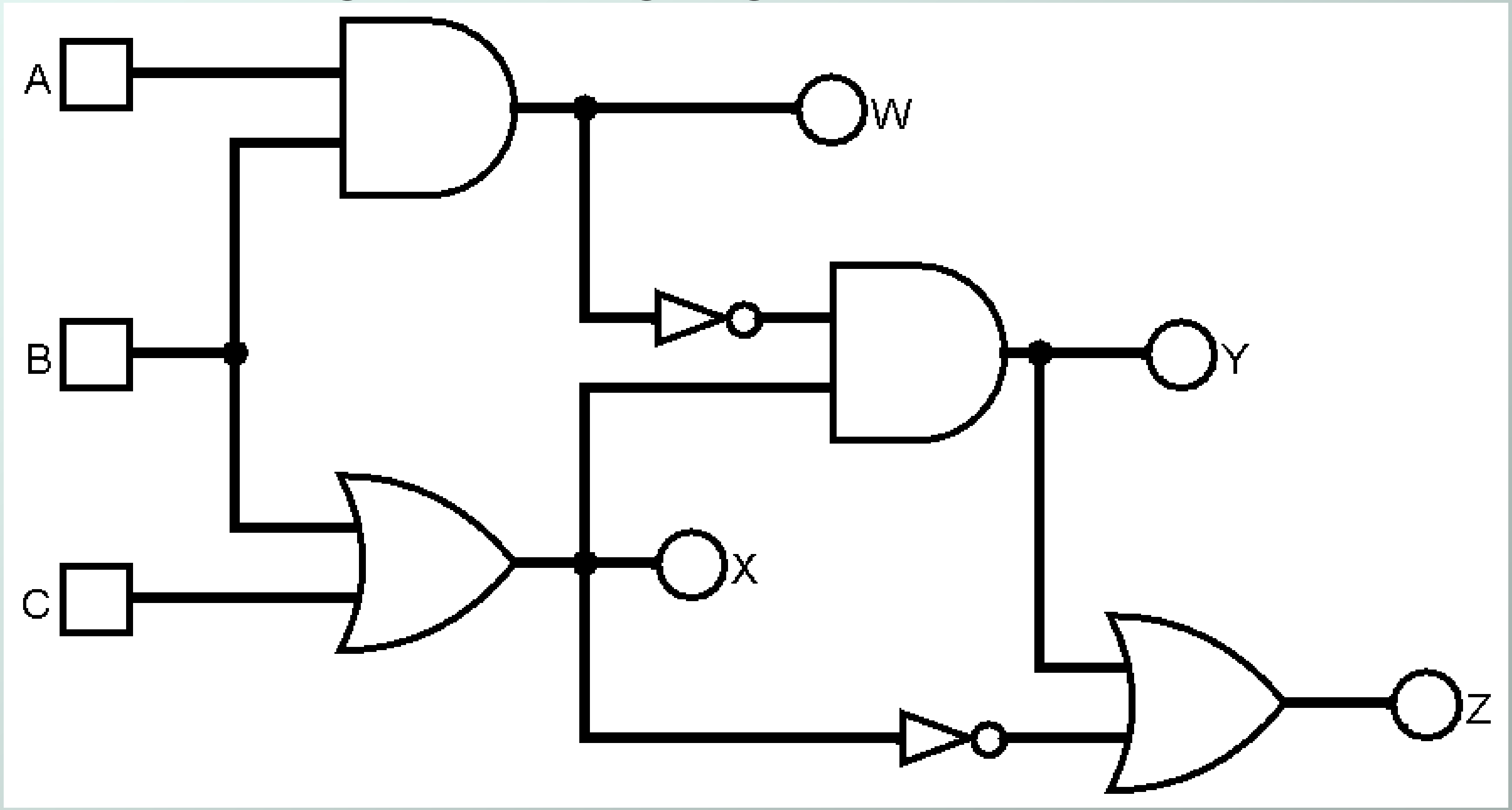
$\boxed{\text{Same as A OR B!}}$

To convert AND to OR (or vice versa), invert inputs and output.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Combinational Logic

- Cascading set of logic gates



What is the truth table?

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Truth Table (from circuit)

- Truth table for circuit on previous slide

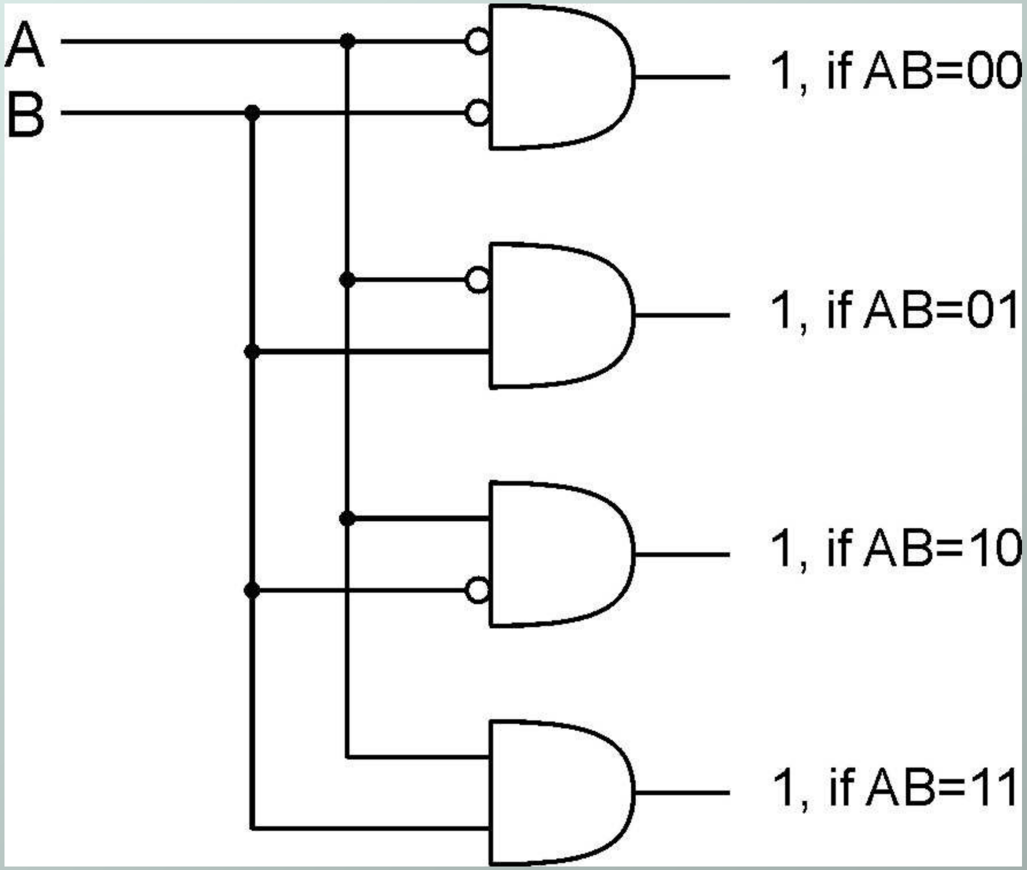
A	B	C	W	X	Y	Z
0	0	0	0	0	0	1
0	0	1	0	1	1	1
0	1	0	0	1	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	1	1	0	0

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Decoder

- *n* inputs, 2*ⁿ* outputs
  - exactly one output is 1 for each possible input pattern

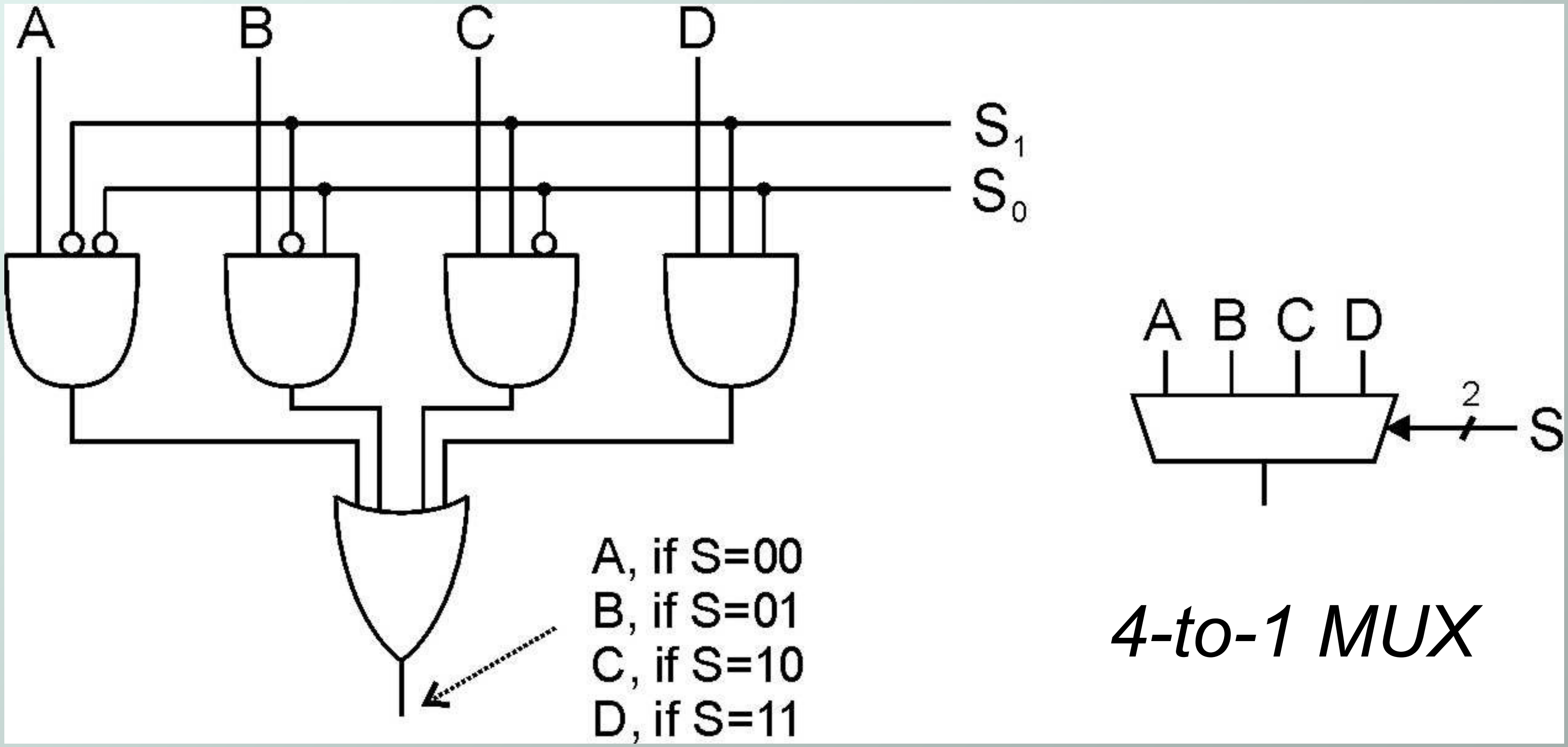
2-bit decoder



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Multiplexer (MUX)

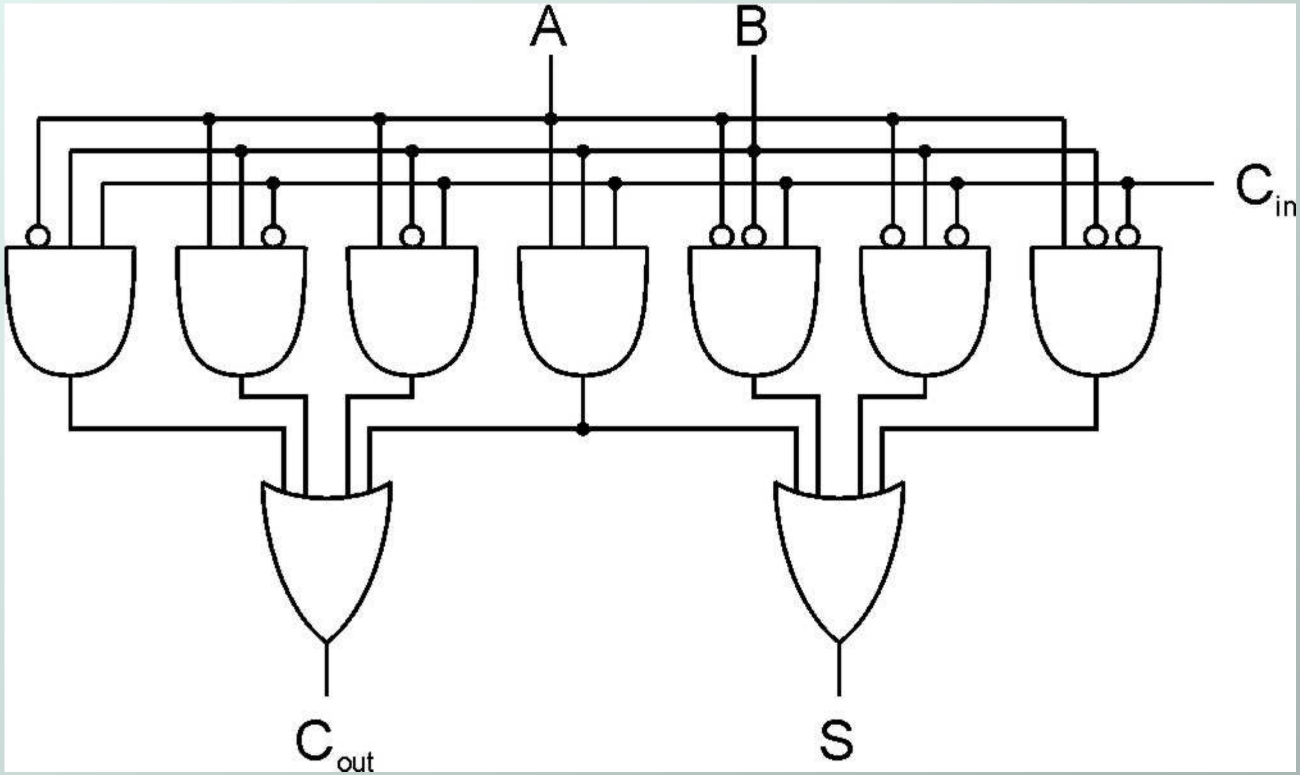
- *n*-bit selector and 2^n inputs, one output
  - output equals one of the inputs, depending on selector



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Full Adder

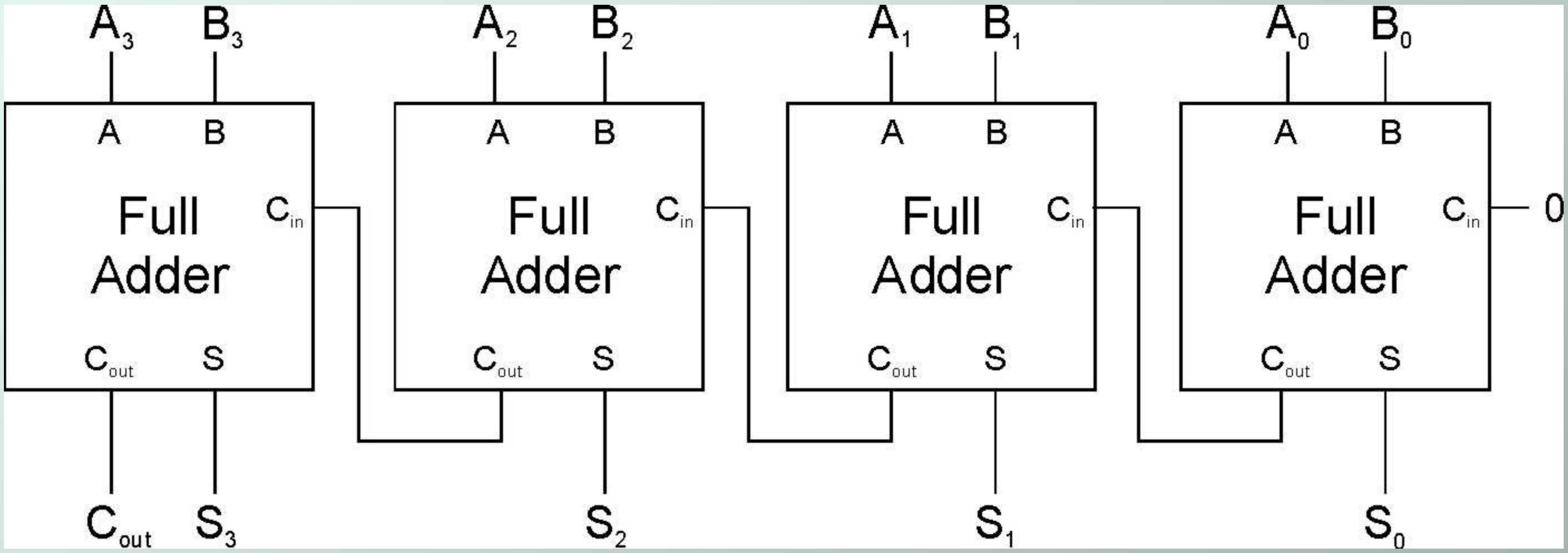
- Add two bits and carry-in, produce one-bit sum and carry-out.



A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

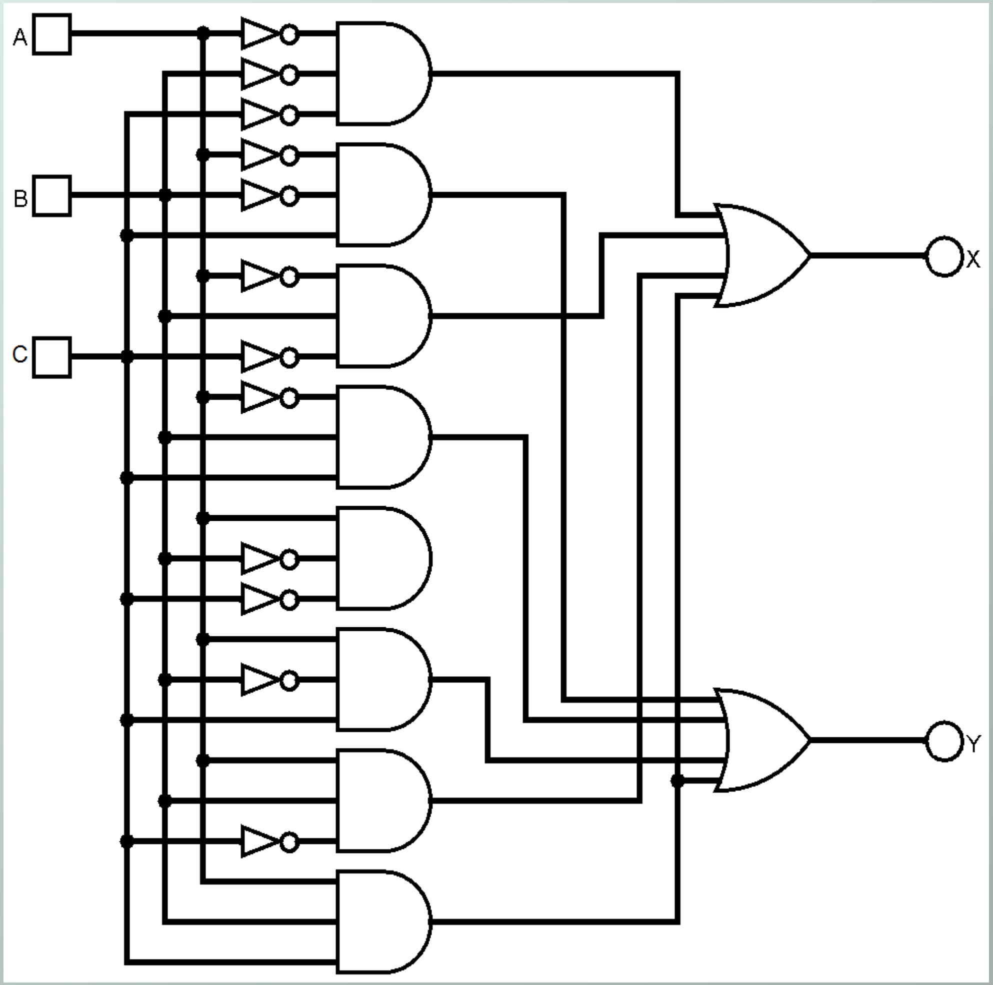
Four-bit Adder



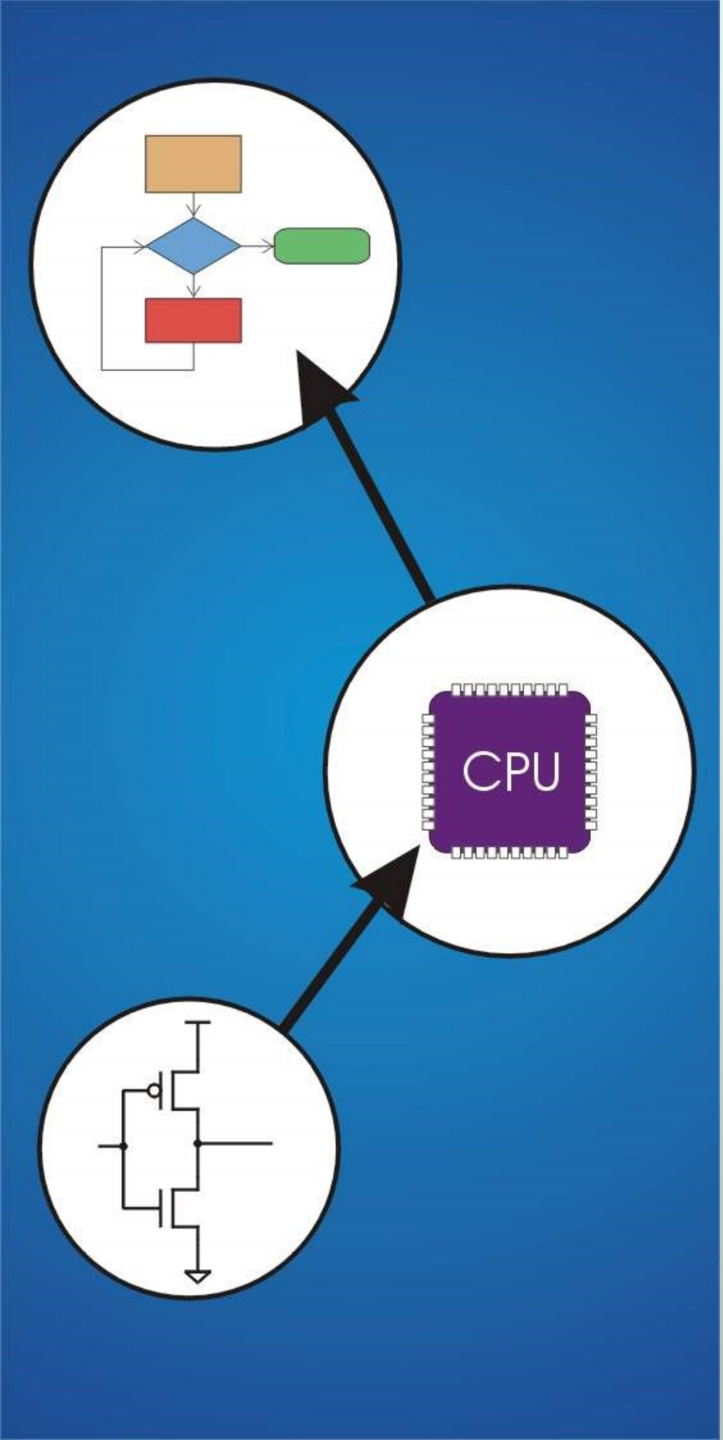
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Programmable Logic Array

- Front end reacts to specific inputs
- Back end defines the outputs
- Any truth table can be built
- Not necessarily minimal circuit!



Requires (at least) ten gates.



Storage Elements & Sequential Circuits

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

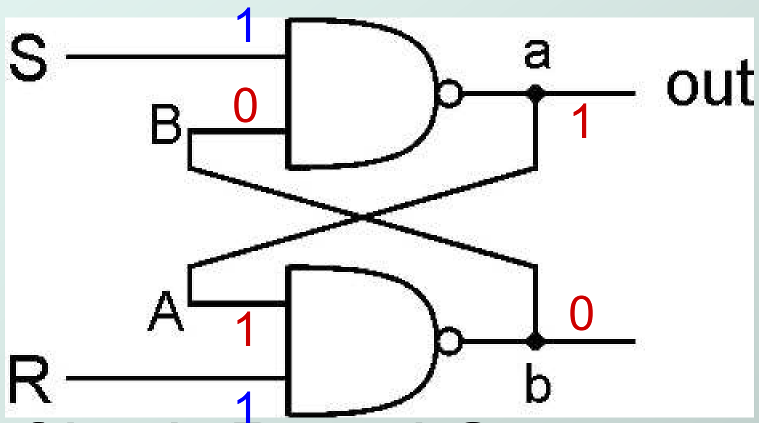
Combinational vs. Sequential

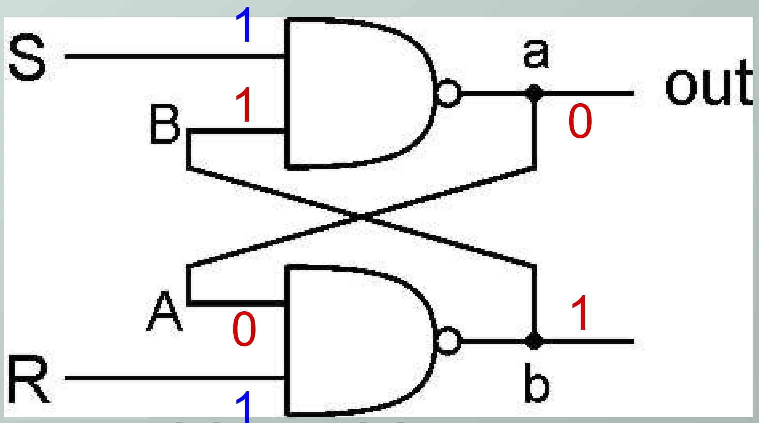
- Combinational Circuit
  - always gives the same output for a given set of inputs
    - ex: adder always generates sum and carry, regardless of previous inputs
- Sequential Circuit
  - stores information
  - output depends on stored information (state) plus input
    - so a given input might produce different outputs, depending on the stored information
  - *example:* ticket counter
    - advances when you push the button

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

R-S Latch: Simple Storage Element

- R is used to "reset" or "clear" the element – set it to zero.
- S is used to "set" the element – set it to one.





- If both R and S are one, out could be <u>either</u> zero or one.
  - "quiescent" state -- holds its previous value
  - note: if a is 1, b is 0, and vice versa

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

R-S Latch Summary

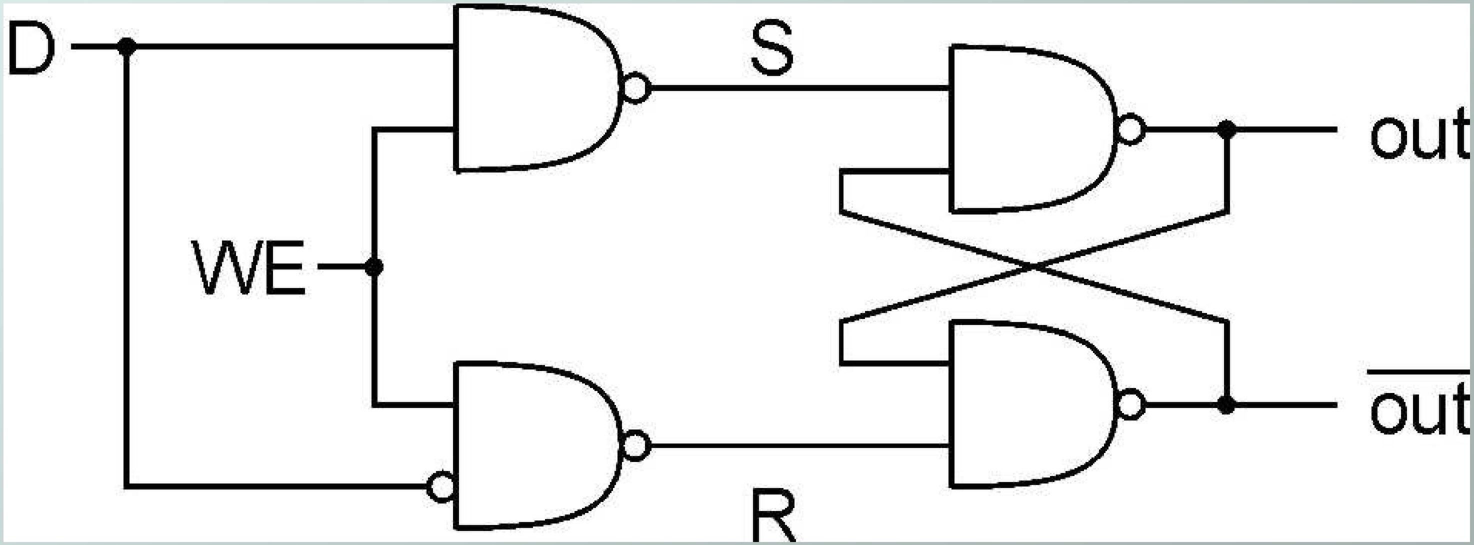
- R = S = 1
  - hold current value in latch
- S = 0, R=1
  - set value to 1
- R = 0, S = 1
  - set value to 0

- R = S = 0
  - both outputs equal one
  - final state determined by electrical properties of gates
  - *Don't do it!*

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

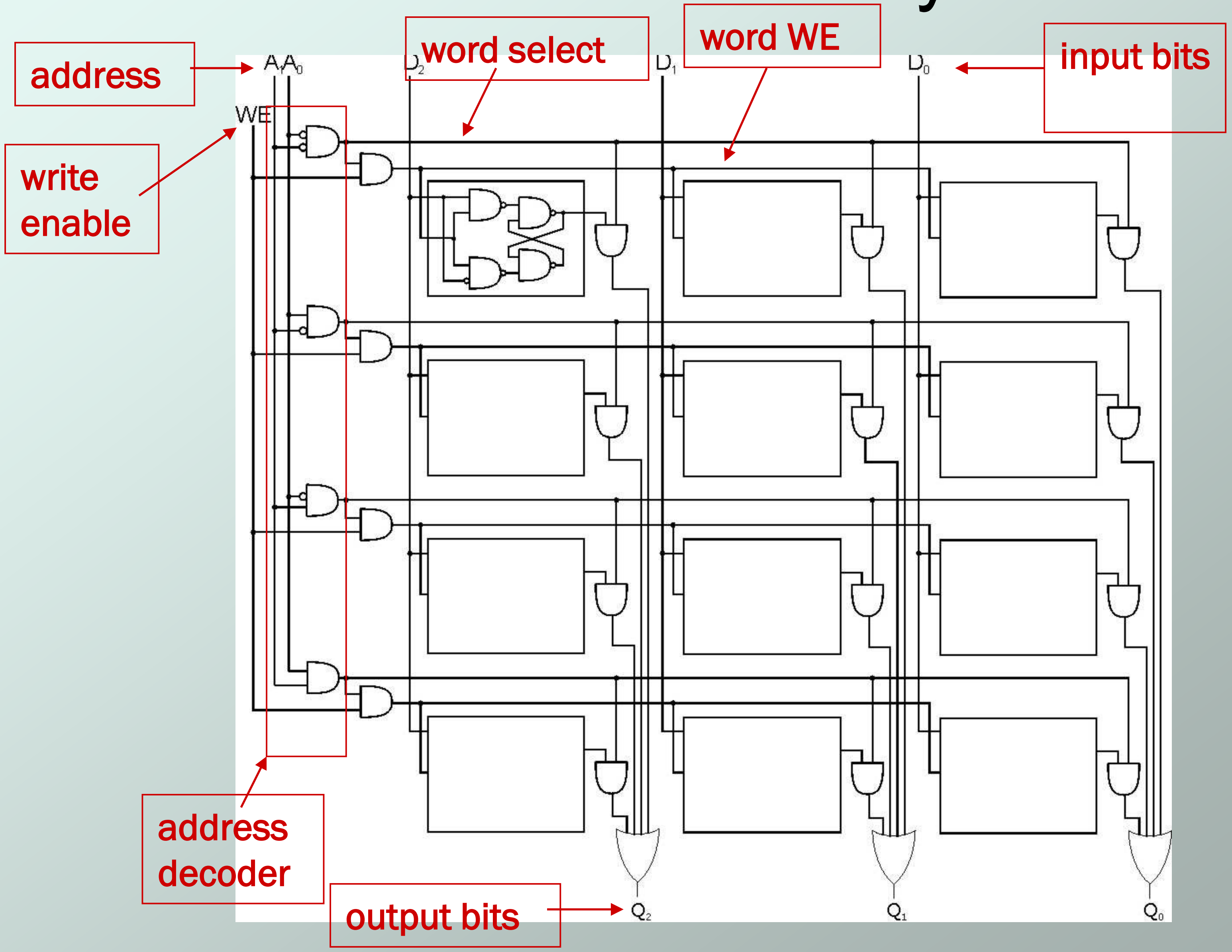
Gated D-Latch

- Two inputs: D (data) and WE (write enable)
  - when WE = 1, latch is set to value of D
    - S = NOT(D), R = D
  - when WE = 0, latch holds previous value



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

2^2 x 3 Memory



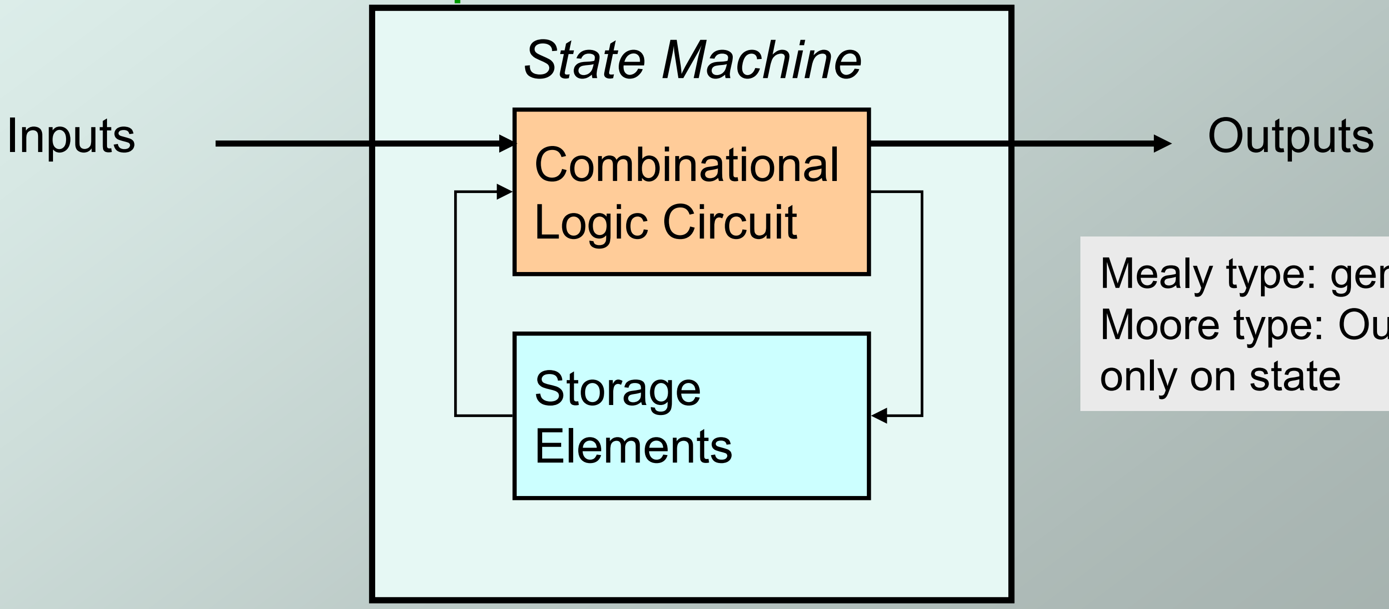
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Finite State Machines

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

State Machine

- A general sequential circuit
  - Combines combinational logic with storage
  - "Remembers" state, and changes output (and state) based on inputs and current state

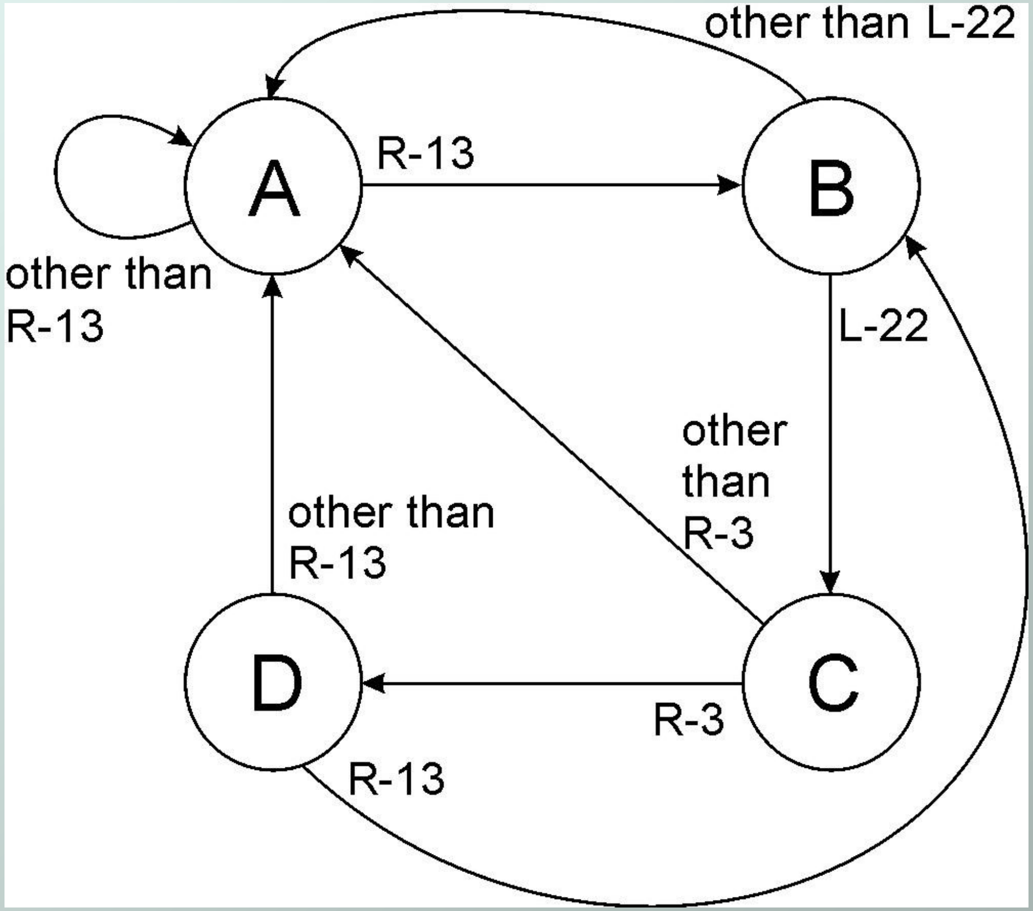


Mealy type: general
Moore type: Output depends only on state

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

State Diagram

- Shows states and actions that cause a transition between states.



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Finite State Machine

- A description of a system with the following components:

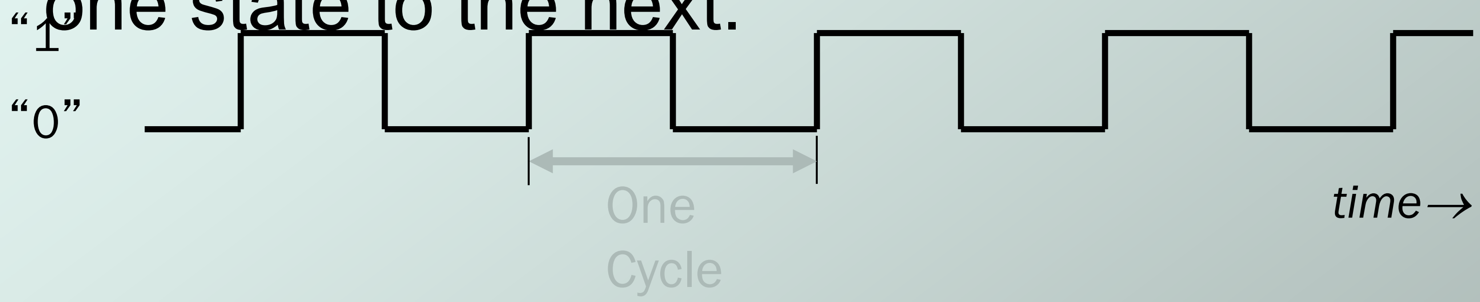
1. A finite number of states
2. A finite number of external inputs
3. A finite number of external outputs
4. An explicit specification of all state transitions
5. An explicit specification of what determines each external output value

- Often described by a state diagram.
  - Inputs trigger state transitions.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

The Clock

- Frequently, a clock circuit triggers transition from one state to the next.

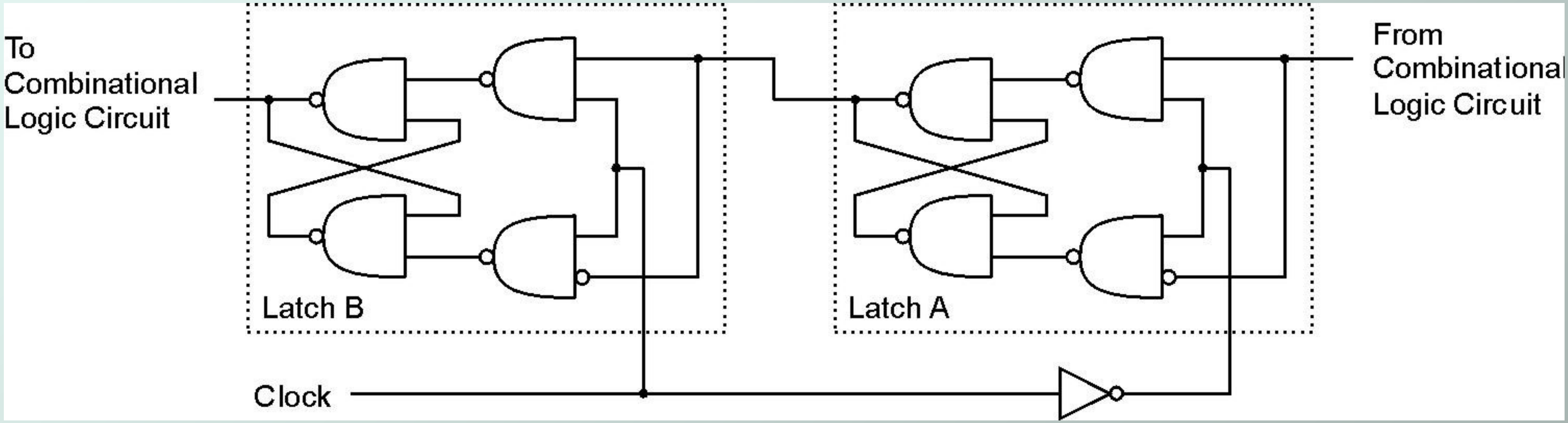


- At the beginning of each clock cycle, state machine makes a transition, based on the current state and the external inputs.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

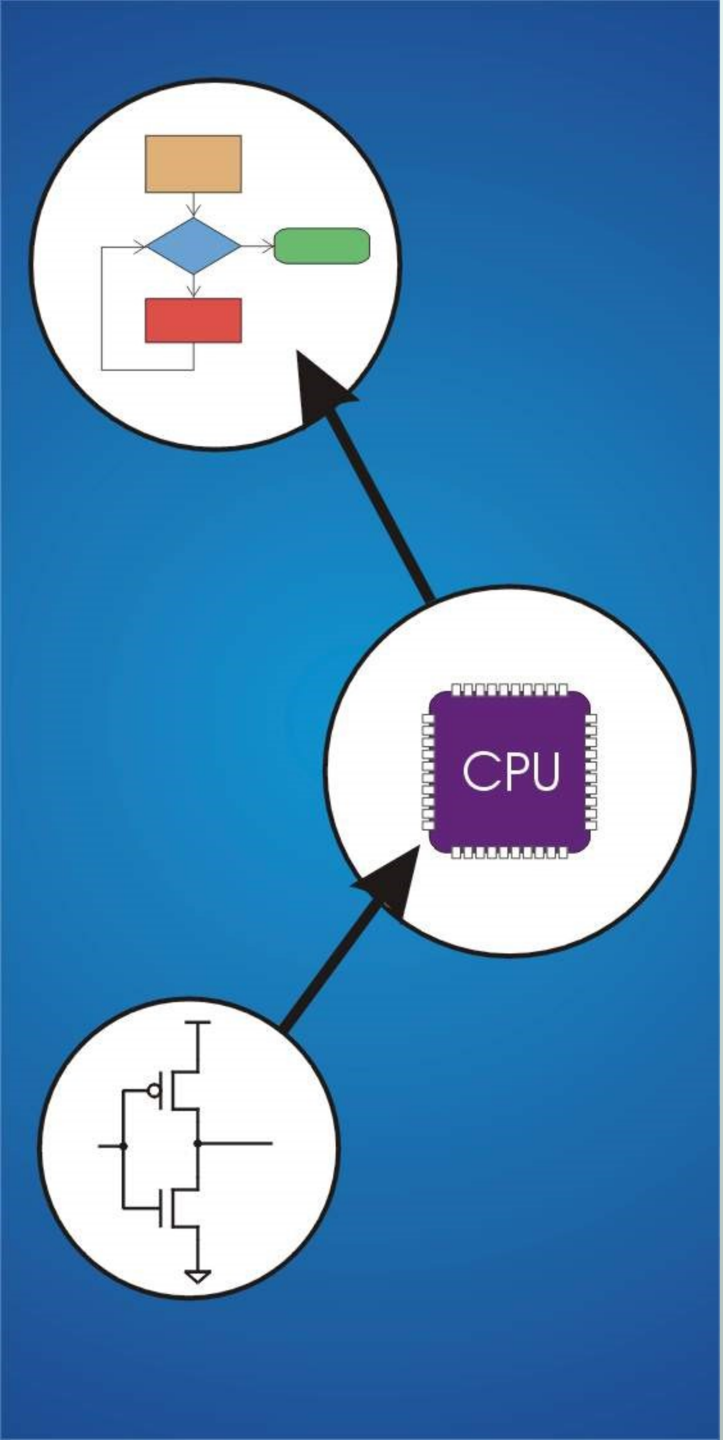
Storage: Master-Slave Flipflop

- A pair of gated D-latches,
  to isolate *next* state from *current* state.


To Combinational Logic Circuit
From Combinational Logic Circuit
Latch B
Latch A
Clock

During 1st phase (clock=1), previously-computed state becomes *current* state and is sent to the logic circuit.

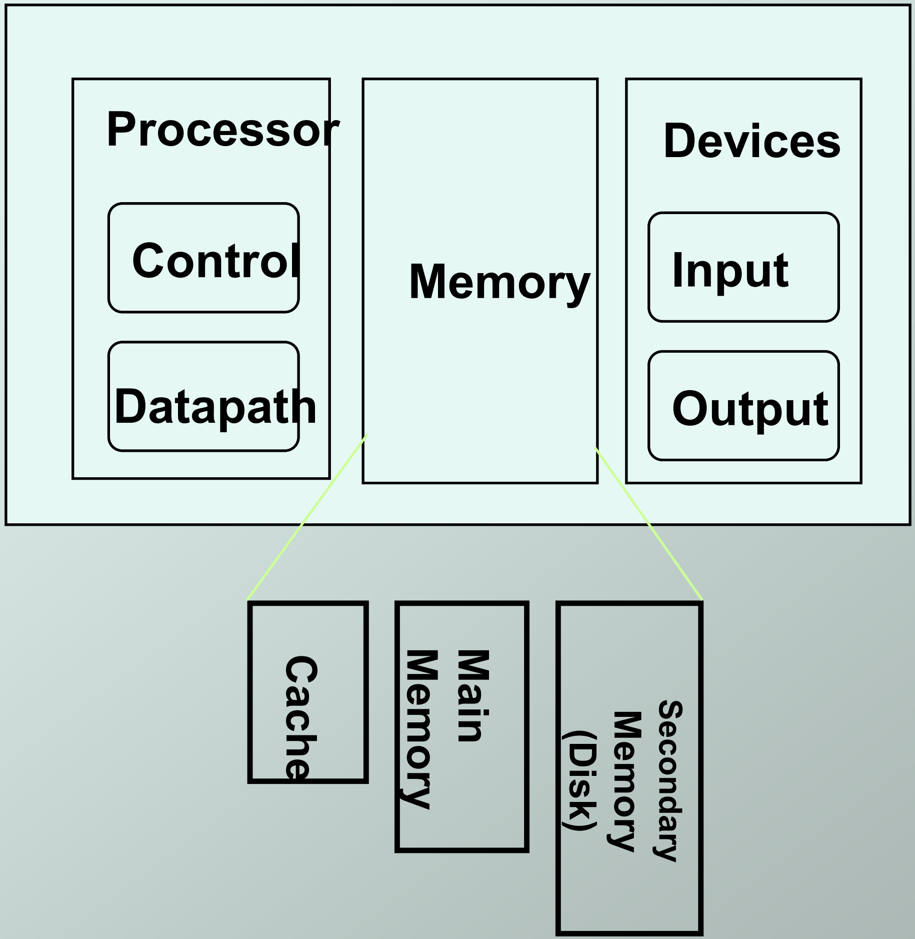
During 2nd phase (clock=0), *next* state, computed by logic circuit, is stored in Latch A.



Memory Hierarchy

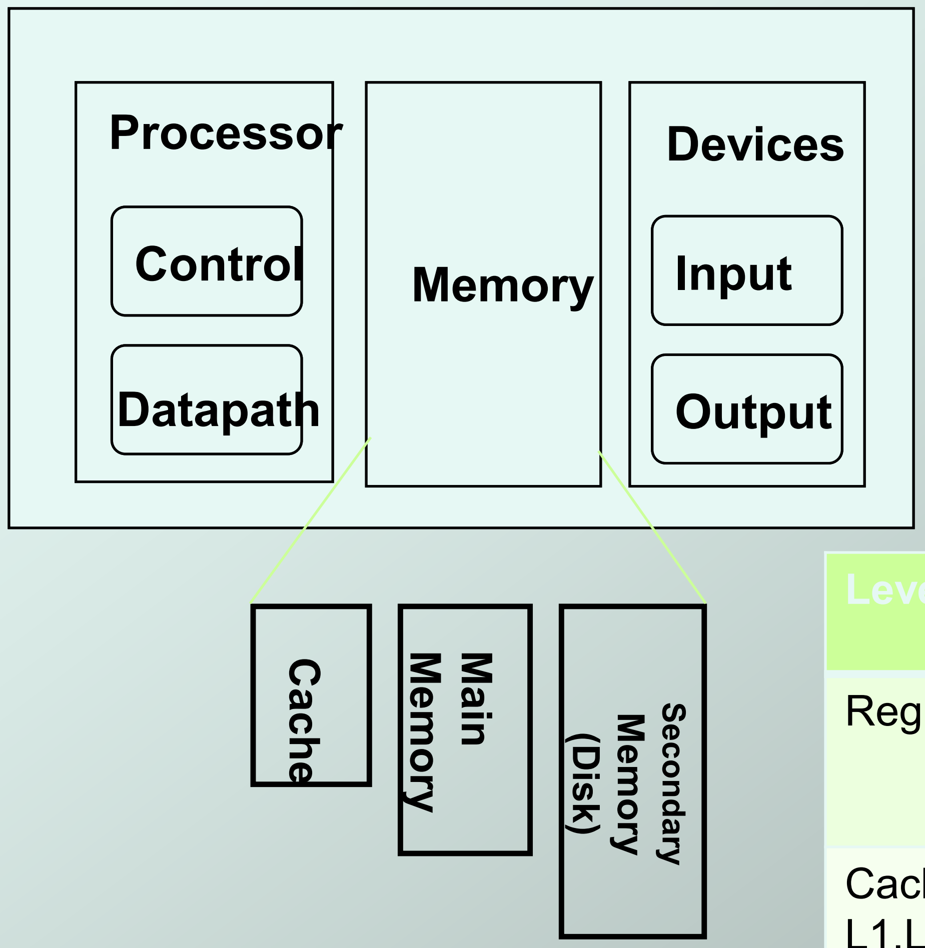
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Review: Major Components of a Computer



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Review: Major Components of a Computer



Level	Access time	Size	Cost/GB
Registers	0.25 ns	48+ 64,128, 32b	-
Cache L1,L2,L3	0.5 ns	5MB	125
Memory	1000 ns	8GB	4.0
SSD	100K ns		0.25
Disk	1000K ns	1 TB	0.02

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

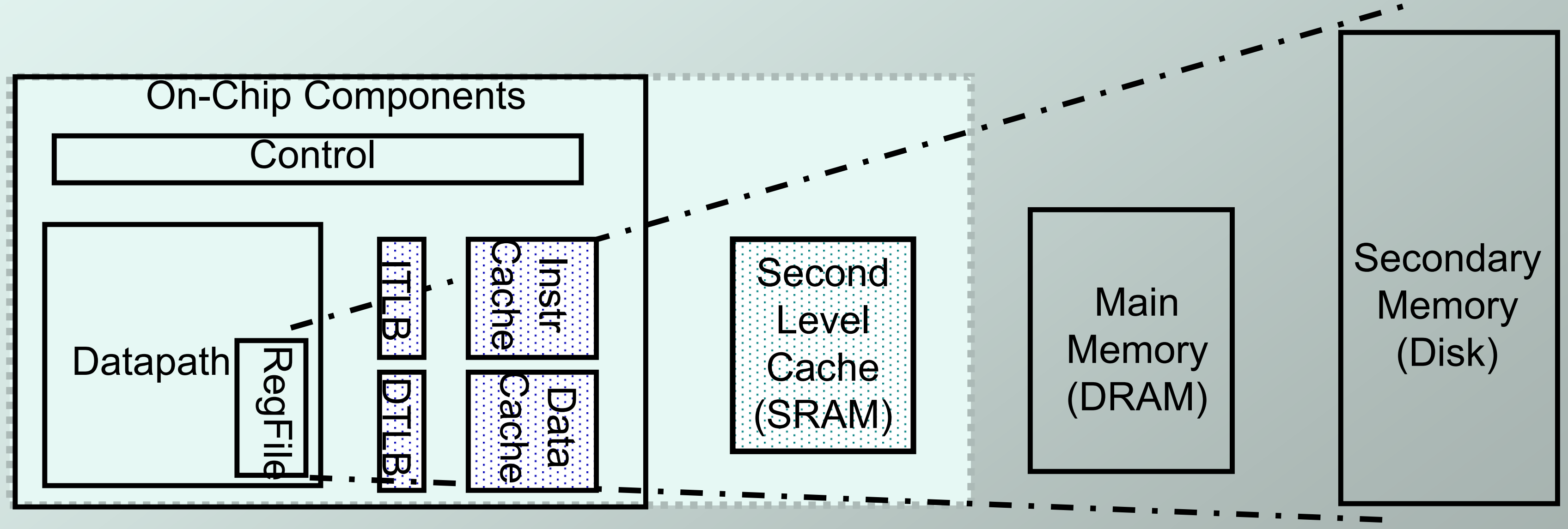
The Memory Hierarchy: Key facts and ideas (1)

- Programs keep getting bigger exponentially.
- Memory cost /bit
  - Faster technologies are expensive, slower are cheaper. Different by orders of magnitude
  - With time storage density goes up driving per bit cost down.
- Locality in program execution
  - Code/data used recently will likely be needed soon.
  - Code/data that is near the one recently used, will likely be needed soon.

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

A Typical Memory Hierarchy

- Take advantage of the principle of locality to present the user with as much memory as is available in the *cheapest* technology at the speed offered by the *fastest* technology



Speed (%cycles):	½'s	1's	10's	100's	10,000's
Size (bytes):	100's	10K's	M's	G's	T's
Cost:	highest				lowest

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Principle of Locality

- Programs access a small proportion of their address space at any time
- Temporal locality
  - Items accessed recently are likely to be accessed again soon
  - e.g., instructions in a loop, induction variables
- Spatial locality
  - Items near those accessed recently are likely to be accessed soon
  - E.g., sequential instruction access, array data

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Cache Misses

- On cache hit, CPU proceeds normally
- On cache miss
  - Stall the CPU pipeline
  - Fetch block from next level of hierarchy
  - Instruction cache miss
    - Restart instruction fetch
  - Data cache miss
    - Complete data access

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Multilevel Caches

- Primary cache attached to CPU
  - Small, but fast
- Level-2 cache services misses from primary cache
  - Larger, slower, but still faster than main memory
- Main memory services L-2 cache misses
- Some systems now include L-3 cache

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Virtual Memory

- Use main memory as a “cache” for secondary (disk) storage
  - Managed jointly by CPU hardware and the operating system (OS)
- Programs share main memory
  - Each gets a private virtual address space holding its frequently used code and data
  - Protected from other programs
- CPU and OS translate virtual addresses to physical addresses
  - VM “block” is called a page
  - VM translation “miss” is called a page fault

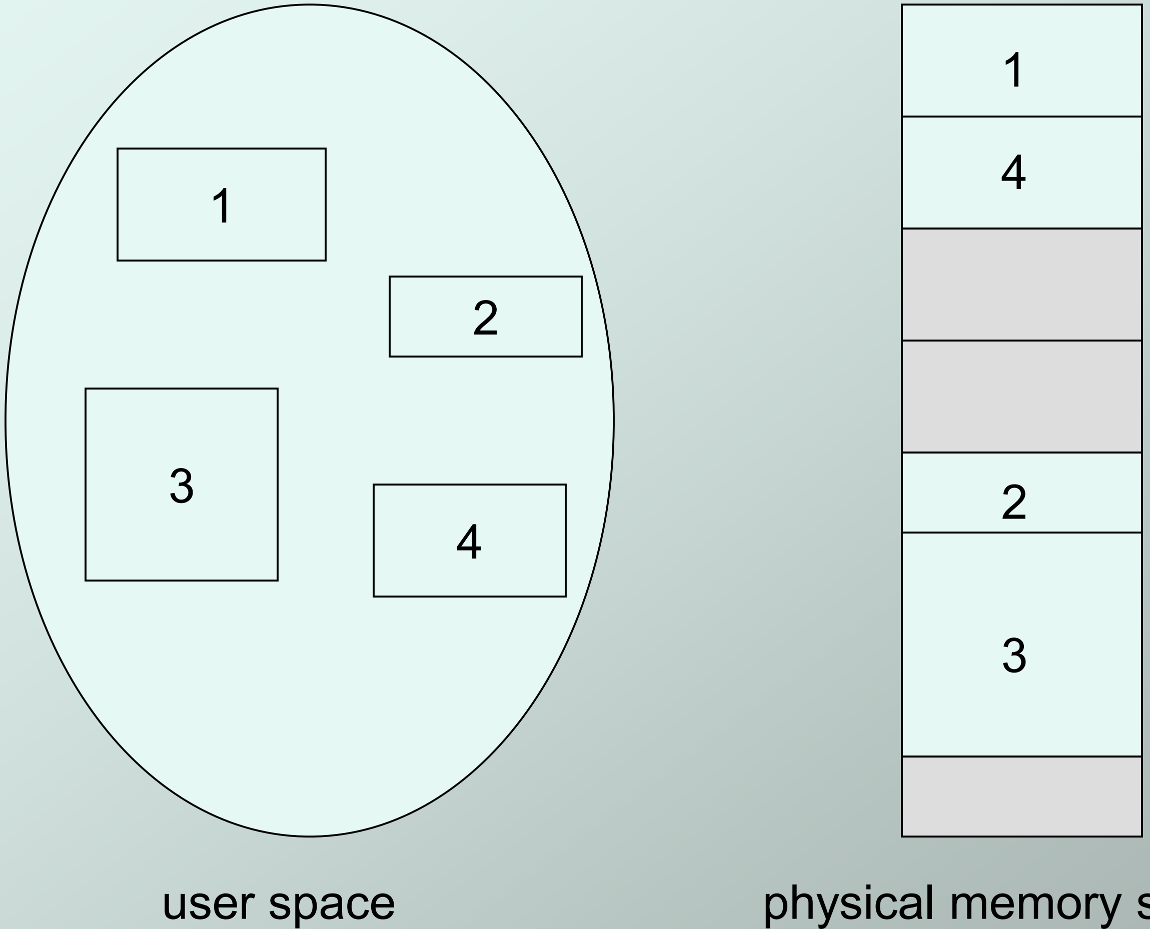
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Virtual vs. Physical Address

- Processor assumes a certain memory addressing scheme:
  - A block of data is called a virtual page
  - An address is called virtual (or logical) address
- Main memory may have a different addressing scheme:
  - Real memory address is called a physical address, MMU translates virtual address to physical address
  - Complete address translation table is large and must therefore reside in main memory
  - MMU contains TLB (translation lookaside buffer), which is a small cache of the address translation table

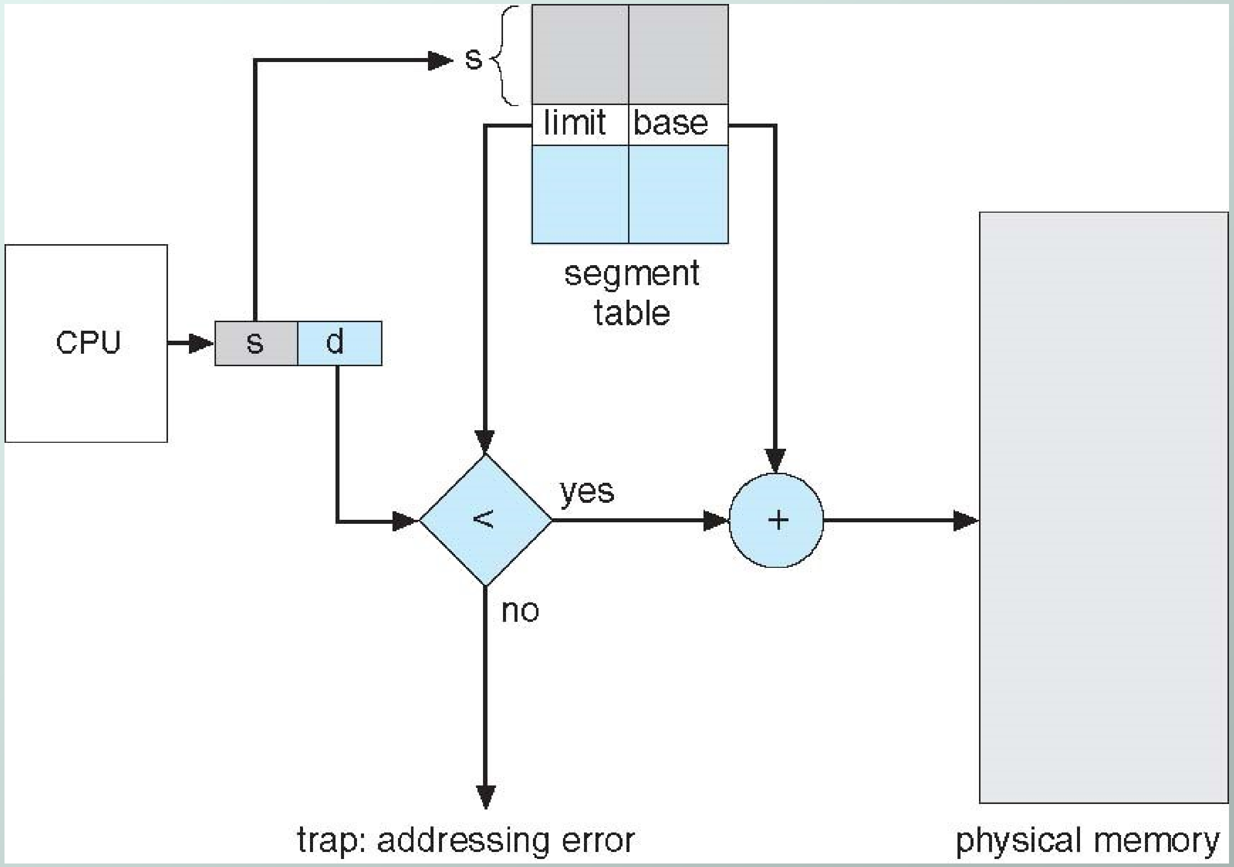
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Logical View of Segmentation



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Segmentation Hardware



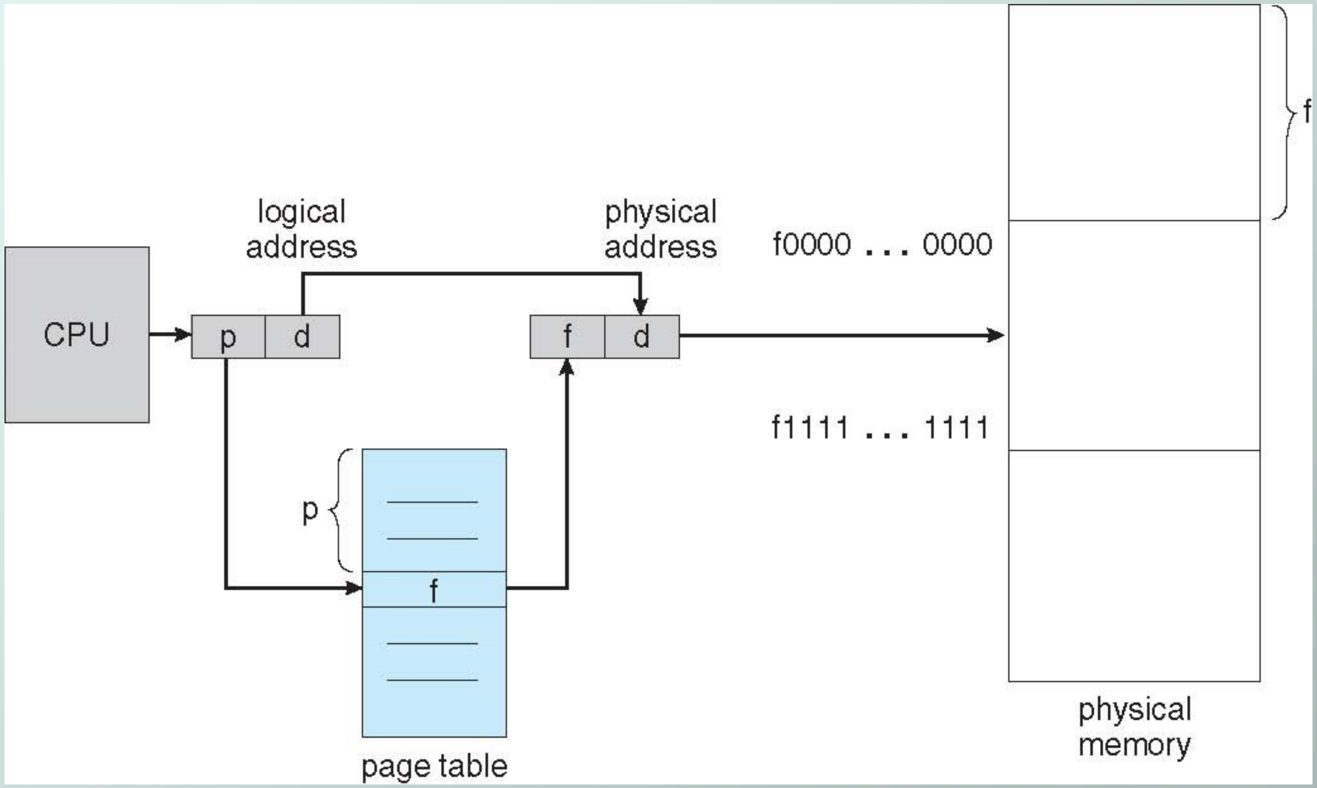
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Paging

- Physical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available
  - Avoids external fragmentation
  - Avoids problem of varying sized memory chunks
- Divide physical memory into fixed-sized blocks called **frames**
  - Size is power of 2, between 512 bytes and 16 Mbytes
- Divide logical memory into blocks of same size called **pages**

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Paging Hardware



Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Page Fault Penalty

- On page fault, the page must be fetched from disk
  - Takes millions of clock cycles
  - Handled by OS code
- Try to minimize page fault rate
  - Smart replacement algorithms

Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

3-Level Cache Organization

	Intel Nehalem	AMD Opteron X4
L1 caches (per core)	L1 I-cache: 32KB, 64-byte blocks, 4-way, approx LRU replacement, hit time n/a L1 D-cache: 32KB, 64-byte blocks, 8-way, approx LRU replacement, write-back/allocate, hit time n/a	L1 I-cache: 32KB, 64-byte blocks, 2-way, LRU replacement, hit time 3 cycles L1 D-cache: 32KB, 64-byte blocks, 2-way, LRU replacement, write-back/allocate, hit time 9 cycles
L2 unified cache (per core)	256KB, 64-byte blocks, 8-way, approx LRU replacement, write-back/allocate, hit time n/a	512KB, 64-byte blocks, 16-way, approx LRU replacement, write-back/allocate, hit time n/a
L3 unified cache (shared)	8MB, 64-byte blocks, 16-way, replacement n/a, write-back/allocate, hit time n/a	2MB, 64-byte blocks, 32-way, replace block shared by fewest cores, write-back/allocate, hit time 32 cycles

n/a: data not available