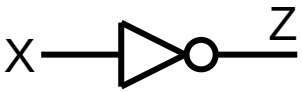
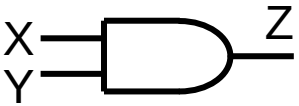
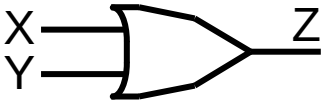
Logic Gates and Typical Functional Blocks

Lecturer: Annie Guo

Logic Gates

- Virtually all problems can be solved by digital circuits and systems
- The basic elements of digital circuits are logic gates
- Logic gates
  - ideally have signals of two levels: high and low
  - perform logic functions, such as NOT, AND, OR, NAND, NOR
- Logic gates can be represented by symbols and their functions can be described using truth tables or logic expressions.
- Some typical examples are given in the next three slides.

NOT, AND & OR Gates

	symbol	*expression*	*function*	
NOT		$Z = \overline{X}$		
AND		$Z = X \bullet Y$		
OR		$Z = X + Y$		

X	Z
0	1
1	0

X	Z
low	high
high	low

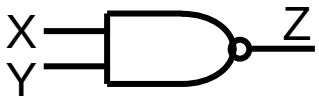
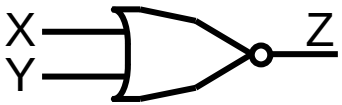
X	Y	X•Y
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	X•Y
low	low	low
low	high	low
high	low	low
high	high	high

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	X+Y
low	low	low
low	high	high
high	low	high
high	high	high

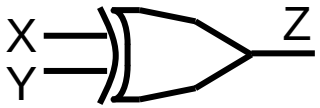
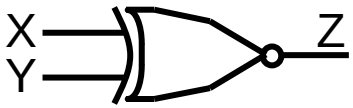
NAND & NOR Gates

	symbol	*expression*	*function*
NAND		$Z = \overline{X \bullet Y}$	
NOR		$Z = \overline{X+Y}$	

X	Y	$\overline{X \bullet Y}$
0	0	1
0	1	1
1	0	1
1	1	0

X	Y	$\overline{X+Y}$
0	0	1
0	1	0
1	0	0
1	1	0

XOR & XNOR Gates

	symbol	*expression*	*function*
XOR		$Z = X \oplus Y$	
XNOR		$Z = \overline{X \oplus Y}$	

X	Y	$X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

X	Y	$\overline{X \oplus Y}$
0	0	1
0	1	0
1	0	0
1	1	1

Functional Blocks

- With basic logic gates we can build up different functional blocks such as
  - Adders
  - Multiplexers
  - Decoders
  - Latches
  - Registers
  - Counters

Adders (1/3)

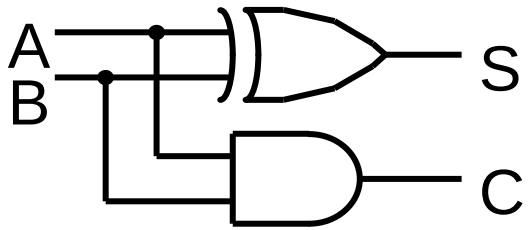
- One bit adder
  - Truth table
  - Logic function

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

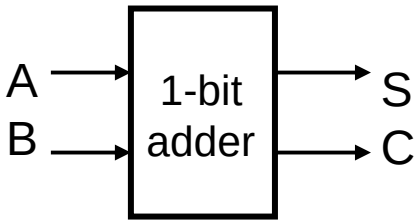
Sum: $S = A \oplus B$

Carry: $C = A \bullet B$

  - Digital circuit

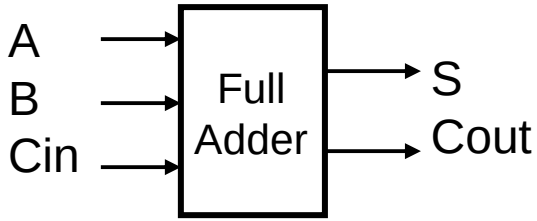


  - Symbol



Adders (2/3)

- One bit adder with carry
  - Called Full Adder
  - Symbol

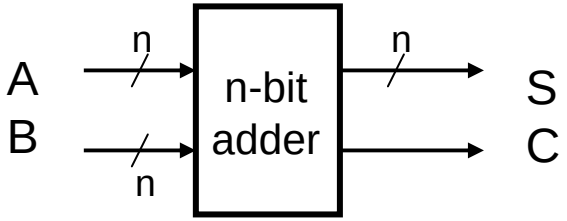


  - Function
    - Adding three 1-bit numbers

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

Adders (3/3)

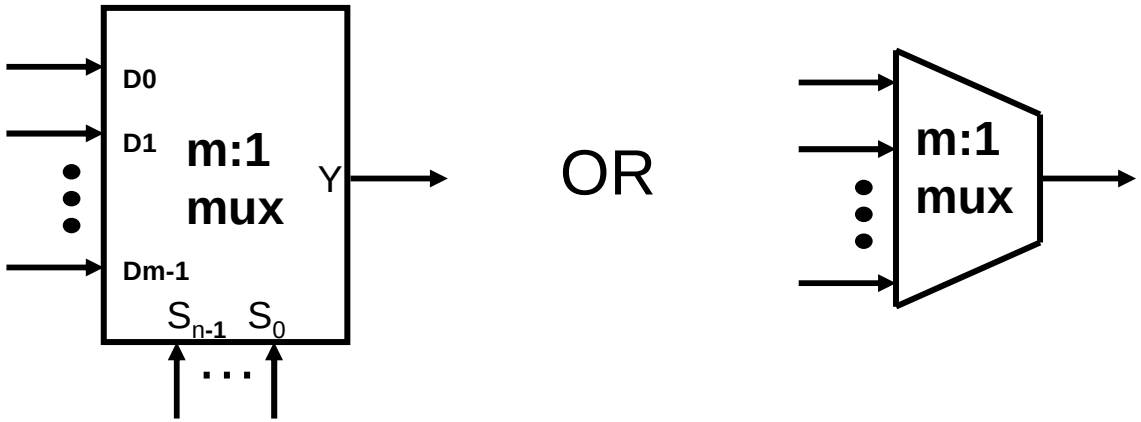
- n-bit adder
  - Symbol



  - Function
    - Adding two n-bit numbers
      - The result is the n-bit sum and 1-bit carry

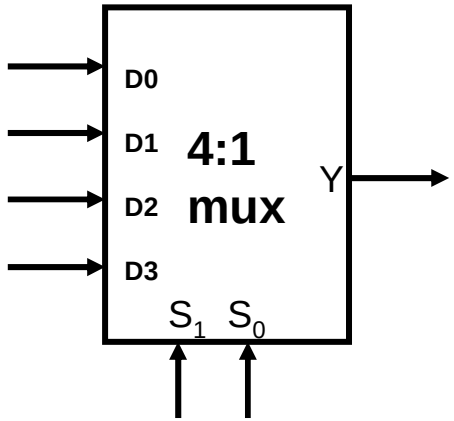
Multiplexers

- Function:
  - A multiplexer selects one input among multiple inputs and passes it to output.
    - The selection is controlled by control signal S_{n-1} ~S_0
- The symbol:



Example

- 4:1 multiplexer

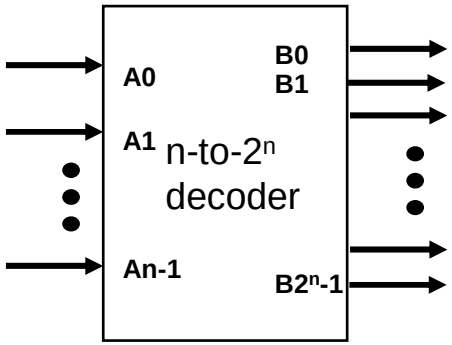


- Function:

When S_1S_0 = 00, Y=D0
When S_1S_0 = 01, Y=D1
When S_1S_0 = 10, Y=D2
When S_1S_0 = 11, Y=D3

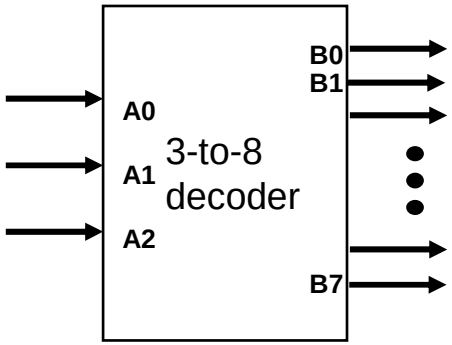
Decoders

- Function:
  - A decoder converts an n-bit input code to an m-bit output code
    - $n \leq m \leq 2^n$
    - each valid input code word produces a unique output code
  - Typical n-to-2^n decoder
    - One line of outputs represents a specific input combination
  - The symbol:



Example

- 3-to-8 register file address decoder

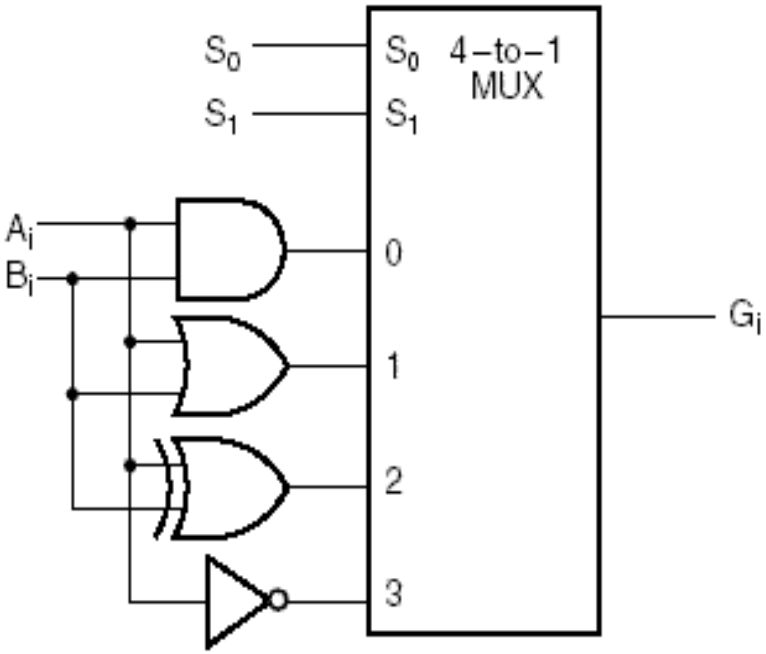


- Function:

Address			Output							
A2	A1	A0	B0	B1	B2	B3	B4	B5	B6	B7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Multi-operation Unit

- Perform following1-bit logic operations:
  - AND, OR, XOR, NOT



(a) Logic Diagram

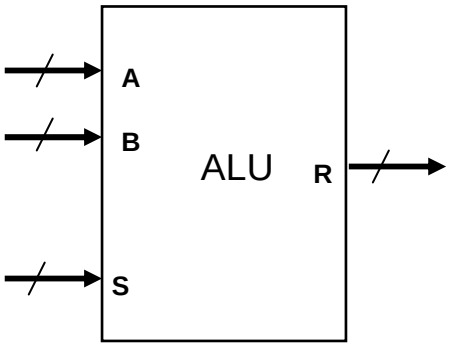
S_1	S_0	Output	Operation
0	0	$G = A \wedge B$	AND
0	1	$G = A \vee B$	OR
1	0	$G = A \oplus B$	XOR
1	1	$G = \overline{A}$	NOT

(b) Function Table

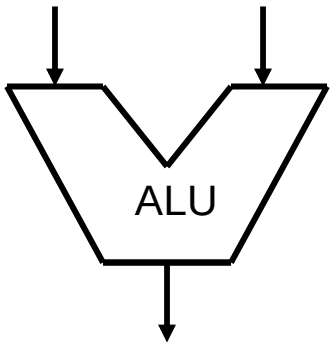
Constructed with functional components

ALU

- Perform arithmetic and logic operations
  - such as addition, subtraction, logic AND, logic OR
- Symbol:
  - A, B are operands, S selects one of operations that can be performed by ALU



OR

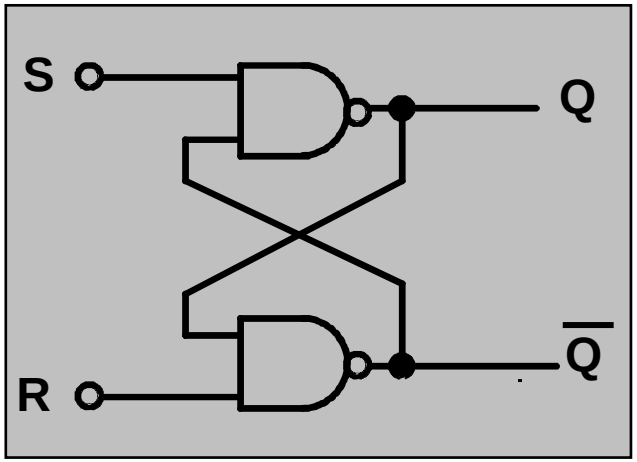


Example

Operation selection S2 S1 S0	Operation
0 0 0	Addition
0 0 1	Subtraction
0 1 0	AND
0 1 1	OR
1 0 0	XOR
1 0 1	NOT
1 1 0	Increment
1 1 1	Transfer

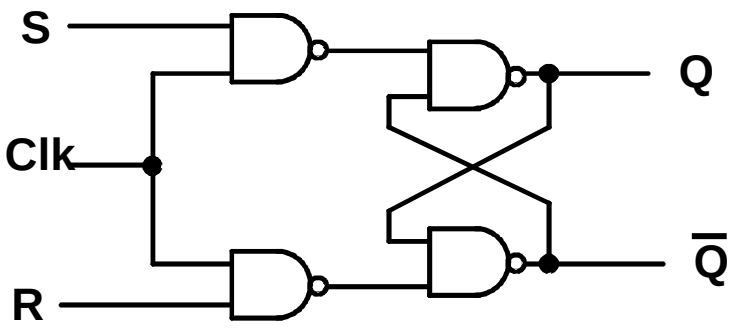
Latches and Flip Flops (1/3)

- Can be constructed in many ways.
- 2-NAND-gate latch (1 bit)
  - R=0, reset the latch
  - S=0, set latch
  - S = R = 1, the data is retained



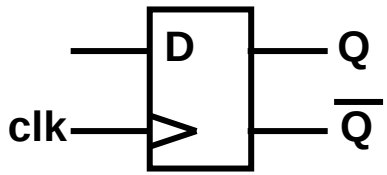
Latches and Flip Flops (2/3)

- Clocked latch uses clock to control the latch operation
  - When Clk=1,
    - S=1, set the latch
    - R=1, reset the latch
    - S=R=0, the data is retained
  - When Clk=0,
    - Data is retained



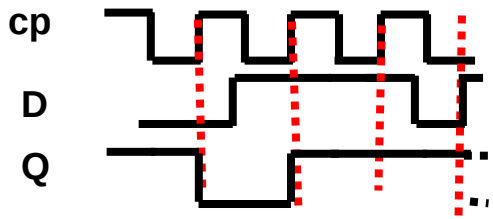
Latches and Flip Flops (3/3)

- Flip Flops use clock edges to trigger the data-store operation.
  - A very commonly used Flip Flop is D FF
    - On the rise edge of clock, the input data D is locked into the D flip flop



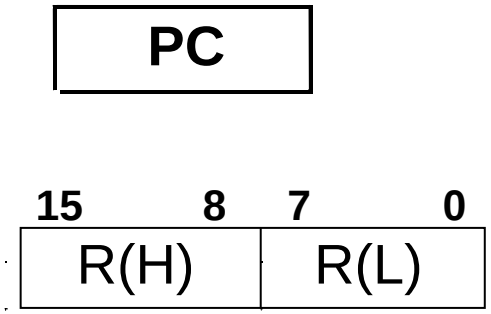
D	Q(n+1)
0	0
1	1

  - Timing diagram



Registers (1/3)

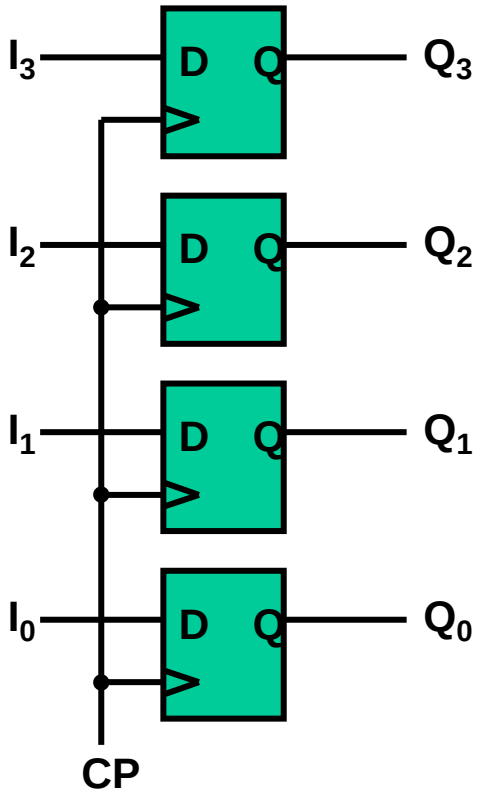
- A register is a collection of latches/FFs
  - storing a vector of bit values
- Symbol



Registers (2/3)

- 4-bit Parallel In Parallel Out (PIPO) registers.

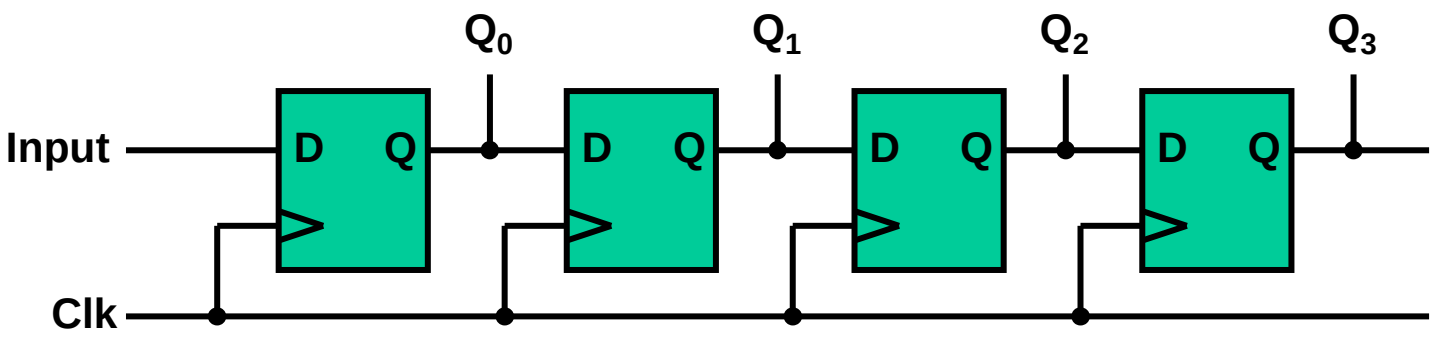
The 4-bit input $\mathbf{I_3I_2I_1I_0}$ is "loaded" (copied to the output $\mathbf{Q_3Q_2Q_1Q_0}$ of the **D FFs**) on the rising clock edge, and that output is held until the next clock edge.



Registers (3/3)

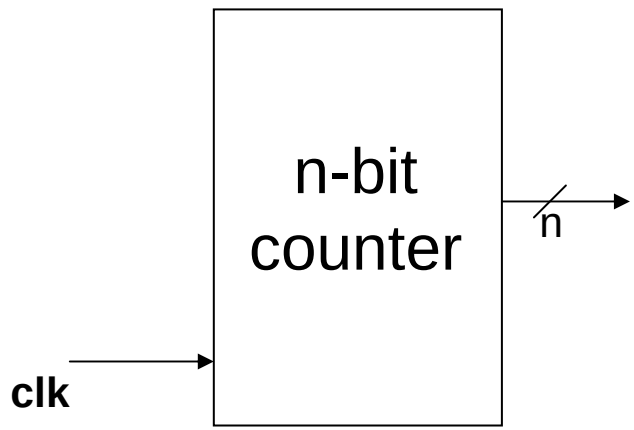
- 4-bit Serial In Parallel Out (SIPO) registers.

On the clock edge, the output of each flip-flop is passed to the next flip-flop in the chain. The input signal is fed serially (one bit at a time) into the first flip-flop. The flip-flop outputs are available in parallel.



Counters (1/2)

- A counter increases/decrease its value every clock cycle.
- Symbol



Counters (2/2)

- 4-bit counter

The counter has

- a **synchronous load**
- an **asynchronous clear**

The counter counts through 0, 1, 2, …, 15, 0