3. Combinational Circuits II

Aims

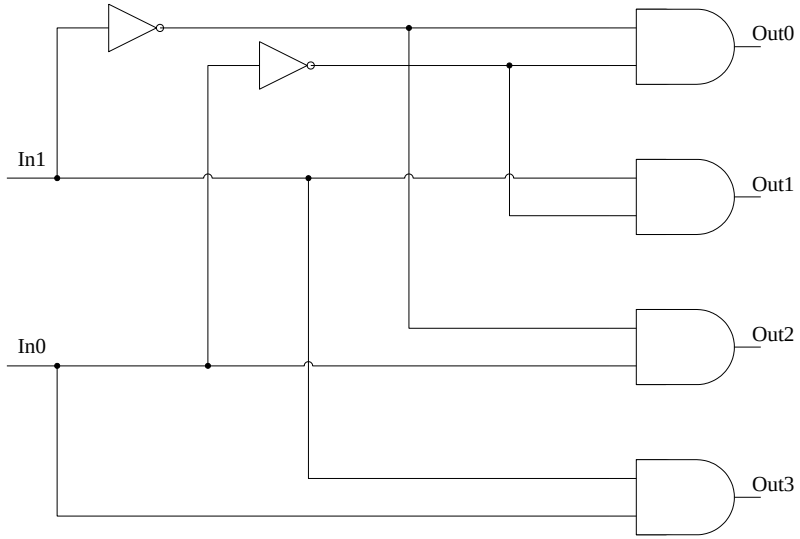
- to show how logic gates can be combined to produce useful circuits such as decoders and adders.

- to show how individual combinational circuits can be combined to produce a simple ALU;

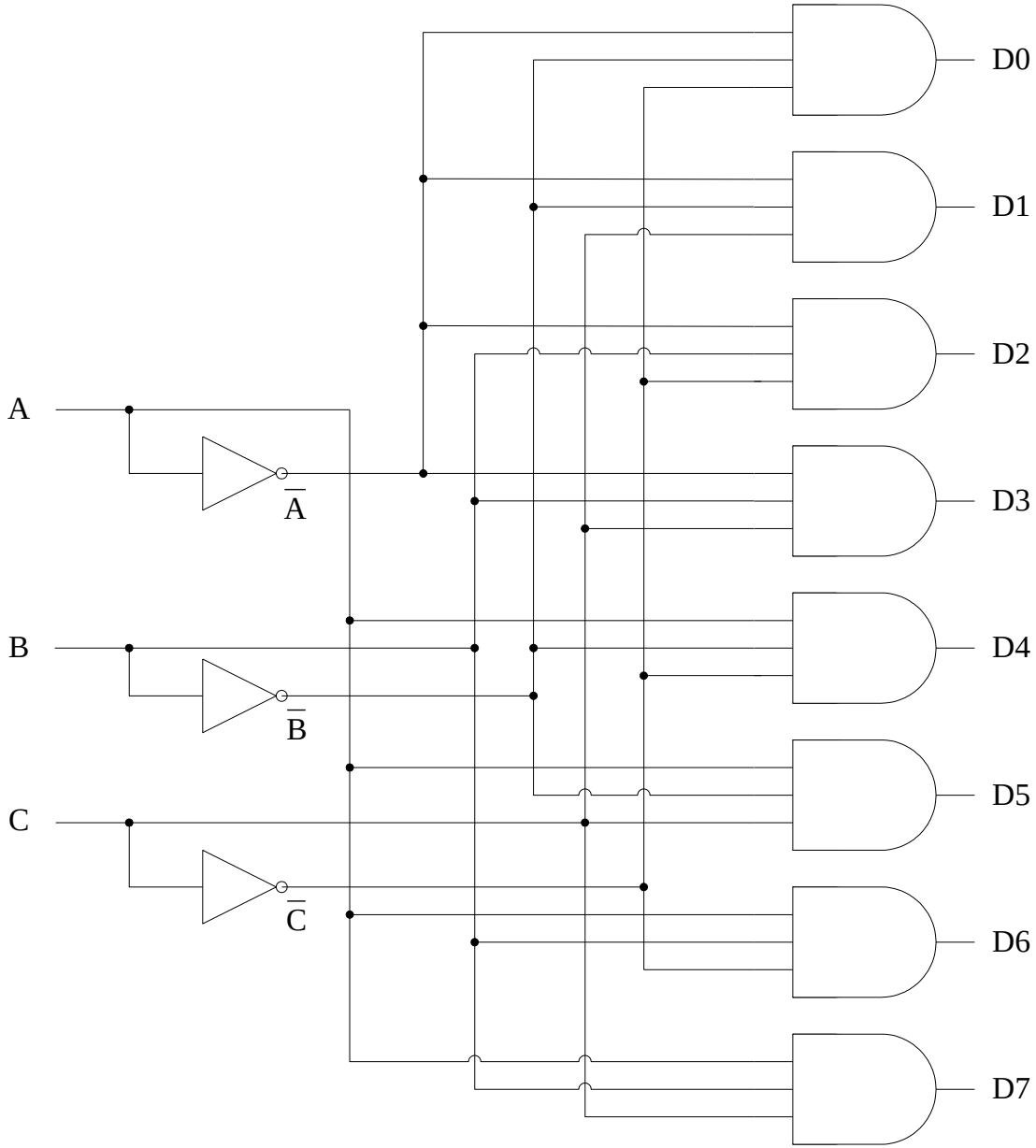
- to introduce a simple memory storage device called a register.

3.1 Decoders

- A decoder is a combinational circuit that converts binary information from *n* inputs to a maximum of 2^n unique outputs. A decoder takes an *n*-bit number as input and uses it to select (set to 1) exactly one of the 2^n outputs.



In1	In0	$\overline{In0}$	$\overline{In1}$	Out0	Out1	Out2	Out3
0	0						
0	1						
1	0						
1	1						


A
B
C
$\overline{A}$
$\overline{B}$
$\overline{C}$
D0
D1
D2
D3
D4
D5
D6
D7

- For each possible input combination there are seven outputs that are equal to 0 and only one that is equal to 1. The output equal to 1 represents the equivalent of the binary number that is applied to the inputs.

A	B	C	D0	D1	D2	D3	D4	D5	D6	D7
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

- Application: *device selector*.

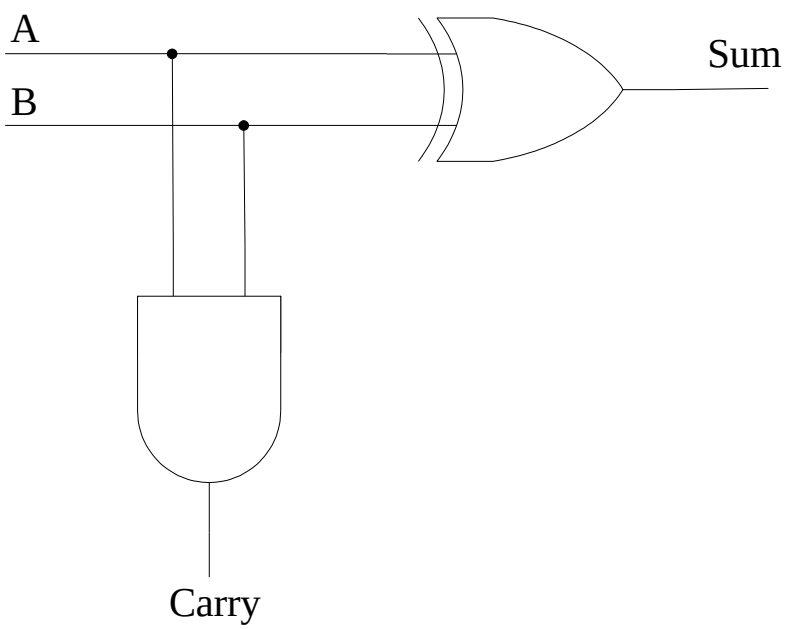
3.2 Half Adder

- A half adder is an arithmetic circuit that performs the addition of two bits. The circuit has two inputs (A, B) and two outputs (Sum, Carry). Two outputs are necessary because the sum of two binary digits ranges from 0 to 2, and the binary equivalent of 2 requires two digits. The sum output represents the sum of the two inputs while the carry output represents the carry to the next leftward position.

- The simple addition performed by a half adder can consist of four possible operations:

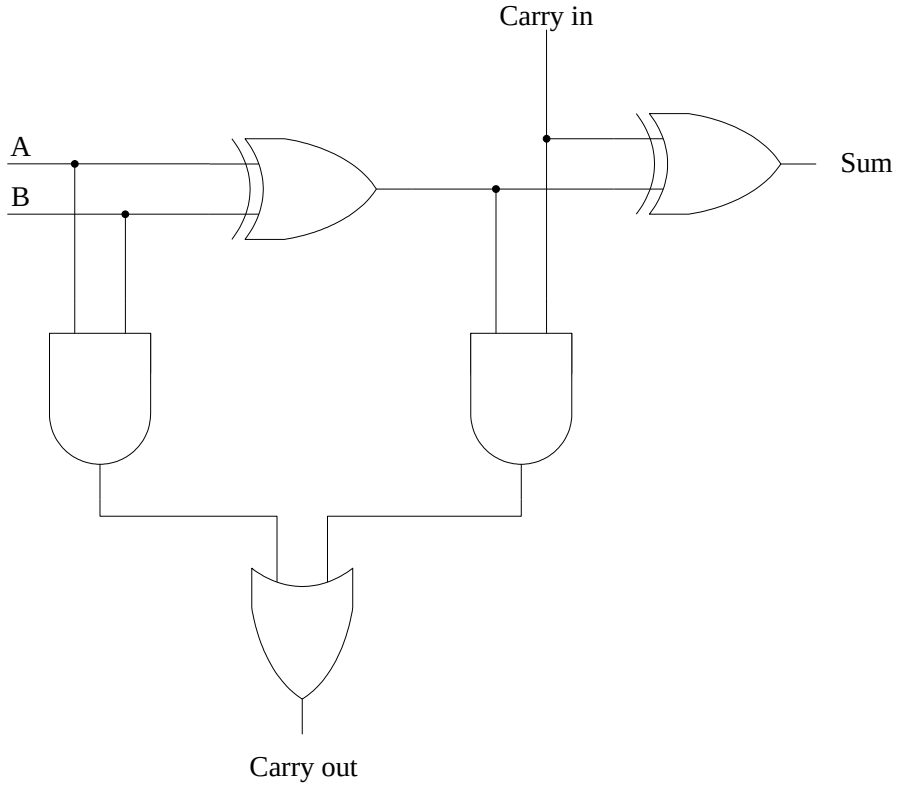
$$0+0 = 0$$
$$0+1 = 1$$
$$1+0 = 1$$
$$1+1 = 10$$

A	B	Carry	Sum
0	0		
0	1		
1	0		
1	1		



3.3 Full Adder

- A full adder is a circuit that performs the addition of three bits. The name of the circuit stems from the fact that two half adders are joined together to create a full adder. The circuit has three inputs (A, B, Carry in) and two outputs (Sum, Carry out). Two outputs are necessary because the sum of three binary digits ranges from 0 to 3, and the binary equivalent of 2 or 3 requires two digits.

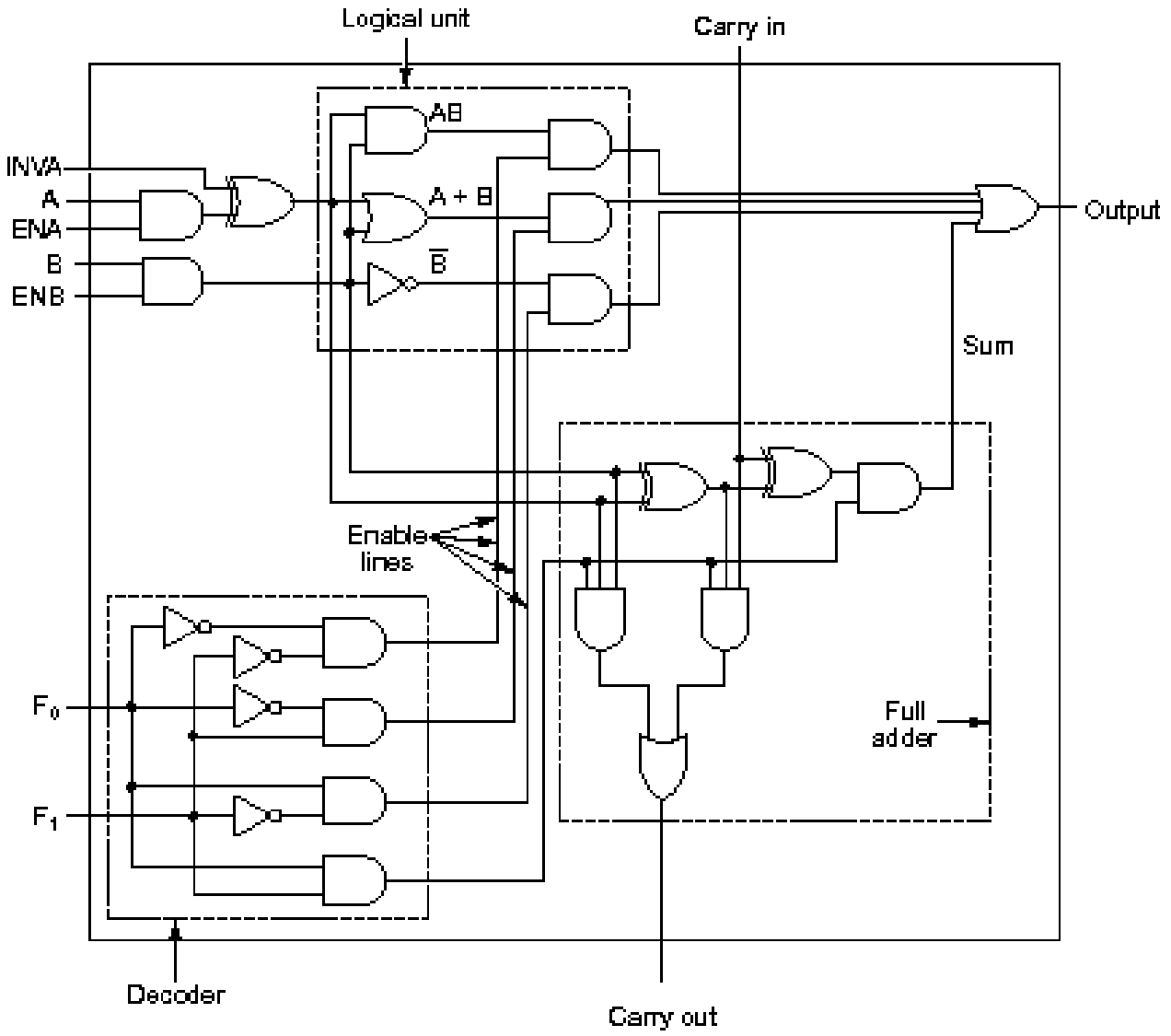


A	B	Carry in	Carry out	Sum
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

- To build an adder for two 16-bit numbers the full adder is replicated sixteen times. The carry out of one full adder is used as the carry in into its left neighbour. The carry into the rightmost bit is set to 0.

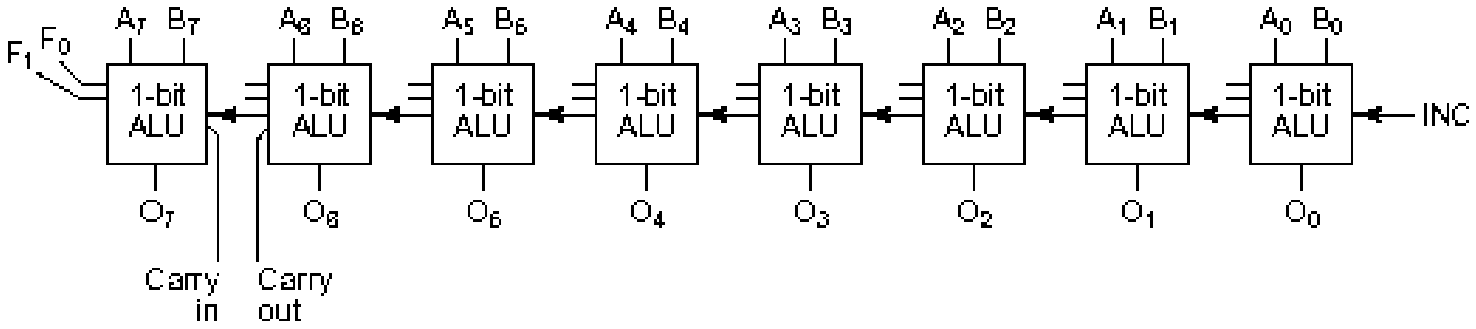
- This type of adder is called a *ripple carry adder* because the addition cannot complete until the carry has rippled all the way across the adders.

4.1 ALU – Arithmetic Logic Unit



- Such 1-bit ALUs allow computer designers to build an ALU of any desired size.

- The INC signal is used for addition operations. When present it increments (adds 1) to the result, allowing the computation of sums such as A+1.



4.2 Memory

- Combinational circuits.
  $\Rightarrow$ outputs dependent on inputs.

- Sequential circuits.
  $\Rightarrow$ outputs depend on present input values, and on past input values.

4.2.1 Flip-Flops

- Sequential circuit that forms the basic building block of memory.
  $\Rightarrow$ stores a single bit.

- *Flip*: output changes from 0 to 1.

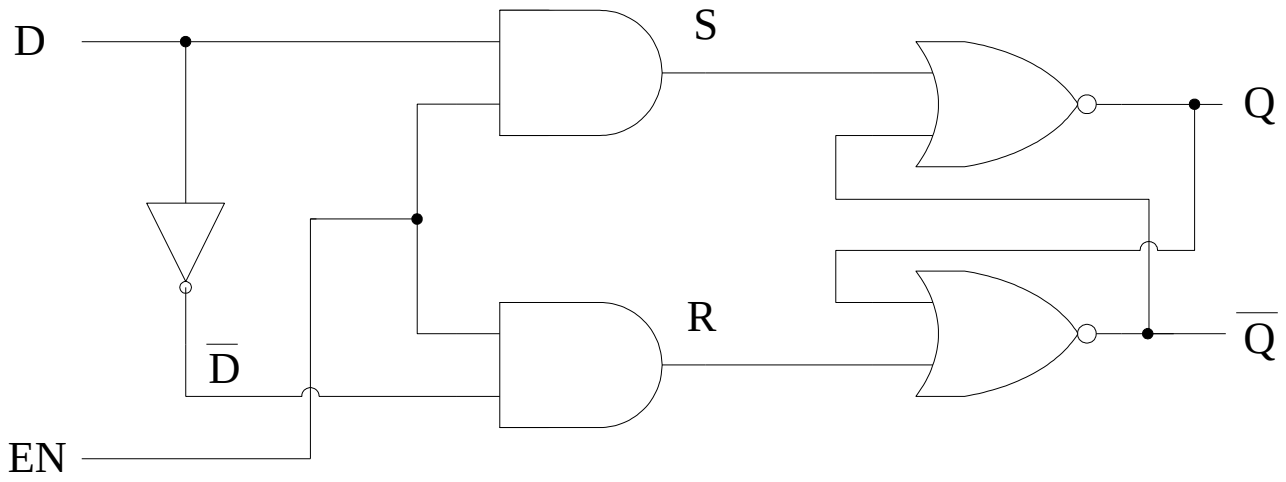
- *Flop*: output changes from 1 to 0.

4.2.2 D-type Flip-Flops

- Two inputs:
  - ⇒ D is the *data* input (the data bit that the flip-flop is to store).
  - ⇒ EN is the *enable* input.

- Output Q only changes on the transition of EN signal from one specified logic level to the other logic level.
  - ⇒ without a transition the output will not change even if the input D does.

- Known as *edge triggering*.
  - ⇒ *positive* (0 to 1) vs *negative* (1 to 0) edge triggering.

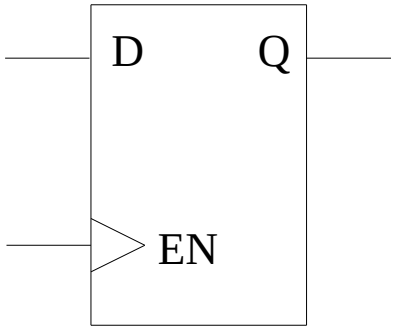


- Operation of the flip-flop can be clarified by a simplified truth table.

  $\Rightarrow$ Q output becomes the value of D after the activating transition of the EN signal.

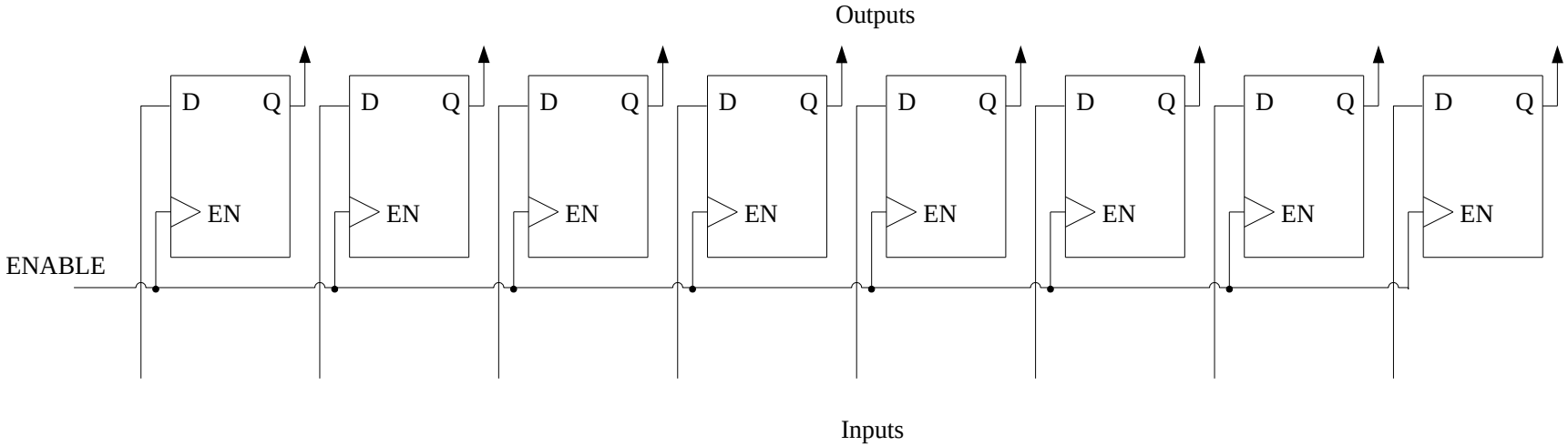
EN	D	Q
0	0	Q
0	1	Q
1	0	0
1	1	1

- Standard symbol:



4.2.3 Registers

- CPU also contains *registers*.
  - ⇒ small, high-speed memory used to store temporary results and control information.
  - ⇒ each register holds one value.
- To store eight bits, eight flip-flops must be combined.



- EN = 0
  - ⇒ flip-flops do not record the data being input to them, Q outputs remain unchanged.
- EN = 1
  - ⇒ flip-flops record the data on their D inputs and output it to their Q outputs.
- EN returns to 0
  - ⇒ Q outputs do not change and the data remains frozen in the flip-flops.