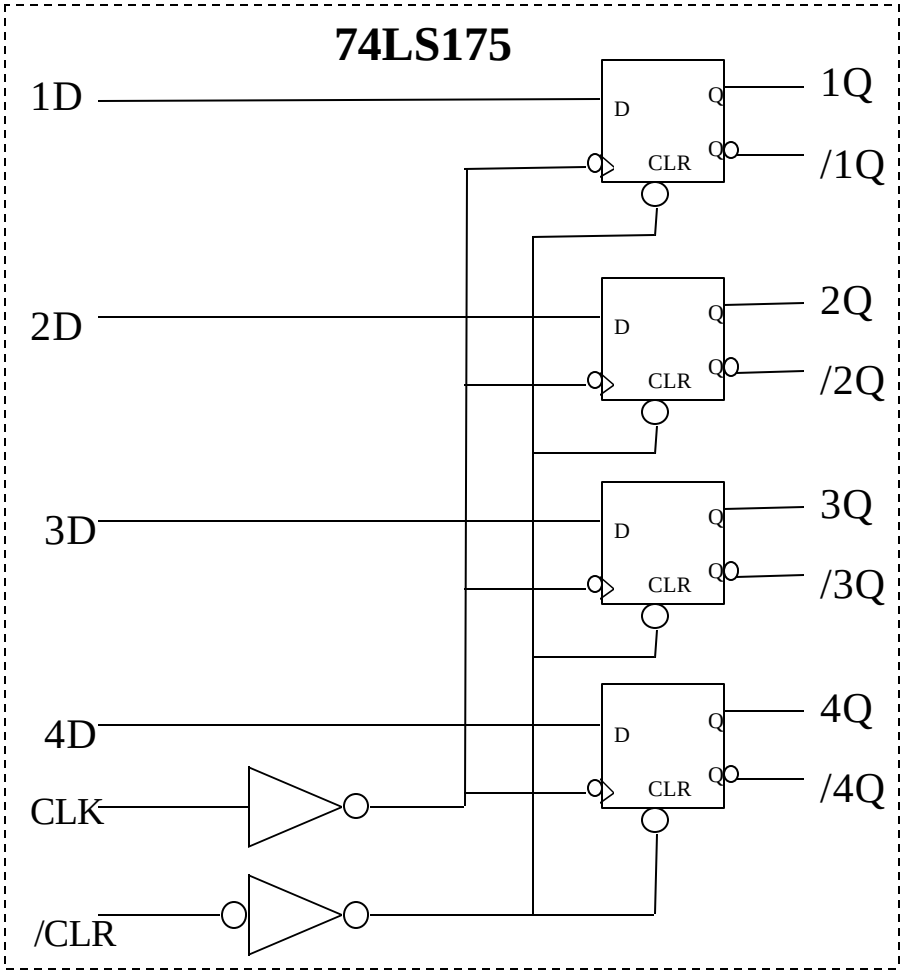
Registers & Counters

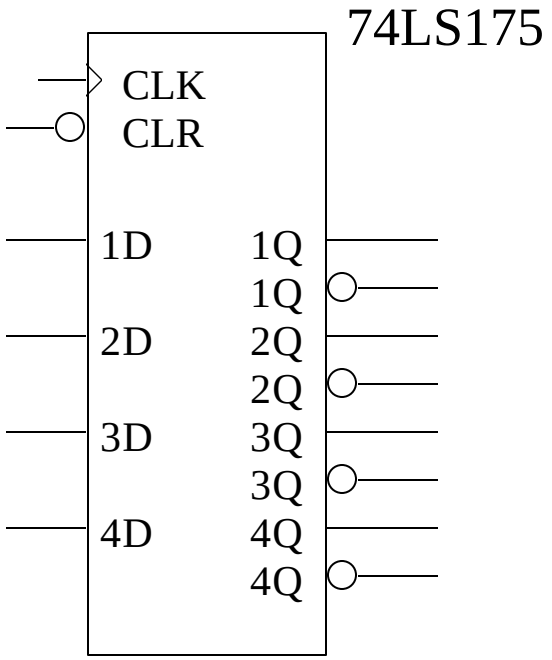
- **Registers.**
- **Shift Registers:**
  - **Serial in, serial out shift register**
  - **Serial in, parallel out shift register**
  - **Parallel in, serial out shift register**
  - **Parallel in, parallel out shift register**
  - **Shift Register Applications**
- **Counters:**
  - **Ripple Counters**
  - **Synchronous Counters**
  - **Counter Applications**

Registers

- **An n-bit register is a collection of n D flip-flops with a common clock used to store n related bits.**

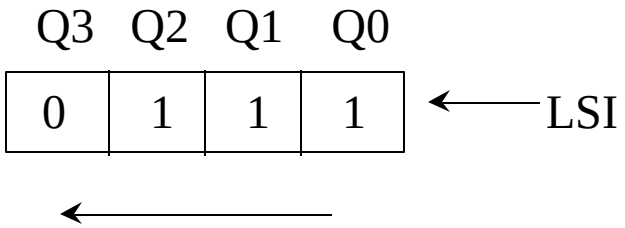


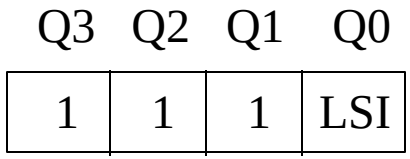
Example: 74LS175 4-bit register



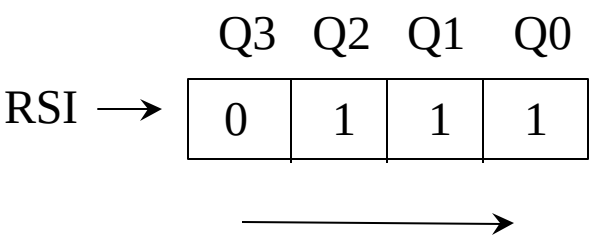
Shift Registers

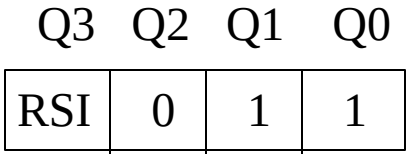
- **Multi-bit register that moves stored data bits left/right (1 bit position per clock cycle)**
  - **Shift Left is towards MSB**



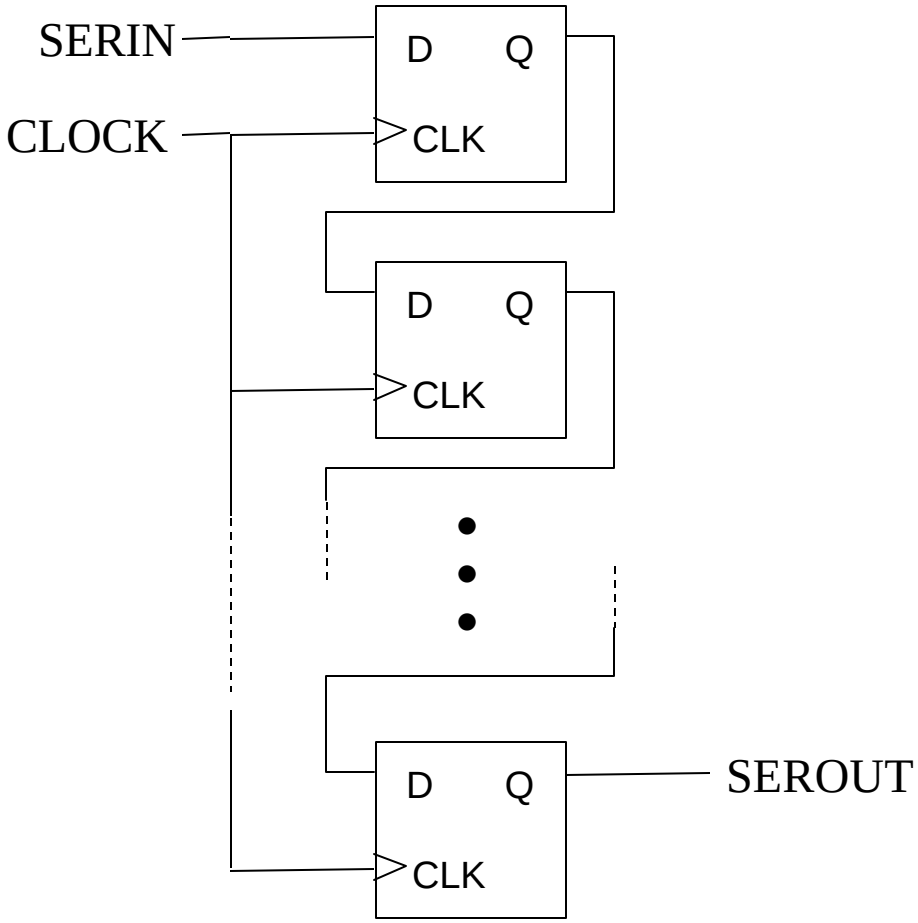


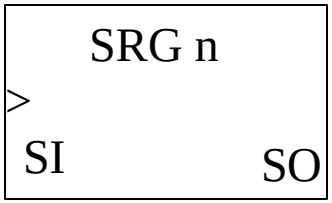
  - **Shift Right (or Shift Up) is towards MSB**





Serial In, Serial Out Shift Register





For a n-bit SRG:
Serial Out = Serial In delayed by n clock period

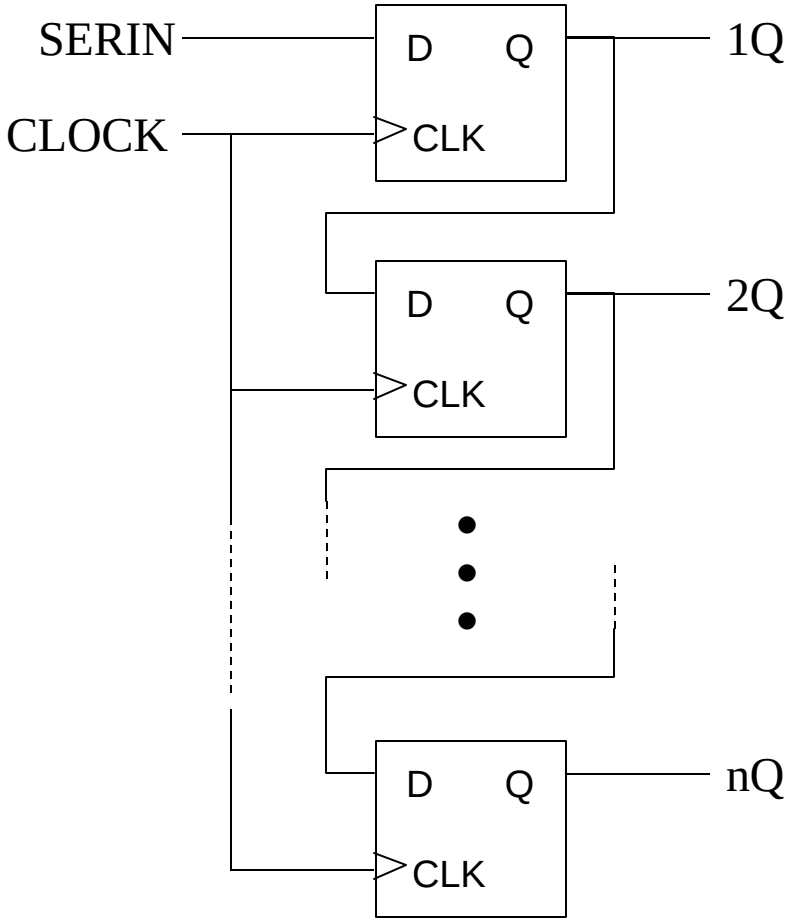
4-bit shift register example:

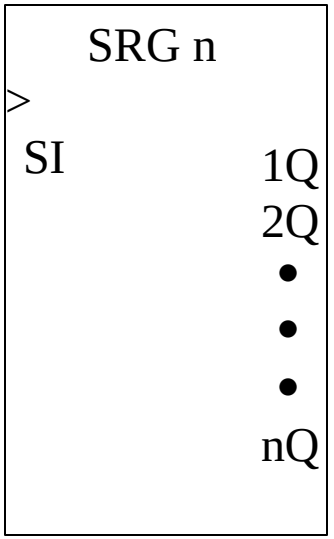
serin: 1 0 1 1 0 0 1 1 1 0

serout: - - - - 1 0 1 1 0 0

clock: 

Serial In, Parallel Out Shift register





(SO)

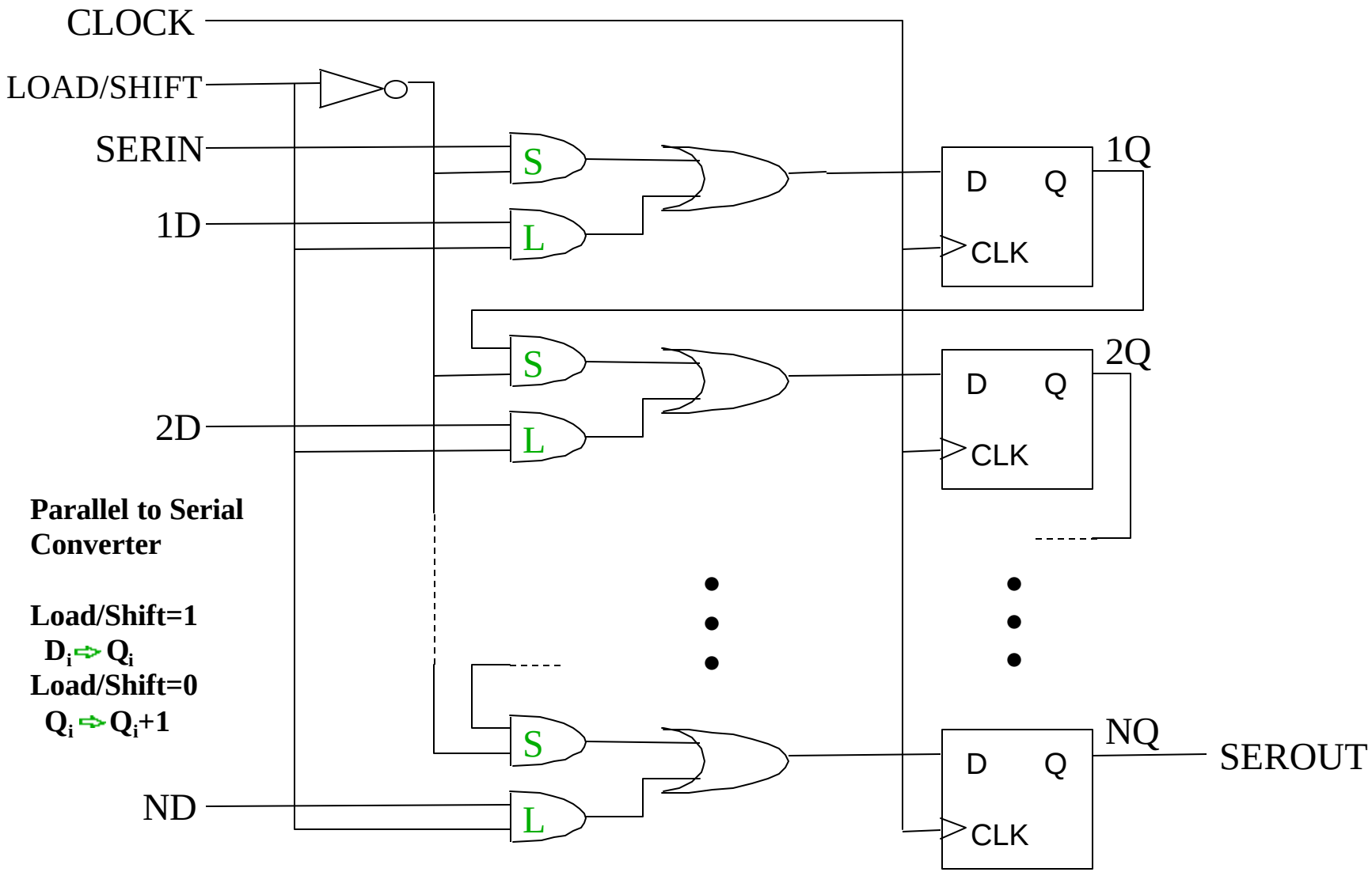
Serial to Parallel Converter

4-bit shift register example:

serin:	1 0 1 1 0 0 1 1 1 0
1Q:	- 1 0 1 1 0 0 1 1 1
2Q:	- - 1 0 1 1 0 0 1 1
3Q:	- - - 1 0 1 1 0 0 1
4Q:	- - - - 1 0 1 1 0 0

clock:

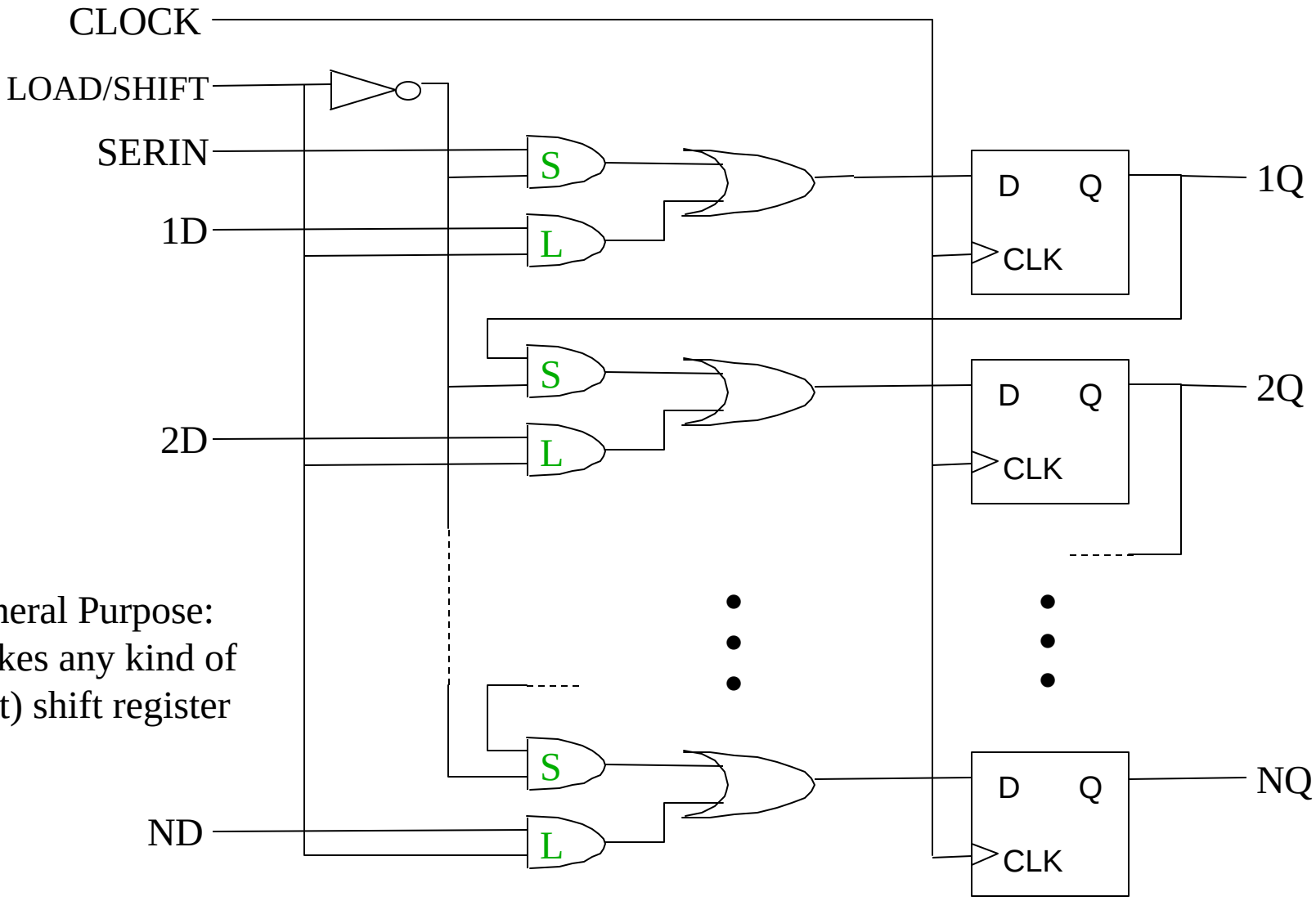
Parallel In, Serial Out Shift Register



Parallel to Serial Converter

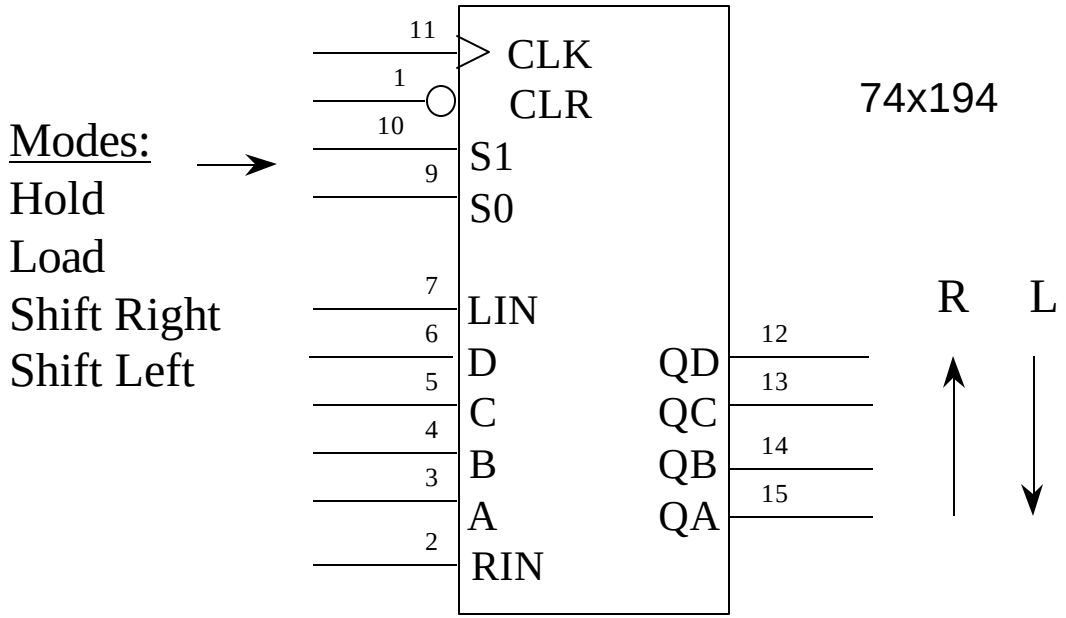
Load/Shift=1
$D_i \Rightarrow Q_i$
Load/Shift=0
$Q_i \Rightarrow Q_i+1$

Parallel In, Parallel Out Shift Register



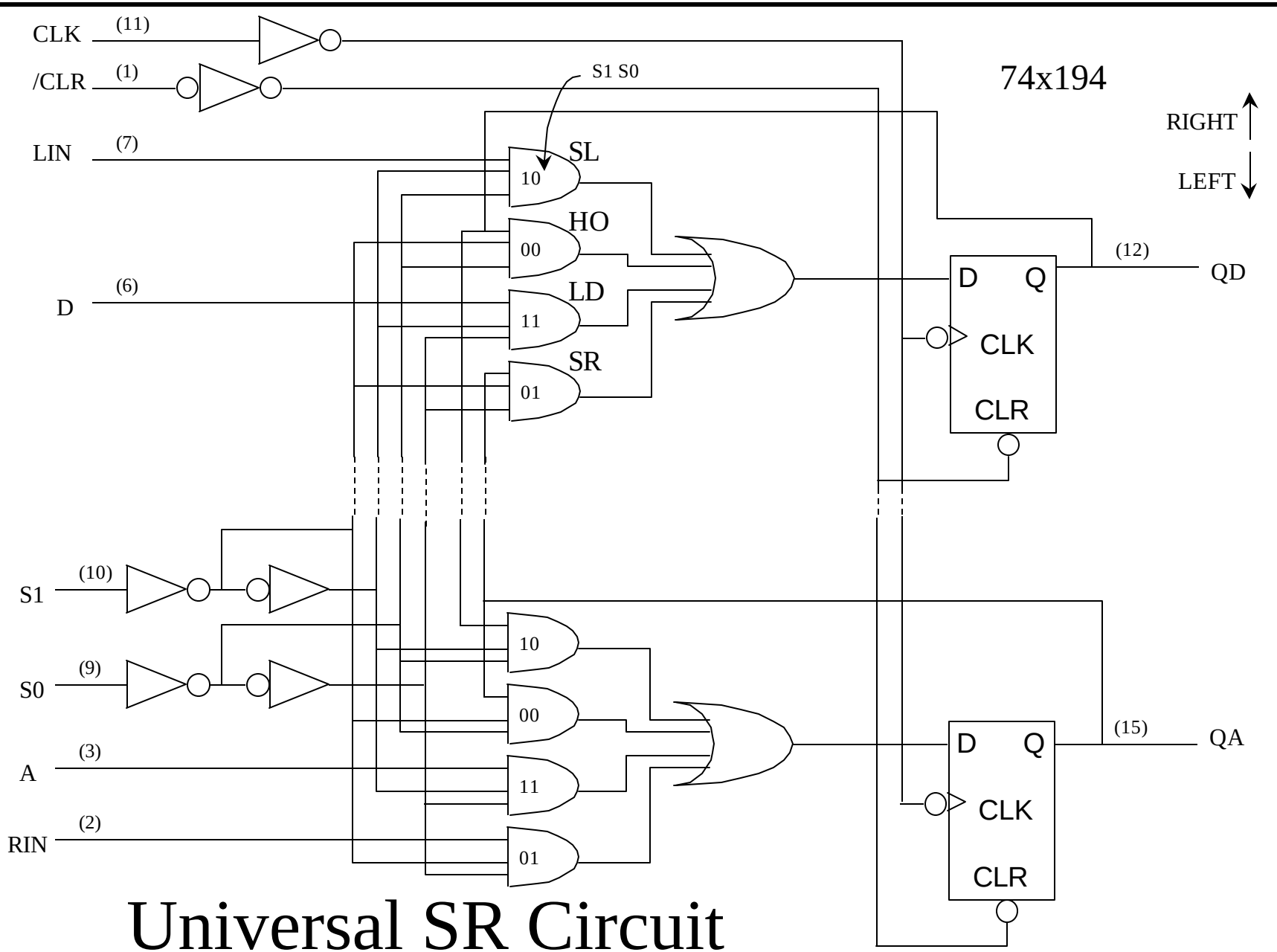
General Purpose:
Makes any kind of
(left) shift register

Bi-directional Universal Shift Registers



4-bit Bi-directional Universal (4-bit) PIPO

	Mode		Next state			
Function	S1	S0	QA*	QB*	QC*	QD*
Hold	0	0	QA	QB	QC	QD
Shift right/up	0	1	RIN	QA	QB	QC
Shift left/down	1	0	QB	QC	QD	LIN
Load	1	1	A	B	C	D



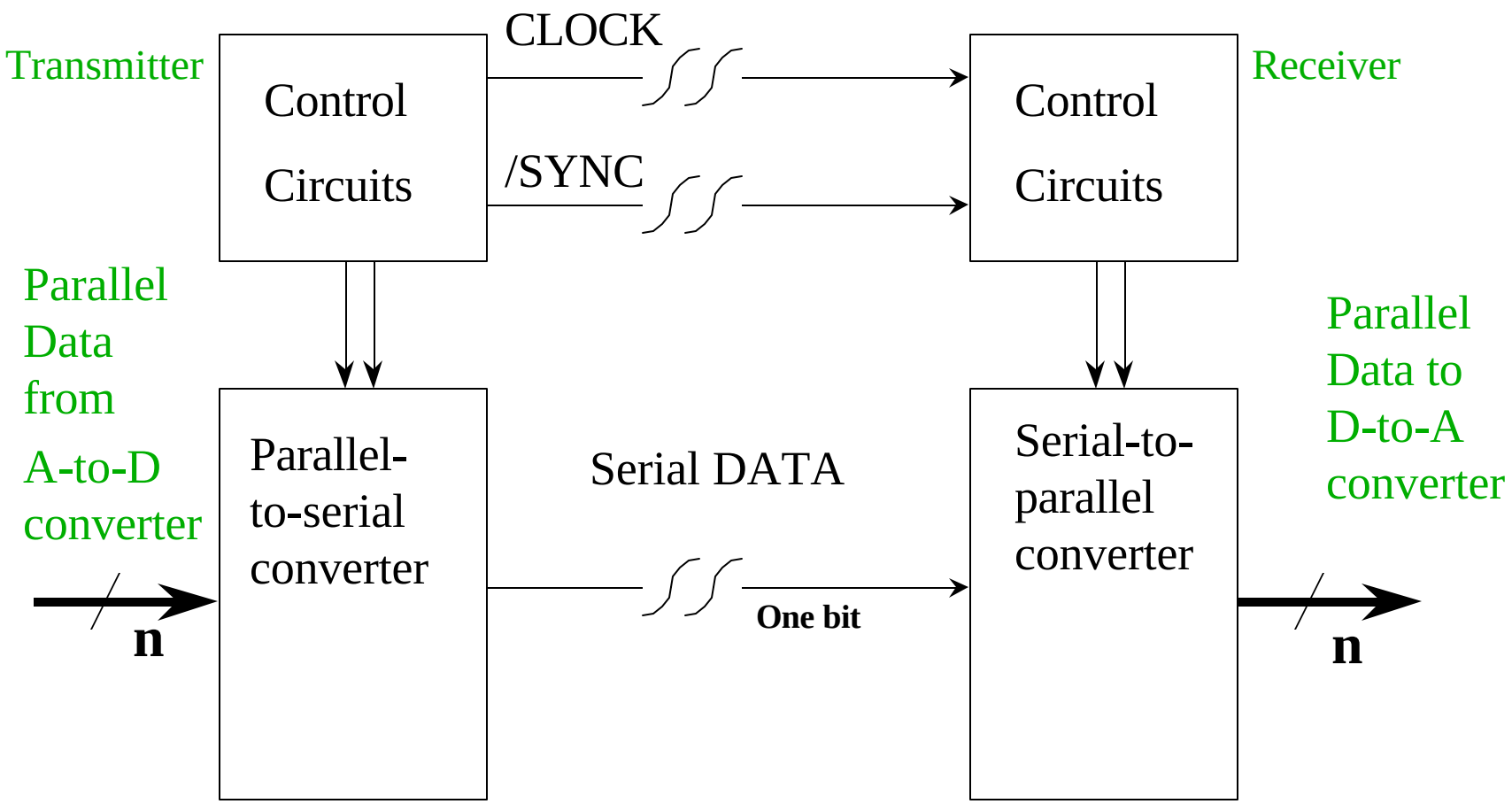
Universal SR Circuit

Shift Register Applications

- **State Registers**
  - Shift registers are often used as the state register in a sequential device. Usually, the next state is determined by shifting right and inserting a primary input or output into the next position (i.e. a finite memory machine)
  - Very effective for sequence detectors
- **Serial Interconnection of Systems**
  - keep interconnection cost low with serial interconnect
- **Bit Serial Operations**
  - Bit serial operations can be performed quickly through device iteration
  - Iteration (a purely combinational approach) is expensive (in terms of # of transistors, chip area, power, etc).
  - A sequential approach allows the reuse of combinational functional units throughout the multi-cycle operation

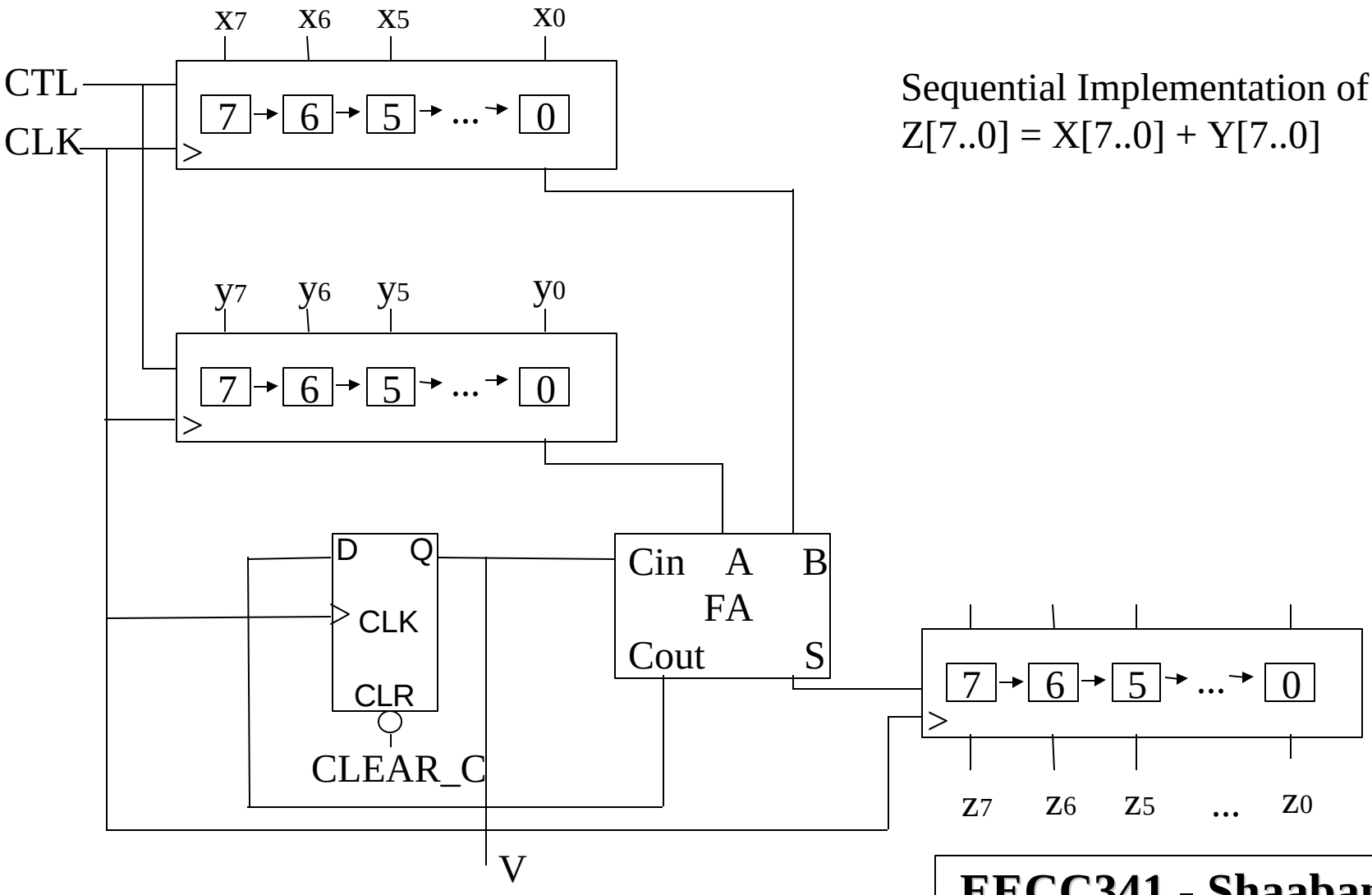
Shift Register Applications:

Serial Interconnection of Systems



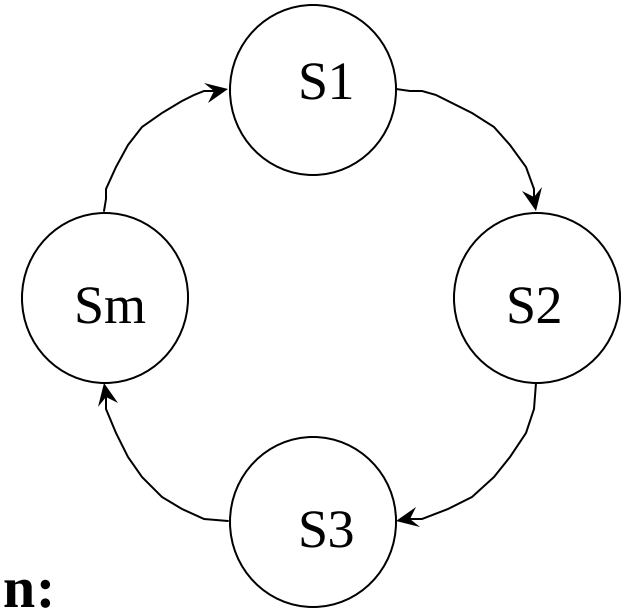
Shift Register Applications Example:

8-Bit Serial Adder



Sequential Implementation of:
Z[7..0] = X[7..0] + Y[7..0]

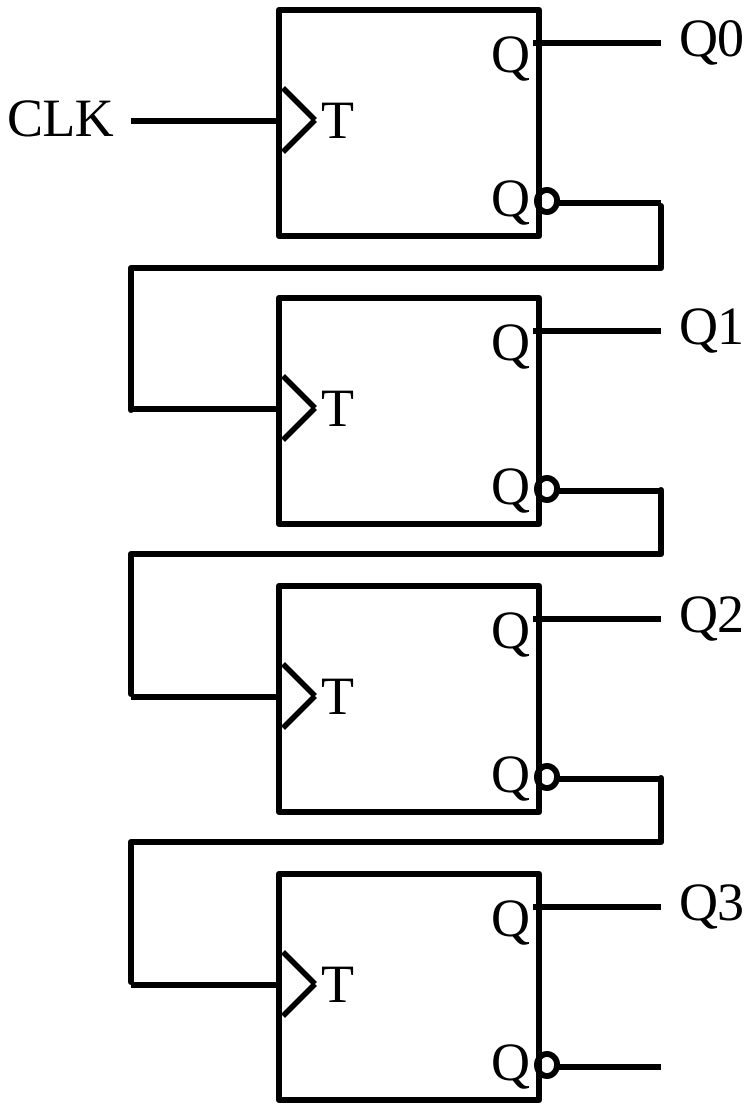
Counters

- **Clocked sequential circuit with single-cycle state diagram**
  - **Modulo-m counter = divide-by-m counter**
  - 

  - **Most Common:**

n-bit binary counter, where m = 2^n ➔ n flip-flops, counts 0 … 2^n-1

4-bit Ripple Counter



1 bit

divide-by-2

2 bit

divide-by-4

3 bit

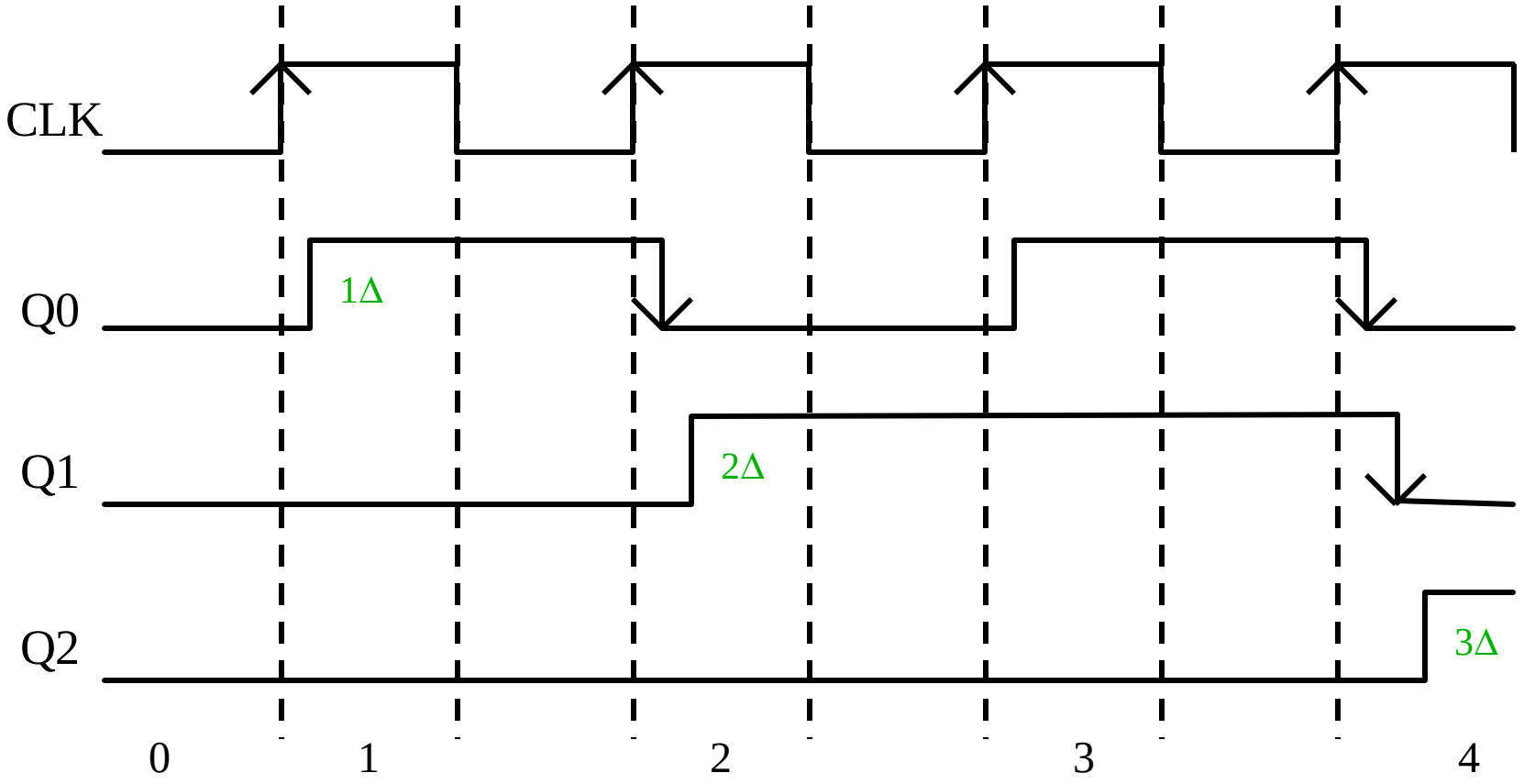
divide-by-8

4 bit

divide-by-16

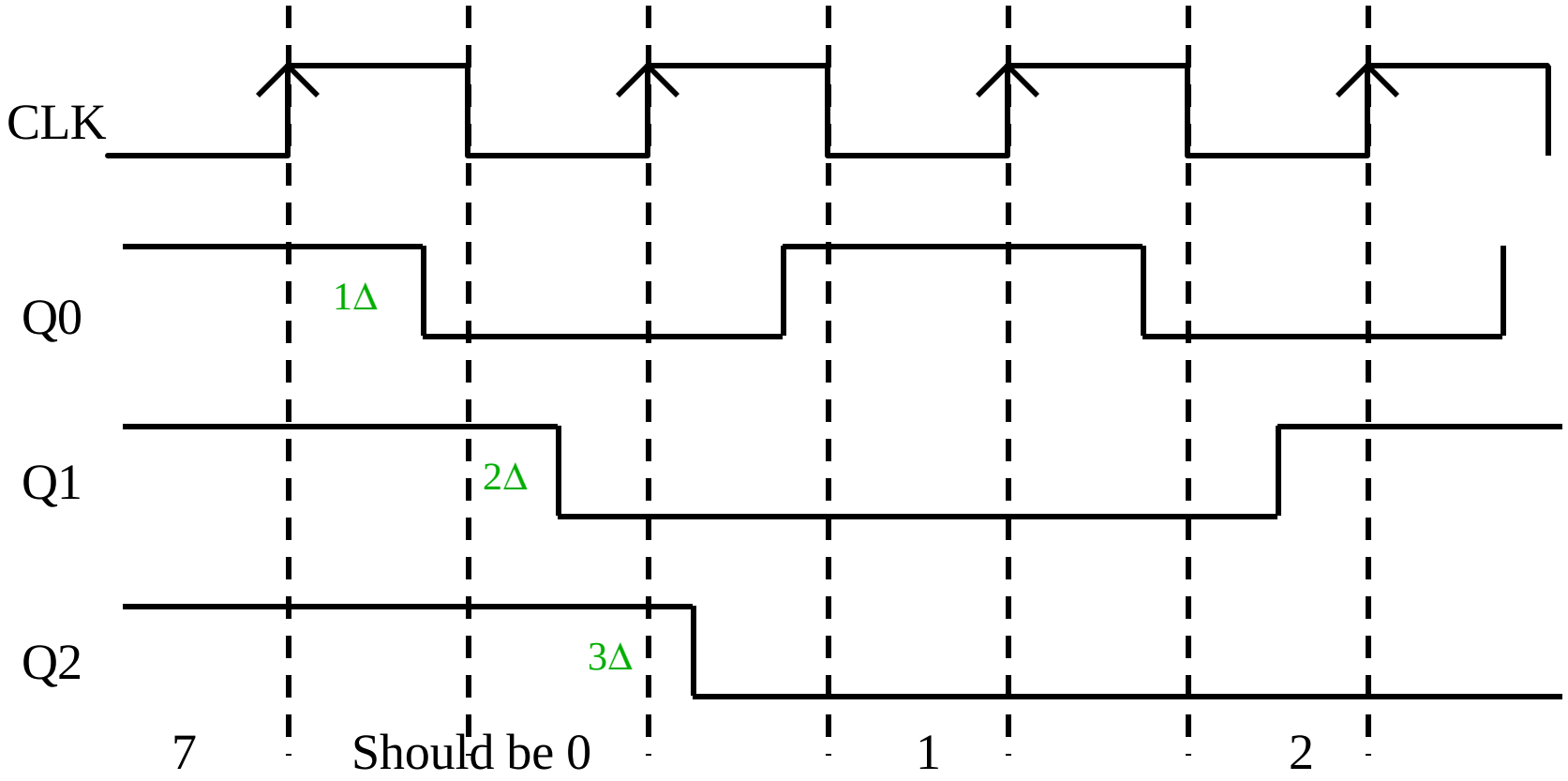
Uses Minimal Logic

Ripple Counter Timing



Ripple Counter Problem

$n \bullet T_{CQ}$ for MSB change for n-bit ripple counter => minimum clk period

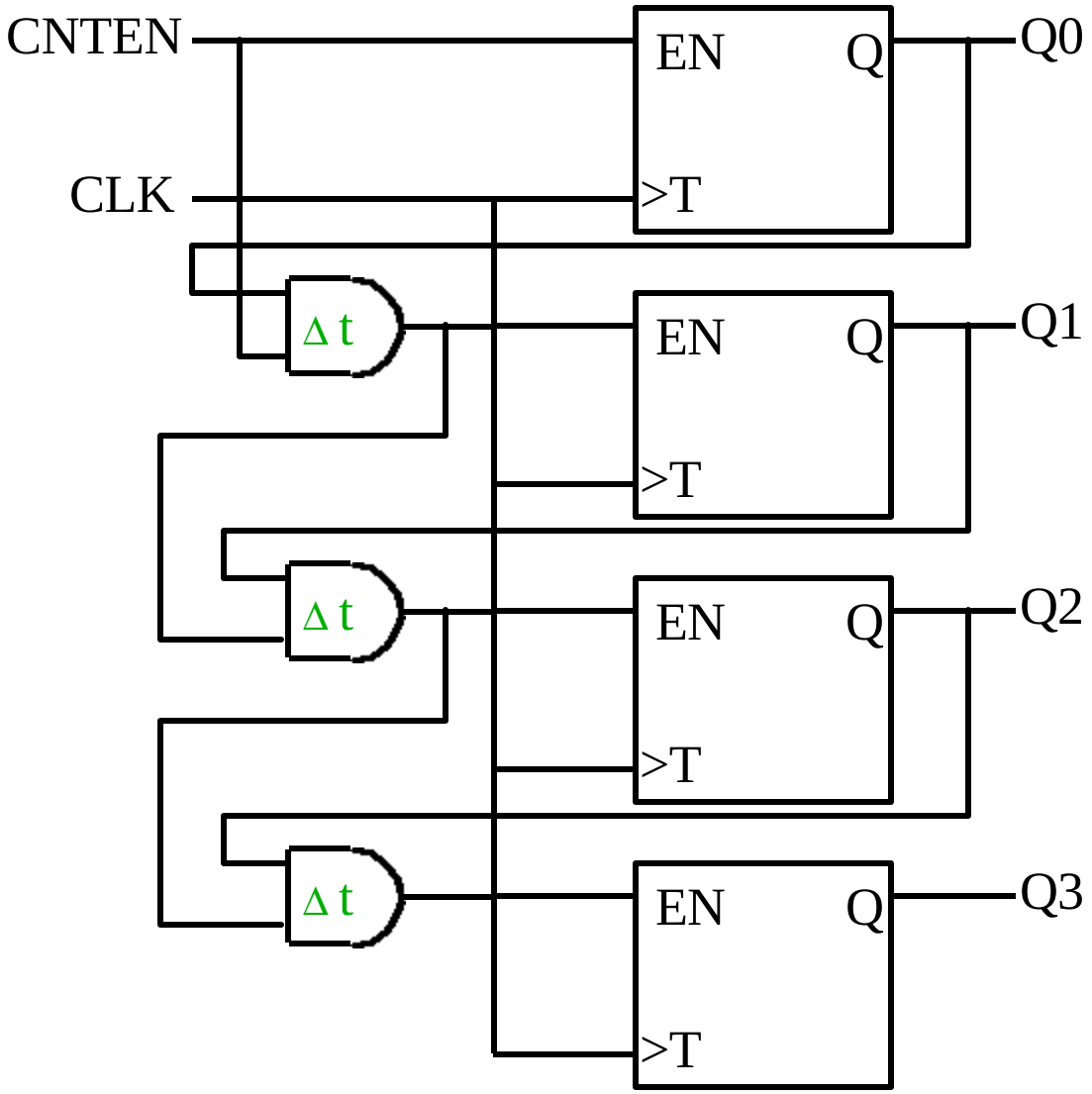


Synchronous Counters

- All clock inputs connected to common CLK signal
  - All flip-flop outputs change simultaneously t_{CQ} after CLK
  - Faster than ripple counters
  - More complex logic
  - Most frequently used type of counter

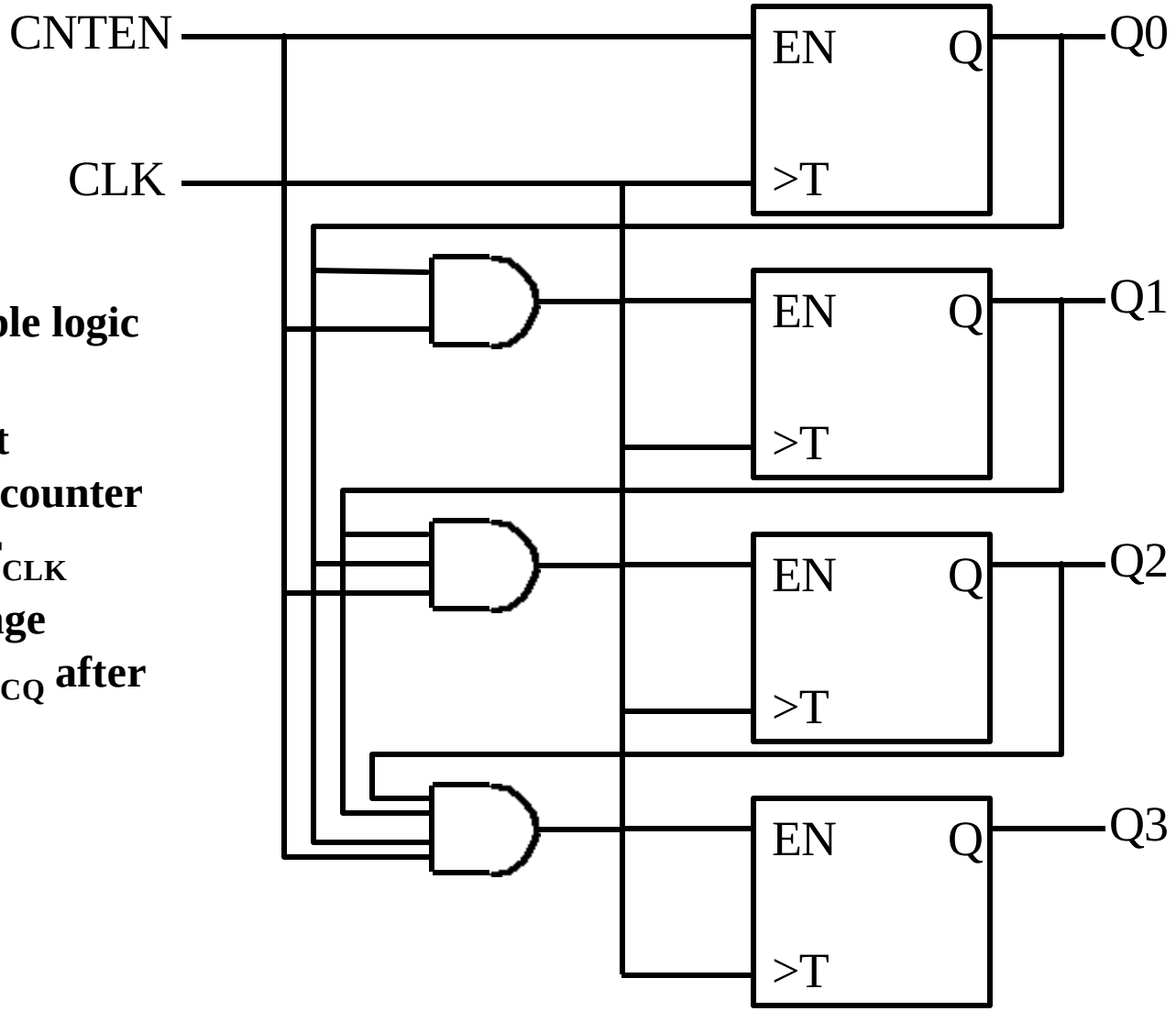
Synchronous Serial Counter

- **Flip-flops enabled when all lower flip-flops = 1.**
- **Enable propagates serially — limits speed**
- **Requires $(n-1) \times t < T_{CLK}$**
- **All outputs change simultaneously t_{CQ} after CLK↑**

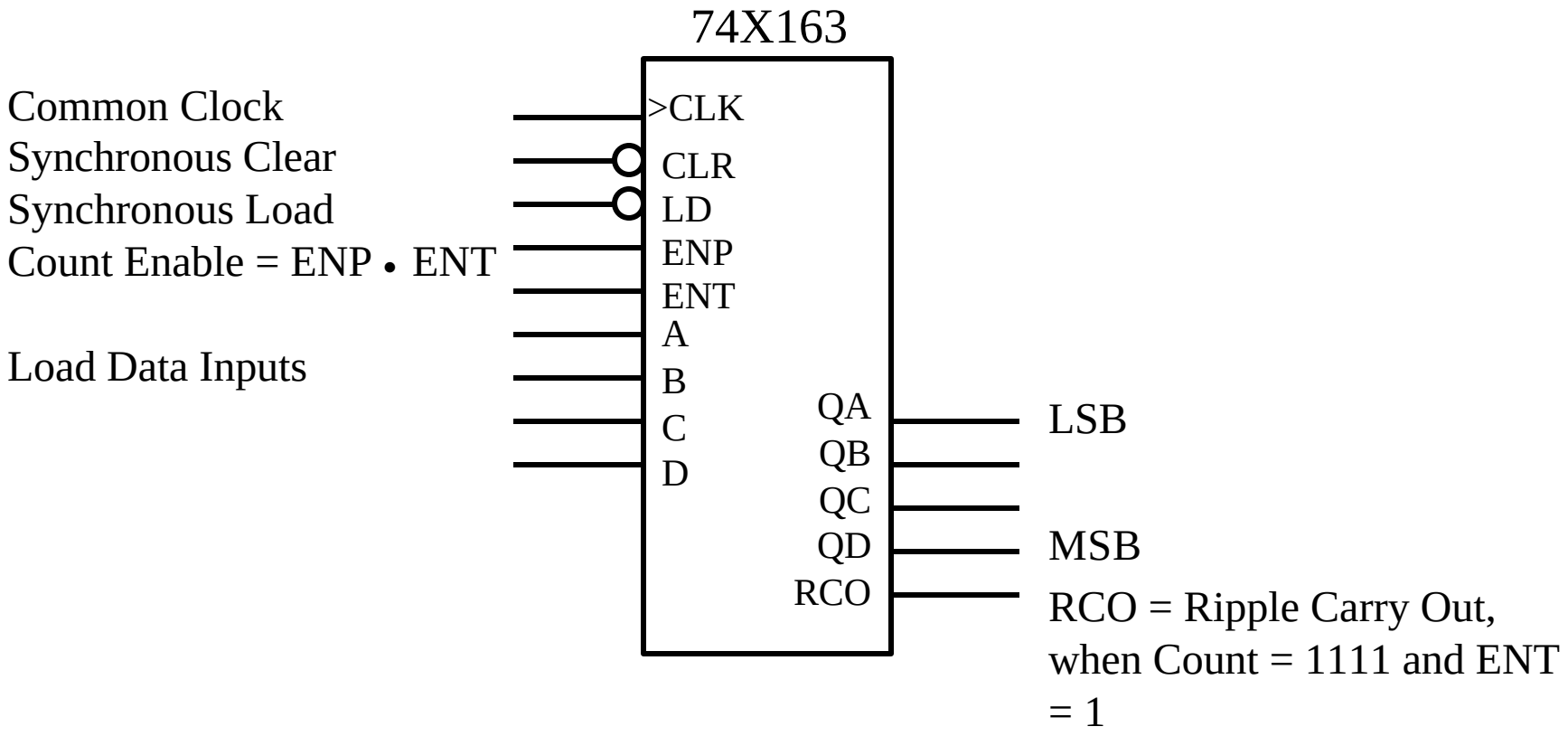


Synchronous Parallel Counter

- **Single-level enable logic per flip-flop**
- **Fastest and most complex type of counter**
- **Requires D t < T_{CLK}**
- **All outputs change simultaneously t_{CQ} after CLK↑**



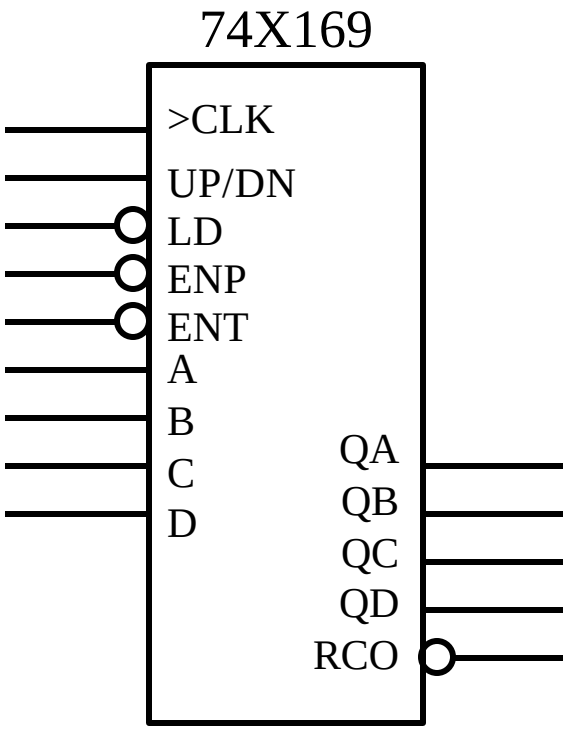
74X163 4-bit Synchronous Parallel Counter



74X163 State Table

Inputs					Current State				Next State			
/CLR	/LD	ENT	ENP		QD	QC	QB	QA	QD*	QC*	QB*	QA*
0	X	X	X	Clear	X	X	X	X	0	0	0	0
1	0	X	X	Load	X	X	X	X	D	C	B	A
1	1	0	X	Hold	X	X	X	X	QD	QC	QB	QA
1	1	1	0	Hold	X	X	X	X	QD	QC	QB	QA
1	1	1	1	Count	0	0	0	0	0	0	0	1
1	1	1	1	.	0	0	0	1	0	0	1	0
1	1	1	1	.	0	0	1	0	0	0	1	1
1	1	1	1	.	0	0	1	1	0	1	0	0
1	1	1	1		0	1	0	0	0	1	0	1
1	1	1	1		0	1	0	1	0	1	1	0
1	1	1	1		0	1	1	0	0	1	1	1
1	1	1	1		0	1	1	1	1	0	0	0
1	1	1	1		1	0	0	0	1	0	0	1
1	1	1	1		1	0	0	1	1	0	1	0
1	1	1	1		1	0	1	0	1	0	1	1
1	1	1	1		1	0	1	1	1	1	0	0
1	1	1	1		1	1	0	0	1	1	0	1
1	1	1	1		1	1	0	1	1	1	1	0
1	1	1	1		1	1	1	0	1	1	1	1
1	1	1	1		1	1	1	1	0	0	0	0

74X169 Up/Down Counter



UP/DN = 1 = up → RCO = 15

UP/DN = 0 = down → RCO = 0

up	down	up
Ex: 0,1,2,	1,0,15,14,	15,0,1,2
	↓ RCO	↓ RCO

Counter Applications

- **Count the number of times an event takes place**
- **Control the number of steps in a sequence of fixed actions (a sequencer)**
- **Generate timing signals (frequency divider, etc.)**