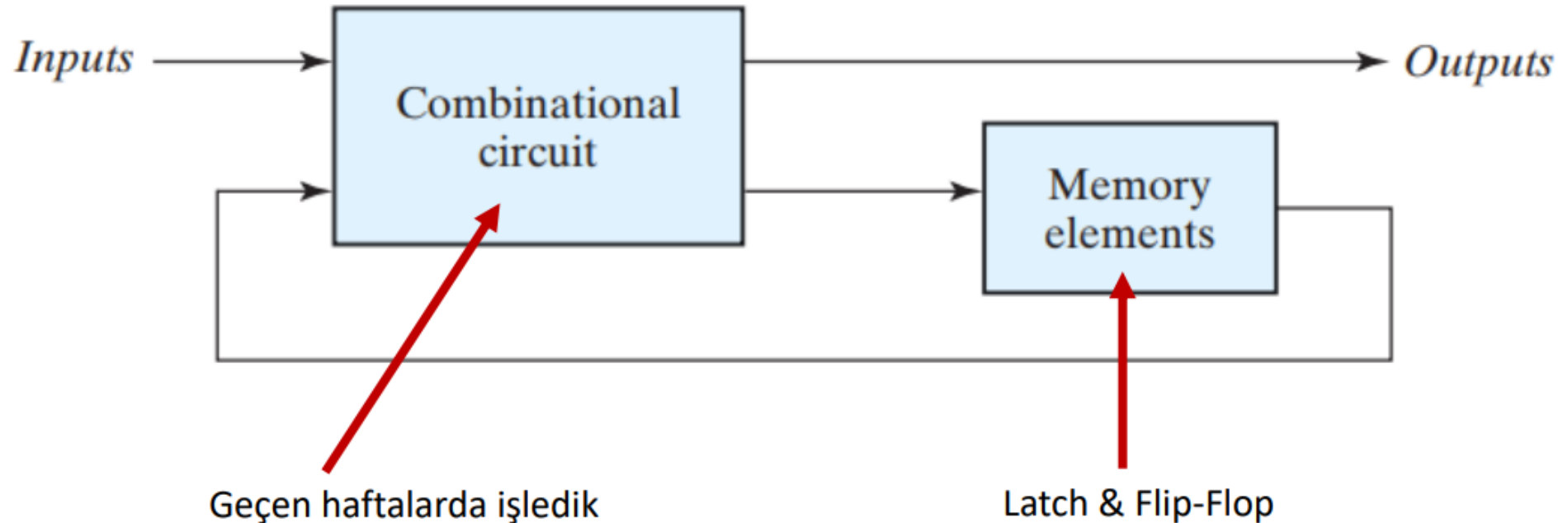


12. HAFTA

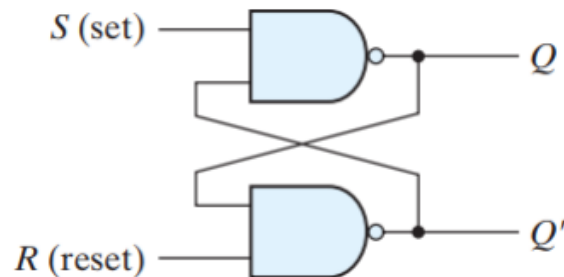
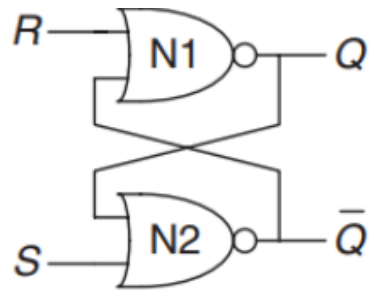
MANTIKSAL TASARIM

Sıralı Devreler

Çıkış sinyali, giriş sinyallerinin hem o anki hem de geçmiş değerine bağlı olan devreler. İçerisinde bellek (memory) birimi bulundurlar. Devre içerisindeki belleğin o anki değeri, devrenin durumu (state) olarak adlandırılır.



SR (Set/Reset) Mandal (Latch)



Durum1: $R=1 \ \& \ S=0 \rightarrow Q = 0, Q\_bar = 1$

Durum2: $R=0 \ \& \ S=1 \rightarrow Q\_bar = 0, Q = 1$

Durum3: $R=1 \ \& \ S=1 \rightarrow Q = 0, Q\_bar = 0$

Durum4: $R=0 \ \& \ S=0 \rightarrow Q = Q\_prev, Q\_bar = Q\_bar\_prev$

Case	S	R	Q	$\bar{Q}$
IV	0	0	Q_{prev}	$\bar{Q}_{prev}$
I	0	1	0	1
II	1	0	1	0
III	1	1	0	0

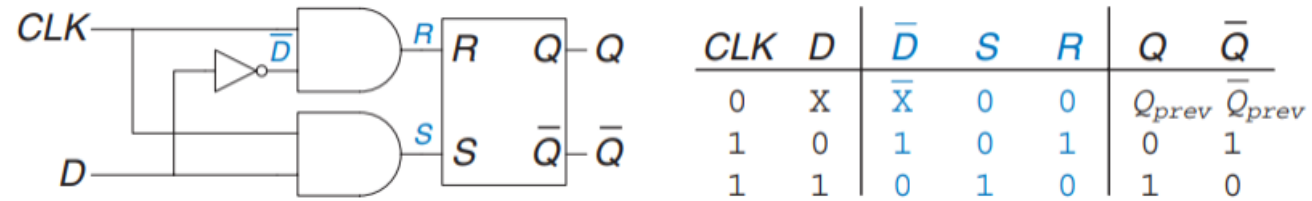
Problem: S ve R aynı anda 1 olduğunda istenmeyen durum oluşuyor

Çözüm: Tasarım öyle düzenlenmeli ki S ve R aynı anda hiç '1' olmasın

Bunu sağlamak mümkün olmayabilir ya da tasarımı zorlaştırabilir

Çözüm: Mandal tipini değiştirmek $\rightarrow$ D-Latch

D Mandal (Latch)



D mandal sayesinde S ve R aynı anda '1' olamıyor

CLK sinyali '1' olduğu zaman Q değeri D'ye eşit oluyor. CLK sinyali '0' olduğu zaman S ve R '0' oluyor ve Q değeri önceki durumunu, yani değerini koruyor

CLK sinyali '1' oldukça Q sinyaline D değeri atanıyor. Atamaların sadece belirli "an"larda yapılması zamanlama analizi ve sistemin senkronize olması açısından daha kolay olacaktır.

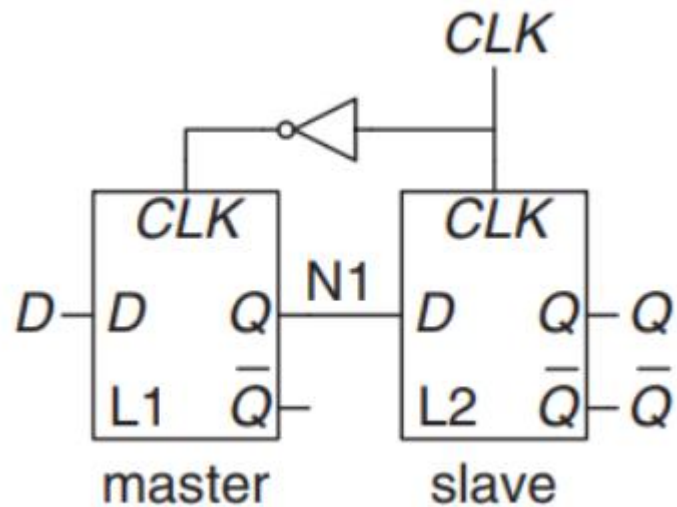
Belirli "an"larda değer ataması yapılması → Flip-Flop

Latch : Level-sensitive

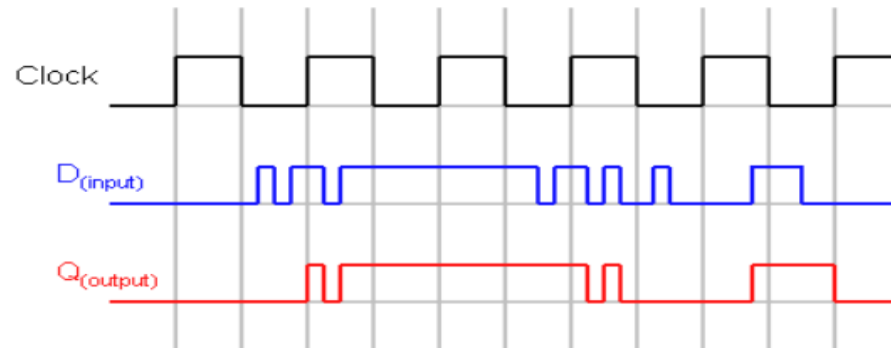
Flip-Flop: Edge-sensitive

D Flip-Flop

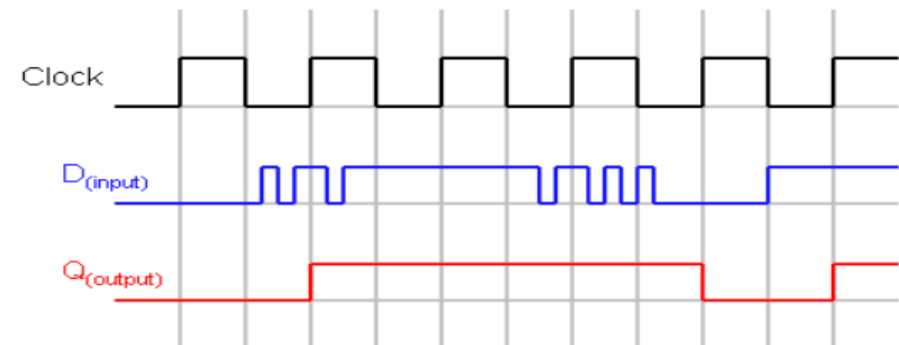
In other words, a D flip-flop copies D to Q on the rising edge of the clock and remembers its state at all other times. Reread this definition until you have it memorized; one of the most common problems for beginning digital designers is to forget what a flip-flop does.



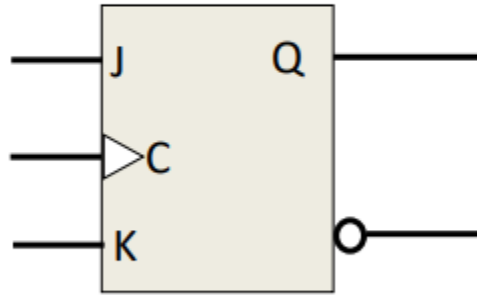
D Latch



D Flip-Flop



JK Flip-Flops



J	K	$Q(t+1)$	next state
0	0	$Q(t)$	no change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

Characteristic Table

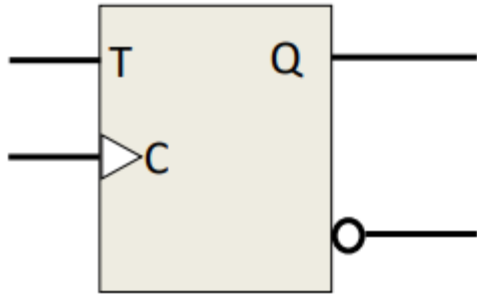
J	K	$Q(t)$	$Q(t+1)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

K,Q(t)		J			
		00	01	11	10
J	0	0	1	0	0
	1	1	1	0	1

$$Q(t+1) = ?$$

T (Toggle) Flip-Flop

- Complementing flip-flop

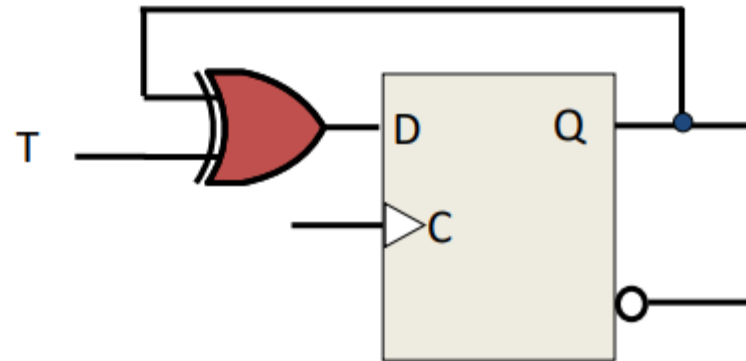
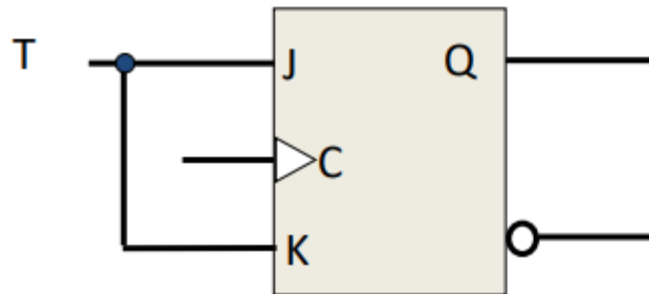


T	$Q(t+1)$	next state
0	$Q(t)$	no change
1	$Q'(t)$	Complement

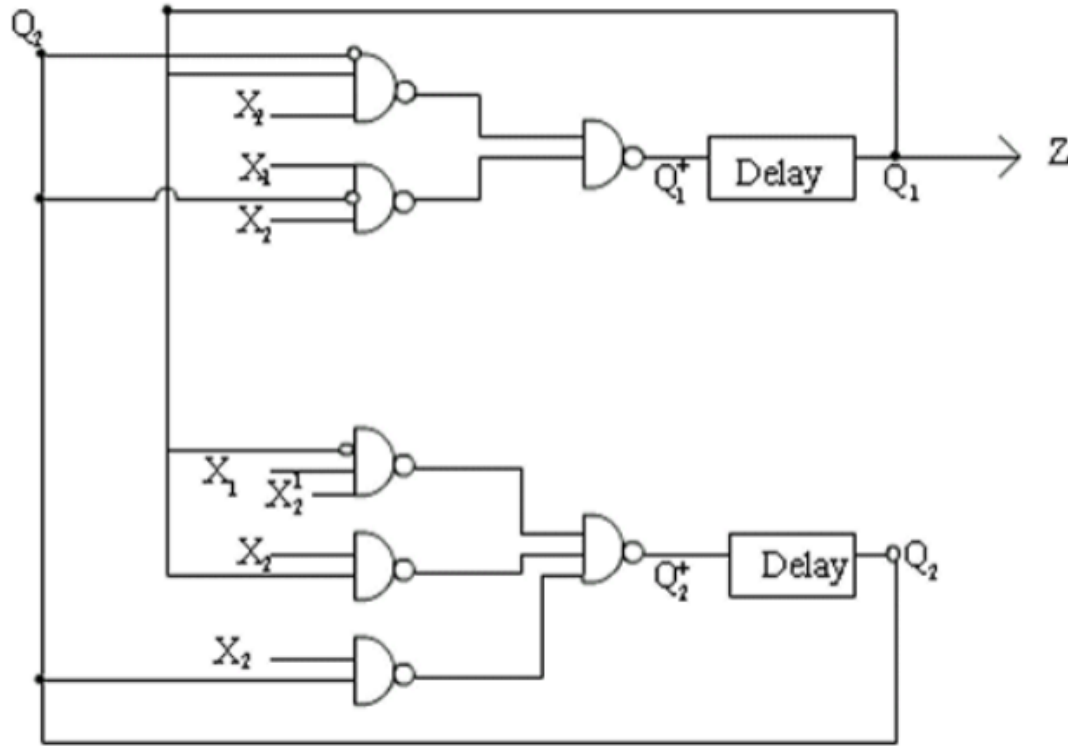
Characteristic Table

- Characteristic equation

$$Q(t+1) = ?$$



ASENKRON vs SENKRON SIRALI DEVRELER (ASYNCHRONOUS vs SYNCHRONOUS SEQUENTIAL CIRCUITS)



Asenkron devrelerde bellek (memory) bulunmaz ve bir saat sinyali ile bu bellek birimleri senkronize edilmez

Asenkron devrelerde saat olmadığı için kapı seviyesinde gecikmelere dayalı ve geribesleme (feedback) yapıları ile durumlar (states) meydana getirir

Teorik olarak asenkron sıralı devrelerin senkron devrelere göre daha hızlı ve daha az güç tüketen olması beklenir

Fakat asenkron devrelerin zamanlama (timing) analizi senkron devrelere göre çok daha zordur

Günümüzde **akademik alanda** asenkron devreler çalışılmaya devam etmektedir

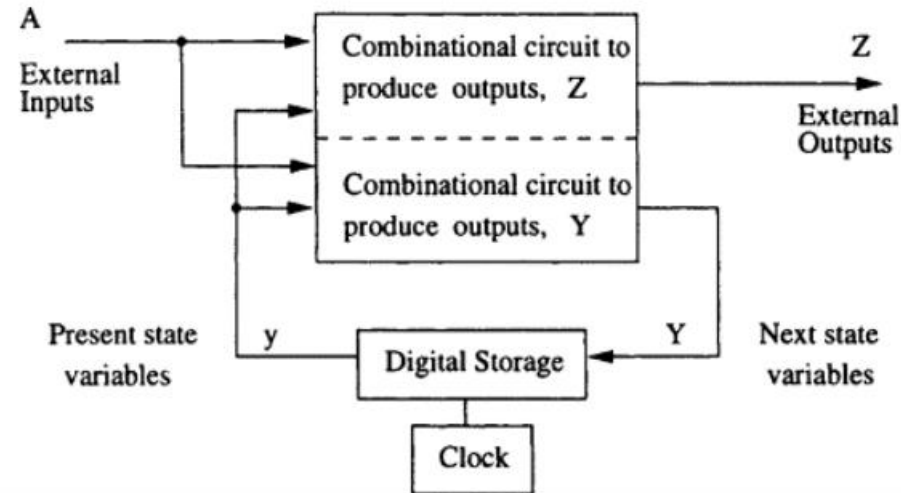
SENKRON SIRALI DEVRELER (SYNCHRONOUS SEQUENTIAL CIRCUITS)

Senkron sıralı devreler birleşik (combinational) ve sıralı (sequential) bloklardan oluşur

Birleşik devre kısmı giriş sinyallerinin değişimine anında tepki veren ve sonuç sinyalleri hesaplanan devrelerdir

Sıralı devre kısmı ise bir saat sinyali (clock signal) ile senkronize olmuş genellikle FF'lerden oluşan ve devrenin o anki durumunu (state) ifade eden bloktur

FF'lar, saat vuruşunun yükselen ya da alçalan kenarında giriş sinyal değerini çıkış sinyaline iletirler



SENKRON SIRALI DEVRELER (SYNCHRONOUS SEQUENTIAL CIRCUITS)

The rules of *synchronous sequential circuit composition* teach us that a circuit is a synchronous sequential circuit if it consists of interconnected circuit elements, such that

- ▶ Every circuit element is either a register or a combinational circuit
- ▶ At least one circuit element is a register
- ▶ All registers receive the same clock signal
- ▶ Every cyclic path contains at least one register

SONLU DURUM MAKİNELERİ (FINITE STATE MACHINES)

Neden “sonlu durum” → Çünkü n adet FF ile 2^n kadar durum meydana getirilebiliyor

Durum tablosu oluşturma → Giriş sinyalleri ve şu anki duruma (present state) göre sonraki durum (next state) ve çıkış sinyallerini hesaplamak

Her bir saat vuruşunda, FF'lar sonraki durum (next state) değerini alırlar ve buna göre de çıkış sinyalleri güncellenir

2 tür sonlu durum makinesi vardır: Moore – Mealy

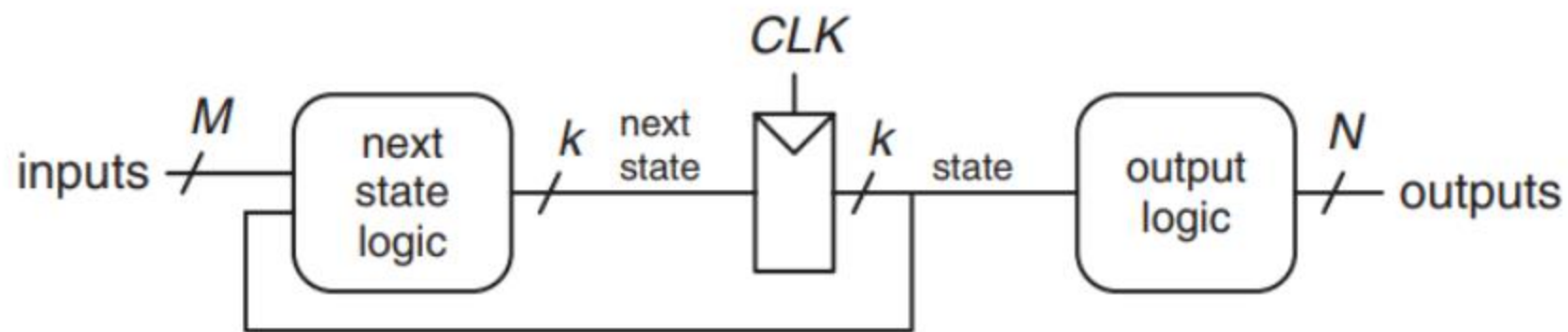
Moore makinelerinde, çıkış sinyalleri sadece devrenin o anki durumuna bağlıdır

Mealy makinelerinde, çıkış sinyalleri hem devrenin o anki durumuna, hem de giriş sinyallerine bağlıdır

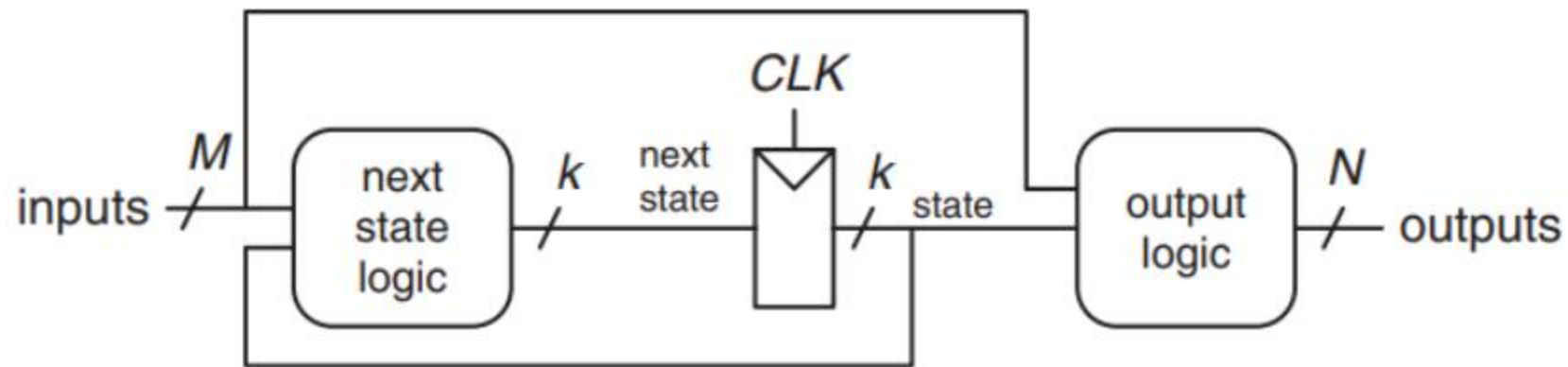
Moore ve Mealy bu yöntemleri 1955/56 yıllarında ortaya atmışlardır

Edward F. Moore (1925-2003) → Bell Lab, University of Wisconsin

George H. Mealy (1927-2010) → Bell Lab, Harvard University



(a)



(b)