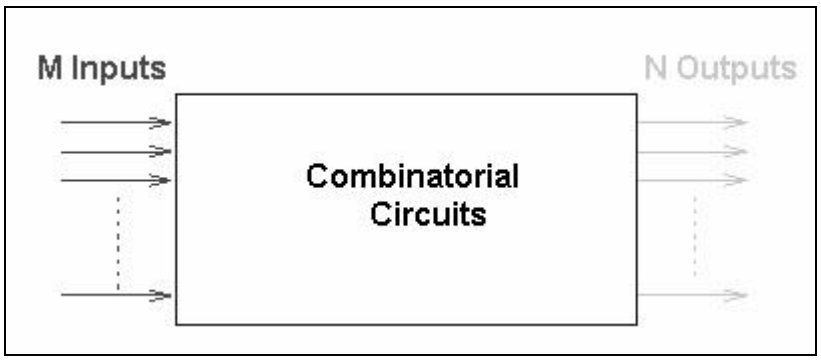
3. Rangkaian Logika Kombinasional dan Sequensial

Rangkaian Logika secara garis besar dibagi menjadi dua, yaitu rangkaian logika Kombinasional dan rangkaian logika Sequensial. Rangkaian logika Kombinasional adalah rangkaian yang kondisi keluarannya (*output*) dipengaruhi oleh kondisi masukan (*input*).

Struktur rangkaian kombinasional secara fisik adalah seperti gambar berikut:



Gambar 3.1.

Sedangkan rangkaian logika Sequensial adalah rangkaian yang kondisi keluarannya dipengaruhi oleh kondisi masukan dan keadaan keluaran sebelumnya atau dapat juga dikatakan rangkaian yang bekerja berdasarkan urutan waktu. Ciri rangkaian logika sequensial yang utama adalah adanya jalur umpan balik (*feed back*) di dalam rangkaiannya.

3.1. Rangkaian Logika Kombinasional

Rangkaian logika kombinasional yang akan dibahas adalah Enkoder, Dekoder, Multiplexer, dan Demultiplexer.

3.1.1. Enkoder

Enkoder adalah rangkaian logika kombinasional yang berfungsi untuk mengubah atau mengkodekan suatu sinyal masukan diskrit menjadi keluaran kode biner.

Enkoder disusun dari gerbang-gerbang logika yang menghasilkan keluaran biner sebagai hasil tanggapan adanya dua atau lebih variabel masukan. Hasil keluarannya dinyatakan dengan aljabar boole, tergantung dari kombinasi-kombinasi gerbang yang digunakan. Sebuah Enkoder harus memenuhi syarat perancangan $m \leq 2^n$. Variabel m adalah kombinasi masukan dan n adalah jumlah bit keluaran sebuah enkoder. Satu kombinasi masukan hanya dapat mewakili satu kombinasi keluaran. Perhatikan contoh tabel fungsi keluaran Enkoder berikut :

Input								Output		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	Y2	Y1	Y0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Tabel 3.1. Fungsi keluaran enkoder 8-ke-3

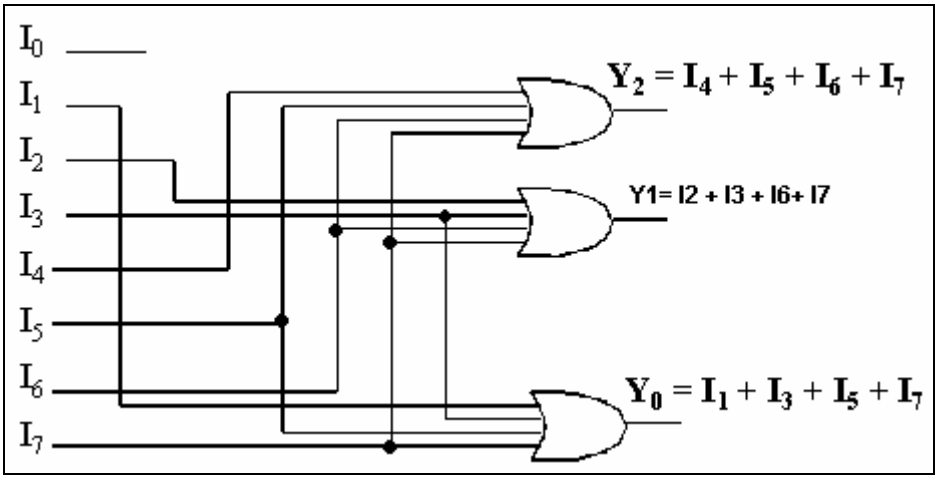
Dari tabel diatas, dapat dibuat fungsi keluaran sebagai berikut :

$Y0 = I_1 + I_3 + I_5 + I_7$

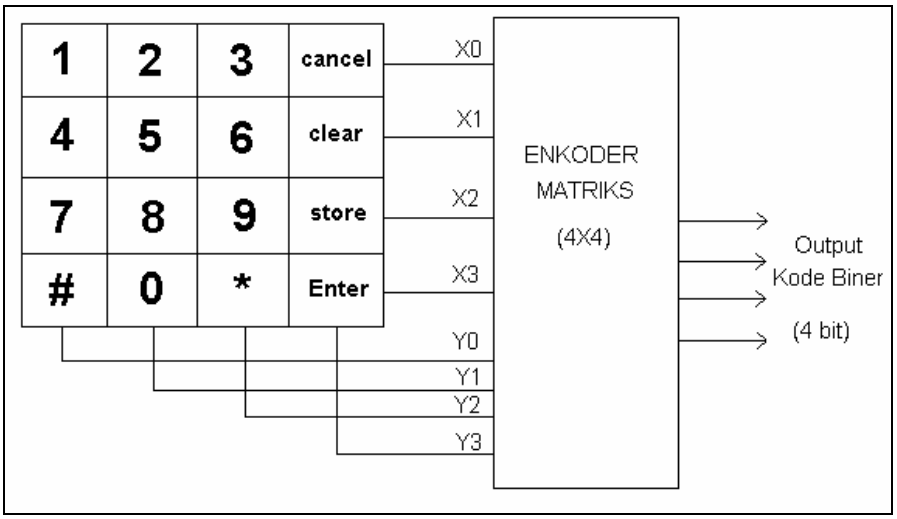
$Y1 = I_2 + I_3 + I_6 + I_7$

$Y2 = I_4 + I_5 + I_6 + I_7$

Dari persamaan tersebut, maka rangkaian gerbangnya dapat dibuat seperti pada gambar berikut :



Gambar 3.2.



Gambar 3.3.

3.1.2. Dekoder

Rangkaian Dekoder mempunyai sifat yang berkebalikan dengan Enkoder yaitu merubah kode biner menjadi sinyal diskrit. Sebuah dekoder harus memenuhi syarat perancangan $m \leq 2^n$. Variabel m adalah kombinasi keluaran dan n adalah jumlah bit masukan. Satu kombinasi masukan hanya dapat mewakili satu kombinasi keluaran. Perhatikan contoh tabel fungsi keluaran dekoder berikut :

X	Y	F0	F1	F2	F3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Tabel 3.2. Fungsi keluaran dekoder 2-ke-4

Dari tabel diatas, dapat dibuat fungsi keluaran sebagai berikut :

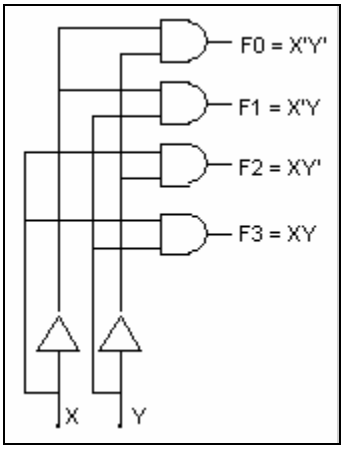
$F0 = \overline{X} \cdot \overline{Y}$

$F1 = \overline{X} \cdot Y$

$F2 = X \cdot \overline{Y}$

$F3 = X \cdot Y$

Dari persamaan tersebut, maka rangkaian gerbangnya dapat dibuat seperti pada gambar berikut :



Gambar 3.4.

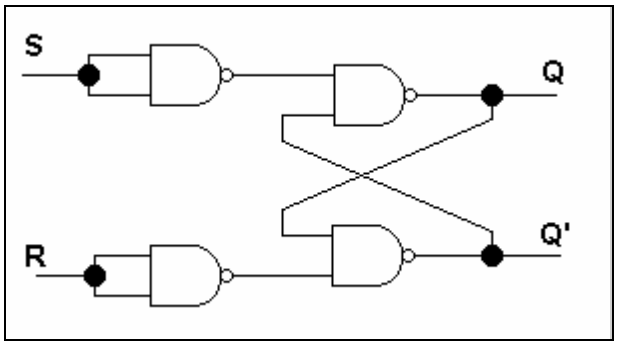
Membuat suatu Enkoder dan dekoder dapat dilakukan dengan dua cara yaitu pertama, menggunakan gerbang-gerbang dasar yang disusun membentuk fungsi Enkoder atau dekoder, kedua, menggunakan IC Enkoder atau dekoder yang banyak terdapat dipasaran. IC dekoder diaplikasikan pada *seven segment*, pengalamatan memori, dan sebagainya.

3.2. Rangkaian Logika Sequensial

Flip-flop adalah rangkaian utama dalam logika sequensial. Counter, Register, Memory, serta rangkaian sequensial lainnya disusun dengan menggunakan flip-flop sebagai komponen utama. Flip-flop adalah rangkaian yang mempunyai fungsi pengingat (*memory*). Artinya rangkaian ini mampu melakukan penyimpanan data sesuai dengan kombinasi masukan yang diberikan kepadanya. Ada beberapa macam flip-flop yang akan dibahas yaitu R-S flip-flop, J-K flip-flop, D flip-flop, dan T flip-flop. Ciri utama dari flip-flop adalah keluaran Q dan $\overline{Q}$ adalah selalu berlawanan / stabil (jika Q = 0 maka $\overline{Q}$ = 1, Jika $\overline{Q}$ = 1 maka Q =0). Karena kondisi dua keadaan stabil ini rangkaian flip-flop dinamakan juga dengan rangkaian bistabil.

3.2.1. R-S flip-flop

flip-flop ini terdiri dari dua masukan, yaitu S (set) dan R (reset). Serta dua keluarannya yaitu Q dan $\overline{Q}$. Kondisi Set adalah kondisi ketika Q berlogika 1. Sedangkan kondisi Reset adalah kondisi ketika Q berlogika 0. Perhatikan gambar berikut :



Gambar 3.5.

Untuk menganalisanya, asumsikan atau ambil permisalan keluaran sebelumnya.

1. Saat S = 0 dan R = 0. Misalkan keluaran sebelumnya $Q_n = 1$ dan $\overline{Q}_n = 0$. maka $Q_{n+1} = 1$ dan $\overline{Q}_{n+1} = 0$.
2. Saat S = 0 dan R = 0. Misalkan keluaran sebelumnya $Q_n = 0$ dan $\overline{Q}_n = 1$. maka $Q_{n+1} = 0$ dan $\overline{Q}_{n+1} = 1$. Dari dua analisa yang ada (1 dan 2), dapat disimpulkan bahwa saat S = 0 dan R = 0, maka keluarannya adalah sama dengan keluaran sebelumnya.
3. Saat S = 0 dan R = 1. Misalkan keluaran sebelumnya $Q_n = 1$ dan $\overline{Q}_n = 0$. maka $Q_{n+1} = 0$ dan $\overline{Q}_{n+1} = 1$.
4. Saat S = 0 dan R = 1. Misalkan keluaran sebelumnya $Q_n = 0$ dan $\overline{Q}_n = 1$. maka $Q_{n+1} = 0$ dan $\overline{Q}_{n+1} = 1$. Dari dua analisa yang ada (3 dan 4), dapat disimpulkan bahwa saat S = 0 dan R = 1, maka keluaran Q = 0.
5. Saat S = 1 dan R = 0. Misalkan keluaran sebelumnya $Q_n = 1$ dan $\overline{Q}_n = 0$. maka $Q_{n+1} = 1$ dan $\overline{Q}_{n+1} = 0$.
6. Saat S = 1 dan R = 0. Misalkan keluaran sebelumnya $Q_n = 0$ dan $\overline{Q}_n = 1$. maka $Q_{n+1} = 1$ dan $\overline{Q}_{n+1} = 0$. Dari dua analisa yang ada (5 dan 6), dapat disimpulkan bahwa saat S = 1 dan R = 0, maka keluaran Q = 1.
7. Saat S = 1 dan R = 1. Misalkan keluaran sebelumnya $Q_n = 1$ dan $\overline{Q}_n = 0$. maka $Q_{n+1} = 1$ dan $\overline{Q}_{n+1} = 1$.
8. Saat S = 1 dan R = 1. Misalkan keluaran sebelumnya $Q_n = 0$ dan $\overline{Q}_n = 1$. maka $Q_{n+1} = 1$ dan $\overline{Q}_{n+1} = 1$. (Ingat ciri utama flip-flop bahwa kondisi keluaran Q dan $\overline{Q}$ harus berlawanan). Dari dua analisa yang ada (7 dan 8), dapat

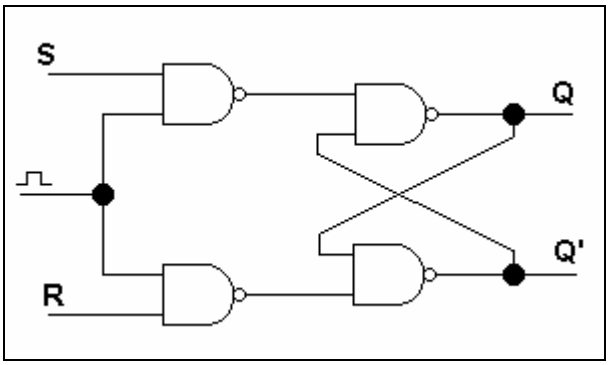
disimpulkan bahwa flip-flop R-S tidak diperbolehkan / dilarang saat S = 1 dan R = 1.

Input		Output	
S	R	Q_{n+1}	$\overline{Q}_{n+1}$
0	0	Q_n	$\overline{Q}_n$
0	1	0	1
1	0	1	0
1	1	Terlarang	

Tabel 3.3.

Perkembangan selanjutnya, flip-flop harus dipasang secara sinkron dengan unit lain dan sesuai dengan *clock*nya.

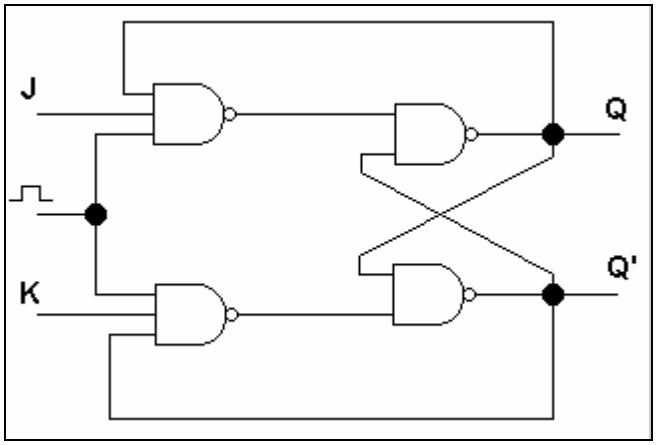
Perhatikan gambar flip-flop R-S dengan clock.



Gambar 3.6.

3.2.2. J-K Flip-flop

Flip-flop J-K merupakan penyempurnaan dari flip-flop R-S terutama untuk mengatasi kondisi terlarang seperti yang telah dijelaskan diatas. Pada kondisi masukan J = 1 dan K = 1 akan membuat kondisi keluaran berlawanan dengan kondisi keluaran sebelumnya. Sementara untuk keluaran berdasarkan kondisi-kondisi masukan yang lain semua sama dengan Flip-flop R-S.



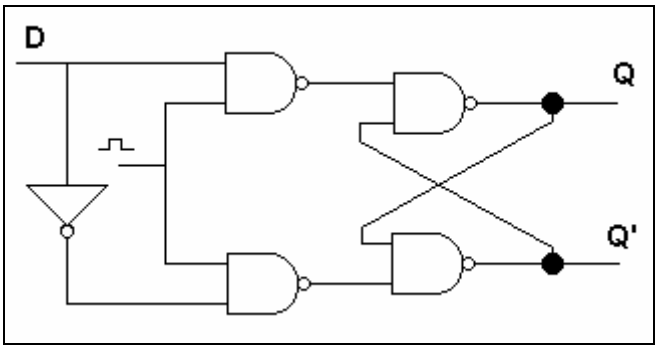
Gambar 3.7.

Input		Output	
J	K	Q_{n+1}	$\overline{Q}_{n+1}$
0	0	Q_n	$\overline{Q}_n$
0	1	0	1
1	0	1	0
1	1	$\overline{Q}_n$	Q_n

Tabel 3.4.

3.2.3. D Flip-Flop

Flip-flop D merupakan Flip-flop R-S yang memaksa untuk memiliki satu masukan dengan R selalu berlawanan dengan S, sehingga kondisi masukan S-R sama tidak akan pernah terjadi. Perhatikan gambar flip-flop D berikut.



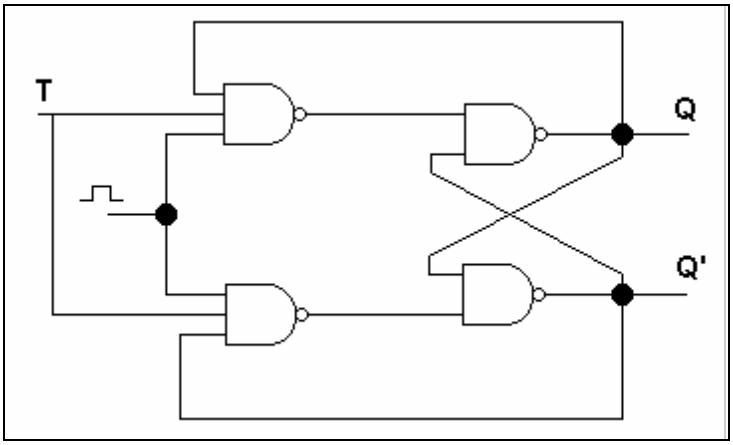
Gambar 3.8.

Input	Output
D	Q_{n+1}
0	0
1	1

Tabel 3.5.

3.2.4. T Flip-flop

Flip-flop T atau flip-flop *toggle* adalah flip-flop J-K yang kedua masukannya (J dan K) digabungkan menjadi satu sehingga hanya ada satu jalan masuk. Karakteristik dari flip-flop ini adalah kondisi keluaran akan selalu *toggle* atau berlawanan dengan kondisi sebelumnya apabila diberikan masukan logika 1. Sementara itu kondisi keluaran akan tetap atau sama dengan kondisi keluaran sebelumnya bila diberi masukan logika 0.



Gambar 3.9.

Input	Output
T	Q_{n+1}
0	Q_n
1	$\overline{Q}_n$

Tabel 3.6.

3.2.5. Register

Register adalah rangkaian logika yang digunakan untuk menyimpan data. Dengan kata lain, register adalah rangkaian yang tersusun dari satu atau beberapa flip-flop yang digabungkan menjadi satu. Flip-flop disebut juga sebagai register 1 bit. Jadi untuk menyimpan 4 bit data, register harus terdiri dari 4 buah flip-flop. Untuk menyimpan data pada register, dapat dilakukan dengan dua cara :

1. Disimpan secara sejajar (*Parallel In*) :

Pada cara ini semua bagian register atau masing-masing flip-flop diisi (dipicu) pada saat yang bersamaan.

2. Disimpan secara seri (*Serial In*) :

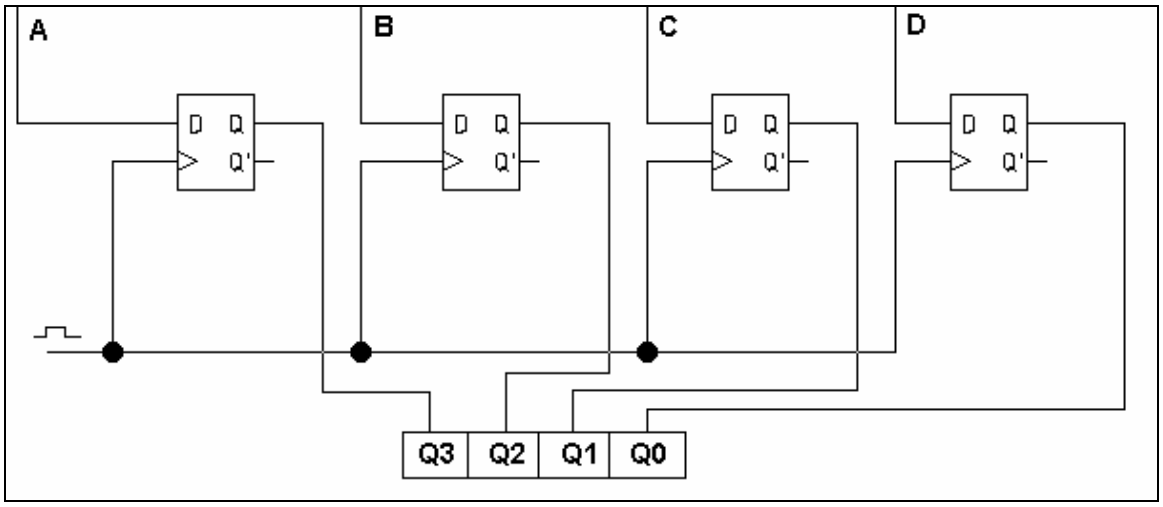
Pada cara ini, data dimasukkan bit demi bit mulai dari flip-flop yang paling ujung (dapat dari kiri atau dari kanan), dan digeser sampai semuanya terisi. Bila data digeser dari kanan kekiri disebut "Register geser kiri" (*Shift Left Register*), sebaliknya bila data digeser dari kiri kekanan disebut "Register geser kanan" (*Shift Right Register*). Register selain digunakan sebagai penyimpan data, juga sering digunakan sebagai *Counter* (lihat modul 2.2.6) dan operasi bilangan / ALU (lihat modul 3).

Seperti pada penyimpanan data, untuk mengeluarkan data juga dapat dilakukan dengan dua cara :

1. Dikeluarkan secara sejajar (*Parallel Out*)
2. Dikeluarkan secara seri (*Serial Out*)

- *Parallel In-Parallel Out* (PIPO)

Perhatikan gambar berikut :

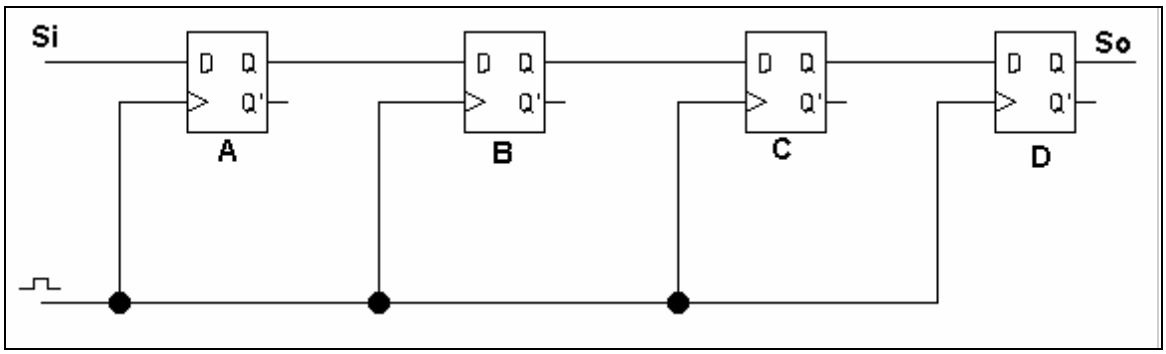


Gambar 3.10.

A, B, C, dan D adalah sinyal masukan. Saat *clock* (pemicu) diaktifkan (Logika 1), maka data yang ada akan dikeluarkan secara bersama-sama ke Q3, Q2, Q1, dan Q0. Saat *clock* kembali tidak dipicu (Logika 0), maka apapun masukannya, keluaran Q akan tetap.

- *Serial In-Serial Out* (SISO)

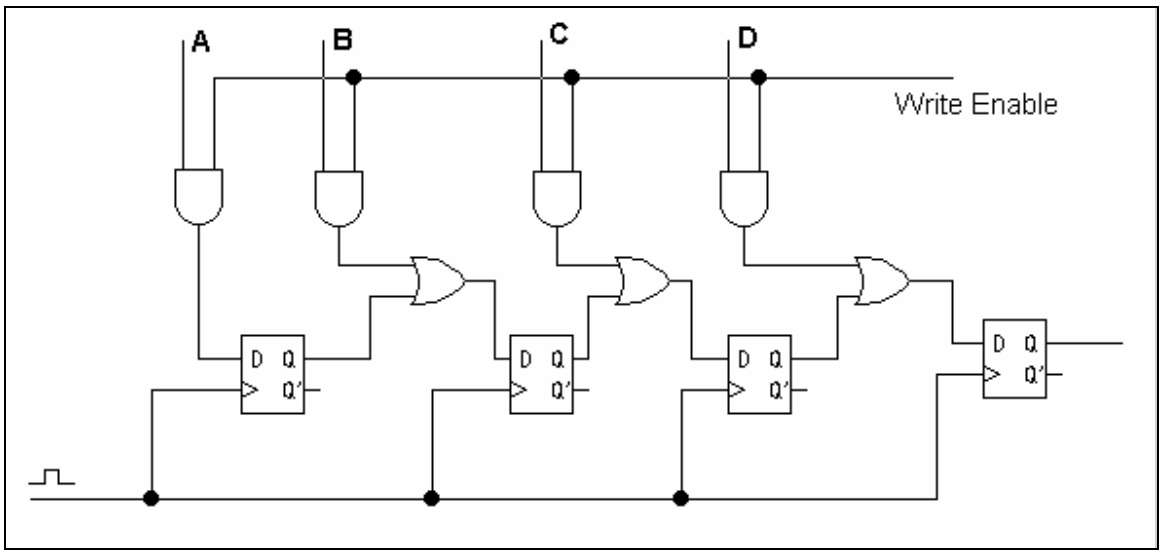
Perhatikan Gambar berikut :



Gambar 3.11.

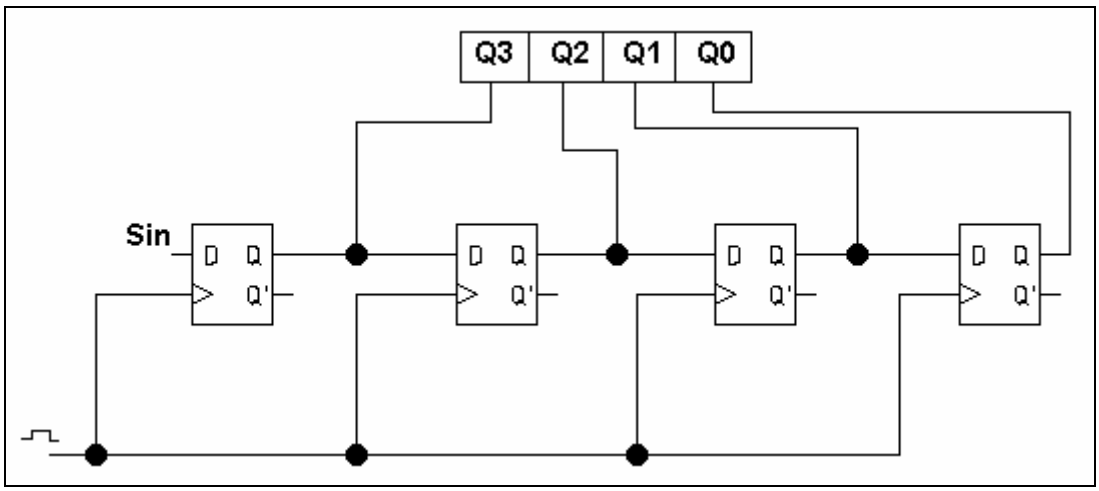
Saat sinyal *clock* diberikan pertama kali, data dari Si masuk ke flip-flop A, pada saat *clock* kedua, data dari flip-flop A masuk ke flip-flop B, demikian seterusnya, sampai keluar ke So. Jadi pada register SISO untuk membaca data pertama kali dibutuhkan jumlah *clock* yang sama banyak dengan jumlah flip-flop yang ada pada register (dalam hal ini adalah empat).

-*Parallel In – Serial Out* (PISO)



Gambar 3.12.

- *Serial In-Parallel Out* (SIPO)



Gambar 3.13