EEM212 - SAYISAL DEVRE TASARIMI DERS NOTLARI

DERS NOTU 8: SAYICILAR

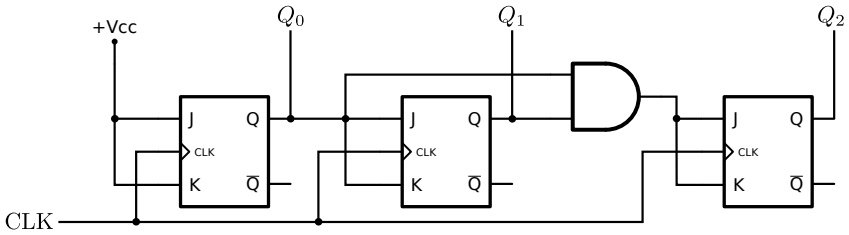
Dr. İsmail Öztürk *

<ismail.ozturk@amasya.edu.tr>

(**ver. 1.1**)

İçindekiler

*Amasya Üniversitesi Teknoloji Fakültesi EEM Bölümü
Daha fazla bilgi için: https://iozturk.com



Şekil 1: 3-bitlik senkron sayıcı.

1 Senkron Sayıcılar

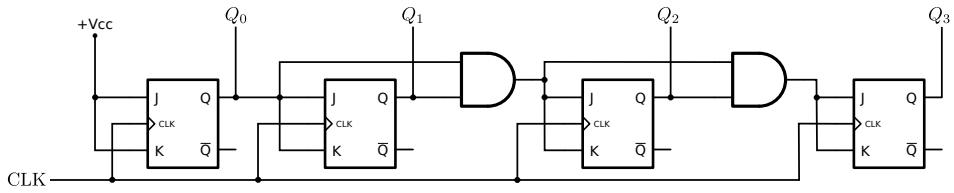
Senkron sayıcılar ikili tabanda sayma işlemini yerine getiren senkron ardışıl devrelerdir. Yani bu tür sayıcıların ortak saat (clock) girişleri vardır ve hafıza elemanlarının güncellenmesi aynı anda olur. Dijital sistemlerde sayma işlemini yerine getiren standart senkron sayıcı tasarımları vardır. Bu kısımda öncelikle bu standart sayıcıları göreceğiz. Bunlar herhangi bir ardışıl devre tasarımı yapmadan gerçekleştirilebildikleri için standart olarak adlandırılırlar. Daha sonra ardışıl devre tasarımıyla elde edilen diğer senkron sayıcıları göreceğiz.

1.1 Standart Sayıcılar

İkili tabanda sıralı sayma işlemine baktığımızda sayma işleminin belli bir paterne dayandığını görürüz. Mesela, üç bitlik sıralı sayma işlemi şu şekildedir: $Q_2Q_1Q_0 = 000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 101 \rightarrow 110 \rightarrow 111 \rightarrow 000$. Baktığımızda Q_0'ın (LSB) her durumda terslendirildiğini görürüz. Q_1 ise $Q_0 = 1$ olduğu zaman terslendirilmektedir. Q_2 (MSB) bitine baktığımızda ise sadece hem Q_1 hem de Q_0 bitleri 1 olduğu zaman terslendirildiğini görürüz ($011 \rightarrow 100$ ve $111 \rightarrow 000$). Bu paterne göre sayıcıyı terslendirme işlemi yapabilen JK FF veya T FF ile oluşturmak son derece kolaydır. Hatta bir önceki ders notunda gördüğümüz ardışıl devre tasarımını uygulamamız bile gerekmez.

Gerçekleştirim için T FF kullandığımızı varsayalım. Paterne baktığımızda tek yapmamız gereken Q_0 için ilk T FF'un girişini 1 yapmak. Bu sayede Q_0her saat darbesinde tam istediğimiz gibi terslendirilecektir. Daha sonra Q_0 çıkışını doğrudan ikinci flip flopun girişine bağlarsak, Q_1 çıkışı yine istediğimiz gibi $Q_0 = 1$ olduğu zaman terslendirilecektir ($Q_0 = 0$ olursa da önceki durumunu koruyacaktır). Son olarak üçüncü flip flop çıkışı olan Q_2'nin $Q_1 = Q_0 = 1$ olduğunda terslendirilmesini istiyoruz. Bunu ise üçüncü flip flopun girişine $T = Q_1Q_0$ (Q_1 VE Q_0) bağlantısını yaparak kolaylıkla elde edebiliriz.

Dolayısıyla, bahsetmiş olduğumuz 3-bitlik senkron sayıcı Şekil 1'deki gibi olacaktır. T FF olarak genellikle girişleri birbirine bağlanmış JK FF kullanıldığı için Şekil 1'deki gerçekleştirimde de JK FF kullanılmıştır.



Şekil 2: 4-bitlik senkron sayıcı.

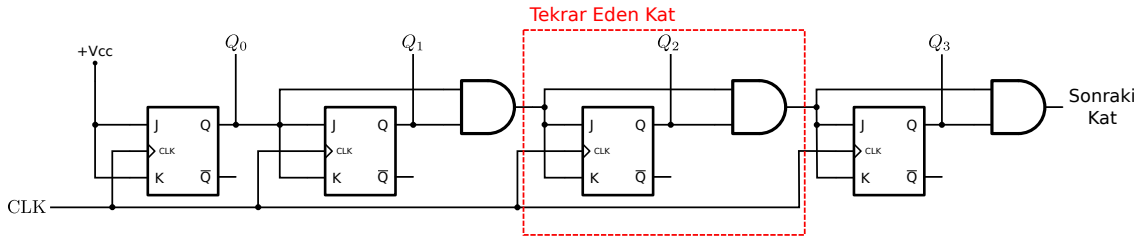
1.1.1 n-Bitlik Senkron Sayıcı

4-bitlik sayma işlemine bakalım:

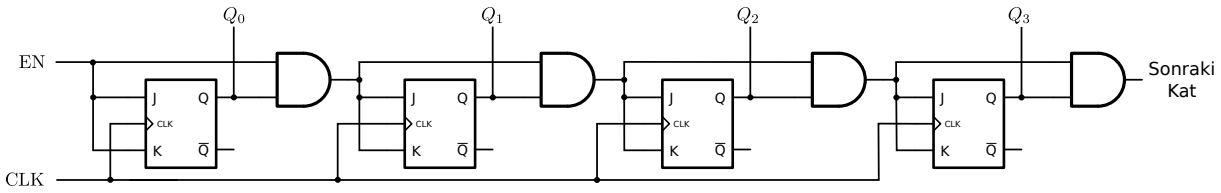
$$Q_3Q_2Q_1Q_0 = 0000 \rightarrow 0001 \rightarrow 0010 \rightarrow 0011 \rightarrow 0100 \rightarrow 0101 \rightarrow 0110 \rightarrow 0111 \rightarrow$$
$$1000 \rightarrow 1001 \rightarrow 1010 \rightarrow 1011 \rightarrow 1100 \rightarrow 1101 \rightarrow 1110 \rightarrow 1111 \rightarrow 0000$$

Görüldüğü üzere $Q_2Q_1Q_0$ bitleri tam olarak az önce görmüş olduğumuz 3-bitlik senkron sayıcı gibi çalışmaktadır. Dolayısıyla, $Q_2Q_1Q_0$ için kurulması gereken devre Şekil 1'deki ile aynı olacaktır. Peki Q_3 biti için kullanacağımız dördüncü flip flopa nasıl bir bağlantı yapmamız lazım? Sayma işlemine baktığımızda, Q_3 bitinin sadece $Q_2 = Q_1 = Q_0 = 1$ olduğu zaman terslendirildiğini görürüz. Diğer tüm durumlarda Q_3 önceki değerini korumaktadır. Buna göre, 4-bitlik sayıcı elde etmek için dördüncü flip flopun girişine $T = Q_2Q_1Q_0$ (Q_2 VE Q_1 VE Q_0) bağlantısı yapmamız yeterlidir. Bunu yaptığımızda Şekil 2'deki 4-bitlik sayıcıyı elde ederiz.

Buradan 5 veya daha fazla bitlik senkron sayıcılar elde etmek için ne yapmanız gerektiğini kayrayabilmiş olmanız lazım. 5 bitlik sayıcı için önceki dört bitin devresi Şekil 2'dekiyle aynı iken Q_4'ün flip flopuna ise $T = Q_3Q_2Q_1Q_0$ bağlanmalıdır. Yani VE kapısı ile flip flop tıpkı 4-bitlik sayıcı elde etmek için 3-bitlik sayıcının sonuna bağladığımız gibi bağlanmalıdır. Bunu **tekrar eden kat** olarak adlandırabiliriz. Dolayısıyla, n-bitlik bir senkron sayıcıyı genel olarak kısaca Şekil 3'deki gibi gösterebiliriz.



Şekil 3: n-bitlik senkron sayıcının genel yapısı.



Şekil 4: Enable girişli senkron sayıcı.

1.1.2 Enable Girişli Sayıcı

Yukarıda görmüş olduğumuz sayıcılar bir kez başlatıldıklarında sürekli sayma işlemi yaparlar. Bu sayıcıları mevcut halleriyle herhangi bir şekilde durduramayız. Bu amaçla, sayma işlemini istediğimiz zaman durdurabilmek için sayıcı devresine "enable" girişi bağlamamız gerekir. Enable girişi 1 olduğunda devre normal sayma işlemi yaparken, enable 0 olduğu müddetçe sayıcı devresi önceki durumunu korumalıdır.

Enable girişini elde etmenin en kolay yolu Şekil 3'deki $+V_{cc}$ ile sürekli 1 yapılan ilk flip flopun girişini enable olarak kullanmaktır. Bu durumda enable 0 olduğunda ilk flip flop istediğimiz gibi önceki durumunu koruyacaktır. Fakat, sadece bunu yapmak diğer flip flopları etkilemeyecektir. Yani diğer flip floplar normal çalışmasını devam ettirecektir.

Enable girişinin diğer flip floplara da etki etmesi için Q_0 çıkışını ikinci flip flopa doğrudan bağlamak yerine bir VE kapısıyla beraber bağlamak gerekir. Bu VE kapısının bir girişi enable'a diğer girişi de Q_0'a bağlanırsa enable 0 yapıldığında tüm flip flop girişleri 0 yapılmış olur. Böylece istediğimiz elde ederiz.

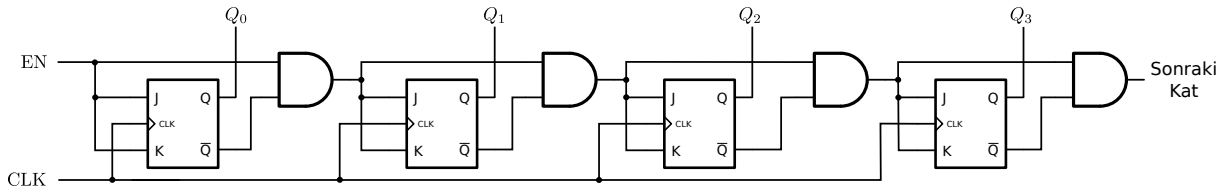
Buna göre enable girişli sayıcının devresi Şekil 4'deki gibi olmalıdır. Şekilden artık tüm katların aynı olduğunu ve enable 0 olduğunda tüm flip flop girişlerinin 0 olması gerektiğini görebilmeniz lazım.

1.1.3 Aşağı Sayıcı

Önceki örnekte görmüş olduğumuz sayıcı yukarı doğru sayıyordu (örneğin 3-bit için $000 \rightarrow 001 \rightarrow \ldots \rightarrow 111 \rightarrow 000$). Sayıcı devrelerinden bazen aşağı doğru da (yani tam tersi şekilde) saymaları istenir. Yine 3-bit için aşağı saymayı incelediğimizde değişimin şu şekilde olduğu görülür:

$$Q_2Q_1Q_0 = 111 \rightarrow 110 \rightarrow 101 \rightarrow 100 \rightarrow 011 \rightarrow 010 \rightarrow 001 \rightarrow 000 \rightarrow 111$$

Görüldüğü üzere Q_0 (LSB) yine her durumda terslendirilmektedir. Fakat, bu sefer Q_1 sadece $Q_0 = 0$ olduğunda terslendirilirken; Q_2 sadece $Q_1 = Q_0 = 0$ olduğu zaman terslendirilmektedir. Yani yukarı sayma durumunda VE kapıları ile 1 olma durumunu kontrol ederken aşağı saymak için VE kapıları ile 0 olma durumunu tespit etmemiz gerekir.



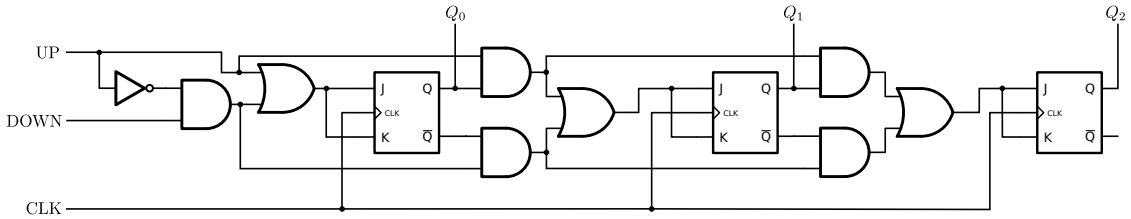
Şekil 5: Enable girişli senkron aşağı sayıcı.

Flip floplarda çıkışların tersi de ($\overline{Q}$) bulunduğu için 0 olma durumları yerine çıkışların tersini kullanıp yine 1 olma durumlarını kontrol edebiliriz. Yani, $Q_0 = 0$ olma durumunu kontrol etmek için $\overline{Q_0} = 1$ olma durumunu kontrol edebiliriz. Benzer şekilde, $Q_1 = Q_0 = 0$ olma durumu yerine $\overline{Q_1} = \overline{Q_0} = 1$ olma durumunu kontrol edebiliriz. Bunun için tek yapmamız gereken Şekil 4'deki yukarı sayıcıdaki VE kapılarının Q çıkışlarına bağlı olan girişlerini $\overline{Q}$ çıkışlarına bağlamaktır. Bu sayede ekstra mantık kapısı kullanmaksızın devreyi Şekil 5'deki gibi aşağı sayıcı devreye rahatlıkla dönüştürebiliriz.

1.1.4 Yukarı / Aşağı Sayıcı

Şekil 4'deki yukarı sayıcı devresi ile Şekil 5'deki aşağı sayıcı devresini katlar arasında iki tane VE kapısı kullanarak birleştirebiliriz. Bu farklı VE kapılarını ise VEYA kapıları ile flip flop girişlerine bağlarız. Elde edilen devre Şekil 6'deki gibi olacaktır. Bu düzenlemede UP girişi 1 olduğunda devre yukarı sayarken, DOWN girişi 1 olduğunda devre aşağı sayacaktır. Girişte DOWN girişinden sonra kullanılan DEĞİL ile VE kapıları hem UP hem de DOWN girişleri 1 yapılırsa devrenin sadece yukarı saymasını garanti altına alır. Hem UP hem de DOWN 0 yapıldığında ise devre saymayı bırakıp önceki durumu koruyacaktır.

Devrenin sayfaya sığması için devreyi 3-bit olarak kurmuş olsak da aynı katları sona ekleyerek sayıcıyı istediğimiz bit uzunluğunda kullanabiliriz. Neticede yaptığımız şey yukarıda elde ettiğimiz sayıcıları birleştirmekten ibaret.



Şekil 6: Senkron yukarı / aşağı sayıcı.

1.2 Diğer Senkron Sayıcılar

Yukarıda görmüş olduğumuz sayıcılar baştan başlayarak sıralı bir şekilde olası bütün durumları saymaktaydı. Dijital uygulamalarda çoğu zaman sadece belli sayılar arasında sayma yapan sayıcılara da ihtiyaç duyulmaktadır. Mesela, 1-3-5-7-1 şeklinde sadece tek sayıları sayan bir sayıcı gibi. Bu tür durumlarda daha önce görmüş olduğumuz senkron ardışıl devre tasarımını kullanırız.

Bu tür devreleri tasarlarken dikkat etmemiz gereken husus kullanılmayan durumları ne yapacağımızdır. Eğer bu tür durumları tasarımda göz önünde bulundurmazsak sistem ilk başlatıldığında veya herhangi bir beklenmedik durumda sayıcı kullanılmayan durumlardan birine geçebilir ve sonrasında ne olacağı tamamen bizim kontrolümüzden çıkmış olur. Bu tür belirsizliklerin önüne geçmek için kullanılmayan durumlar mutlaka kullanılan durumlardan birine yönlendirilmelidir. Aksi söylenmediği müddetçe kullanılmayan durumları başlangıç durumuna yönlendirebilirsiniz.

Örnek 1.1:

0-3-5-1-6-0 şeklinde sayan bir sayıcıyı en sade haliyle JK FF kullanarak tasarlayınız.

—

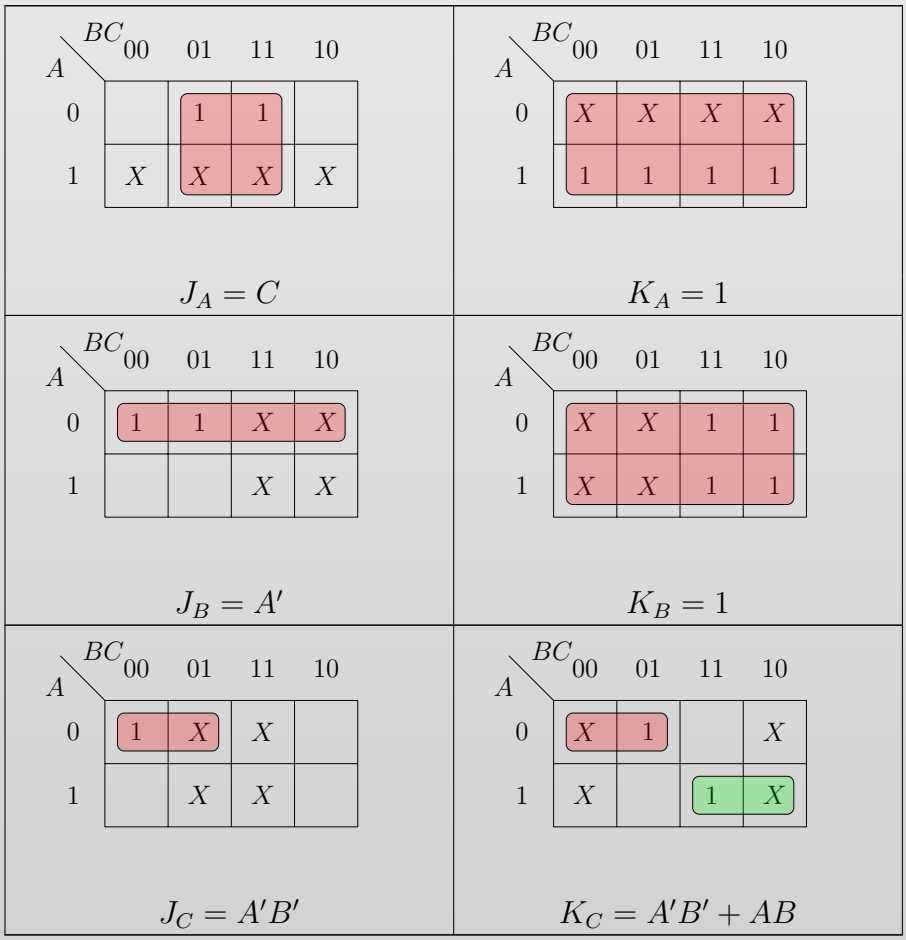
Sayılar 3-bitle ifade edilebileceği için sayıcı 3-bit olmalı. MSB A olmak üzere 3-bitlik durumları ABC ile ifade edersek durum tablosu aşağıdaki gibi olacaktır (kullanılmayan tüm durumlar 0 başlangıç durumuna yönlendirilmektedir):

Şimdiki Durum			**Sonraki Durum**			**Flip Flop Girişleri**					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_B	K_B
0	0	0	0	1	1	0	X	1	X	1	X
0	0	1	1	1	0	1	X	1	X	X	1
0	1	0	0	0	0	0	X	X	1	0	X
0	1	1	1	0	1	1	X	X	1	X	0
1	0	0	0	0	0	X	1	0	X	0	X
1	0	1	0	0	1	X	1	0	X	X	0
1	1	0	0	0	0	X	1	X	1	0	X
1	1	1	0	0	0	X	1	X	1	X	1

Buradan flip flop girişlerine ait kombinasyonel devreleri miniterimlerin toplamı cinsinden aşağıdaki gibi elde ederiz:

$$J_A = \sum(1,3) + d(4,5,6,7)$$
$$K_A = \sum(4,5,6,7) + d(0,1,2,3)$$
$$J_B = \sum(0,1) + d(2,3,6,7)$$
$$K_B = \sum(2,3,6,7) + d(0,1,4,5)$$
$$J_C = \sum(0) + d(1,3,5,7)$$
$$K_C = \sum(1,7) + d(0,2,4,6)$$

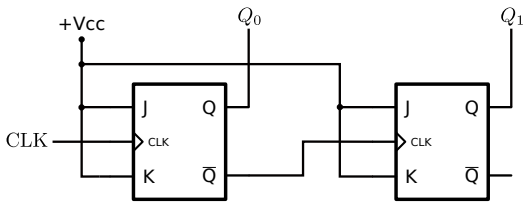
Kombinasyonel devreleri en sade halleriyle aşağıdaki gibi buluruz:



Elde ettiğimiz bu Boole eşitlikleri ile sayıcı devresinin nasıl kurulacağını önceki ders notunda görmüştük.

2 Asenkron Sayıcılar

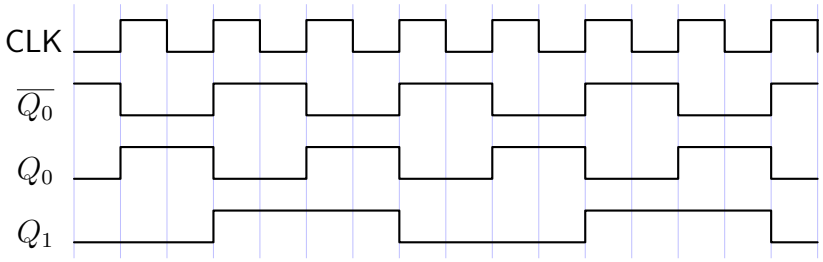
Asenkron sayıcılarda sadece bir veya birkaç tane flip flopa periyodik saat darbesi uygulanırken diğer flip floplar saat darbesi yerine başka flip flopların çıkışlarıyla güncellenirler. Yani tüm flip floplara aynı saat darbesi uygulanmaz ve bu nedenle flip floplar aynı anda güncellenmez. Asenkron sayıcıların en yaygın kullanılan türü **ripple sayıcı**lardır. Ripple sayıcılarda sadece LSB değerini taşıyan flip flopa saat darbesi uygulanırken diğer tüm flip flopların saat girişleri kendisinden bir önceki flip flopun çıkışına bağlanır. Bu sayede LSB'ye yakın olan flip floplar hemen güncellenirken MSB'ye yakın olan flip flopların güncellenmesi için belli bir süre geçer. Bu bir nevi dalgalanma etkisi yaratır. İşte ripple (ing. dalgalanma) ismi buradan gelir.



Şekil 7: 2-bit asenkron (ripple) sayıcı.

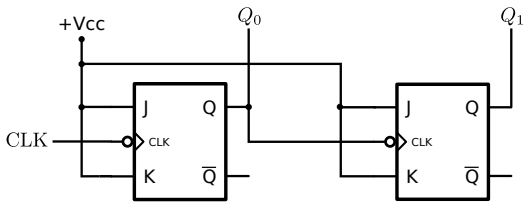
Dijital devrenin zaman diyagramı saat darbelerine karşı giriş ve çıkışların değişiminin alt alta çizilmesidir. Yani saatin yükselen ya da düşen kenarında giriş ve çıkışların yeni değerinin ne olacağı (H veya L) çizilir.

Ripple sayıcıları D veya T FF (ya da girişleri birbirine bağlanmış JK FF) ile kurabiliriz. En basit 2-bitlik ripple sayıcı örneği Şekil 7'deki gibidir. Şekildeki ripple sayıcının çalışmasını anlamanın en iyi yolu sayıcıya ait zaman diyagramını oluşturmaktır. Sayıcının zaman diyagramı aşağıdaki gibidir:

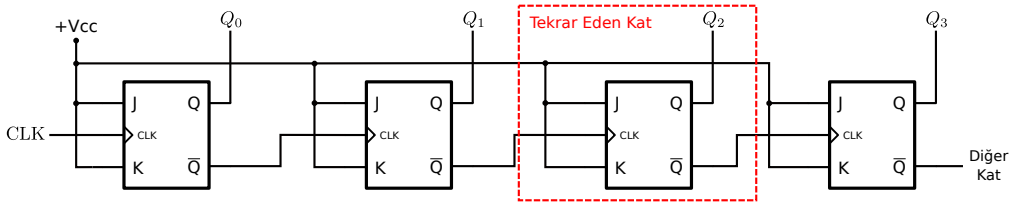


Zaman diyagramına baktığımızda $Q_0 = Q_1 = 0$ başlangıç durumu için hazırlandığını görürüz. Yani, Q_0 LSB ve Q_1 MSB olduğundan sayıcının başlangıç durumu ondalık olarak 0'dır. Flip floplar yükselen kenar tetiklemeli olduğu için güncellemeler için yükselen kenarları takip etmemiz gerekir. İlk flip flop doğrudan saate (CLK) bağlı olduğu için saat darbelerinin her yükselen kenarında Q_0 (ve dolayısıyla $\overline{Q_0}$) güncellenecektir. İkinci flip flopun saat girişi ise $\overline{Q_0}$ çıkışına bağlı olduğundan, Q_1 çıkışı $\overline{Q_0}$ sinyalinin her yükselen kenarında güncellenmelidir.

Buna göre, JK FF girişleri 1 olduğu için saat darbelerinin ilk yükselen kenarında Q_0 terslendirilir. Yani $Q_0 = 1$ ve $\overline{Q_0} = 0$ olur. Bu durumda $\overline{Q_0}$ sinyali yüksekten düşüğe geçiş yaptığı için düşen kenar oluşturur ve ikinci flip flop güncellenmez (Q_1 aynı



Şekil 8: Düşen kenar tetiklemeli 2-bit asenkron (ripple) sayıcı.



Şekil 9: n-bitlik asenkron (ripple) sayıcının genel yapısı.

kalır). Saatin sonraki yükselen kenarında ise terslendirme sonucu $Q_0 = 0$ ve $\overline{Q_0} = 1$ olur. Bu durumda $\overline{Q_0}$ alçaktan yükseğe geçiş yaptığı için ikinci flip flop yükselen kenar nedeniyle güncellenir ve $Q_1 = 0$ terslendirilerek $Q_1 = 1$ olur. Diğer durum değişiklikleri benzer şekilde belirlenir. Elde ettiğimiz zaman diyagramından Q_1Q_0 bitlerinin sırasıyla $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \ldots$ şeklinde değiştiğini görürüz. Yani ondalık olarak sayma işlemi $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 0 \rightarrow \ldots$ şeklindedir.

Eğer kullanılan JK flip flop düşen kenar tetiklemeli ise sonraki flip flopun saat girişi önceki flip flopun Q_0 çıkışına bağlanmalıdır. Bu durumda yeni bağlantı Şekil 8'de görüldüğü gibi olacaktır. Bu bağlantı şekline ait zaman diyagramı yukarıda gördüğümüz yükselen kenar tetiklemeli bağlantıya ait zaman diyagramıyla birebir aynı olacaktır. Yani sayma işlemi yine $00 \rightarrow 01 \rightarrow 10 \rightarrow 11 \rightarrow 00 \rightarrow \ldots$ şeklinde olacaktır. Dikkat etmeniz gereken, zaman diyagramı oluşturulurken Q_1'in artık $\overline{Q_0}$'ın her yükselen kenarı yerine Q_0 değerinin her düşen kenarında güncelleniyor olmasıdır.

2.1 n-bitlik Asenkron Sayıcı

Şekil 7'deki 2-bitlik sayıcıyı n-bit için genelleştirmek son derece kolaydır. Tek yapmamız gereken sona yeni bir flip flop ekleyip saat girişine bir önceki flip flopun $\overline{Q}$ çıkışını bağlamaktır. Buna göre, n-bitlik ripple sayıcının genel yapısını Şekil 9'deki gibi gösterebiliriz. n-bit olarak kuracağınız bu tür ripple sayıcılar ondalık olarak 0 ile $2^n - 1$ arasında sayacaktır.

Eğer düşen kenarlı flip flop kullanıyorsanız bu durumda $\overline{Q}$ çıkışı yerine Q çıkışı bir sonraki flip flopun saat girişine bağlanmalıdır. Bunun şeklini ayrıca vermiyoruz. Şekil 9'de $\overline{Q}$ çıkışları yerine Q çıkışlarını düşen kenar tetiklemeli saat girişine bağlayarak (yani Şekil 8'i genelleştirerek) düşen kenar tetiklemeli n-bit asenkron sayıcının şeklini elde edebilirsiniz.

Asenkron ripple sayıcılara enable fonksiyonu eklemek son derece kolaydır. Tek yapmamız gereken Şekil 9'deki $+V_{cc}$ bağlantısını "enable" girişi olarak kullanmaktır. Bu durumda enable 0 olduğunda sayıcı mevcut durumunu koruyacaktır. Enable 1 olduğunda ise (yani $+V_{cc}$ bağlantısı yaptığımızda) sayıcı normal çalışmasını sergileyecektir.

2.2 Asenkron Aşağı Sayıcı

Şekil 9'deki devrenin aşağı doğru sayması için yapabileceğimiz iki tane alternatifimiz vardır:

1. Sayıcı çıkışlarını Q yerine $\overline{Q}$'dan almak.
2. Asenkron saat girişlerini $\overline{Q}$ çıkışına bağlamak yerine Q çıkışına bağlamak.

Aşağı sayıcı elde etmek için bunlardan sadece birini yapmanız gerekir. Her ikisini birden yaparsanız tekrar yukarı sayıcı elde etmiş olursunuz.

Eğer düşen kenar tetiklemeli flip flop kullanıyorsanız madde 1 yine aynı olacaktır, fakat madde 2 için tam tersini yapmanız gerekir.

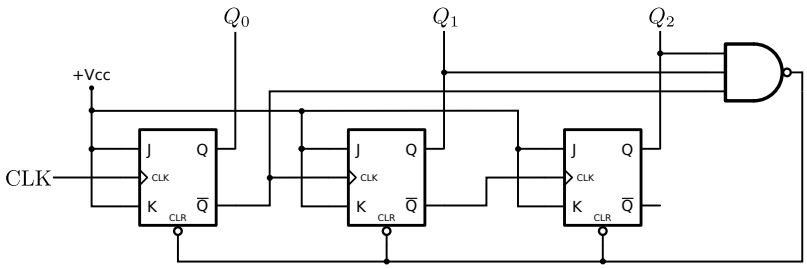
Çalışma

3-bitlik asenkron (ripple) aşağı sayıcı tasarlayıp zaman diyagramını oluşturunuz. Elde ettiğiniz zaman diyagramından sayma sırasını belirleyiniz.

2.3 Mod-N Sayıcılar

Şimdiye kadar görmüş olduğumuz n-bitlik sayıcılar 0 ile $2^n - 1$ arasında saymaktaydı. Bunlar aynı zamanda mod-2^n sayıcılar olarak da adlandırılabilir. Fakat, bazen elimizdeki n-bitlik sayıcının $2^n - 1$ değerine kadar saymak yerine daha düşük bir N değerine kadar saymasını isteriz. Mesela, elimizde 3-bitlik bir asenkron sayıcı olsun. Bu sayıcı normalde 0 ile 7 arasında sayacaktır. Yani sayıcı mod-8 bir sayıcıdır. Bu sayıcıyı mantık kapısı kullanarak mod-6 sayıcıya dönüştürmemiz mümkündür. Yani sayıcı 0 ile 7 arasında değil de 0 ile 5 arasında sayabilir.

Bir mod-2^n sayıcıyı ($N < 2^n$ olmak üzere) mod-N sayıcıya dönüştürmek için yapmamız gereken sayıcı N değerine gelir gelmez flip flopların clear (reset) girişlerini aktifleştirerek tüm flip flop çıkışlarını 0 yaparak sayma işlemini başa döndürmektir. Bunu yapmak için sadece N değerinde çıkışı 1 olan bir mantık kapısı bağlantısını clear girişine bağlamamız gerekir.



Şekil 10: Asenkron mod-6 sayıcı devresi.

Bunu yukarıda bahsettiğimiz 3-bitlik mod-6 sayıcı örneğiyle açıklayalım. Devreyi Şekil 9'deki gibi 3-bit olarak kurduğumuzda sayma işlemi normalde $Q_2Q_1Q_0 = 000 \rightarrow 001 \rightarrow 010 \rightarrow 011 \rightarrow 100 \rightarrow 101 \rightarrow 110 \rightarrow 111 \rightarrow 000$ şeklinde olacaktır. Mod-6 sayıcı elde etmek için sayıcı $(000)_2 = 0$ ile $(101)_2 = 5$ arasında saymalıdır. Yani $(110)_2 = 6$ değerini görür görmez sayıcı resetlenerek başa dönmelidir.

Flip flop clear girişlerinin "aktif düşük" olduğunu varsayalım. Yani clear girişi 0 olduğunda flip flop resetlenecektir. 0 ile 6 arasındaki üç bitlik ikili sayılara baktığımızda $(110)_2 = 6$ değerinde $Q_2 = Q_1 = 1$ ve $Q_0 = 0$ olduğunu görürüz. Dolayısıyla, flip flopları $(110)_2 = 6$ değerinde resetleyebilmek için Q_2, Q_1 ile $\overline{Q_0}$ çıkışlarını alıp VEDEĞİL kapısına sokarsak, VEDEĞİL kapısının çıkışı sadece $Q_2 = Q_1 = 1$ ve $Q_0 = 0$ olduğunda sıfırlanacaktır. Tahmin edebileceğiniz üzere bu VEDEĞİL kapısının çıkışını da tüm flip flopların aktif düşük clear girişine bağlarız. Böylelikle, sayıcıda $(110)_2 = 6$ değeri görülür görülmez sayıcı resetlenir ve başa döner.

Buna göre, mod-6 ripple sayıcı devresi Şekil 10'deki gibi olacaktır. Bu sayıcı $(000)_2 = 0$'dan başlayıp $(101)_2 = 5$'e kadar sayacaktır. Sayıcı $(110)_2 = 6$ olur olmaz VEDEĞİL kapısı çıkışı 0 olacağı için sayıcı $(110)_2$ durumunda kalamaz ve resetleme işlemi asenkron olduğu için hemen $(000)_2$ başlangıç durumuna döner. Sonuç olarak mod-6 sayıcı 0 ile 5 arasında sayma işlemi yapar. Diğer mod-N sayıcılar da benzer şekilde bir mantık kapısı ile flip flopların resetlenmesi prensibine göre tasarlanır.

0'dan 6'ya kadar olan $(Q_2Q_1Q_0)_2$ şeklindeki ikili sayılarda sadece 6 değerinde $Q_2 = Q_1 = 1$ olduğuna dikkat edin. Buna göre, Şekil 10'deki mod-6 sayıcıyı sadece iki girişli VEDEĞİL kapısı kullanarak da tasarlayabileceğinizi fark edebildiniz mi?

3 Senkron ve Asenkron Sayıcıların Kıyaslanması

Her iki tür sayıcıları da görmüş olduğumuza göre bunların bir kıyaslamasını yapabiliriz. Görmüş olduğunuz üzere bit sayısı arttıkça senkron sayıcılarda kullanılması gereken eleman sayısı da artmaktadır. Yani devre karmaşıklığı artmaktadır. Asenkron sayıcılarda bit sayısını ekstra eleman kullanmadan sadece flip flop ekleyerek

arttırabildiğimiz için asenkron sayıcılarda bit sayısını arttırmak devre karmaşıklığını aynı oranda arttırmaz.

Senkron sayıcılarda senkron güncelleme yapılır yapılmaz sayıcı yeni durumuna geçer. Yeni duruma geçme süresi tek bir flip flopun güncellenmesi için gerekecek yayılma gecikmesi kadardır. Asenkron sayıcılarda ise sayma işlemi baştan başlayarak sırasıyla en sondaki flip flopun güncellenmesiyle sona erer. Yani sayıcının yeni duruma geçmesi için gerekecek süre kullanılan tüm flip flopların güncellenmesi için gereken toplam yayılma gecikmesine eşittir. Bu nedenle, asenkron sayıcılar senkron sayıcılara göre daha yavaştırlar. İlk flip flopun saat periyodu bu toplam yayılma gecikmesinden daha düşük olamaz. Dolayısıyla, asenkron sayıcıların bir maksimum çalışma frekansı vardır ve sayıcının bit sayısı arttıkça bu maksimum çalışma frekansı düşer.

Versiyon Notları

ver. 1.1:

- Düşen kenar tetiklemeli asenkron sayıcılar nota eklendi. Zaman diyagramına düşey çizgiler eklendi.