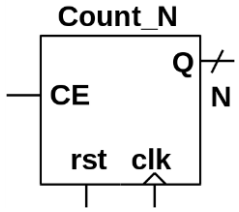
Deney 9: Ardışıl Lojik Devrelerde Blok Tasarım

Genel Bilgiler:

Lojik devrelerin karmaşıklığı arttıkça matematiksel model, durum tablosu ve durum diyagramı gibi klasik devre sentezi yöntemleri yetersiz kalmaktadır. Bu nedenle karmaşık devrelerin tasarlanabilmesi için problemin alt problemlere parçalandığı blok tasarım yöntemleri geliştirilmiştir. Bu deney kapsamında tasarlanması istenen devreyi uç uca (kaskad) hiyerarşik bağlanabilen alt devrelere bölme yöntemi ele alınmıştır.

Şekil 9.1'deki N bitlik sayıcının devresinde rst ucu reset girişiyken CE de 1 ve 0 değerlerinde sırasıyla ileri saymaya izin veren veya Q'nun değerini korumasını sağlayan bir kontrol girişidir.

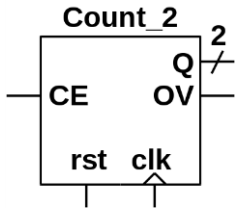


Şekil 9.1 Tasarlanmak istenen N bitlik sayıcı devresinin uç blok diyagramı

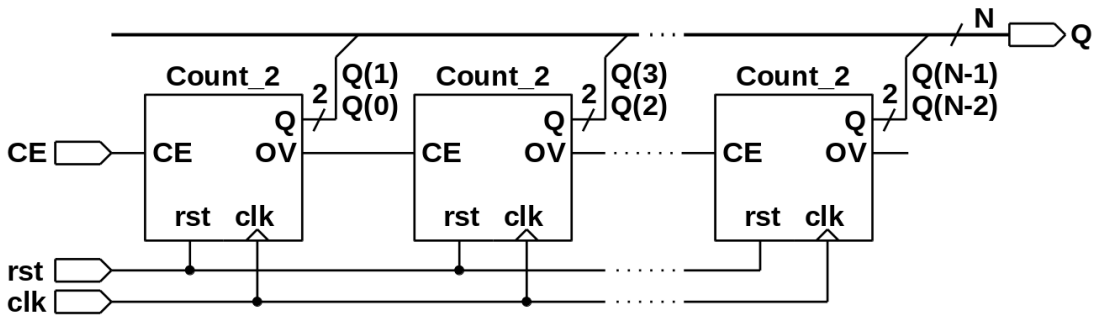
N=10 için problemi parçalamadan klasik bir tasarım yapılırsa 2 giriş ve 10 durum için klasik tasarım yöntemleri aşağıdaki darboğazlara girer:

- Durum tablosu 2^{12} = 4096 satırdan oluşur;
- Durum diyagramı 2^{10} = 1024 durum ve her bir durumdan çıkan 2^2 = 4 oktan oluşur;
- Problem Boole cebri ifadesini doğrudan yazmak için fazlasıyla karmaşıktır.

Oysaki problem "N bitlik sayıcıyı 2 bitlik sayıcıların uç uca bağlanması" şeklinde parçalanırsa Şekil 9.1'deki N bitlik sayıcı Şekil 9.2'deki 2 bitlik sayıcılardan N/2 tane kullanarak Şekil 9.3'teki gibi tasarlanabilir. Burada OV çıkışı yalnızca Q = 11'den 00'a taştığı saat kenarında değeri 1, diğer tüm kenarlarda ise 0 olan bir kontrol çıkışıdır. OV'nin bu şekilde tanımlanmasıyla 2 bitlik sayıcıya her taştığında kendisinden sonraki sayıcıyı CE ucu üzerinden aktif hale getirme (tetikleme) yeteneği kazandırılmıştır. Böylece her 2 bitlik taşma bir sonraki 2 bitin bir artmasını tetikleyerek zincirleme bir sayıcı donanımı ortaya çıkartır.

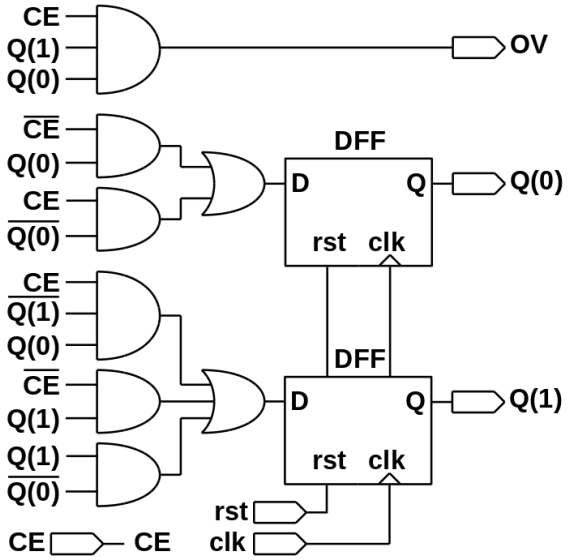


Şekil 9.2 N bitlik sayıcı tasarımında kullanılan 2 bitlik uç uca bağlanabilen bir sayıcı devresinin uç blok diyagramı



Şekil 9.3 2 bitlik sayıcıların uç uca bağlanmasıyla oluşturulmuş N bitlik sayıcı devresinin blok diyagramı

Devrenin tamamlanmasındaki son adım Şekil 9.2'deki 2 bitlik uç uca bağlanabilen sayıcı bloğunun içinin devre olarak tasarlanmasıdır. CE ve rst girişleri ile 2 bitlik Q durumuna sahip bu devre bilinen tasarım yöntemleriyle kolaylıkla tasarlanabilir. Reset girişi olan D bellek elemanlarının ve bilinen tasarım yöntemlerinin kullanılmasıyla elde edilmiş 2 bitlik sayıcı devresi Şekil 9.4'te verilmiştir.



Şekil 9.4 2 bitlik uç uca bağlanabilen sayıcı devresi

N=10 için örnek bir PAL programı deneyin sonunda verilmiştir.

Deney Öncesi Yapılacak İşlemler:

1- GAL22V10 tümleşik devresinin özelliklerini daha ayrıntılı bir şekilde araştırınız.
2- Atmel firmasının ücretsiz sağladığı WinCUPL geliştirme ortamını edinerek bilgisayarınıza kurunuz.
3- Deneyin sonunda yer alan ve Şekil 9.1'i N=10 için gerçekleyen örnek PAL kodunu inceleyiniz, WinCUPL'da yazınız ve hatasız olarak derleyiniz.
4- WinSIM programıyla bir önceki adımda derlediğiniz kodun doğru çalıştığını onaylayınız.
5- Hazır verilen örnekten yararlanarak Şekil 9.2'deki devrenin PAL kodunu kendiniz yazınız, WinCUPL'da derleyiniz ve WinSIM'de simülasyonunu yapınız.
6- Duyurularda grup numaranız için tanımlanmış N değerine göre Şekil 9.1'deki devrenin PAL kodunu kendiniz yazınız, WinCUPL'da derleyiniz ve WinSIM'de simülasyonunu yapınız.

7- Yukarıdaki tüm adımları raporlayarak çıktısını alınız ve deneye gelirken yanınızda getiriniz.
8- Derleme sonucu WinCUPL programının ürettiği ".jed" uzantılı programlama dosyalarını en geç deney günü sabahı 08:00'a kadar dersin LAB sorumlusuna e-posta ile ulaştırınız. N ve 2 bitlik sayıcılar için üretilen JED dosyalarına sırasıyla "grup_<numara>_N.jed" ve "grup_<numara>_2.jed" isimlerini veriniz.

Deneyde Yapılacak İşlemler:

1- PAL elemanınızın üzerine kurşun kalemle grup numaranızı yazarak LAB sorumlunuza veriniz, PAL elemanınıza sizin e-posta ile gönderdiğiniz 2 bitlik sayıcı kodu yüklenecektir.
2- 2 bitlik sayıcı koduna sahip PAL elemanınızı uygun şekilde anahtarlara, LED'lere ve güç kaynağına bağlayarak çalışmasının doğruluğunu gösteriniz.
3- PAL elemanınızı tekrar LAB sorumlunuza vererek içine yine sizin hazırladığınız N bitlik sayıcı kodunun yüklenmesini sağlayınız.
4- N bitlik sayıcı koduna sahip PAL elemanınızı uygun şekilde anahtarlara, LED'lere ve güç kaynağına bağlayarak çalışmasının doğruluğunu gösteriniz.

Sorular:

1- Verilen devrenin analizini teorik olarak yapınız.
2- WinCUPL programının kullandığı kodlama dilini açıklayınız. Örnek PAL programını satır satır analiz ediniz.
3- WinSIM programında simülasyon yapma adımlarını açıklayınız.
4- GAL22V10 elemanının giriş ve çıkış sayısı açısından tasarım sınırlamaları nelerdir? Açıklayınız.
5- GAL22V10 elemanının içindeki flip-flop belleklerin tasarım sınırlamaları nelerdir? Açıklayınız.
6- GAL22V10 elemanının çıkış pinlerine tanımlanan fonksiyonların minterim sayısıyla ilgili sınırlamalar nelerdir? Açıklayınız.

Malzeme Listesi:

- 1 adet GAL22V10 PAL elemanı (Deneyden önce programlanmış olmalı)

N=10 için Örnek PAL Programı:

```
Name     deney9 ;
PartNo   00 ;
Date     08.05.2015 ;
Revision 01 ;
Designer Engineer ;
Company  ytu ;
Assembly None ;
Location  ;
Device   g22v10 ;

/* *************** INPUT PINS *********************/
PIN 1 = clk ; /*                                  */
PIN 2 = CE  ; /*                                  */
PIN 3 = rst ; /*                                  */
```

```
/* *************** OUTPUT PINS *********************/
PIN 19 = Q14 ; /*                                   */
PIN 22 = Q04 ; /*                                   */
PIN 21 = Q13 ; /*                                   */
PIN 20 = Q03 ; /*                                   */
PIN 23 = Q12 ; /*                                   */
PIN 18 = Q02 ; /*                                   */
PIN 17 = Q11 ; /*                                   */
PIN 16 = Q01 ; /*                                   */
PIN 15 = Q10 ; /*                                   */
PIN 14 = Q00 ; /*                                   */

CE0   = CE;
OV0   = CE0 & Q10 & Q00;
Q00.D = (!CE0 & Q00) # (CE0 & !Q00);
Q10.D = (CE0 & !Q10 & Q00) # (!CE0 & Q10) # (Q10 & !Q00);

CE1   = OV0;
OV1   = CE1 & Q11 & Q01;
Q01.D = (!CE1 & Q01) # (CE1 & !Q01);
Q11.D = (CE1 & !Q11 & Q01) # (!CE1 & Q11) # (Q11 & !Q01);

CE2   = OV1;
OV2   = CE2 & Q12 & Q02;
Q02.D = (!CE2 & Q02) # (CE2 & !Q02);
Q12.D = (CE2 & !Q12 & Q02) # (!CE2 & Q12) # (Q12 & !Q02);

CE3   = OV2;
OV3   = CE3 & Q13 & Q03;
Q03.D = (!CE3 & Q03) # (CE3 & !Q03);
Q13.D = (CE3 & !Q13 & Q03) # (!CE3 & Q13) # (Q13 & !Q03);

CE4   = OV3;
OV4   = CE4 & Q14 & Q04;
Q04.D = (!CE4 & Q04) # (CE4 & !Q04);
Q14.D = (CE4 & !Q14 & Q04) # (!CE4 & Q14) # (Q14 & !Q04);

Q14.ar = rst;
Q04.ar = rst;
Q13.ar = rst;
Q03.ar = rst;
Q12.ar = rst;
Q02.ar = rst;
Q11.ar = rst;
Q01.ar = rst;
Q10.ar = rst;
Q00.ar = rst;
```