SAYISAL TASARIM

Sayısal Tasarım

Ders notları ve alıştırmalar

0 Sayısal sistemler ve ikilik kodlar

1 Boole cebri

2 Lojik fonksiyonlar

3 Lojik kapılar

4 Karnaugh haritaları

5 Toplama ve çıkarma devreleri

6 Seçiciler, kod çözücüler, kodlayıcılar

7 ALU tasarımı

8 Mandallar ve flip-floplar

9 Yazmaçlar ve sayıcılar

10 Sonlu durum makineleri

11 Bellekler ve programlanabilir lojik

İçindekiler

Bu notlar, YouTube'daki Sayısal Tasarım dersinin yazılı hâlidir. Her konunun yanındaki zaman, uzun videoda o konunun başladığı yeri gösterir. Her bölümün sonunda sorular var; cevaplarını ve çözümlerini en sonda bulabilirsin. Aynı soruları hekzon.com/dersler adresinde, cevabını hemen göreceğin alıştırmalar olarak da çözebilirsin.

0 **Sayısal sistemler ve ikilik kodlar** 3
Bir buçuk volt · Tek yol mu?

1 **Boole cebri** 9
Bir artı bir · Bozulan eşitlik · Turnike

2 **Lojik fonksiyonlar** 17
Kaç farklı iş? · Jüri

3 **Lojik kapılar** 23
Tek tür kapı · Ters fiyat · Aynı kapı, farklı hız

4 **Karnaugh haritaları** 30
Uyarı sesi · Yedi sayı · Yuvarlama

5 **Toplama ve çıkarma devreleri** 38
Sayı bilmeyen kapılar · Bekleyen basamak · Bir tel

6 **Seçiciler, kod çözücüler, kodlayıcılar**
Tek hücre · Tuştan ışığa · Dört giriş, bir ekran

7 **ALU tasarımı** 56
Ayrı devre mi? · Eşit mi?

8 **Mandallar ve flip-floplar** 62
Duracak ışığı · Her tıkta bir kez · Çakışan iki emir

9 **Yazmaçlar ve sayıcılar** 70
Tek hat · Yanlış sayı · Tuhaf sayı

10 **Sonlu durum makineleri** 79
Sıradaki renk · Telden gelen bitler · Etiketsiz devre

11 **Bellekler ve programlanabilir lojik** 88
Hangi kutu? · Kendini tutan bit · Bir çip, birçok devre

Cevaplar ve çözümler 96

SAYISAL TASARIM

0 Sayısal sistemler ve ikilik kodlar

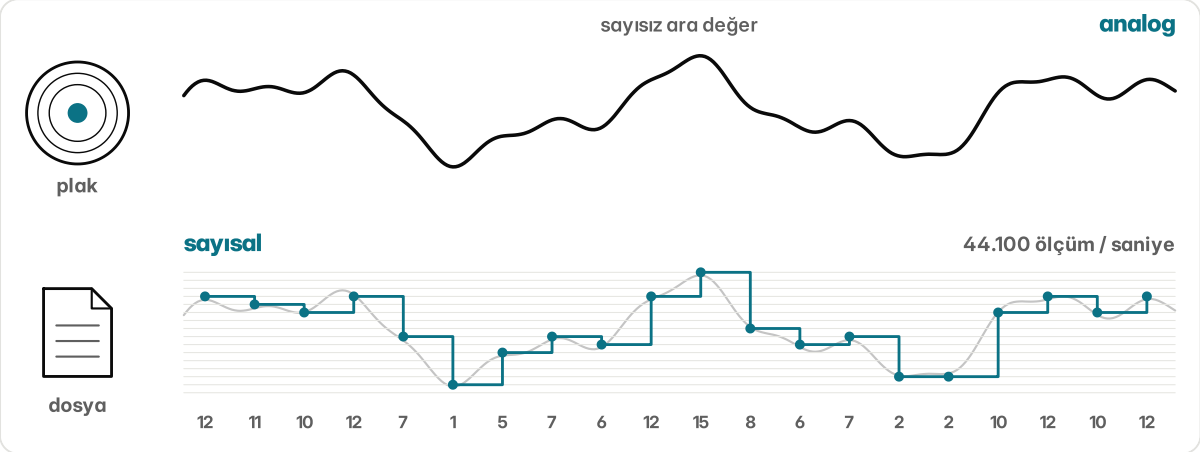
Bu bölüm, bilgisayarın neden yalnız iki değerle çalıştığını ve bu iki değerin bir telde nasıl temsil edildiğini anlatır. Ardından aynı bitlerin sayı olmaktan başka işler için nasıl kodlandığını üç örnekle gösterir: BCD, Gray kodu ve eşlik biti.

Bir buçuk volt

videoda 0:10

Bir tel ölçülüyor ve voltmetre 1,5 V gösteriyor. Bu bir mi, sıfır mı? Ölçümde sorun yok; asıl soru, devrenin bu değeri nasıl okuyacağı. Cevap için önce neden yalnız iki değer kullanıldığına bakmak gerekir.

Aynı şarkı iki biçimde saklanabilir. Plakta oluğun kıvrımı sesin dalgasını kesintisiz izler; iki değer arasında sayısız ara değer vardır. Böyle sürekli değerlerle çalışan yönteme **analog** denir. Ses dosyasında ise ses saniyede binlerce kez, örneğin 44.100 kez ölçülür ve her ölçüm belirli adımlarla yazılmış bir sayı olur. Sayılabilir değerlerle çalışan bu yönteme **sayısal** (dijital) denir.



Şekil 0.1 Üstte sesin kesintisiz dalgası; altta aynı dalganın eşit aralıklarla ölçülüp 0-15 arası basamaklara yuvarlanmış hâli.

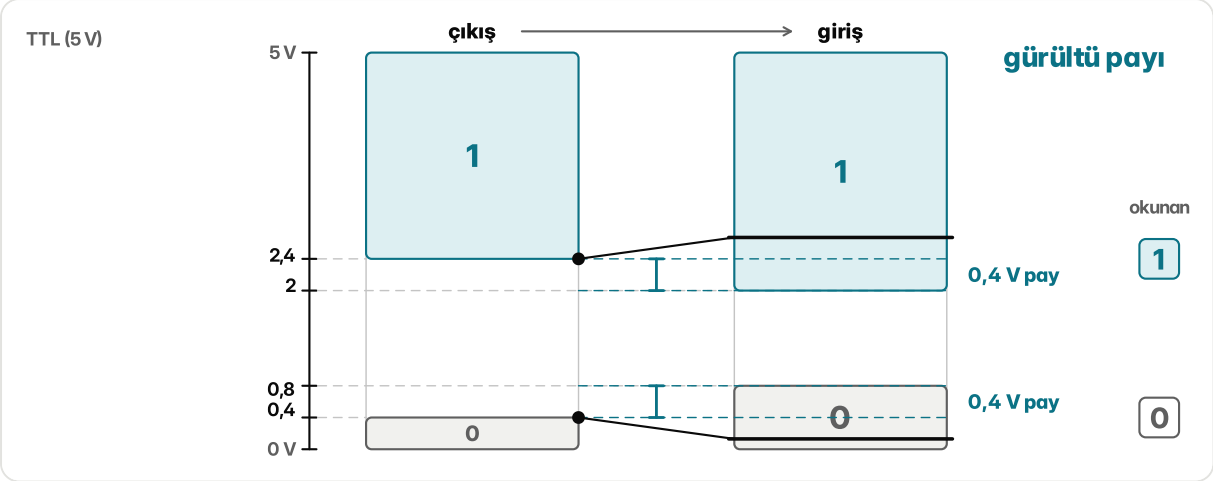
Fark kopyalarken ortaya çıkar. Plaktan kasete, kasetten kasete her kopya biraz daha cızırtı ekler, çünkü ara değerlerden hangisinin doğru olduğu bilinemez. Dosyada ise her bit yalnız "1 mi, 0 mı" diye okunur; küçük sapmalar okunurken silinir. Dosya 1000 kez kopyalansa da 1000. kopya ilkinin aynısıdır.

Bir teldeki gerilim hiçbir zaman tam sabit durmaz; yandaki teller, ısı ve besleme onu sarsar. Bu istenmeyen oynamaya **gürültü** denir. 5 V on seviyeye bölünseydi her seviyeye 0,5 V'luk bir dilim düşerdi; 0,25 V'luk bir sarsıntı değeri komşu seviyeye kaydırabilirdi. İki seviyede dilimler geniştir ve aralarında boşluk bırakılır.

5 V ile çalışan eski bir kapı ailesinde (TTL) kural şöyledir:

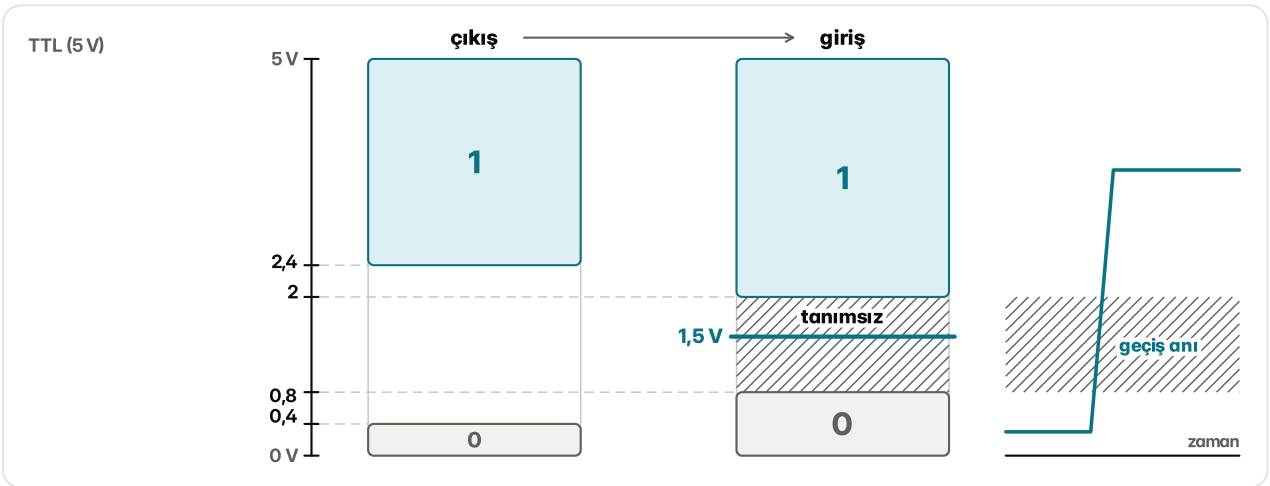
```
çıkış verir          giriş okur
0: 0 - 0,4 V         0: 0 - 0,8 V
1: 2,4 - 5 V         1: 2 - 5 V
                     0,8 - 2 V: tanımsız
```

Çıkış, girişten daha dar bantlarda çalışır. Çıkış 0'ı en çok 0,4 V ile gönderir, giriş 0,8 V'a kadar 0 sayar; arada 0,8 − 0,4 = 0,4 V'luk bir pay kalır. 1 tarafında da 2,4 ile 2 V arasında aynı 0,4 V vardır. Tele binen gürültü bu payı aşmadıkça hiçbir bit değişmez. Bu paya **gürültü payı** denir.



Şekil 0.2 Çıkışın ve girişin gerilim bantları; iki yandaki 0,4 V'luk gürültü payı ve titreşen telde okunan bitlerin değişmediği görülüyor.

0,8 ile 2 V arası **tanımsız bölgedir**: kapı bu gerilimi 0 da sayabilir, 1 de; hangisini sayacağı garanti değildir. Bu boşluk bilerek bırakılır, böylece 0'ın ve 1'in bantları birbirine hiç değmez. Açılıştaki sorunun cevabı da buradadır: 1,5 V ne 1'dir ne 0. Doğru çalışan bir devrede tel bu bölgeden yalnız bir değerden ötekine geçerken, çok kısa bir anda geçer.



Şekil 0.3 Girişin 0,8-2 V arasındaki taralı tanımsız bölgesi ve 1,5 V çizgisi; sağda 0'dan 1'e geçen sinyal bu bölgeyi kısa bir anda kesiyor.

İki seviyede anlaşınca volt ayrıntısı alt katta kalır. Transistör bir anahtar gibi açılıp kapanır; birkaç transistör bir kapı kurar; kapılar toplayıcı, seçici gibi blokları, bloklar da işlemciyi. Her katman alttakinin ayrıntısını gizler: kapıyla çalışan volt düşünmez, yalnız 0 ve 1 düşünür; blokla çalışan da

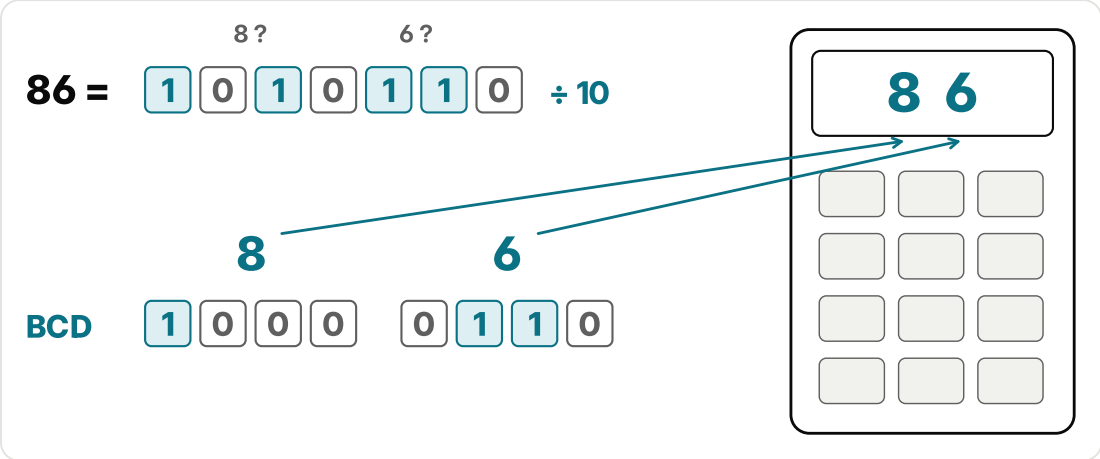
kapıları değil, toplamayı ve seçmeyi düşünür. Bugünkü bir işlemcide milyarlarca transistör vardır; hiçbiri tek tek tasarlanmaz, katman katman kurulur.

Tek yol mu?

videoda 2:14

86 sayısını bitlerle yazmanın tek doğru yolu var mı? Sayının ikilik değeri bir yoldur, ama bitlere hangi anlamın verileceği bir seçimdir. Bitlere verilen bu anlama **kod** denir. Aşağıda üç ayrı iş için üç kod var.

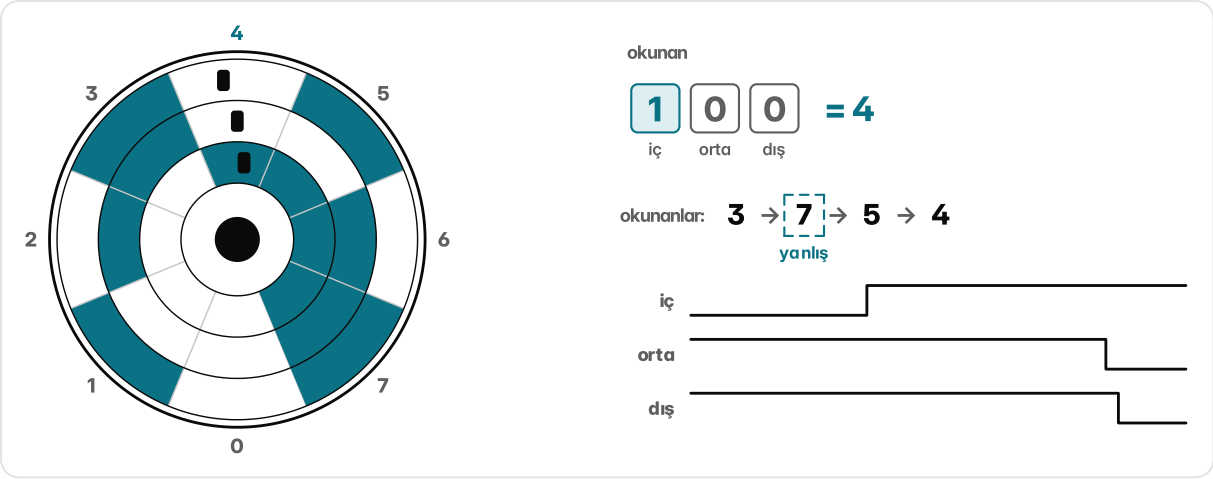
86'nın ikilik değeri 1010110'dır. Bu bitlerin içinde ne 8 görünür ne 6; sayıyı ekrana rakam rakam yazmak için önce 10'a bölmek gerekir. **BCD** (ikilik kodlanmış onluk, binary coded decimal) kodunda her onluk basamak ayrı 4 bitle yazılır: 8 → 1000, 6 → 0110, yani 86 → 1000 0110. Her dörtlü doğrudan kendi rakamına gider; pek çok cep hesap makinesi sayıları böyle tutar.



Şekil 0.4 86'nın ikilik değeri 1010110 ve BCD yazımı 1000 0110; her dörtlü hesap makinesi ekranındaki kendi rakamına gidiyor.

BCD'nin bir bedeli var: 4 bit 16 birleşim taşır, bir rakama 10 tanesi yeter. 1001'den, yani 9'dan sonraki 6 birleşim (1010-1111) hiç kullanılmaz. Bu yüzden BCD daha çok bit ister: iki basamaklı her sayı ikilikte 7 bite sığar (99 = 1100011), BCD'de 8 bit gerekir (99 → 1001 1001).

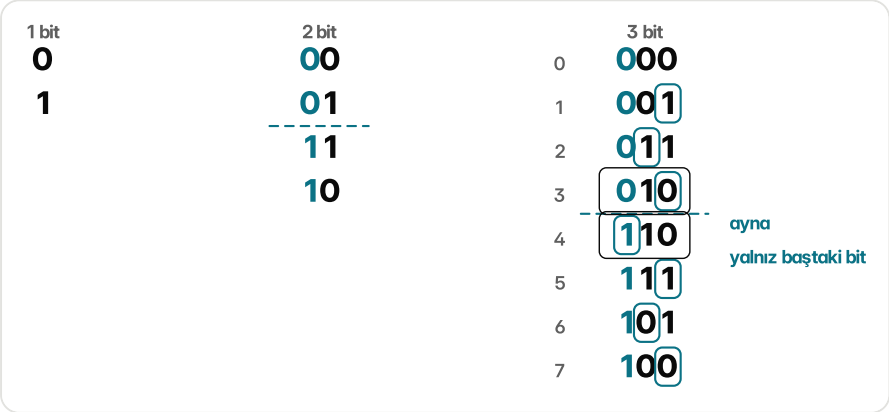
İkinci iş için 8 konumlu bir döner düğme düşünelim. Diskte 3 halka vardır; her halkada iletken ve yalıtkan dilimler, her halkanın üstünde bir kontak bulunur. Konum 3 bitlik bir sayı olarak okunur. Konumlar ikilikle numaralanırsa 3 numaralı konum 011, 4 numaralı konum 100 olur: geçişte üç bitin üçü birden değişmelidir. Kontaklar hiçbir zaman tam aynı anda değişmez; bir an 111, yani 7 okunabilir.



Şekil 0.5 Sekiz konumlu disk 3'ten 4'e dönerken kontaklar ayrı anlarda değişiyor; okunan değerler 3, 7, 5, 4 ve 7 yanlış okuma olarak işaretli.

Gray kodunda ardışık iki sayı yalnız tek bitle ayrılır. Kuruluşu yansıtmaya dayanır: 0 ve 1 yazılır, altına aynalanarak 1 ve 0 eklenir; üst yarının başına 0, alt yarının başına 1 konur: 00, 01, 11, 10. Aynı işlem bir kez daha yapılınca 3 bitlik sıra çıkar. Yöntemin neden işe yaradığı açıktır: her yarının içinde adımlar zaten tek bitliktir; aynanın birleştiği yerde iki satır aynıdır, yalnız baştaki bit değişir.

konum	ikilik	Gray
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100



Şekil 0.6 Gray kodunun yansıtmayla kuruluşu: 1, 2 ve 3 bitlik sütunlar, ayna çizgileri ve her satırda değişen tek bit işaretli.

Düğme Gray koduyla okunursa 3 numaralı konum 010, 4 numaralı konum 110 olur; yalnız soldaki bit değişir. Kontak geç kalsa bile okunan değer ya 3 ya 4'tür. Son konumdan başa dönerken de (100 → 000) tek bit değişir. Bir milin açısını ölçen pek çok algılayıcı bu yüzden Gray kodu kullanır.

Üçüncü iş hata yakalamaktır. 7 bitlik 1010001 verisi gönderilecek. Birler sayılır: 3, tek sayı. Sona bir **eşlik biti** eklenir; birlerin toplam sayısı çift olsun diye bu bit 1 olur (çift eşlik). Alıcı da birleri sayar:

```
gönderilen      1010001 1   1'ler: 4, çift
3. bit bozuk    1000001 1   1'ler: 3, tek   -> hata yakalanır
3. ve 6. bit    1000011 1   1'ler: 4, çift  -> hata görünmez
```

Tek bir bit bozulursa birlerin sayısı tekleşir ve hata yakalanır. İki bit birden bozulursa sayı yine çift kalır ve hata görünmez. Eşlik biti hatanın yerini de söylemez; yalnız bir şeyin bozulduğunu haber verir ve veri yeniden istenir. Tek bir bitle bu kadarı yapılır; seri haberleşme hatlarında ve eski bellek modüllerinde bu yüzden kullanıldı.

Kısacası bir sayıyı bitlerle yazmanın tek doğru yolu yoktur: rakamı ekrana göndermek için BCD, dönen bir diski okumak için Gray kodu, hatayı yakalamak için eşlik biti.

Özet

- Analog değerler süreklidir; sayısal değerler sayılabilir adımlarla yazılır ve kopyalandıkça bozulmaz.
- Gürültüye dayanmak için yalnız iki seviye kullanılır; 5 V'luk eski ailede giriş 0-0,8 V'u 0, 2-5 V'u 1 okur, aradaki bölge tanımsızdır.
- Çıkışın bantları daha dardır (0 en çok 0,4 V, 1 en az 2,4 V); iki yanda 0,4 V'luk gürültü payı kalır.
- BCD her onluk basamağı 4 bitle yazar (86 → 1000 0110); 1010-1111 kullanılmaz, bu yüzden daha çok bit ister.
- Gray kodunda ardışık sayılar tek bitle ayrılır; döner disklerde geçiş anında yanlış sayı okunmaz.
- Çift eşlik biti tek bitlik hatayı yakalar, iki bitlik hatayı göremez ve hatanın yerini söylemez.

Sorular

1 **Plakta saklanan sesin analog sayılmasının nedeni nedir?**

- **A** Ses saniyede 44.100 kez ölçülerek saklanır.
- **B** Oluğun kıvrımı dalgayı kesintisiz izler; iki değer arasında sayısız ara değer vardır.
- **C** Her ölçüm belirli adımlarla yazılmış bir sayıdır.
- **D** Kopyalandıkça hiç bozulmaz.

2 **5 V ile çalışan eski kapı ailesinde (giriş 0-0,8 V'u 0, 2-5 V'u 1 okur) bir girişe 1,2 V gelirse ne olur?**

A 0 okunur, çünkü 2 V'tan küçüktür.

B 1 okunur, çünkü 0,8 V'tan büyüktür.

C Tanımsız bölgededir; kapı 0 da sayabilir, 1 de.

D Gürültü payının içinde kaldığı için bit değişmez.

3 **Aynı ailede çıkış 0'ı en çok 0,4 V ile gönderiyor, giriş 0,8 V'a kadar 0 sayıyor. 0 tarafındaki gürültü payı kaç volttur?**

Cevap:

4 **57 sayısını BCD koduyla yaz (8 bit).**

Cevap:

5 **3 bitlik Gray kodunda 5 numaralı konumun kodu hangisidir? (Sıra: 000, 001, 011, 010, 110, ...)**

A 101

B 110

C 100

D 111

6 **Döner bir diskin konumunu okurken Gray kodunun ikilik koda üstünlüğü nedir?**

A Komşu konumlar tek bitle ayrıldığı için geçiş anında yanlış bir sayı okunmaz.

B Aynı sayıda bitle daha çok konum gösterir.

C Her konumda birlerin sayısı çifttir, bu yüzden hata yakalar.

D Onluk rakamları doğrudan ekrana gönderir.

7 **7 bitlik 1100100 verisi çift eşlikle gönderilecek. Eklenecek eşlik biti nedir?**

Cevap:

8 **Çift eşlikle gönderilen 8 bitlik bir çerçeve alıcıya 01101101 olarak ulaşıyor. Alıcı ne sonuca varır?**

A Çerçeve doğrudur, çünkü son bit 1'dir.

B Hatalı bit 3. bittir.

C Birlerin sayısı tek; bir hata yakalanır ve veri yeniden istenir.

D İki bit bozulmuştur.

SAYISAL TASARIM

1 Boole cebri

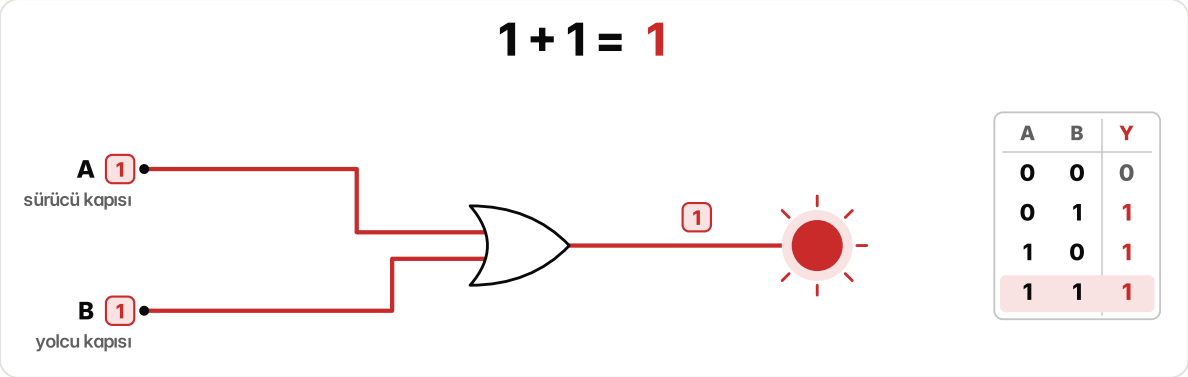
Sayısal devreler yalnız iki değerle çalışır: 0 ve 1. Bu bölümde bu iki değerin cebrini, Boole cebrini öğreneceksin: üç işlemi ve sıralarını, her zaman doğru olan kuralları, bir eşitliği bütün durumları deneyerek ispat etmeyi, De Morgan kurallarını ve bir ifadeyi kural kural küçültmeyi.

Bir artı bir

videoda 4:21

Hangi matematikte 1 + 1 = 1 olur? Arabanın tavan lambasına bakalım. Lamba sürücü kapısı açılınca yanar, yolcu kapısı açılınca da yanar. İki kapı birden açılınca iki kat parlak yanmaz; yine yalnızca yanar. Açık kapıya 1, kapalı kapıya 0 dersek bu lamba için 1 + 1 = 1'dir. Sürücü kapısı A, yolcu kapısı B, lamba Y olsun:

```
A  B | Y = A + B
0  0 |     0
0  1 |     1
1  0 |     1
1  1 |     1
```



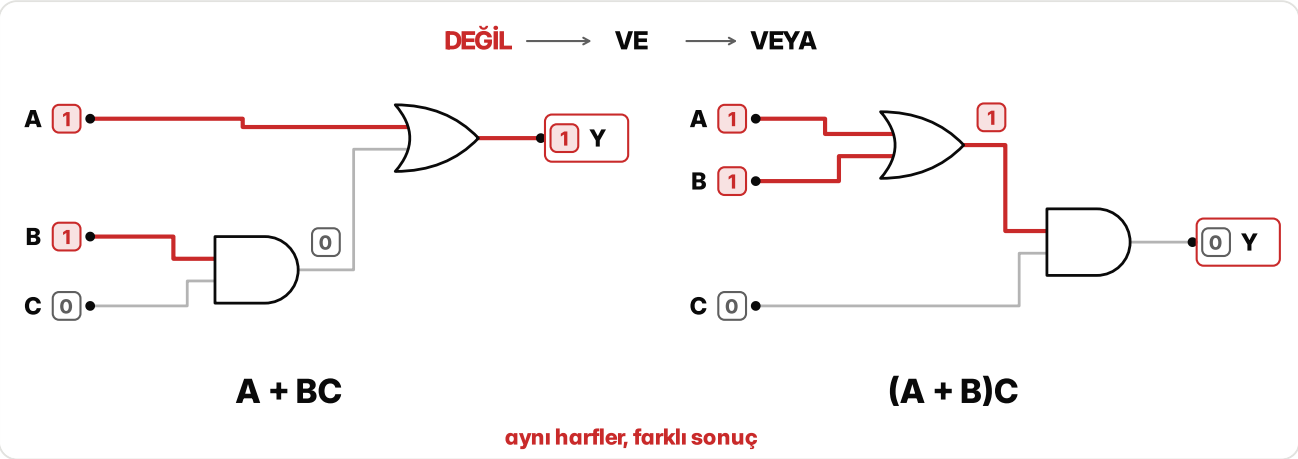
Şekil 1.1 Tavan lambası: A (sürücü kapısı) ve B (yolcu kapısı) VEYA kapısına giriyor; iki giriş de 1 iken çıkış 1, lamba yanıyor.

Bu hesabın adı **Boole cebri**dir. George Boole 1854'te yayımladığı "An Investigation of the Laws of Thought" adlı kitabında düşünmenin kurallarını cebirle yazdı. Boole cebrinde bir değişken yalnız iki değer alır, 0 ya da 1; işlem de üç tanedir:

- **VE** (AND): iki giriş de 1 ise 1 verir. 0 ve 1 ile tıpkı çarpma gibi çalıştığı için çarpım gibi yazılır: A·B ya da kısaca AB.
- **VEYA** (OR): girişlerden en az biri 1 ise 1 verir. Toplama gibi + ile yazılır; toplamadan tek farkı 1 + 1'in 1 etmesidir.
- **DEĞİL** (NOT): 0'ı 1, 1'i 0 yapar. Harfin üstüne çizgi çekilir: $\overline{A}$. $\overline{A}$, A'nın **tümleyeni**dir.

A	B	AB	A + B	$\overline{A}$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

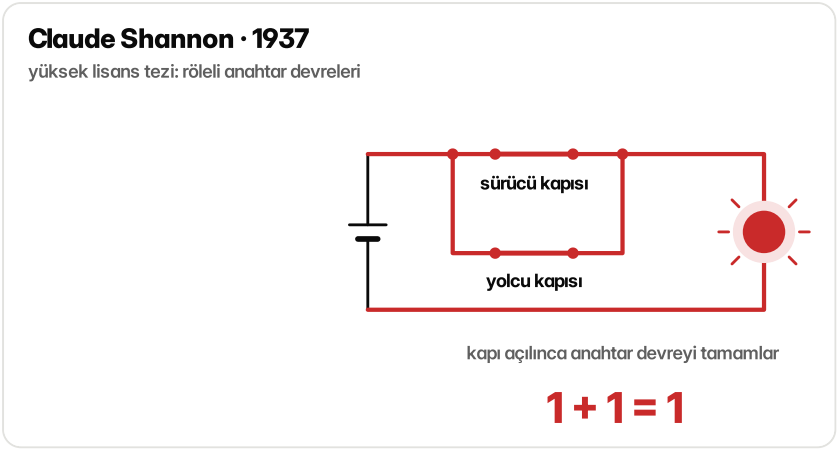
Sıradan cebirdeki gibi bir **işlem önceliği** vardır: önce DEĞİL, sonra VE, en son VEYA. A + BC yazılmışsa önce B ile C'nin VE'si, sonra sonucun A ile VEYA'sı alınır. A = 1, B = 1, C = 0 için BC = 0, A + 0 = 1 olur. Parantez sırayı değiştirir: (A + B)C'de önce A + B = 1 bulunur, sonra 1·C = 0. Aynı harfler, farklı sonuç.



Şekil 1.2 A + BC ve (A + B)C devreleri A = 1, B = 1, C = 0 için: ilkinin çıkışı 1, ikincisinin 0.

Şimdi kurallar. A ne olursa olsun A·0 = 0 ve A·1 = A'dır, çarpmadaki gibi. A + 0 = A'dır, ama A + 1 her zaman 1'dir: kapılardan biri açıksa öteki ne olursa olsun lamba yanar. Bu kuralları ispat etmek kolaydır: A yalnız iki değer alabildiği için ikisini de denemek yeter. Bir değişken kendisiyle birleşince değişmez: A + A = A ve A·A = A; sıradan cebirdeki 2A ya da A^2 burada yoktur. Tümleyeniyle birleşince sonuç hep aynıdır: A + $\overline{A}$ = 1 ve A·$\overline{A}$ = 0. Sürücü kapısı ya açıktır ya kapalı; ikisinden biri mutlaka doğrudur, ikisi birden asla. Tümleyenin tümleyeni de değişkenin kendisidir.

Boole'dan 83 yıl sonra, 1937'de, Claude Shannon MIT'deki yüksek lisans tezinde ("A Symbolic Analysis of Relay and Switching Circuits") bu cebrin röleli anahtar devrelerini anlattığını gösterdi: seri bağlı iki anahtar VE gibi, paralel bağlı iki anahtar VEYA gibi çalışır. Tavan lambasının iki kapı anahtarı da paralel bağlıdır; iki kapı birden açılınca akım iki yoldan gelir ama lamba yine aynı yanar. Bugün her sayısal devre bu cebirle tasarlanır.



Şekil 1.3 Tavan lambasının paralel bağlı iki kapı anahtarı: ikisi de kapalıyken akım iki yoldan geliyor, lamba aynı yanıyor; 1 + 1 = 1.

Bozulan eşitlik

videoda 6:13

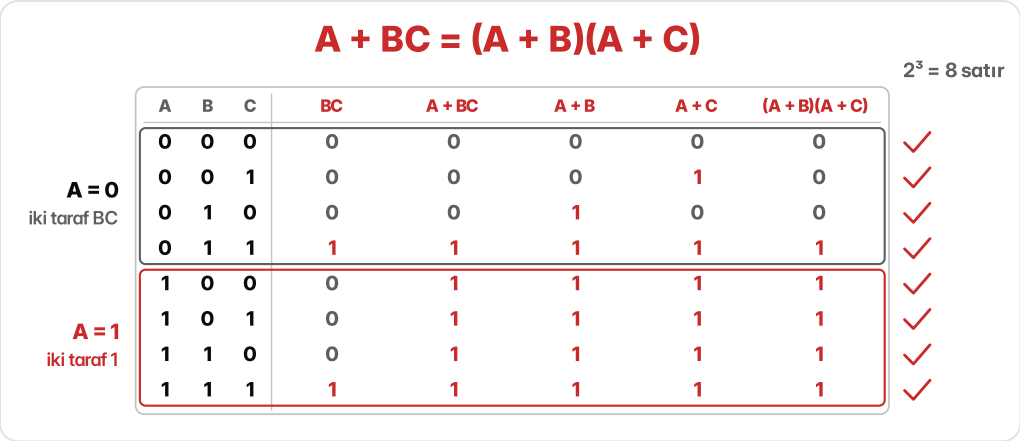
Sıradan aritmetikte 1 + 2·3 = 7, buna karşılık (1 + 2)(1 + 3) = 12'dir; bu biçimdeki iki ifade genellikle eşit değildir. Boole cebrinde ise A + BC = (A + B)(A + C) her zaman doğrudur. Nasıl olduğunu görmeden önce değişmeyen kurallara bakalım.

Sıra önemsizdir: AB = BA ve A + B = B + A; buna **değişme** denir. Üç değişken birleştirilirken hangi ikisiyle başlandığı da önemsizdir: (AB)C = A(BC) ve (A + B) + C = A + (B + C); bu da **birleşme**dir. VE, parantezin içindeki VEYA'ya dağılır: A(B + C) = AB + AC; bu **dağılma**dır. Buraya kadar her şey okuldaki cebirle aynıdır.

Boole cebrinde bir dağılma daha vardır: VEYA da VE'nin üstüne dağılır, A + BC = (A + B)(A + C). Sıradan sayılar sonsuz olduğu için bir eşitliği bütün sayılarla deneyemeyiz. Burada ise üç değişkenin yalnız $2^3 = 8$ birleşimi vardır; sekizini de denemek bir ispattır. Buna **doğruluk tablosuyla ispat** denir:

```
A  B  C | BC  A + BC  | A + B  A + C  (A + B)(A + C)
0  0  0 | 0     0     |   0      0          0
0  0  1 | 0     0     |   0      1          0
0  1  0 | 0     0     |   1      0          0
0  1  1 | 1     1     |   1      1          1
1  0  0 | 0     1     |   1      1          1
1  0  1 | 0     1     |   1      1          1
1  1  0 | 0     1     |   1      1          1
1  1  1 | 1     1     |   1      1          1
```

A + BC ve (A + B)(A + C) sütunları sekiz satırın sekizinde aynıdır. Nedeni de görülür: A = 1 ise iki taraf da 1'dir; A = 0 ise iki taraf da BC'ye iner.



Şekil 1.4 A + BC = (A + B)(A + C) eşitliğinin sekiz satırlık tablosu: A = 0 satırlarında iki taraf BC'ye, A = 1 satırlarında 1'e eşit.

Kuralları yan yana koyunca bir düzen görünür: A + 0 = A ile A·1 = A, A + $\overline{A}$ = 1 ile A·$\overline{A}$ = 0, A + A = A ile A·A = A eşleşir; iki dağılma da birbirinin aynasıdır. Bir kuralda VE ile VEYA, 0 ile 1 yer değiştirince yine doğru bir kural çıkar. Buna **ikillik** (dualite) denir: bir kuralı ispat etmek, ikizini de ispat eder.

Gündelik bir cümle: "Yağmur yağıyorsa, ya da yağmur yağıyor ve rüzgâr esiyorsa şemsiye alırız." İkinci koşul bir şey katmaz; yağmur varsa şemsiye zaten alınır. Yağmur A, rüzgâr B ise A + AB = A'dır: A, B'yi yutar. Cebirle: A dışarı alınır, A(1 + B) olur; 1 + B hep 1 olduğu için sonuç A·1 = A. Bu **yutma** kuralıdır.

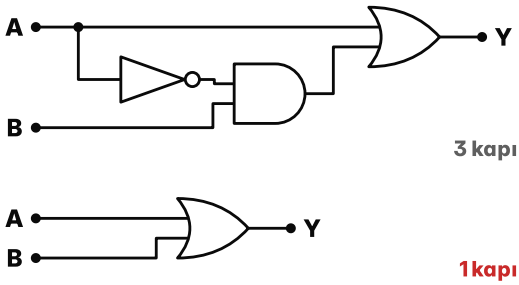
```
A  B | AB  A + AB
0  0 | 0     0
0  1 | 0     0
1  0 | 0     1
1  1 | 1     1
```

Yutmanın bir yüzü daha var: "Yağmur yağıyorsa, ya da yağmur yağmıyor ama güneş yakıyorsa şemsiye alırız." "Yağmur yağmıyor" kısmı da fazladır, çünkü yağsaydı şemsiye zaten alınırdı; kısası "yağmur VEYA güneş". Güneş B ise A + $\overline{A}$B = A + B. Bunu ters yöndeki dağılma gösterir: A + $\overline{A}$B = (A + $\overline{A}$)(A + B); ilk parantez hep 1'dir, geriye A + B kalır. DEĞİL, VE ve VEYA'dan oluşan üç kapılık devre tek bir VEYA kapısına iner.

yağmur (A) VEYA ~~yağmur yok (A değil) VE güneş~~ (B) → şemsiye

kısası: yağmur VEYA güneş

$$A + \overline{A}B$$
$$= (A + \overline{A})(A + B)$$
$$= 1 \cdot (A + B)$$
$$= A + B$$



iki devrenin tablosu aynı: 0, 1, 1, 1

Şekil 1.5 $A + \overline{A}B = (A + \overline{A})(A + B) = 1 \cdot (A + B) = A + B$; üç kapılık devre ile tek VEYA kapısı aynı tabloyu veriyor (0, 1, 1, 1).

Turnike

videoda 7:55

Metro turnikesi kart okutulduğunda ve bakiye yettiğinde açılır. Kart okutuldu A, bakiye yetti B olsun; turnike Y = AB olunca açılır. Peki ne zaman açılmaz? "Kart okutulmadı VE bakiye yetmedi" mi, yoksa "kart okutulmadı VEYA bakiye yetmedi" mi?

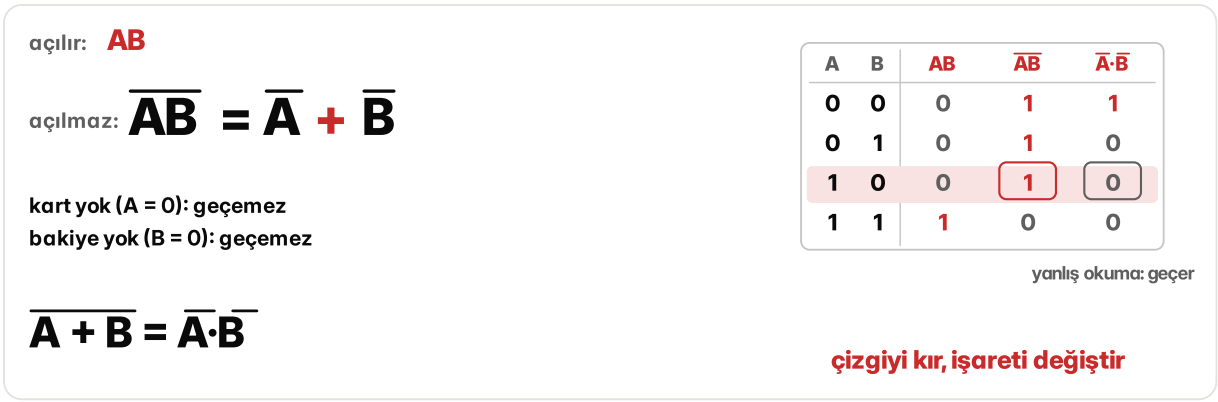
Açılmaması Y'nin tümleyenidir, $\overline{Y}$: AB'nin tümleyeni. Kart okutulmadıysa geçilemez; kart okutulup bakiye yetmediyse de geçilemez; ikisinden birinin olmaması yeter. Yani AB'nin tümleyeni $\overline{A} + \overline{B}$'dir. VE deseydik, A = 1, B = 0 iken $\overline{A} \cdot \overline{B} = 0$ çıkar, kartını okutup bakiyesi yetmeyen yolcu da geçerdi. Bu, **De Morgan** kuralıdır; ikizi de doğrudur:

```
AB'nin tümleyeni       = Ā + B̄
A + B'nin tümleyeni    = Ā·B̄
```

Kısa yolu: uzun çizgiyi kır, işareti değiştir. İspat için iki girişin dört birleşimini denemek yeter ($\overline{Y}$, Y = AB'nin tümleyeni):

```
A  B | AB  Ȳ  | Ā  B̄  Ā + B̄
0  0 | 0   1  | 1  1    1
0  1 | 0   1  | 1  0    1
1  0 | 0   1  | 0  1    1
1  1 | 1   0  | 0  0    0
```

$\overline{Y}$ ve $\overline{A} + \overline{B}$ sütunları dört satırda da aynıdır (1, 1, 1, 0).

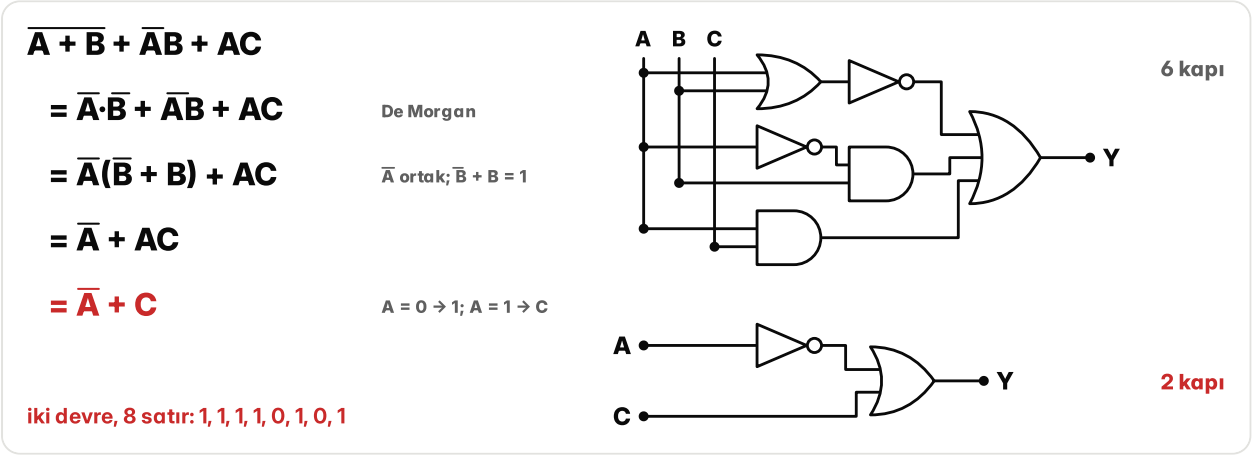


Şekil 1.6 AB'nin tümleyeni $\overline{A} + \overline{B}$'ye eşit; tabloda A = 1, B = 0 satırında yanlış okuma $\overline{A} \cdot \overline{B} = 0$ veriyor, yani yolcuyu geçiriyor.

Bu kurallar bir ifadeyi küçültmeye yarar. A + B'nin tümleyeni, VEYA $\overline{A}B$, VEYA AC ifadesinin devresi altı kapıdır. Kural kural:

```
= Ā·B̄ + ĀB + AC     De Morgan
= Ā(B̄ + B) + AC     Ā ortak çarpan
= Ā + AC            B̄ + B = 1
= Ā + C             A = 0 ise 1, A = 1 ise C
```

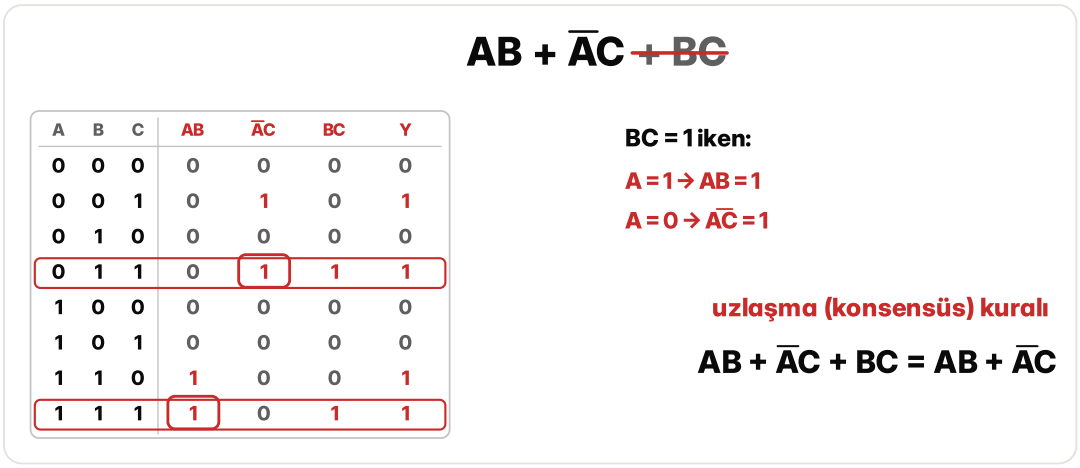
Son adım yutmanın öteki yüzüdür. Sonuç DEĞİL ve VEYA'dan oluşan iki kapılık devredir. Sekiz satırın hepsinde iki devrenin tablosu aynıdır (000'dan 111'e: 1, 1, 1, 1, 0, 1, 0, 1); sinyalin geçtiği en uzun yol da üç kapıdan ikiye iner.



Şekil 1.7 Altı kapılık (A + B)'nin tümleyeni + $\overline{A}B$ + AC devresi kural kural $\overline{A}$ + C'ye, iki kapılık devreye iniyor.

Bazen bir terim hiçbir şey katmaz ama bunu görmek zordur: AB + $\overline{A}$C + BC. BC'nin 1 olduğu satırlarda A ya 1'dir ya 0: A = 1 ise AB, A = 0 ise $\overline{A}$C zaten 1'dir. BC hiçbir satırı tek başına 1 yapamaz ve silinir: AB + $\overline{A}$C + BC = AB + $\overline{A}$C. Buna **uzlaşma** (konsensüs) kuralı denir.

A	B	C	AB	$\overline{A}C$	BC	Y	AB + $\overline{A}C$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	1
0	1	0	0	0	0	0	0
0	1	1	0	1	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
1	1	0	1	0	0	1	1
1	1	1	1	0	1	1	1



Şekil 1.8 Uzlaşma kuralı: BC = 1 olan 011 ve 111 satırlarında $\overline{A}C$ ya da AB zaten 1; BC terimi silinebiliyor.

Silinen her terim devreden kapı ve tel eksiltir; devre küçülür ve hızlanır. Bugün milyonlarca kapılık çipleri tasarlayan programlar da devreyi aynı kurallarla küçültür.

Özet

- Boole cebrinde değişkenler yalnız 0 ya da 1 olur; işlemler VE (AB), VEYA (A + B) ve DEĞİL ($\overline{A}$), öncelik sırası DEĞİL, VE, VEYA'dır.
- 1 + 1 = 1, A + A = A, A + 1 = 1, A + $\overline{A}$ = 1 ve A·$\overline{A}$ = 0 sıradan cebirde olmayan, Boole cebrinde her zaman doğru kurallardır.
- VEYA da VE'nin üstüne dağılır: A + BC = (A + B)(A + C); her kuralın VE ile VEYA, 0 ile 1 yer değiştirmiş ikizi de doğrudur (ikillik).
- Değişken sayısı sonlu olduğu için bir eşitlik bütün satırlar denenerek ispat edilir.
- De Morgan: AB'nin tümleyeni $\overline{A}$ + $\overline{B}$, A + B'nin tümleyeni $\overline{A}$·$\overline{B}$; çizgi kırılır, işaret değişir.
- Yutma (A + AB = A, A + $\overline{A}$B = A + B) ve uzlaşma kuralı ifadeyi, dolayısıyla devreyi küçültür.

Sorular

1 **Boole cebrinde 1 + 1 kaçtır?**

A 0 **B** 1

C 2 **D** 10

2 **A = 1, B = 0, C = 0 iken A + BC ve (A + B)C sırasıyla kaçtır?**

A 0 ve 0 **B** 0 ve 1

C 1 ve 1 **D** 1 ve 0

3 **Aşağıdakilerden hangisi Boole cebrinde her zaman doğru değildir?**

A $A + A = A$ **B** $A \cdot \overline{A} = 0$

C $A + 1 = A$ **D** $A \cdot 1 = A$

4 **A + BC = (A + B)(A + C) kuralının ikizi (ikillik) hangisidir?**

A $A(B + C) = AB + AC$ **B** $A + BC = AB + AC$

C $A \cdot BC = (A \cdot B)(A \cdot C)$ **D** $A(B + C) = (A + B)(A + C)$

5 **$A + \overline{A}B$ ifadesini sadeleştir.**

Cevap:

6 **AB'nin tümleyeni (turnikenin açılmaması) neye eşittir?**

A $\overline{A} \cdot \overline{B}$ **B** $A + B$

C $\overline{A}\,B + A\,\overline{B}$ **D** $\overline{A} + \overline{B}$

7 **$AB + \overline{A}C + BC$ ifadesinden hangi terim, sonucu değiştirmeden silinebilir?**

A AB **B** $\overline{A}\,C$

C BC **D** AB ve $\overline{A}\,C$

8 **Y = A + BC'nin doğruluk tablosunda Y sütununu, satırlar 000'dan 111'e sırayla, bitişik yaz.**

Cevap:

SAYISAL TASARIM

2 Lojik fonksiyonlar

Bu bölümde bir devrenin yaptığı işin doğruluk tablosuyla nasıl tam olarak yazıldığını, tablodan minterim ve maksterim yoluyla iki kanonik biçimin nasıl çıktığını, satırları birleştirip standart biçime ve iki katlı devreye nasıl inildiğini, son olarak bir fonksiyonun tümleyenini göreceksin.

Kaç farklı iş?

videoda 9:35

İki girişi, bir çıkışı olan bir kutunun içine istediğimiz kadar kapı koyabiliriz. Kutuyu dışarıdan izleyen biri ise yalnız hangi girişe hangi çıkışın geldiğini görür. İki girişin 4 birleşimi vardır: 00, 01, 10, 11. Her birinin karşısına çıkışı yazınca **doğruluk tablosu** çıkar. **Lojik fonksiyon** da budur: girişlerin her birleşimine 0 ya da 1 karşılık getiren kural. İçi bambaşka iki kutunun tablosu aynıysa ikisi aynı fonksiyondur. Tek VEYA kapısıyla kurulan A + B ile DEĞİL, VE ve VEYA kapılarıyla kurulan $A + \overline{A}B$ aynı tabloyu verir:

A	B	A + B	$A + \overline{A}B$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

Peki iki girişle kaç farklı fonksiyon vardır? Tablonun 4 satırının her birinde çıkış ya 0 ya 1 olabilir: $2^4 = 16$ farklı tablo. VE (0001), VEYA (0111) ve ÖZEL VEYA (0110) bunların içindedir; girişlere hiç bakmadan hep 0 veren fonksiyon da öyle. Kutunun içine ne konursa konsun, bu 16 fonksiyondan biri çıkar. Genel olarak n girişli tablo 2^n satırdır ve 2 üzeri 2^n fonksiyon vardır: 3 girişte 8 satır ve 256 fonksiyon, 4 girişte 16 satır ve 65.536 fonksiyon.

AB	0	AB	$A\overline{B}$	A	$\overline{A}B$	B	A ⊕ B	A + B	$\overline{A+B}$	$\overline{A \oplus B}$	$\overline{B}$	$A+\overline{B}$	$\overline{A}$	$\overline{A}+B$	$\overline{AB}$	1
00	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
01	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
10	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
11	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$2 \times 2 \times 2 \times 2 = 16$ tablo

kutuya ne konursa konsun, bu 16 fonksiyondan biri

3 giriş → 8 satır → 256 fonksiyon
4 giriş → 16 satır → 65.536 fonksiyon

n giriş → 2^n satır → 2^{2^n} fonksiyon

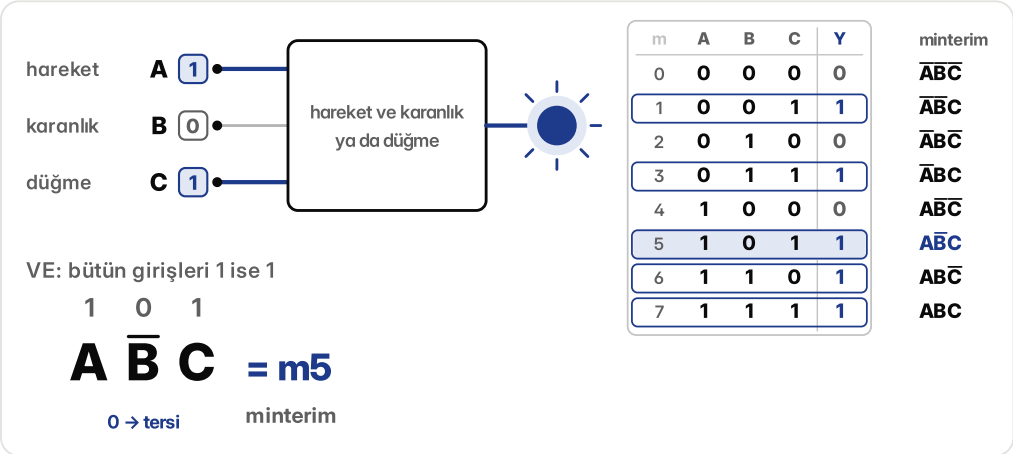
Şekil 2.1 İki girişin 16 fonksiyonu: her sütun bir doğruluk tablosu; VE, VEYA, ÖZEL VEYA ve hep 0 veren sütun çerçeveli.

Üç girişli bir örnek alalım: bahçe lambası, hareket varsa ve hava kararmışsa yanar, düğmeye basılırsa da yanar. Hareket A, karanlık B, düğme C olsun. Satırları 0'dan 7'ye numaralayınca lamba 1, 3, 5, 6 ve 7. satırlarda yanar:

no	A	B	C	Y	minterim	maksterim
0	0	0	0	0	$\overline{A}\overline{B}\overline{C}$	$A + B + C$
1	0	0	1	1	$\overline{A}\overline{B}C$	$A + B + \overline{C}$
2	0	1	0	0	$\overline{A}B\overline{C}$	$A + \overline{B} + C$
3	0	1	1	1	$\overline{A}BC$	$A + \overline{B} + \overline{C}$
4	1	0	0	0	$A\overline{B}\overline{C}$	$\overline{A} + B + C$
5	1	0	1	1	$A\overline{B}C$	$\overline{A} + B + \overline{C}$
6	1	1	0	1	$AB\overline{C}$	$\overline{A} + \overline{B} + C$
7	1	1	1	1	ABC	$\overline{A} + \overline{B} + \overline{C}$

5. satırda A = 1, B = 0, C = 1'dir. VE kapısı yalnız bütün girişleri 1 olunca 1 verir; bu yüzden 0 olan B'nin tersi alınır: $A\overline{B}C$. Bu terim yalnız 5. satırda 1'dir. Tablonun tek bir satırında 1 olan böyle VE terimine **minterim** denir; her satırın kendi minterimi vardır ve satırın numarasını taşır (m0 ... m7). Lambanın yandığı beş satırın minterimleri VEYA ile bağlanınca fonksiyon yazılmış olur, çünkü her satırda yalnız o satırın minterimi 1'dir. Bu yazıma **minterimlerin toplamı** denir:

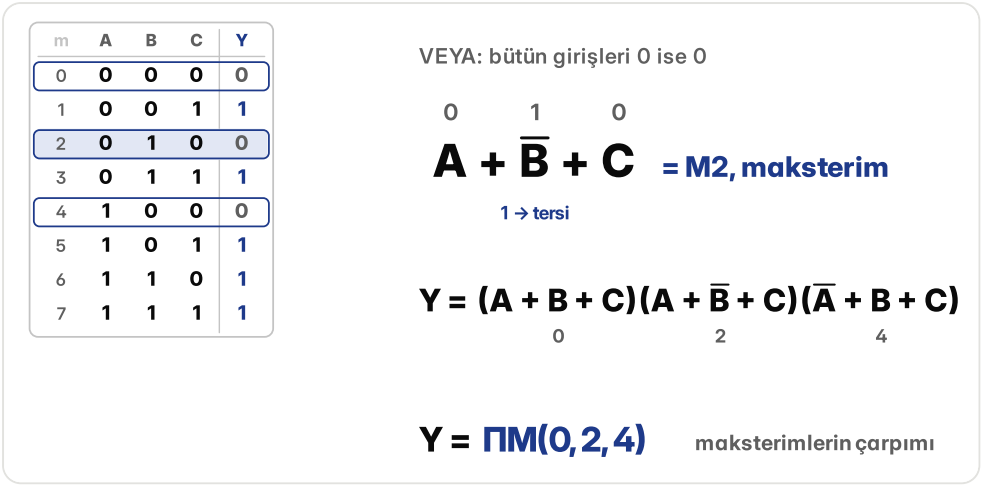
$$Y = \overline{A}\overline{B}\overline{C} + \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC = \Sigma m(1, 3, 5, 6, 7)$$



Şekil 2.2 Bahçe lambası: 5. satırda A = 1, B = 0, C = 1 için minterim $A\overline{B}C$; tabloda her satırın minterimi.

Aynı lamba söndüğü satırlardan da yazılabilir. Bu kez VEYA kullanılır, çünkü VEYA yalnız bütün girişleri 0 olunca 0 verir. 2. satırda A = 0, B = 1, C = 0'dır; 1 olan B'nin tersi alınır: $A + \overline{B} + C$. Yalnız tek bir satırda 0 olan böyle VEYA terimine **maksterim** denir (M0 ... M7). Lamba 0., 2. ve 4. satırlarda sönüktür; bu üç maksterim VE ile bağlanır. Sönük bir satırda o satırın maksterimi 0 olur ve VE de 0 verir. Bu yazıma **maksterimlerin çarpımı** denir:

$$Y = (A + B + C)(A + \overline{B} + C)(\overline{A} + B + C) = \Pi M(0, 2, 4)$$



Şekil 2.3 Maksterimlerin çarpımı: 2. satırın maksterimi A + $\overline{B}$ + C ve 4. satır için çarpanların değerleri 1, 1, 0.

İki yazım arasında geçmek için hesap gerekmez: 0'dan 7'ye numaralardan minterim listesinde olmayanlar maksterim listesini oluşturur. 1'ler azsa minterimle, 0'lar azsa maksterimle yazmak kısadır; lambaya 3 maksterim yetti. Her terimde her değişkenin bulunduğu bu iki yazıma **kanonik biçim** denir. Bir fonksiyonun minterim listesi tektir; bu yüzden içleri ne kadar farklı görünürse görünsün, listesi aynı iki devre aynı işi yapar. Baştaki A + B ve A + $\overline{A}$B kutularının ikisi de Σm(1, 2, 3)'tür.

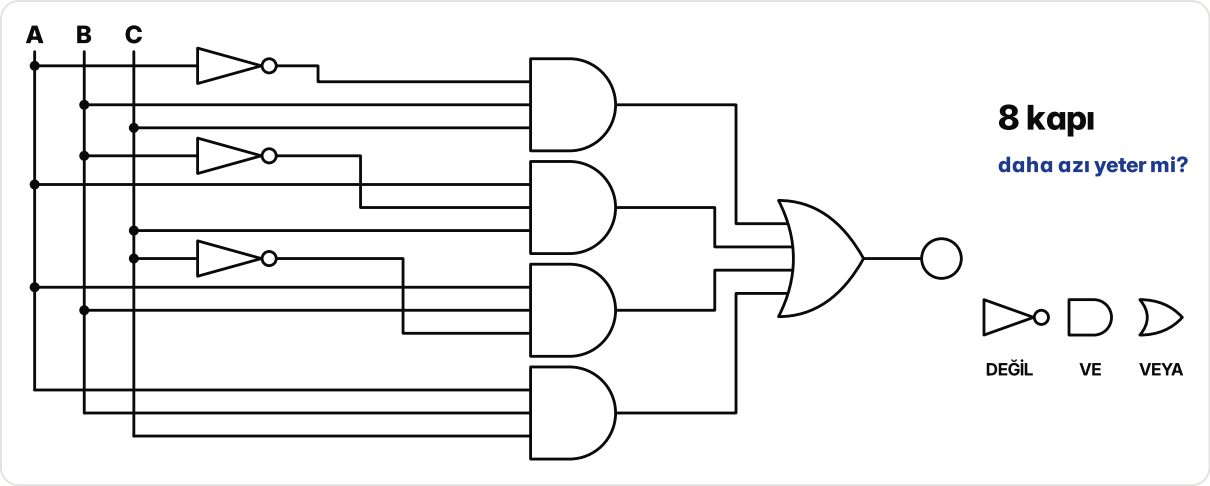
Jüri

videoda 11:41

Üç kişilik bir jüride her üyenin önünde bir düğme var; en az ikisi evet derse ışık yanacak. Tasarım sözle değil, tabloyla başlar. Üyeler A, B, C; evet 1, hayır 0. Her satırda evetler sayılır, 2 ya da 3 evet varsa ışık 1 olur. Tümleyen sütunu $\overline{Y}$ bölümün sonunda kullanılacak:

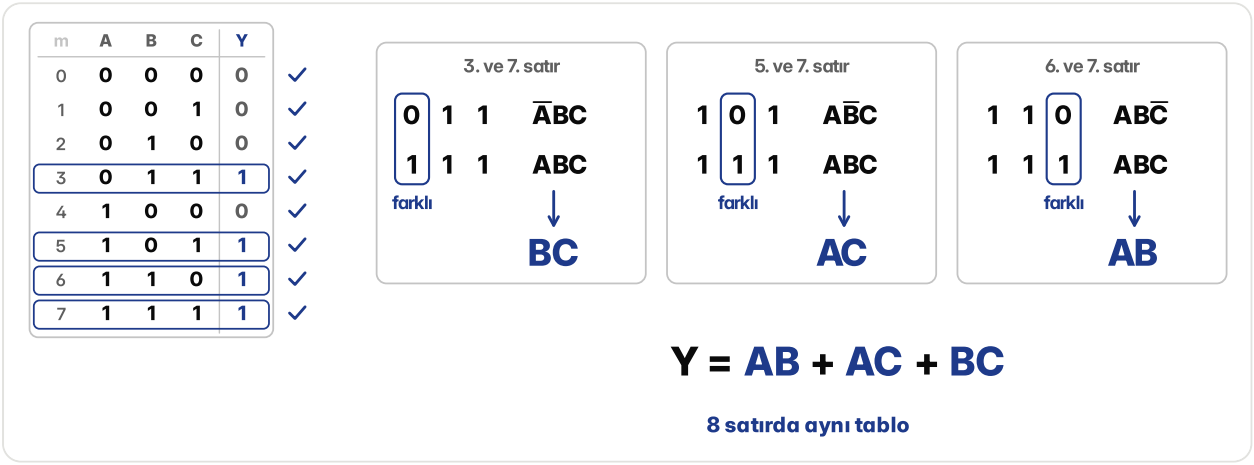
```
no  A  B  C | evet  Y  Ȳ
 0  0  0  0 |   0   0  1
 1  0  0  1 |   1   0  1
 2  0  1  0 |   1   0  1
 3  0  1  1 |   2   1  0
 4  1  0  0 |   1   0  1
 5  1  0  1 |   2   1  0
 6  1  1  0 |   2   1  0
 7  1  1  1 |   3   1  0
```

Işık 3, 5, 6 ve 7. satırlarda yanar. Bu satırların minterimleri VEYA ile bağlanınca kanonik biçim çıkar: Y = $\overline{A}$BC + A$\overline{B}$C + AB$\overline{C}$ + ABC = Σm(3, 5, 6, 7). İfade olduğu gibi devreye dönüşür: girişlerin tersi için 3 DEĞİL kapısı, 4 tane 3 girişli VE kapısı ve sonunda 4 girişli bir VEYA; toplam 8 kapı. Devre doğru çalışır, ama daha azı yeter.



Şekil 2.4 Kanonik biçimden kurulan çoğunluk devresi: 3 DEĞİL, 4 VE ve 1 VEYA, toplam 8 kapı.

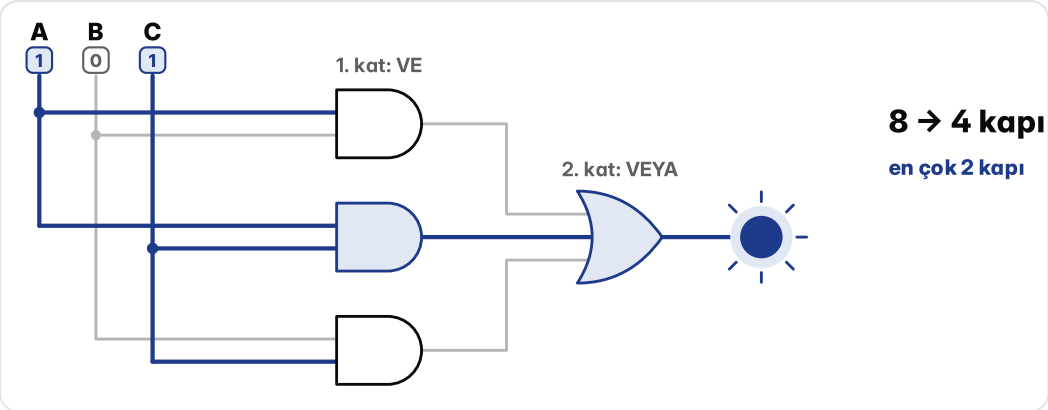
3. ve 7. satırlara bakalım: ikisinde de B ve C evettir, yalnız A farklıdır. A ne derse desin ışık yanar; bu iki terim tek terimde birleşir: $\overline{A}BC + ABC = BC$. Aynı yoldan 5. ve 7. satırlar AC'yi, 6. ve 7. satırlar AB'yi verir. 7. satır üç kez kullanıldı; VEYA'da aynı terimi tekrar saymak sonucu değiştirmez. Sonuç $Y = AB + AC + BC$'dir ve 8 satırın hepsinde aynı tabloyu verir.



Şekil 2.5 Satırları birleştirmek: yalnız bir değişkeni farklı olan satır çiftleri BC, AC ve AB terimlerini verir.

AB + AC + BC de VE terimlerinin VEYA ile bağlanmasıdır; buna **çarpımların toplamı** (sum of products) denir. Kanonik biçimden farkı, her terimde her değişkenin bulunmamasıdır; bu yüzden adı **standart biçim** olur. İkizi **toplamların çarpımı** (product of sums) biçimidir: VEYA terimleri VE ile bağlanır. Jüri böyle de yazılır: $Y = (A + B)(A + C)(B + C)$; yani her ikiliden en az biri evet demelidir. Tablo yine aynıdır.

Çarpımların toplamı doğrudan **iki katlı devre** olur: önce VE kapıları, sonra tek bir VEYA. 3 tane 2 girişli VE ve 1 tane 3 girişli VEYA, toplam 4 kapı: ilk devrenin yarısı. Her giriş ışığa en çok 2 kapıdan geçerek ulaşır. A = 1, B = 0, C = 1 için AB = 0, AC = 1, BC = 0 olur; VEYA kapısına 1 gelir ve ışık yanar.



Şekil 2.6 İki katlı devre: A = 1, B = 0, C = 1 iken yalnız AC kapısı 1 verir, VEYA çıkışı 1 olur ve ışık yanar.

Çoğunluk hayır derse yanacak ret ışığı, fonksiyonun **tümleyeni** olur: çıkış sütununda her 1 0, her 0 1 olur. Ret ışığı 0, 1, 2 ve 4. satırlarda yanar: $\overline{Y} = \Sigma m(0, 1, 2, 4)$, yani çoğunluğun listesinde olmayan numaralar. İfadesi sözünden de okunur, en az ikisi hayır derse: $\overline{Y} = \overline{AB} + \overline{AC} + \overline{BC}$. De Morgan kuralı da aynı tabloya götürür: AB + AC + BC'nin tümleyeninde her değişkenin tersi alınır, VE ile VEYA yer değiştirir ve $(\overline{A} + \overline{B})(\overline{A} + \overline{C})(\overline{B} + \overline{C})$ çıkar.

Aynı çoğunluk devresi, uzay araçları gibi hataya yer olmayan sistemlerde de kullanılır: bir hesap üç ayrı devrede yapılır, sonuçlar bu devreye girer. Biri yanlış sonuç verse bile öteki ikisi çoğunluğu oluşturur ve çıkış doğru kalır.

Özet

- Lojik fonksiyon, girişlerin her birleşimine 0 ya da 1 karşılık getiren kuraldır; n girişte 2^n satır ve 2 üzeri 2^n fonksiyon vardır.
- Minterim tek bir satırda 1 olan VE terimidir; maksterim tek bir satırda 0 olan VEYA terimidir.
- Minterimlerin toplamı (Σm) ve maksterimlerin çarpımı (ΠM) kanonik biçimlerdir; birinin listesinde olmayan numaralar ötekinin listesidir.
- Yalnız bir değişkeni farklı olan iki satırın terimi birleşir ve o değişken düşer; çoğunluk Y = AB + AC + BC olur.
- Çarpımların toplamı iki katlı devre olur: önce VE'ler, sonra tek VEYA; çoğunlukta 8 kapı yerine 4 kapı.
- Tümleyende her 1 0, her 0 1 olur; minterim listesi, asıl fonksiyonun listesinde olmayan numaralardır.

Sorular

1 İki girişli bir devreyle kaç farklı lojik fonksiyon yazılabilir?

A 4 **B** 8

C 16 **D** 256

2 Bir maksterim, doğruluk tablosunun kaç satırında 0 olur?

A Hiçbirinde **B** 1 olan satırların hepsinde

C Yarısında **D** Tam bir satırda

3 Üç girişli bir tabloda A = 0, B = 1, C = 1 satırının (3. satır) minterimi hangisidir?

A $A\,\overline{B}\,\overline{C}$ **B** $\overline{A}\,BC$

C $A + \overline{B} + \overline{C}$ **D** $\overline{A} + B + C$

4 Üç girişli bir fonksiyon Σm(0, 2, 3, 7) ile verilmiş. Aynı fonksiyonu maksterimlerin çarpımı olarak yazarken hangi numaralar kullanılır? Virgülle ayırarak yaz.

Cevap:

5 Çoğunluk fonksiyonunda 3. satırın minterimi $\overline{A}BC$, 7. satırınki ABC'dir. İkisi VEYA ile birleşince hangi terim kalır?

A A **B** ABC

C BC **D** $\overline{A}\,B$

6 Y = AB + AC + BC iki katlı devresinde A = 0, B = 1, C = 1 iken AB, AC ve BC kapılarının çıkışlarını bu sırayla yaz.

Cevap:

7 Çoğunluk fonksiyonu kanonik biçimden 8 kapıyla kurulur. Y = AB + AC + BC ile kurulan iki katlı devre kaç kapı gerektirir?

A 4 **B** 3

C 6 **D** 8

8 AB + AC + BC'nin tümleyeni hangisidir?

A $\overline{A}\,\overline{B}\,\overline{C}$ **B** $\overline{A} + \overline{B} + \overline{C}$

C $(A + B)(A + C)(B + C)$ **D** $\overline{A}\,\overline{B} + \overline{A}\,\overline{C} + \overline{B}\,\overline{C}$

SAYISAL TASARIM

3 Lojik kapılar

Bir bilgisayarın içinde milyarlarca kapı çalışır. Bu bölümde kapı ailesinin bütün üyelerini, tek bir kapı türünün bütün devreleri kurmaya nasıl yettiğini, kapıların çipin içinde transistörlerle nasıl yapıldığını ve bir devrenin hızını neyin belirlediğini göreceğiz.

Tek tür kapı

videoda 13:51

Soruyla başlayalım: bir bilgisayardaki milyarlarca kapının hepsi tek bir türden olsaydı, yine de bilgisayar kurabilir miydik? Cevaba varmak için önce elimizdeki kapılara bakalım.

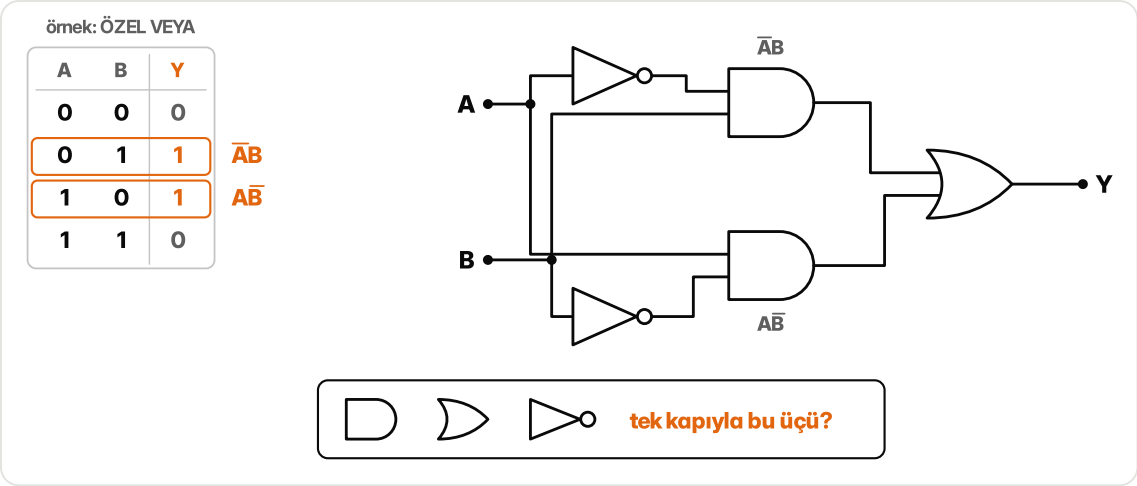
VE, VEYA ve DEĞİL kapıları tanıdık. DEĞİL kapısının ucundaki küçük daireye **balon** denir; balon tersini almak demektir. Balonsuz üçgen **tampon**dur: çıkışı girişin aynısıdır. İşe yaramaz gibi görünür, ama zayıflayan sinyali tazeler ve bir çıkışın birçok girişi beslemesini sağlar. VE kapısının ucuna balon koyunca **VE DEĞİL** (NAND) olur: girişlerin hepsi 1'se 0, yoksa 1 verir. VEYA kapısının ucuna balon koyunca **VEYA DEĞİL** (NOR) olur: girişlerden hiçbiri 1 değilse 1 verir. **ÖZEL VEYA** (XOR, $\oplus$) girişler farklıysa 1 verir; balonlusu **ÖZEL VEYA DEĞİL** (XNOR) girişler aynıysa 1 verir.

```
A  B | VE  VE DEĞİL  VEYA  VEYA DEĞİL
0  0 |  0      1       0       1
0  1 |  0      1       1       0
1  0 |  0      1       1       0
1  1 |  1      0       1       0

A  B | ÖZEL VEYA  ÖZEL VEYA DEĞİL
0  0 |     0             1
0  1 |     1             0
1  0 |     1             0
1  1 |     0             1
```

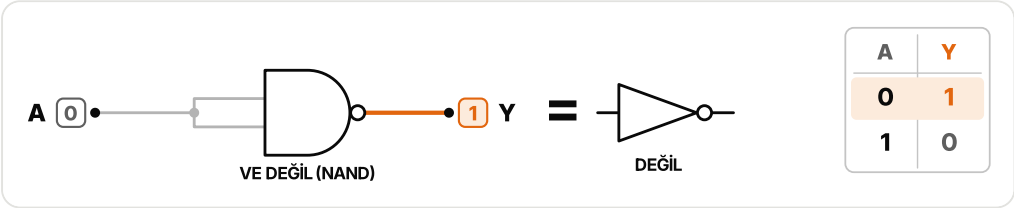
Bu kapılar ikiden fazla girişle de yapılır. Üç girişli VEYA DEĞİL, sekiz giriş birleşiminden yalnız birinde, üç girişin üçü de 0 iken 1 verir.

Peki neden tek bir kapı türü yetebilir? Her doğruluk tablosu VE, VEYA ve DEĞİL ile yazılabilir: tablonun 1 olan her satırı için bir VE terimi kurulur, sonunda hepsi bir VEYA ile bağlanır. Örneğin ÖZEL VEYA yalnız 01 ve 10 satırlarında 1'dir, bu yüzden $\overline{A}B + A\overline{B}$ olarak yazılır. Öyleyse tek bir kapı türüyle bu üç işi kurabilirsek, her devreyi kurabiliriz.



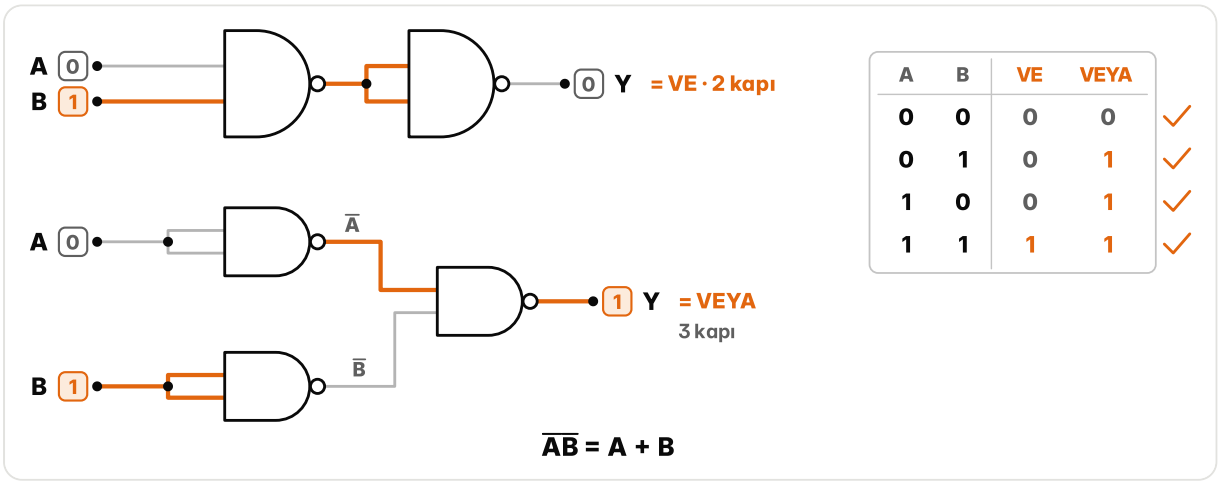
Şekil 3.1 ÖZEL VEYA tablosunun 1 olan iki satırı $A\bar{B}$ ve $\bar{A}B$ terimlerini verir; iki DEĞİL, iki VE ve bir VEYA ile kurulur.

VE DEĞİL kapısının iki girişini birleştirelim. Giriş 1 olunca iki giriş de 1 olur, çıkış 0; giriş 0 olunca çıkış 1. Bu tam olarak DEĞİL kapısıdır.



Şekil 3.2 Girişleri birleştirilmiş VE DEĞİL kapısı DEĞİL gibi çalışır: A = 0 iken çıkış 1.

VE kurmak kolaydır: VE DEĞİL'in çıkışı, girişleri birleştirilmiş ikinci bir VE DEĞİL ile ters çevrilir; iki kapı yeter. VEYA için önce A ve B ayrı ayrı ters çevrilir, sonra ikisi bir VE DEĞİL'e verilir. $\bar{A}\cdot\bar{B}$'nin tümleyeni, De Morgan kuralına göre A + B'dir. Deneyelim: A = 0, B = 1. Ters çevrilince 1 ve 0 olurlar; VE DEĞİL bu ikisine 1 verir, tıpkı VEYA gibi. Bu devre üç kapıdır ve dört satırın dördünde de VEYA'nın tablosunu verir.



Şekil 3.3 Yalnız VE DEĞİL ile VE (2 kapı) ve VEYA (3 kapı); A = 0, B = 1 için değerler ve iki devrenin tablosu.

Aynı yol VEYA DEĞİL kapısıyla da işler: girişleri birleştirince DEĞİL, ardına DEĞİL ekleyince VEYA, girişleri ters çevirince VE çıkar. Bu iki kapıya **evrensel kapı** denir; her devre yalnız birinden kurulabilir.

Bu fikir kâğıt üstünde kalmadı. Apollo araçlarını Ay'a götüren rehber bilgisayarın bütün mantık devreleri tek bir kapıdan kuruldu: üç girişli VEYA DEĞİL. Her yongada bu kapıdan iki tane vardı; mantık için tek bir yonga türü üretip sınamak yetiyordu. Baştaki sorunun cevabı evet: tek tür kapı, bütün bir bilgisayara yeter.

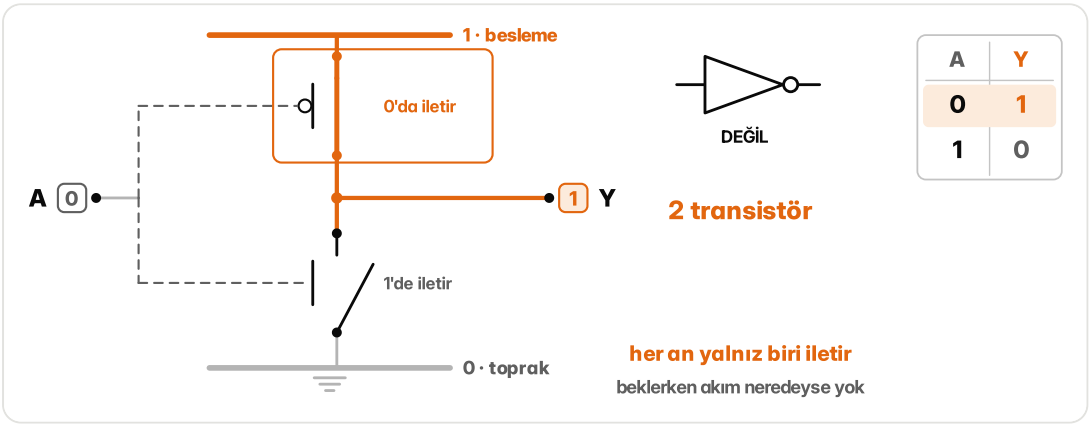
Ters fiyat

videoda 15:36

VE DEĞİL kapısı, VE kapısının ardına bir DEĞİL eklenmiş gibi görünür; öyleyse VE daha ucuz olmalı. Çipte durum tam tersidir: VE DEĞİL daha az transistörle kurulur. Nedenini görmek için kapının içine bakalım.

Bugünkü çipler **CMOS** denen yapıyla üretilir. Transistör, girişine gelen gerilimle açılıp kapanan bir anahtardır. CMOS'ta iki türü birlikte kullanılır: biri girişi 1 olunca iletir, öteki girişi 0 olunca (bunlara n kanallı ve p kanallı transistör denir). Kapının çıkışı ya 1'e, yani besleme gerilimine, ya da 0'a, yani toprağa bağlanır.

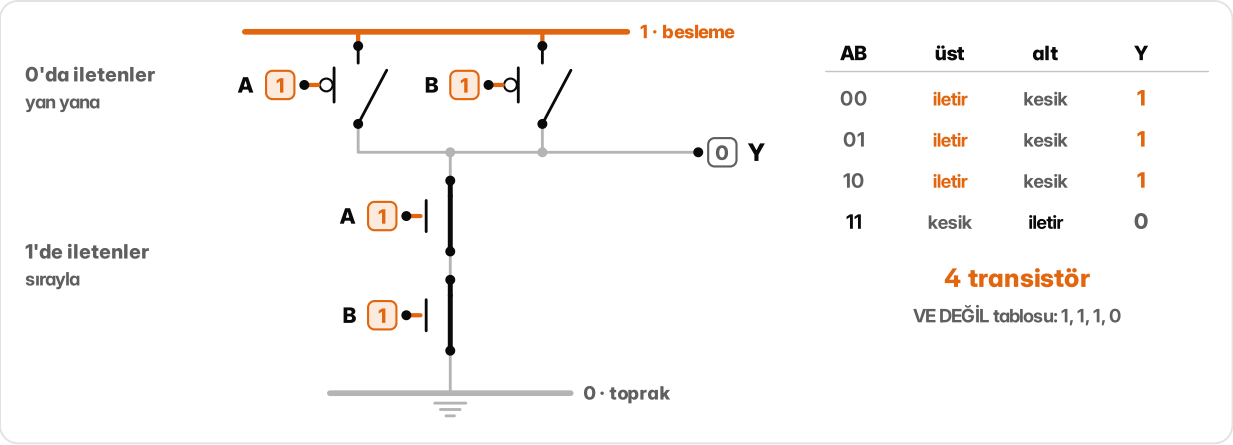
En basiti DEĞİL kapısıdır ve iki transistörle kurulur. 0'da ileten transistör çıkışı 1'e, 1'de ileten transistör çıkışı 0'a bağlar. Giriş 1 olunca alttaki iletir, çıkış 0 olur; giriş 0 olunca üstteki iletir, çıkış 1 olur. Her an yalnız biri iletir; bu yüzden kapı beklerken neredeyse hiç akım çekmez.



Şekil 3.4 CMOS DEĞİL kapısı: üstte 0'da ileten, altta 1'de ileten transistör; ikisinin arası çıkış.

VE DEĞİL için dört transistör yeter. Alta, 1'de iletenlerden ikisi sırayla konur: çıkış ancak iki giriş de 1 olunca 0'a bağlanır. Üste, 0'da iletenlerden ikisi yan yana konur: girişlerden biri 0 olunca çıkış 1'e bağlanır. Dört satırın dördünde VE DEĞİL'in tablosu çıkar:

```
A  B | üst     alt    | Y
0  0 | iletir  kesik  | 1
0  1 | iletir  kesik  | 1
1  0 | iletir  kesik  | 1
1  1 | kesik   iletir | 0
```

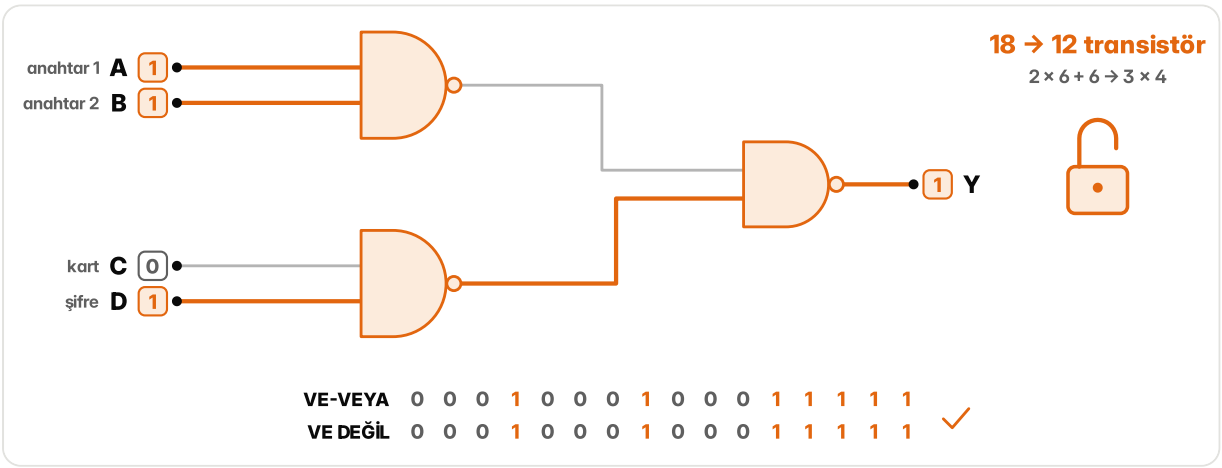


Şekil 3.5 CMOS VE DEĞİL: üstte yan yana iki, altta sırayla iki transistör; yanında hangisinin ilettiğini gösteren tablo.

Peki VE doğrudan kurulamaz mı? CMOS'ta 1'de ileten transistörler hep altta, 0'da iletenler hep üstte durur; giriş 1'e çıkınca çıkış ancak 0'a inebilir. Bu yüzden tek katlı her CMOS kapısı tersler. VE kapısı aslında VE DEĞİL artı DEĞİL'dir: 4 + 2 = 6 transistör. VEYA DEĞİL de 4 transistör, VEYA 6 transistördür. Çipte en ucuz kapılar VE DEĞİL ile VEYA DEĞİL'dir. İkisinden VE DEĞİL çoğu zaman öne geçer: sırayla bağladığı transistörler daha hızlı türdendir.

Ucuz kapıyı kullanmak için devreyi dönüştürmeyi öğrenelim. Bir kasa, iki yönetici anahtarlarını birlikte çevirince ya da kartla şifre birlikte girilince açılsın. Anahtarlar A ve B, kart C, şifre D ise Y = AB + CD olur: iki VE ve bir VEYA ile iki katlı bir devre. Şimdi her VE kapısının çıkışına bir balon, VEYA kapısının her girişine bir balon ekleyelim. Aynı teldeki iki balon birbirini götürür, devre değişmez. VE kapıları VE DEĞİL oldu. Girişleri balonlu VEYA ise De Morgan kuralına göre ($\overline{A} + \overline{B}$, $A \cdot B$'nin tümleyenidir) VE DEĞİL'in ta kendisidir. Üç kapının üçü de VE DEĞİL olur. Bu yönteme **balon itme** denir.

Deneyelim: A = 1, B = 1, C = 0, D = 1. Üstteki VE DEĞİL 0, alttaki 1 verir; sondaki VE DEĞİL 1 verir ve kasa açılır. On altı satırın hepsinde eski devreyle aynıdır. Transistör sayısı 2 × 6 + 6 = 18'den 3 × 4 = 12'ye iner. Önce VE, sonra VEYA kullanan her iki katlı devre bu yoldan yalnız VE DEĞİL ile kurulur.



Şekil 3.6 Yalnız VE DEĞİL ile kurulan kasa devresi: A = 1, B = 1, C = 0, D = 1 iken kasa açık; iki devrenin 16 satırlık çıkışı aynı.

Aynı kapı, farklı hız

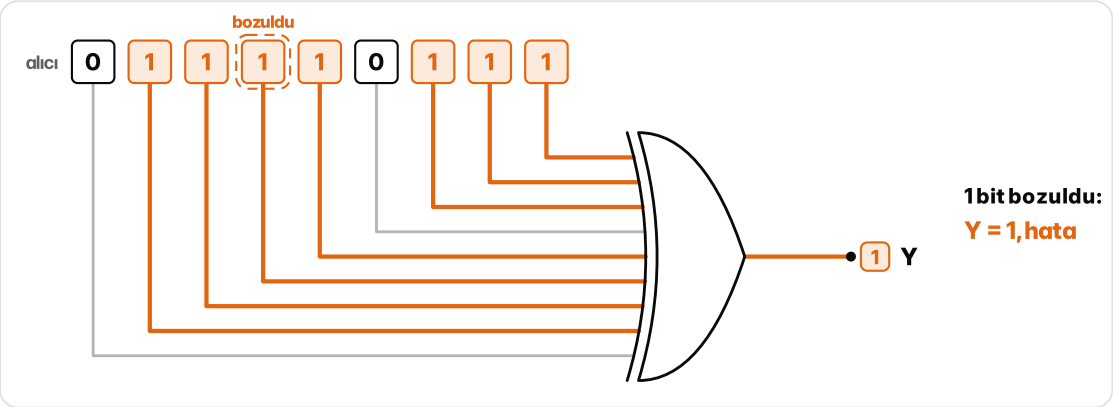
videoda 17:32

Aynı işi yapan, aynı sayıda kapıyla kurulmuş iki devreden biri ötekinden iki kattan daha hızlı olabilir. Nedenini ÖZEL VEYA kapısıyla bulacağız.

ÖZEL VEYA, iki girişi farklıysa 1, aynıysa 0 verir. Yani iki biti karşılaştırır: çıkış 1 ise bitler farklıdır. Balonlusu, ÖZEL VEYA DEĞİL, tam tersini söyler: bitler eşitse 1 verir.

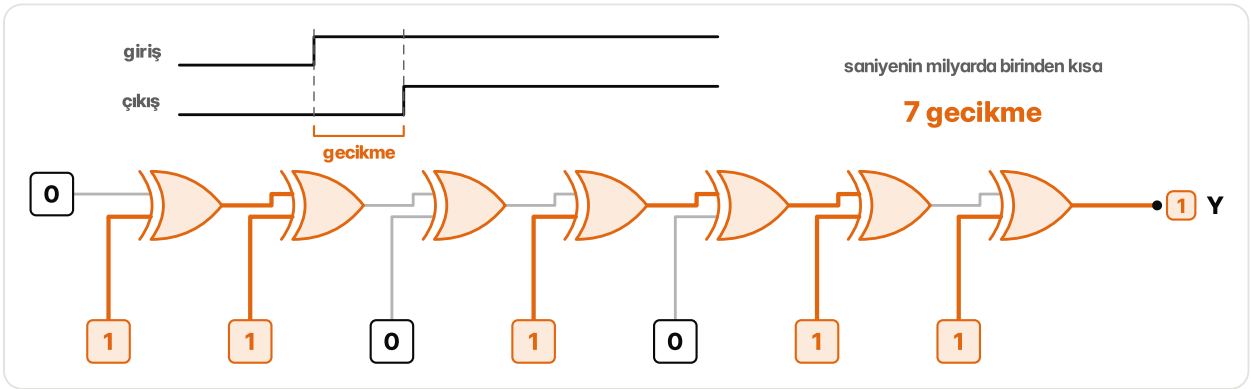
ÖZEL VEYA'nın bir girişine **denetim teli** diyelim. Denetim 0'sa çıkış öteki girişin aynısıdır, 1'se tersidir. Dört bitlik bir sayının her bitine bir ÖZEL VEYA koyup denetim tellerini birleştirelim. 1011, denetim 0 iken olduğu gibi geçer; denetim 1 olunca 0100 olur. Tek bir telle bütün bitler istenen anda ters çevrilir.

Çok girişli ÖZEL VEYA, 1'lerin sayısı tekse 1, çiftse 0 verir. 0110 1011 verisinde beş tane 1 var; çıkış 1. Bu çıkış **eşlik biti**dir: veriye eklenince 1'lerin sayısı çift olur (altı). Alıcı dokuz biti aynı kapıdan geçirir; sonuç 0 ise sorun yoktur. Tek bir bit bozulursa 1'lerin sayısı tek olur, sonuç 1 çıkar ve hata yakalanır. İki bit birden bozulursa sayı yine çift kalır; bu yöntem onu görmez.



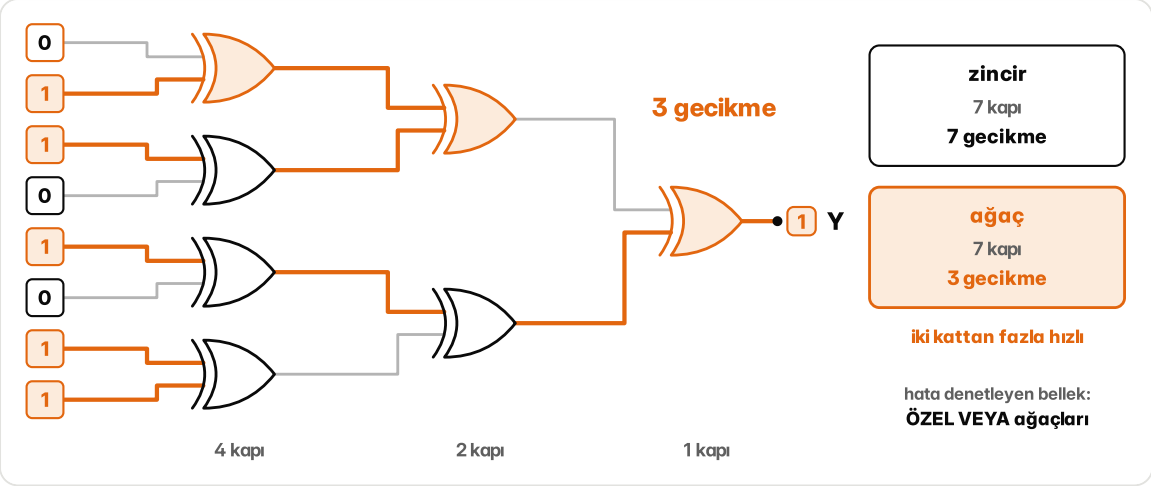
Şekil 3.7 Alıcı dokuz biti çok girişli ÖZEL VEYA'dan geçirir; bozulan tek bit yüzünden çıkış 1, hata yakalandı.

Sekiz girişli ÖZEL VEYA, yedi tane iki girişli ÖZEL VEYA'dan kurulur. İlk akla gelen, onları bir zincir gibi arka arkaya dizmektir. Ama hiçbir kapı anında cevap vermez: giriş değişince çıkış biraz sonra değişir. Buna **yayılma gecikmesi** denir. Bugünkü çiplerde bu süre saniyenin milyarda birinden de kısadır, ama zincirde birikir: en baştaki bit yedi kapıdan geçer, yedi gecikme beklenir.



Şekil 3.8 Yedi ÖZEL VEYA'lık zincir: çıkış girişten bir gecikme sonra değişir, en baştaki bit yedi kapıdan geçer.

Aynı yedi kapıyı ağaç gibi dizelim: önce dört kapı bitleri ikişer ikişer alır, sonra iki kapı, en sonda bir kapı. Her bit yalnız üç kapıdan geçer. Kapı sayısı aynıdır, ama devre 7 / 3 ≈ 2,3 kat, yani iki kattan fazla hızlıdır. Hata denetleyen bellekler de denetim bitlerini böyle ÖZEL VEYA ağaçlarıyla hesaplar. Bir devrenin hızını kapı sayısı değil, en uzun yolu belirler.



Şekil 3.9 Aynı yedi ÖZEL VEYA ağaç düzeninde: en uzun yol üç kapı; zincirle karşılaştırma.

Özet

- VE DEĞİL ve VEYA DEĞİL evrensel kapılardır: DEĞİL, VE ve VEYA, dolayısıyla her devre yalnız birinden kurulabilir.
- Apollo rehber bilgisayarının bütün mantık devreleri üç girişli VEYA DEĞİL kapılarıyla kuruldu.
- CMOS'ta DEĞİL 2, VE DEĞİL ve VEYA DEĞİL 4, VE ve VEYA 6 transistördür; tek katlı her CMOS kapısı tersler.
- Balon itme ile iki katlı VE-VEYA devresi yalnız VE DEĞİL kapılarıyla kurulur; kasa örneğinde transistör sayısı 18'den 12'ye indi.
- ÖZEL VEYA iki biti karşılaştırır, denetim teliyle istenince ters çevirir ve çok girişli hâliyle eşlik biti üretir.
- Yayılma gecikmesi yol boyunca birikir: aynı yedi kapı zincirde 7, ağaçta 3 gecikme verir.

Sorular

1 Hangi kapının çıkışı yalnız bütün girişleri 1 olunca 0 olur?

A VE
B VE DEĞİL
C VEYA DEĞİL
D ÖZEL VEYA

2 **Üç girişli VEYA DEĞİL kapısı hangi durumda 1 verir?**

A En az bir girişi 1 iken

B Üç girişi de 1 iken

C Üç girişi de 0 iken

D 1'lerin sayısı tekken

3 **VEYA yalnız VE DEĞİL kapılarıyla kuruluyor: A ve B ayrı ayrı ters çevriliyor ($\overline{A}$, $\overline{B}$), sonra ikisi bir VE DEĞİL'e giriyor. A = 1, B = 0 iken $\overline{A}$, $\overline{B}$ ve çıkış hangi değerleri alır? Üç biti bu sırayla yaz.**

Cevap:

4 **CMOS'ta iki girişli bir VE kapısı kaç transistörle kurulur?**

A 4

B 2

C 8

D 6

5 **CMOS VE DEĞİL kapısında girişi 1 iken ileten iki transistör nasıl bağlanır?**

A Altta, sırayla

B Altta, yan yana

C Üstte, sırayla

D Üstte, yan yana

6 **Y = AB + CD devresi balon itmeyle üç VE DEĞİL kapısından kuruldu: üstteki A ve B'yi, alttaki C ve D'yi alıyor, sondaki bu ikisinin çıkışını alıyor. A = 0, B = 1, C = 1, D = 1 iken üstteki kapının, alttaki kapının ve Y'nin değerlerini bu sırayla yaz.**

Cevap:

7 **1001 0111 verisine, 1'lerin sayısı çift olsun diye eklenecek eşlik biti nedir?**

Cevap:

8 **16 girişli bir ÖZEL VEYA, iki girişli ÖZEL VEYA kapılarıyla ağaç düzeninde kuruluyor. Bir bitin çıkışa varırken geçtiği en uzun yol kaç kapıdır?**

A 15

B 4

C 8

D 3

SAYISAL TASARIM

4 Karnaugh haritaları

Bir doğruluk tablosundan doğrudan kurulan devre doğru çalışır ama çoğu zaman gereğinden çok kapı kullanır. Bu bölümde tabloyu en kısa formüle indiren Karnaugh haritası anlatılıyor: haritanın neden böyle dizildiği, birlerin nasıl gruplandığı, hangi grupların zorunlu olduğu ve hiç gelmeyecek girişlerin işe nasıl katıldığı.

Uyarı sesi

videoda 18:57

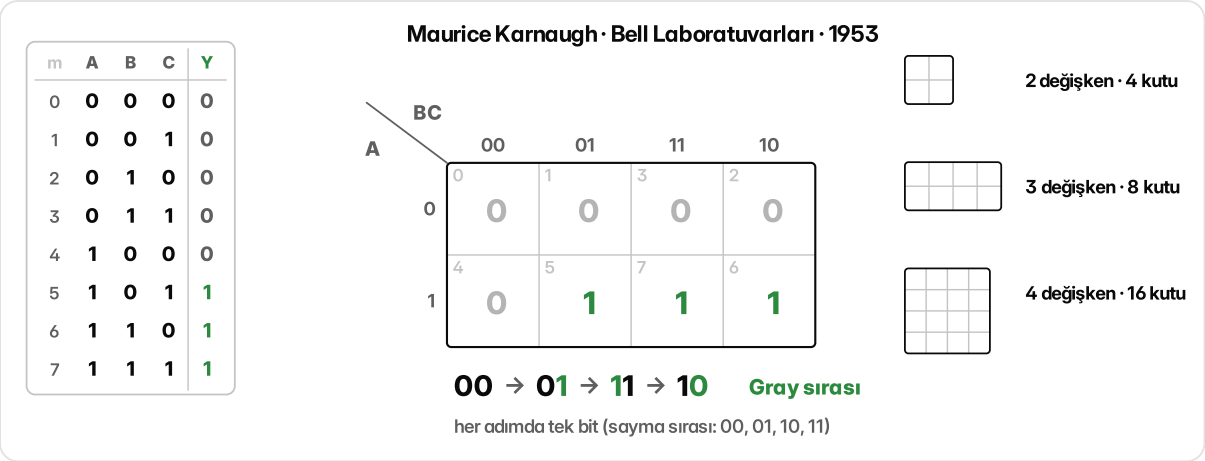
Arabadan inerken çalan uyarı sesi üç durumda çalsın: kapı açık, anahtar kontakta, farlar kapalı; kapı açık, anahtar çıkarılmış, farlar açık; kapı açık, anahtar kontakta, farlar açık. Kapı açık A, anahtar kontakta B, farlar açık C olsun. Üç durum tablonun 6, 5 ve 7. satırlarıdır:

```
m  A  B  C | Y
0  0  0  0 | 0
1  0  0  1 | 0
2  0  1  0 | 0
3  0  1  1 | 0
4  1  0  0 | 0
5  1  0  1 | 1
6  1  1  0 | 1
7  1  1  1 | 1
```

Her satır bir **minterim** verir, yani yalnız o satırda 1 olan VE terimi. Üçünü VEYA ile bağlayınca Y = $A\overline{B}C$ + $AB\overline{C}$ + ABC çıkar. Bu ifade iki DEĞİL kapısı, üç tane üç girişli VE ve bir VEYA ile kurulur: 6 kapı, sinyal girişten çıkışa 3 kat geçer. Devre çalışır, ama her kapı birkaç transistör demektir: çipte yer kaplar, güç harcar ve her kat biraz gecikme ekler. Aynı devre bir çipte binlerce kez tekrarlanınca fazladan her kapı binlerce kez ödenir.

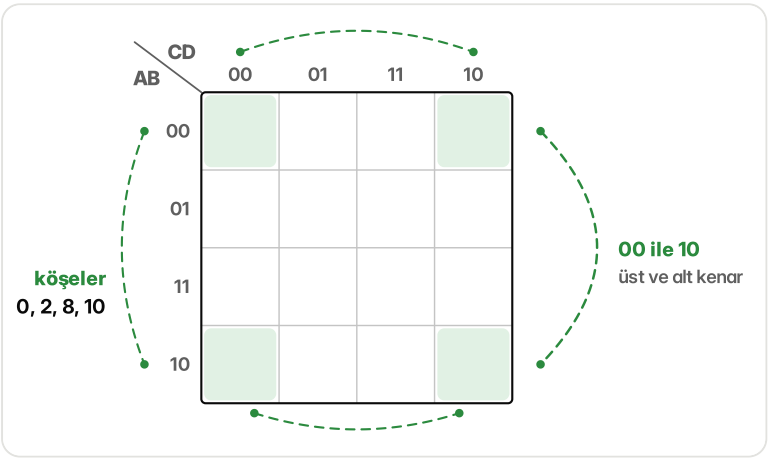
Kısaltmanın tek kuralı şudur: yalnız bir değişkende ayrılan iki minterim birleşir, o değişken düşer. 5 (101) ile 7 (111) yalnız B'de farklıdır. Kapı açık ve farlar açıkken anahtar ne olursa olsun ses çalar; bu yüzden $A\overline{B}C$ + ABC = AC. Zor olan böyle çiftleri bulmaktır, çünkü tablo onları saklar: 5 ile 6 komşu satırlardır ama iki bitte ayrılır; 0 ile 4 tek bitte ayrılır ama aralarında 3 satır vardır.

1953'te Bell Laboratuvarları'ndan Maurice Karnaugh tabloyu bir ızgaraya dizmeyi önerdi: yan yana her iki kutu tek bitle ayrılsın. Üç değişkende iki satır A'ya, dört sütun BC'ye ayrılır ve sütunlar 00, 01, 11, 10 sırasıyla dizilir. Bu alışılmış sayma sırası değildir; her adımda yalnız bir bit değişir. Gray kodunun sırası da budur. Her kutuya tablodaki satırın çıkışı yazılır. Harita iki değişkende 4, üçte 8, dörtte 16 kutudur.



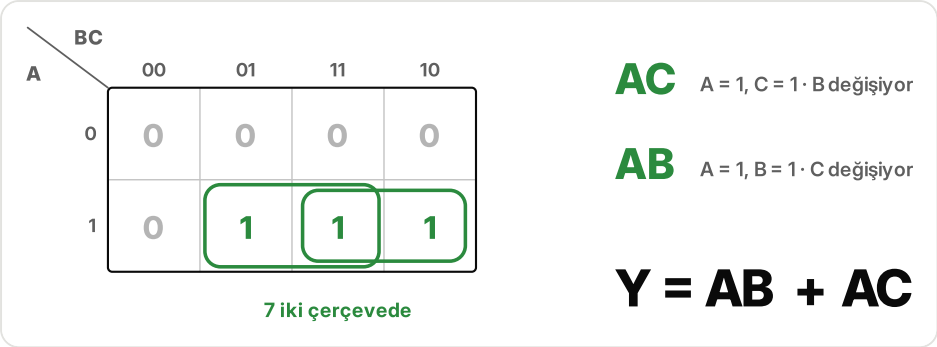
Şekil 4.1 Üç değişkenli Karnaugh haritası: sütunlar 00, 01, 11, 10 sırasıyla dizili, kutularda uyarı sesinin çıkışları.

Komşuluk kenarda bitmez. En sağdaki sütun (10) ile en soldaki (00) de tek bitle ayrılır; harita kâğıttan kesilip sağ kenarı sol kenara yapıştırılsa bir boru olurdu. Dört değişkende satırlar da aynı sırayla dizilir ve üst kenar alt kenara komşudur. Dört köşe de birbirine bağlıdır: her köşe, aynı satırdaki ve aynı sütundaki köşeyle tek bitle ayrılır.



Şekil 4.2 Dört değişkenli haritada karşılıklı kenarlar ve dört köşe (0, 2, 8, 10) birbirine komşu.

Uyarı sesinde birler 5, 7 ve 6 numaralı kutularda, A = 1 satırında yan yanadır. 5 ile 7'yi bir çerçeveye alınca A ve C hep 1, B değişir: terim AC. 7 ile 6'da A ve B hep 1, C değişir: terim AB. Bir kutu iki çerçeveye birden girebilir. Sonuç Y = AB + AC: iki tane iki girişli VE ve bir VEYA, yani 6 yerine 3 kapı. DEĞİL kapısı gerekmez, sinyal 2 kattan geçer. Sekiz satırın hepsinde tablo aynıdır. Harita durum listesini tek cümlelik bir kurala çevirdi: kapı açıkken anahtar kontaktaysa ya da farlar açıksa ses çalar.



Şekil 4.3 Haritada iki çerçeve: 5 ile 7'den AC, 7 ile 6'dan AB; Y = AB + AC.

Yedi sayı

videoda 21:10

0 ve 1 ile başlayıp her adımda son iki sayıyı toplayınca 1, 2, 3, 5, 8, 13 gelir: Fibonacci dizisi. 15'e kadar dizide 7 sayı vardır: 0, 1, 2, 3, 5, 8, 13. Dört bitlik bir sayının bu dizide olup olmadığını söyleyen devrenin haritasında bu yedi kutuya 1, öteki dokuzuna 0 yazılır. Sayının bitleri A, B, C, D'dir; A en büyük basamaktır (8, 4, 2, 1). Haritada satırlar AB'ye, sütunlar CD'ye ayrılır.

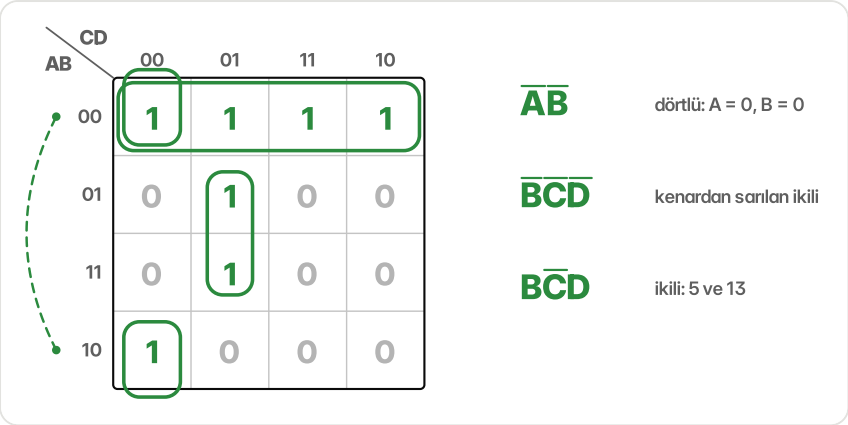


Şekil 4.4 Dört değişkenli harita: üst satırın tamamı, ikinci sütunda üst üste iki kutu (5 ve 13) ve sol alt köşe (8) 1.

Birleri çerçevelemenin kuralları kısadır:

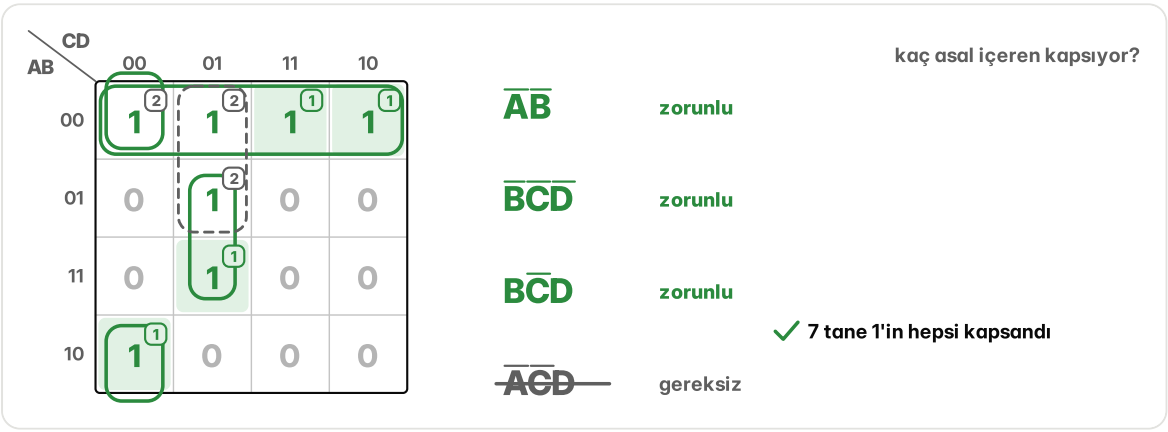
- Bir çerçeve 1, 2, 4 ya da 8 kutu alır ve dikdörtgendir; 3 ya da 6 kutu olmaz. Çerçevede bazı değişkenler sabit kalırken ötekiler bütün değerlerini almalıdır: bir değişken serbestse 2 kutu, ikisi serbestse 4, üçü serbestse 8.
- Çerçeve olabildiğince büyük olmalı: boyu iki katına çıkınca terimden bir harf düşer. Dört değişkende 1 kutu 4 harf, 2 kutu 3, 4 kutu 2, 8 kutu 1 harf verir.
- Çerçeve sayısı olabildiğince az olmalı: her çerçeve bir terimdir.
- Terim, çerçevede değişmeyen değişkenlerin VE'sidir.

Üst satırın dört kutusunda A ve B hep 0, C ile D her değeri alır: terim $\overline{AB}$. Sol alt köşedeki 8 (1000), en üstteki 0 numaralı kutuya komşudur, çünkü harita üst ve alt kenardan birleşir. Bu çerçeve kenardan taşar ve iki parça görünür ama tek gruptur; içinde yalnız A değişir: terim $\overline{BCD}$. İkinci sütunda üst üste duran 5 ile 13'ün terimi $B\overline{C}D$'dir.



Şekil 4.5 Üç çerçeve: üst satırın dörtlüsü $\overline{AB}$, kenardan sarılan 0 ile 8 $\overline{BCD}$, 5 ile 13 $B\overline{C}D$.

5'in bir komşusu daha vardır: üstündeki 1. Onunla da bir ikili kurulur, terimi $A\overline{C}D$. Bu ikili de büyütülemez, çünkü genişleyebileceği her yönde bir 0 vardır (4, 7 ve 9). Büyütülemeyen her çerçeveye **asal içeren** (prime implicant) denir; haritada dört asal içeren var. Hangilerinin gerektiğini bulmak için her 1'e onu kaç asal içerenin kapsadığı sorulur. 2 ve 3 yalnız üst satırın dörtlüsünde, 8 yalnız köşe ikilisinde, 13 yalnız kendi ikilisindedir. Bir 1'i kapsayan tek asal içeren **zorunlu asal içeren** (essential prime implicant) olur ve formüle mutlaka girer. Bu üç zorunlu içeren yedi birin hepsini kapsar; 1 ile 5'in ikilisi doğru bir grup olduğu hâlde gereksiz kalır.



Şekil 4.6 Her birin köşesinde onu kapsayan asal içeren sayısı; zorunlu üç terim ve gereksiz $A\overline{C}D$.

Sonuç Y = $\overline{AB}$ + $\overline{BCD}$ + $B\overline{C}D$: 7 uzun terim ve 28 harf yerine 3 terim ve 8 harf. Örneğin 8 (1000) için $\overline{BCD}$ = 1, yani çıkış 1; 4 (0100) için üç terim de 0. On altı sayının hepsinde devre diziyle aynı cevabı verir. Yöntem hep aynıdır: önce zorunlu çerçeveler alınır, kalan 1 varsa en az çerçeveyle kapatılır. Fazladan bir çerçeve devreyi bozmaz ama boşa harcanmış bir kapıdır.

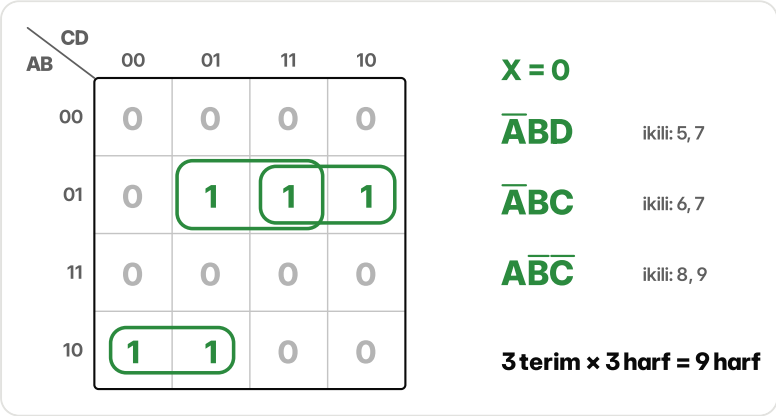
Yuvarlama

videoda 23:12

Pek çok hesap makinesi sayıları **BCD** ile saklar: her onluk rakam 4 bitte durur. Ekrana sığmayan basamağı atarken yuvarlar: atılan rakam 5 ya da büyükse bir önceki rakam 1 artar (3,47 → 3,5; 3,42 → 3,4). Bu kararı veren devre rakamın 4 bitini (A, B, C, D) alır. 4 bit 16 birleşim verir ama rakam 10 tanedir; 1010'dan 1111'e kadar 6 birleşim, yani 10-15, girişe hiç gelmez. Bu satırlarda çıkışın ne olduğu fark etmez; tabloya X yazılır ve bunlara **önemsiz durum** (don't care) denir:

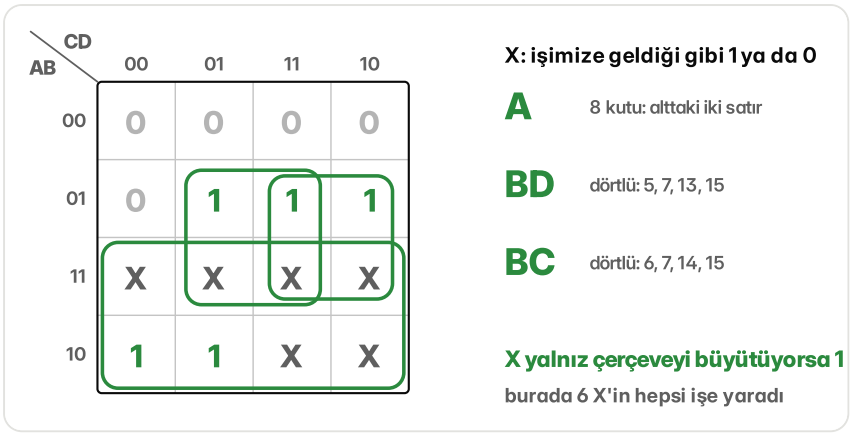
```
 n  A  B  C  D | Y
 0  0  0  0  0 | 0
 1  0  0  0  1 | 0
 2  0  0  1  0 | 0
 3  0  0  1  1 | 0
 4  0  1  0  0 | 0
 5  0  1  0  1 | 1
 6  0  1  1  0 | 1
 7  0  1  1  1 | 1
 8  1  0  0  0 | 1
 9  1  0  0  1 | 1
10  1  0  1  0 | X
11  1  0  1  1 | X
12  1  1  0  0 | X
13  1  1  0  1 | X
14  1  1  1  0 | X
15  1  1  1  1 | X
```

X'ler önce 0 sayılırsa birler üç ikiliye ayrılır: 5 ile 7 ($\bar{A}BD$), 6 ile 7 ($\bar{A}BC$), 8 ile 9 ($A\bar{B}\bar{C}$). Formül üç terim, her biri üç harf: 9 harf.



Şekil 4.7 X'ler 0 sayılınca üç ikili: $\bar{A}BD$, $\bar{A}BC$ ve $A\bar{B}\bar{C}$.

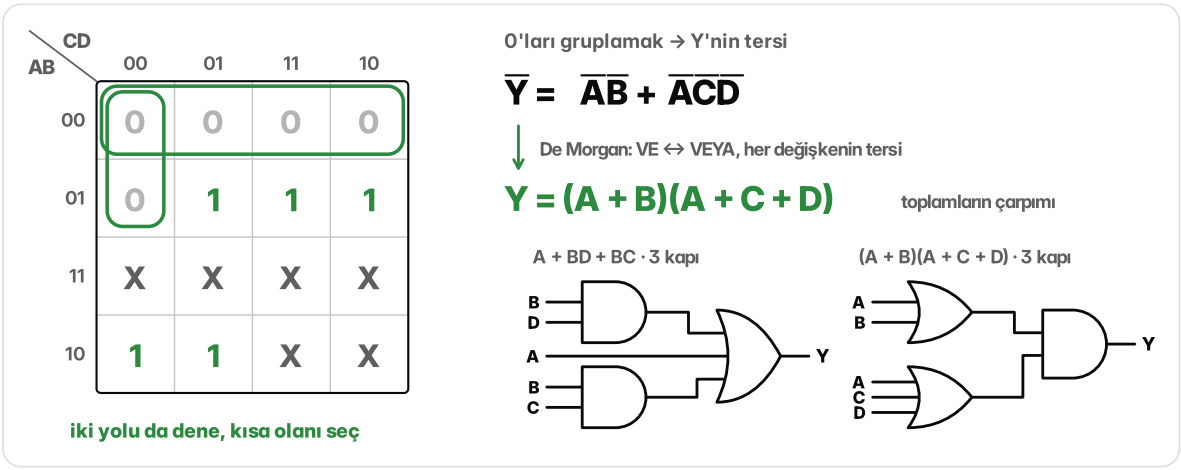
X'ler işe göre 1 ya da 0 sayılabilir. 8 ile 9'un yanındaki ve üstündeki X'ler 1 sayılınca alttaki iki satır 8 kutuluk tek çerçeve olur; içinde yalnız A sabittir: terim A. 5 ve 7, altlarındaki 13 ve 15 ile bir dörtlü kurar: BD. 6 ve 7, 14 ve 15 ile bir dörtlü daha: BC. Bir X yalnız bir çerçeveyi büyütüyorsa 1 sayılır; burada altısı da işe yaradı.



Şekil 4.8 X'ler kullanılınca üç büyük çerçeve: 8 kutuluk A, dörtlüler BD ve BC.

Sonuç Y = A + BD + BC: üç harfli üç terim yerine tek harf ve iki ikili, 16 satır tek satıra indi. Sağlama: 4 (0100) için A = 0, BD = 0, BC = 0, çıkış 0; 5 (0101) için BD = 1, çıkış 1. 0'dan 9'a on rakamın hepsi doğrudur.

Aynı haritada sıfırlar da gruplanabilir; bu kez çıkan, fonksiyonun tersidir. Üst satırdaki dört 0 bir dörtlüdür: $\overline{AB}$. Kalan 0, 4'ün kutusudur ve üstündeki 0 ile bir ikili kurar: $\overline{ACD}$. Böylece $\overline{Y} = \overline{AB} + \overline{ACD}$. Tersi De Morgan kuralıyla çevrilince VE ile VEYA yer değiştirir, her değişkenin tersi alınır: $Y = (A + B)(A + C + D)$. Buna **toplamların çarpımı** denir; bu devre de 3 kapıdır. İki yol da denenip kısa olanı seçilir.



Şekil 4.9 Sıfırların gruplanması: $\overline{AB}$ ve $\overline{ACD}$; De Morgan ile $Y = (A + B)(A + C + D)$ ve iki eşit devre.

Harita küçük işler içindir. Beş değişkende harita 16 kutuluk iki haritaya, altı değişkende dörde bölünür ve komşuları gözle aramak zorlaşır. Daha büyük işlerde aynı birleştirmeyi tablo üzerinde adım adım yapan, bilgisayara uygun yöntemler kullanılır; bunların bilineni Quine-McCluskey yöntemidir (ör. 0101 ile 0111 birleşip 01-1 olur, C düşer). Bugün çip tasarım programları milyonlarca kapılık devreleri sadeleştirirken de aynı fikirden yola çıkar: tek bitle ayrılan iki satır birleşir.

Özet

- Tek bir değişkende ayrılan iki minterim birleşir ve o değişken düşer; Karnaugh haritası bu çiftleri yan yana getirir.
- Haritada satır ve sütunlar 00, 01, 11, 10 sırasıyla dizilir; karşılıklı kenarlar ve dört köşe de komşudur.
- Çerçeve 1, 2, 4 ya da 8 kutuluk bir dikdörtgendir; olabildiğince büyük ve olabildiğince az çerçeve seçilir, terim çerçevede değişmeyen değişkenlerin VE'sidir.
- Büyütülemeyen çerçeve asal içerendir; bir 1'i tek başına kapsayan asal içeren zorunludur. Önce zorunlular alınır, kalan 1'ler en az çerçeveyle kapatılır.
- Önemsiz durumlar (X) yalnız çerçeveyi büyütüyorsa 1 sayılır; BCD'de rakam 5 ya da büyük mü sorusu Y = A + BD + BC'ye iner.
- Sıfırları gruplamak fonksiyonun tersini verir; De Morgan ile toplamların çarpımı biçimine geçilir.

Sorular

1 **Karnaugh haritasında iki değişkenli sütun başlıkları hangi sırayla dizilir?**

A 00, 01, 10, 11

B 00, 01, 11, 10

C 00, 10, 01, 11

D 01, 00, 10, 11

2 **Bir Karnaugh çerçevesi kaç kutu alabilir?**

A 1, 2, 3 ya da 4

B 2, 4, 6 ya da 8

C 1, 2, 4 ya da 8

D Dikdörtgen olduğu sürece istediği kadar

3 **Üç değişkenli bir tabloda 5 (101) ve 6 (110) satırlarının çıkışı 1. Bu iki minterim neden tek terimde birleşemez?**

A Tabloda komşu satırlar oldukları için

B İkisinin de çıkışı 1 olduğu için

C Biri tek, öteki çift sayı olduğu için

D İki bitte ayrıldıkları için

4 **Dört değişkenli (AB satırda, CD sütunda) bir haritada yalnız dört köşedeki kutular 1. Bu dört kutu tek çerçeveye alınırsa terim ne olur?**

A $\overline{B}\ \overline{D}$

B BD

C $\overline{A}\ \overline{C}$

D AC

5 **Bir haritada sıfırlar gruplanıp terimler VEYA ile bağlanırsa ne elde edilir?**

A Fonksiyonun kendisi

B Önemsiz durumların listesi

C Fonksiyonun tersi

D Minterimlerin toplamı

6 **Zorunlu asal içeren hangisidir?**

A Haritadaki en büyük çerçeve

B Bir 1'i kapsayan tek asal içeren

C Kenardan sarılan her çerçeve

D Önemsiz durum içeren her çerçeve

7 **Üç değişkenli (A, B, C) bir fonksiyon yalnız 3, 5 ve 7 numaralı satırlarda 1. En kısa çarpımların toplamını yaz.**

Cevap:

8 **BCD girişte (A en büyük basamak) rakam 8 ya da 9 ise 1 veren devreyi önemsiz durumları (10-15) kullanarak en kısa biçimde yaz.**

Cevap:

SAYISAL TASARIM

5 Toplama ve çıkarma devreleri

Kapılar sayı bilmez, yine de bilgisayar toplar ve çıkarır. Bu bölümde iki biti toplayan en küçük devreden başlayıp çok bitli toplayıcıya, eldeyi beklemeyen toplayıcıya, aynı devreyle çıkarmaya, taşmanın yakalanmasına, iki sayıyı karşılaştırmaya ve onluk toplamaya gideceksin.

Sayı bilmeyen kapılar

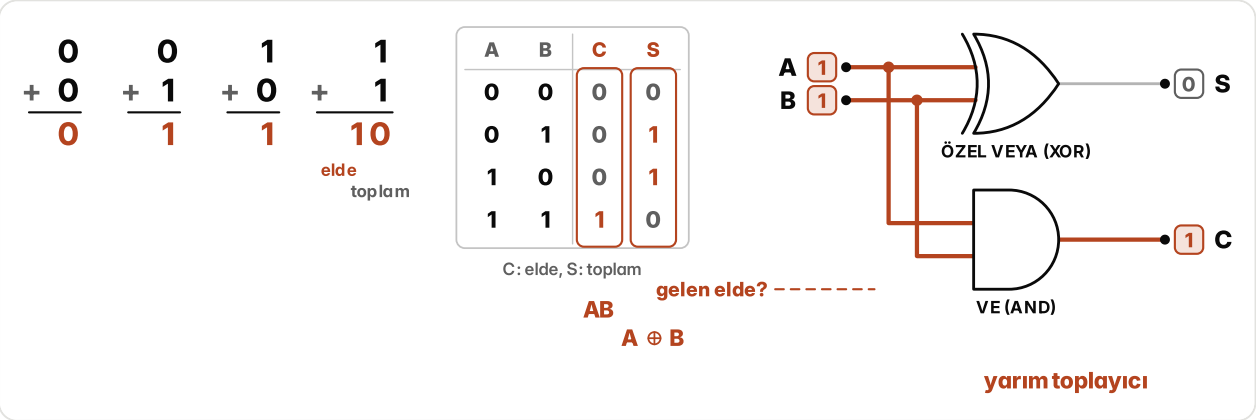
videoda 25:15

Kapılar yalnız 0 ve 1 alır, 0 ve 1 verir; toplama yapan devre de bu kapılardan kurulur. Toplayıcının çıkışı yalnız o anki girişlere bağlıdır: aynı iki sayı her zaman aynı toplamı verir, devre önceki hesabı hatırlamaz. Hafızası olmayan böyle devrelere **kombinezonsal devre** denir. Tasarımı hep aynı yoldan gider: istenen iş sözle söylenir, doğruluk tablosu yazılır, tablo Karnaugh haritasıyla sadeleştirilir, çıkan ifade devreye dönüşür.

En küçük iş iki biti toplamaktır. 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1; 1 + 1 ise 2'dir, ikilikte 10: toplam basamağına 0 yazılır, sola 1 elde gider. Tablonun iki çıkışı vardır, C (elde) ve S (toplam):

```
A  B | C  S
0  0 | 0  0
0  1 | 0  1
1  0 | 0  1
1  1 | 1  0
```

S sütunu ÖZEL VEYA (XOR) kapısının tablosudur, C sütunu VE (AND) kapısının. Bir ÖZEL VEYA ve bir VE kapısından oluşan bu devreye **yarım toplayıcı** denir. Yarımdır, çünkü sağdaki basamaktan gelen eldeyi alacak girişi yoktur.



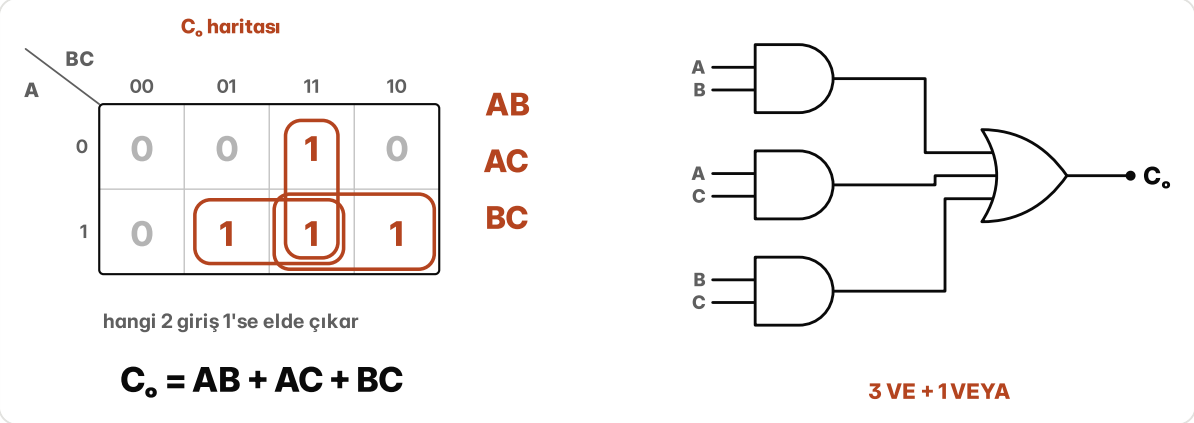
Şekil 5.1 Dört tek bitlik toplama, A B | C S tablosu ve yarım toplayıcı: A = 1, B = 1 için S = 0, C = 1; sağda gelen elde için yer yok.

Çok bitli toplamada her basamağa iki bit ve sağdan gelen elde girer: A, B ve C (gelen elde). Üç girişin 8 satırı vardır. Toplam S, girişlerdeki 1'lerin sayısı tekse 1'dir; giden elde C_o, en az iki giriş 1 ise 1'dir. Örneğin A = 1, B = 1, C = 0 satırında 1 + 1 + 0 = 2, ikilikte 10: S = 0, C_o = 1. Üç girişi

toplayan bu devreye **tam toplayıcı** denir.

```
A  B  C | S  Cₒ
0  0  0 | 0  0
0  0  1 | 1  0
0  1  0 | 1  0
0  1  1 | 0  1
1  0  0 | 1  0
1  0  1 | 0  1
1  1  0 | 0  1
1  1  1 | 1  1
```

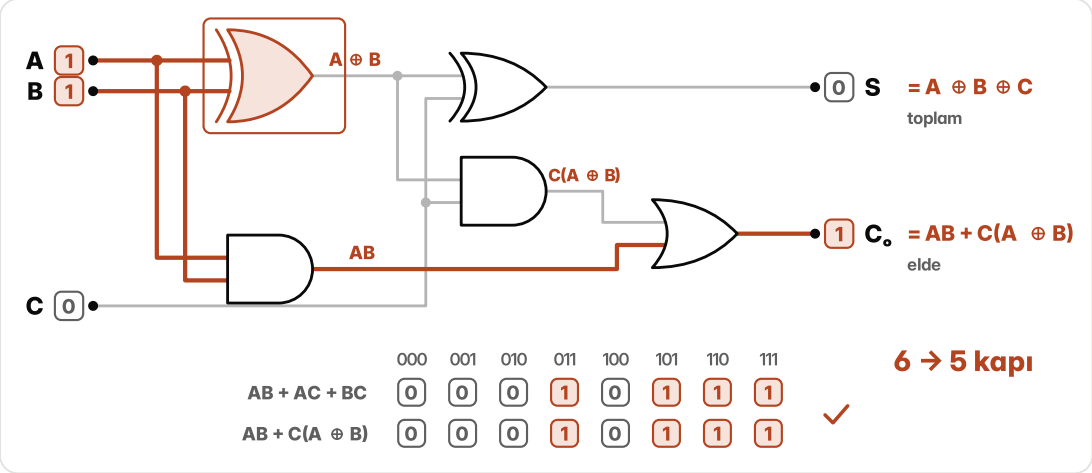
C_o sütunu haritaya dökülünce komşu 1'ler üç ikili grup oluşturur: AB, AC ve BC. Her grup eldenin bir sebebidir: hangi iki giriş 1 ise elde çıkar. Böylece C_o = AB + AC + BC olur; üç VE ve bir VEYA (OR) kapısı.



Şekil 5.2 C_o haritasında üç ikili grup (AB, AC, BC), C_o = AB + AC + BC ifadesi ve 3 VE ile 1 VEYA kapısından devresi.

S haritasında dört tane 1 vardır ama hiçbiri yan yana değildir; dama tahtası gibi dizilirler. Gruplanacak komşu olmadığı için harita kısaltamaz. Bu desen ÖZEL VEYA'nın imzasıdır: S = A $\oplus$ B $\oplus$ C. İki ÖZEL VEYA kapısı daha eklenince tam toplayıcı 6 kapıda biter.

Tasarımın tersi de vardır: verilen bir devrenin ne yaptığını bulmaya **analiz** denir. İki yarım toplayıcı ve bir VEYA kapısından kurulu bir devreyi soldan sağa izleyelim. Birinci yarım toplayıcı A $\oplus$ B ile AB'yi verir, ikincisi bu toplama gelen eldeyi ekler. Üst çıkış A $\oplus$ B $\oplus$ C, yani toplamdır. Alt çıkış AB + C(A $\oplus$ B)'dir: elde ya A ile B'nin ikisi de 1 olduğu için doğar ya da ikisinden yalnız biri 1 ve gelen elde de 1 olduğu için. 8 satırda bu ifade AB + AC + BC ile aynı sütunu verir; devre bir tam toplayıcıdır. Üstelik toplam için kurulan A $\oplus$ B elde için de kullanıldığından 6 yerine 5 kapı yeter. Çok bitli toplayıcı da buradan başlar: her basamağa bir tam toplayıcı.



Şekil 5.3 İki yarım toplayıcı ve VEYA kapısından tam toplayıcı: A = 1, B = 1, C = 0 için S = 0, C_o = 1; iki ifade 8 satırda aynı, kapı sayısı 6'dan 5'e iniyor.

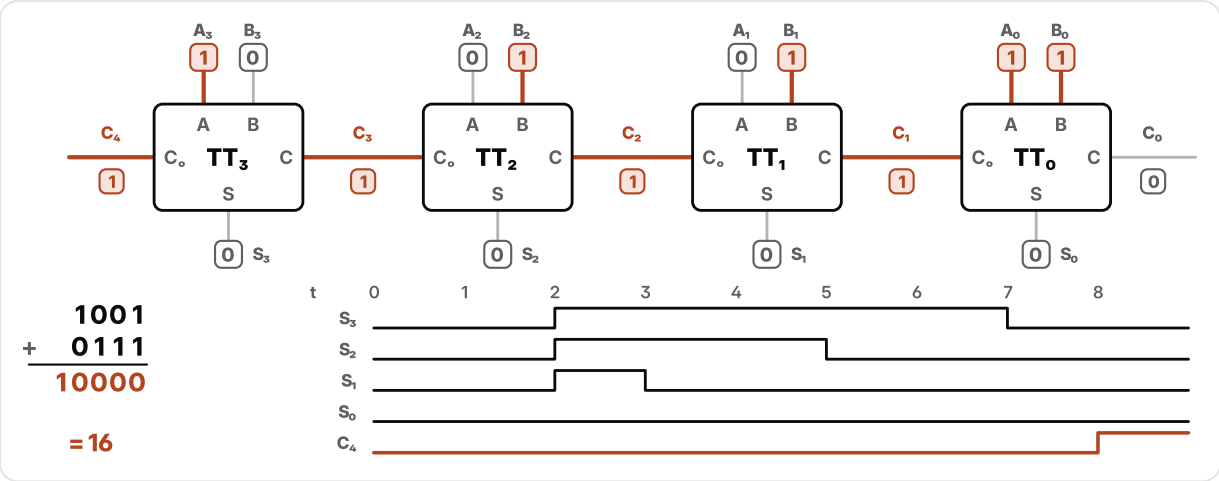
Bekleyen basamak

videoda 27:14

64 bitlik iki sayı toplanırken bütün basamaklar aynı anda çalışmaya başlar, yine de son basamak ilk basamağı beklemek zorundadır. Nedeni 4 bitlik bir toplayıcıda görülür. Dört tam toplayıcı yan yana dizilir; her basamağın giden eldesi solundakinin gelen eldesine bağlanır, en sağa gelen elde C_o = 0'dır. Elde basamaktan basamağa yürüdüğü için buna **elde yürüyen toplayıcı** denir.

Her kapı sonucunu bir an gecikmeyle verir; her kapıyı bir birim gecikme sayalım. Örnek 9 + 7: A = 1001, B = 0111. Bitler aynı anda girer. 2 birim sonra toplam çıkışlarında 1110 görünür; bu yanlıştır, çünkü eldeler henüz yoldadır. En sağ basamakta 1 + 1 = 10: elde doğar ve sola geçer. Orada 0 + 1 + 1 yine 2 eder ve elde bir sonrakine geçer. Elde ulaştığı her basamakta bir 1'i 0'a çevirir ve sonunda dışarı çıkar: 10000, yani 16. Benzetimde değerler şöyle değişir (t: kapı gecikmesi):

```
t | S3 S2 S1 S0 | C1 C2 C3 C4
0 |  0  0  0  0 |  0  0  0  0
1 |  0  0  0  0 |  0  0  0  0
2 |  1  1  1  0 |  1  0  0  0
3 |  1  1  0  0 |  1  0  0  0
4 |  1  1  0  0 |  1  1  0  0
5 |  1  0  0  0 |  1  1  0  0
6 |  1  0  0  0 |  1  1  1  0
7 |  0  0  0  0 |  1  1  1  0
8 |  0  0  0  0 |  1  1  1  1
```



Şekil 5.4 Dört tam toplayıcılı zincirde 1001 + 0111: elde sağdan sola yürüyor, zaman çizelgesinde toplam bitleri 2. gecikmeden sonra teker teker 0'a dönüyor ve C_4 8. gecikmede 1 oluyor.

Elde her basamakta bir VE ve bir VEYA kapısından geçer: 2 kapı gecikmesi. 4 bitte 8, 64 bitte 128 kapı gecikmesi eder. Toplam ancak elde son basamağa varınca doğru olur ve işlemci en yavaş yolu beklemek zorundadır.

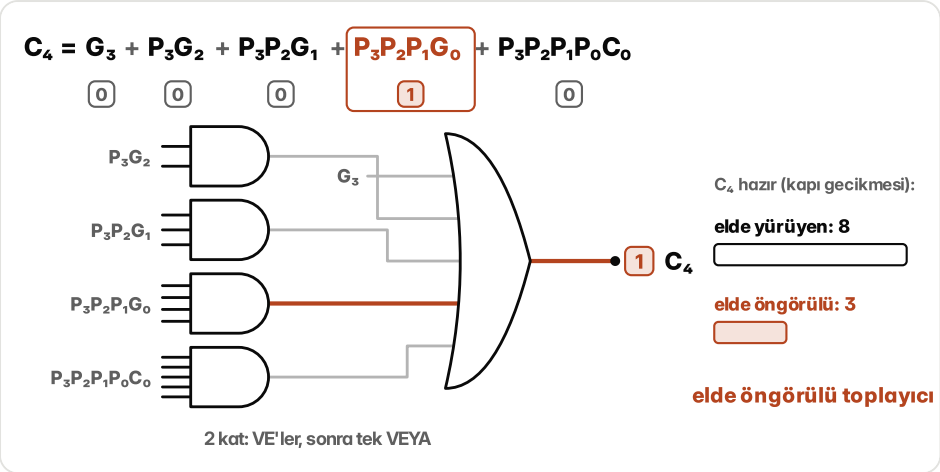
Eldeyi beklemeden bilmenin yolu her basamağa iki soru sormaktır. A_i ve B_i ikisi de 1 ise basamak, gelen elde ne olursa olsun elde **üretir**: $G_i = A_iB_i$. Yalnız biri 1 ise eldeyi **geçirir**: elde gelirse elde çıkar, $P_i = A_i \oplus B_i$. İkisi de 0 ise gelen eldeyi yutar. G_i ve P_i girişler gelir gelmez, tek kapı gecikmesinde hazırdır. 9 + 7 için:

```
bit | 3  2  1  0
A   | 1  0  0  1
B   | 0  1  1  1
G   | 0  0  0  1
P   | 1  1  1  0
```

En soldan çıkan eldenin beş yolu vardır: ya en sol basamak üretti, ya sağındaki üretti ve en sol geçirdi, ya daha sağdakiler üretti ve üstlerindekilerin hepsi geçirdi, ya da en sağa elde geldi ve dört basamak da geçirdi:

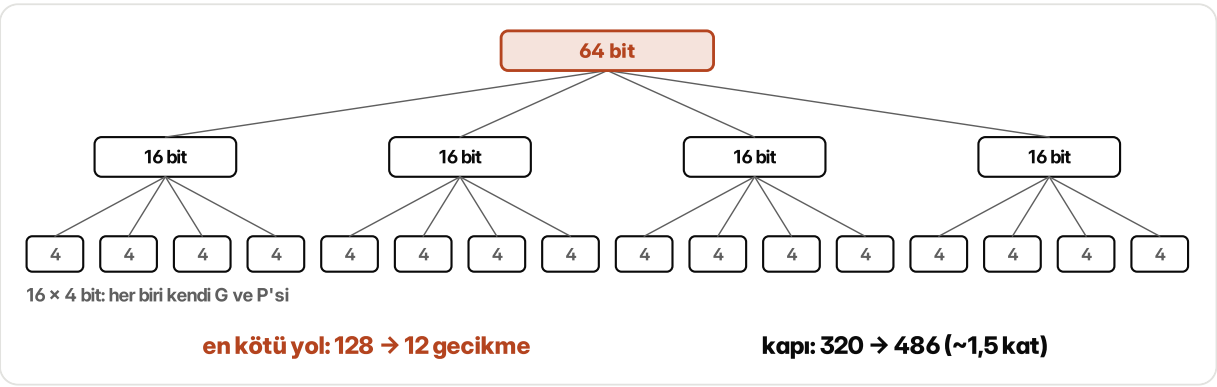
$$C_4 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_0$$

Her yol bir VE terimidir ve beşi tek bir VEYA kapısına girer. 9 + 7'de en sağ basamak üretiyor, üstündeki üç basamak geçiriyor: $P_3P_2P_1G_0 = 1$, dolayısıyla $C_4 = 1$. Bütün eldeler 8 yerine 3 kapı gecikmesinde hazırdır. Buna **elde öngörülü toplayıcı** (carry lookahead) denir.



Şekil 5.5 C_4'ün beş terimi 9 + 7 için 0, 0, 0, 1, 0; iki katlı devre (VE kapıları ve tek VEYA) C_4 = 1 veriyor; C_4 elde yürüyende 8., öngörülüde 3. gecikmede hazır.

64 bitte bu formül çok uzar: son terim 65 girişli bir VE kapısı ister. Bu yüzden bitler dörder dörder gruplanır; her grup kendi G ve P bitini verir, gruplar da aynı yoldan 16 bitlik, sonra 64 bitlik üst gruplara bağlanır. Her kapıyı bir gecikme sayınca en kötü durum 128 gecikmeden 12'ye iner; bedeli yaklaşık 1,5 kat kapıdır (bu tasarımda 320 yerine 486). Bugünkü işlemcilerin toplayıcıları da aynı fikre dayanır: biraz daha çok kapı harcayıp eldeyi beklememek.



Şekil 5.6 16 tane 4 bitlik blok, 4 tane 16 bitlik grup ve tek 64 bitlik grup; en kötü yol 128'den 12 gecikmeye, kapı sayısı 320'den 486'ya.

Bir tel

videoda 29:11

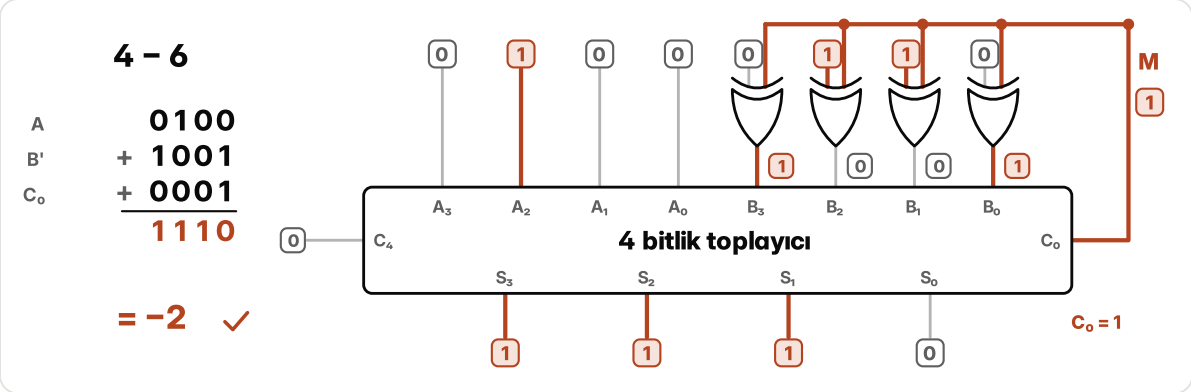
4 bitlik bir toplayıcıya tek bir tel ve 4 kapı ekleyerek çıkarma da yaptırılabilir. Anahtar ÖZEL VEYA kapısının bir özelliğidir: bir girişine 0 verilirse öteki giriş olduğu gibi geçer, 1 verilirse ters çevrilir.

```
M  B | M ⊕ B
0  0 |   0
0  1 |   1
1  0 |   1
1  1 |   0
```

B'nin her biti bir ÖZEL VEYA'dan geçer ve dört kapının öteki girişleri tek bir tele bağlanır: **işlem teli** M. M = 0 ise toplama, M = 1 ise çıkarma yapılır. −B ikinin tümleyeniyle yazılır: bitler ters çevrilir, 1 eklenir. Ters çevirmeyi ÖZEL VEYA'lar yapar; 1 eklemeyi en sağdaki gelen elde C_0 yapar, M oraya

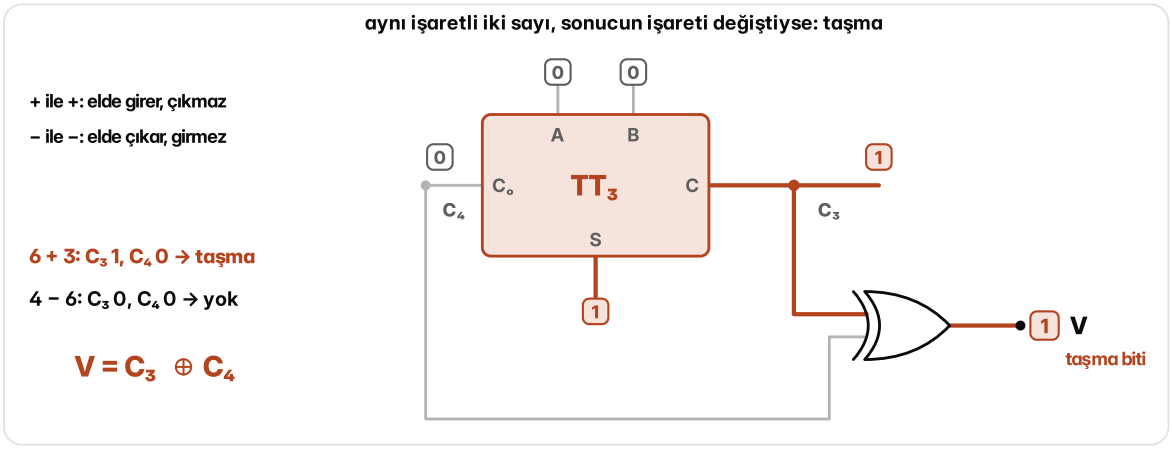
da bağlanır. M = 0 iken devre A + B'yi, M = 1 iken A + $\overline{B}$ + 1'i, yani A − B'yi hesaplar. Örnek 4 − 6:

```
  0100     A = 4
+ 1001     B = 0110'ın ters çevrilmişi
+    1     C₀ = M = 1
  1110     işaretli okununca −2
```



Şekil 5.7 Toplayıcı-çıkarıcıda 4 − 6: M = 1, ÖZEL VEYA çıkışları 1001, C_0 = 1, sonuç 1110 = −2.

4 bitle işaretli sayılar −8'den +7'ye kadardır. 6 + 3 = 0110 + 0011 = 1001 eder; işaretli okununca −7. İki pozitif sayının toplamı negatif çıktı, çünkü 9 dört bite sığmaz. Buna **taşma** denir. Kural şudur: aynı işaretli iki sayının toplamında sonucun işareti değiştiyse taşma vardır. Devrede daha kısa bir yol var: en sol basamağa giren elde C_3 ile çıkan elde C_4 karşılaştırılır. İki pozitif sayı taşınca en sol basamağa elde girer ama çıkmaz; iki negatif taşınca elde çıkar ama girmez. 6 + 3'te C_3 = 1, C_4 = 0; 4 − 6'da ikisi de 0. Giren ile çıkan farklıysa taşma vardır: $V = C_3 \oplus C_4$.



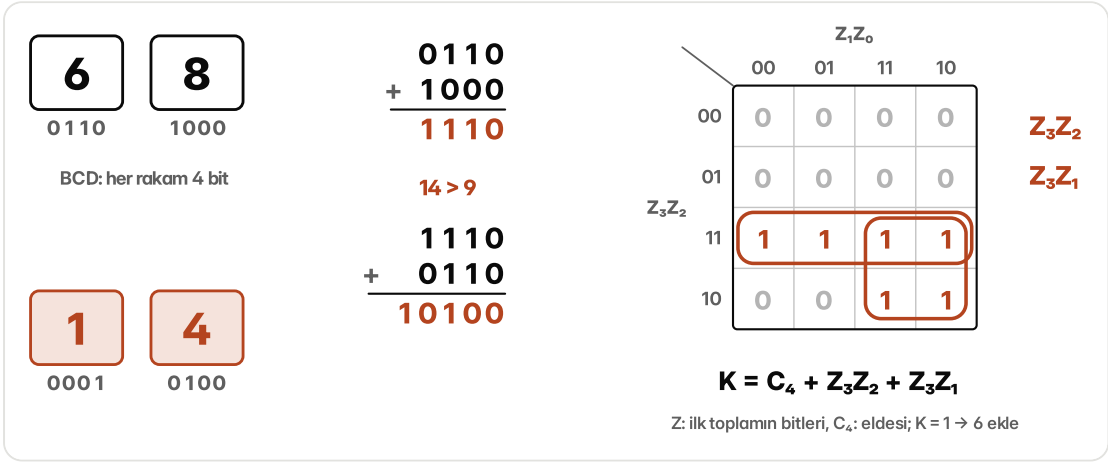
Şekil 5.8 En sol tam toplayıcıda 6 + 3: C_3 = 1 giriyor, C_4 = 0 çıkıyor, ÖZEL VEYA kapısı taşma biti V = 1 veriyor; 4 − 6'da V = 0.

Toplayıcının yakın bir komşusu **karşılaştırıcıdır**. Eşitlik için her bit çifti bir ÖZEL VEYA DEĞİL (XNOR) kapısına girer; bitler aynıysa 1 verir. Dört çıkışın VE'si 1 ise sayılar eşittir. Büyüklük için soldan bakılır: ilk farklı bitte A 1, B 0 ise A büyüktür. 10 ile 9, yani 1010 ve 1001: ilk iki bit aynı, üçüncüde A 1, B 0; 10 büyüktür. Formülü, x_i i. bit çiftinin ÖZEL VEYA DEĞİL çıkışı olmak üzere, $A > B = A_3\overline{B}_3 + x_3A_2\overline{B}_2 + x_3x_2A_1\overline{B}_1 + x_3x_2x_1A_0\overline{B}_0$'dır.

Hesap makinesi gibi bazı cihazlar sayıyı rakam rakam tutar: BCD kodunda her onluk rakam 4 bittir. 4 bit 16 değer tutar ama rakam 9'da biter; toplam 9'u geçerse 6 eklenir ve kullanılmayan 6 kod (1010 ... 1111) atlanır. Örnek 6 + 8:

```
  0110      6
+ 1000      8
  1110      14, 9'u geçti
+ 0110      6 ekle
1 0100      elde 1, rakam 0100: 14
```

"9'u geçti mi?" sorusunu haritadan çıkan küçük bir devre cevaplar: ilk toplamın bitleri $Z_3Z_2Z_1Z_0$ ve eldesi C_4 ise $K = C_4 + Z_3Z_2 + Z_3Z_1$; K = 1 olunca 6 eklenir. Toplama, çıkarma, taşma ve onluk toplama aynı toplayıcının çevresine birkaç kapı eklenerek kuruldu. İşlemcideki çıkarma da böyle yapılır: ayrı bir devre değil, aynı toplayıcı ve bir tel.



Şekil 5.9 BCD toplayıcıda 6 + 8: ilk toplam 1110, 6 eklenince elde 1 ve rakam 0100; haritada Z_3Z_2 ve Z_3Z_1 grupları, $K = C_4 + Z_3Z_2 + Z_3Z_1$.

Özet

- Kombinezonsal devrenin çıkışı yalnız o anki girişlere bağlıdır; tasarım istenen işten doğruluk tablosuna, haritaya, ifadeye ve devreye gider.
- Yarım toplayıcı iki biti toplar ($S = A \oplus B$, $C = AB$); tam toplayıcı gelen eldeyi de alır ($S = A \oplus B \oplus C$, $C_o = AB + AC + BC = AB + C(A \oplus B)$) ve 5 kapıyla kurulur.
- Elde yürüyen toplayıcıda elde her basamakta 2 kapı gecikmesi ekler; 64 bitte 128 gecikme eder.
- Elde öngörülü toplayıcı her basamağın üretir ($G = AB$) ve geçirir ($P = A \oplus B$) bitlerinden eldeleri önceden hesaplar; daha çok kapıyla gecikmeyi kısaltır.
- B'nin bitleri işlem teli M ile ÖZEL VEYA'dan geçirilip $C_o = M$ yapılınca aynı toplayıcı A − B'yi hesaplar; işaretli toplamada taşma biti $V = C_3 \oplus C_4$'tür.
- Karşılaştırıcı eşitliği ÖZEL VEYA DEĞİL ve VE ile, büyüklüğü ilk farklı bitle bulur; BCD toplayıcı 9'u geçen toplama 6 ekler.

Sorular

1 **Kombinezonsal devre için hangisi doğrudur?**

- **A** Çıkışı önceki hesapları da hatırlar.
- **B** Yalnız VE ve VEYA kapılarıyla kurulabilir.
- **C** Çıkışı yalnız o anki girişlere bağlıdır; aynı girişe her zaman aynı çıkışı verir.
- **D** Aynı girişe zamana göre farklı çıkışlar verebilir.

2 **Yarım toplayıcıya neden "yarım" denir?**

- **A** Yalnız bir çıkışı olduğu için.
- **B** Sağdaki basamaktan gelen eldeyi alacak girişi olmadığı için.
- **C** Kapılarından biri VE, öteki ÖZEL VEYA olduğu için.
- **D** 1 + 1 işlemini yanlış topladığı için.

3 **Tam toplayıcıda A = 1, B = 0, C = 1 (C gelen elde) iken toplam S ve giden elde C_o nedir?**

- **A** S = 1, C_o = 0
- **B** S = 0, C_o = 0
- **C** S = 1, C_o = 1
- **D** S = 0, C_o = 1

4 **Elde öngörülü toplayıcıda bir basamakta A_i = 0 ve B_i = 1 ise bu basamak için hangisi doğrudur?**

- **A** Eldeyi geçirir: sağdan elde gelirse bu basamaktan da elde çıkar.
- **B** Elde üretir, çünkü B_i = 1.
- **C** Eldeyi yutar: bu basamaktan hiç elde çıkmaz.
- **D** Gelen elde ne olursa olsun elde çıkar.

5 **Elde yürüyen toplayıcıda elde her basamakta bir VE ve bir VEYA kapısından, yani 2 kapı gecikmesinden geçiyor. 16 bitlik bir elde yürüyen toplayıcıda en sağa giren elde en soldan kaç kapı gecikmesi sonra çıkar?**

Cevap:

6 **4 bitlik toplayıcı-çıkarıcıda işlem teli M = 1, A = 0101 ve B = 0010. Toplayıcının 4 bitlik çıkışı nedir?**

Cevap:

7 **4 bitlik işaretli sayılarla yapılan toplamalardan hangisinde taşma olur?**

- **A** 0011 + 0010
- **B** 1110 + 0101
- **C** 0101 + 0100
- **D** 1100 + 0010

8 **BCD toplayıcıda 8 ile 7 toplanıyor. Onluk elde 1 olur; sonucun rakamını tutan 4 bit nedir?**

Cevap: _____

SAYISAL TASARIM

6 Seçiciler, kod çözücüler, kodlayıcılar

Bu bölümde bir adresi tek bir hatta açan kod çözücüyü, birçok hattan biri bir olduğunda onu kısa bir koda çeviren kodlayıcıyı, dört bitlik rakamı yedi parçalı göstergeye çeviren sürücüyü ve birçok girişten birini çıkışa geçiren seçiciyi göreceksin. Seçici ve kod çözücünün yalnız seçmekle kalmayıp her fonksiyonu kurabildiğini de göreceksin.

Tek hücre

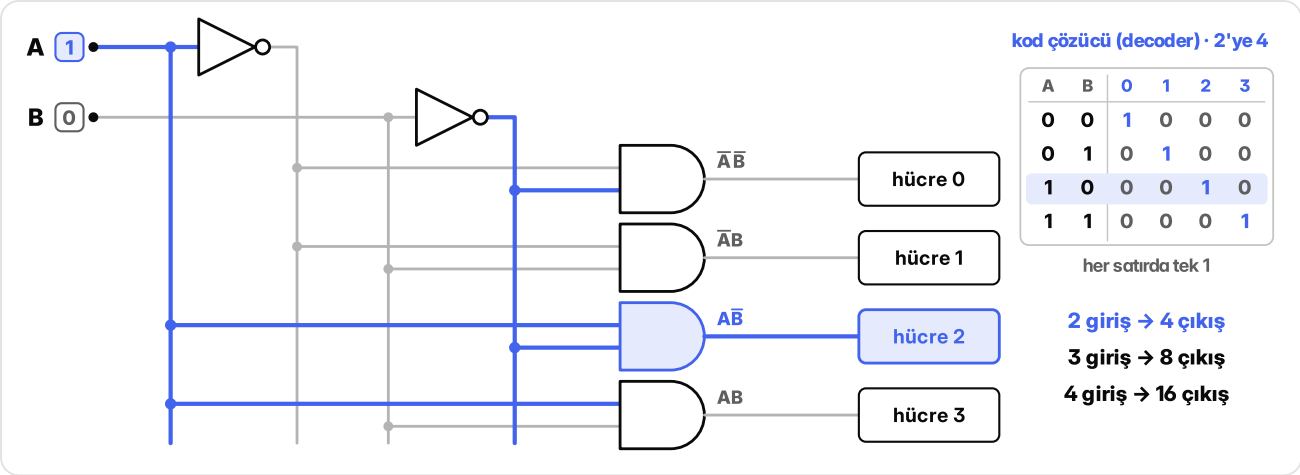
videoda 31:22

Bellekte milyarlarca hücre vardır ve işlemci her okumada yalnız birine ulaşmak ister. Elindeki tek bilgi adrestir: birkaç tel üstünde 0'lar ve 1'ler. Bu tellerden o tek hücreye nasıl gidildiğini görmek için küçük başlayalım: 4 hücreli bir bellek ve 2 adres teli, A ile B. Her hücrenin bir seçme teli vardır; bu tel yalnız kendi adresi gelince 1 olmalıdır. 3 numaralı hücrenin adresi 11'dir; A ile B'yi alan tek bir VE kapısı yeter. 2 numaralı hücrenin adresi 10'dur. VE kapısı yalnız bütün girişleri 1 olunca 1 verdiği için 0 olan B'nin tersi alınır: $A \cdot \overline{B}$. Dört hücrenin her biri böyle kendi VE kapısını alır; $\overline{B}$ ve $\overline{A}$ için iki DEĞİL kapısı yeter, toplam 6 kapı.

Bu devreye **kod çözücü** (decoder) denir: 2 giriş, 4 çıkış ve her an çıkışlardan yalnız biri 1:

```
A  B | Y0  Y1  Y2  Y3
0  0 |  1   0   0   0
0  1 |  0   1   0   0
1  0 |  0   0   1   0
1  1 |  0   0   0   1
```

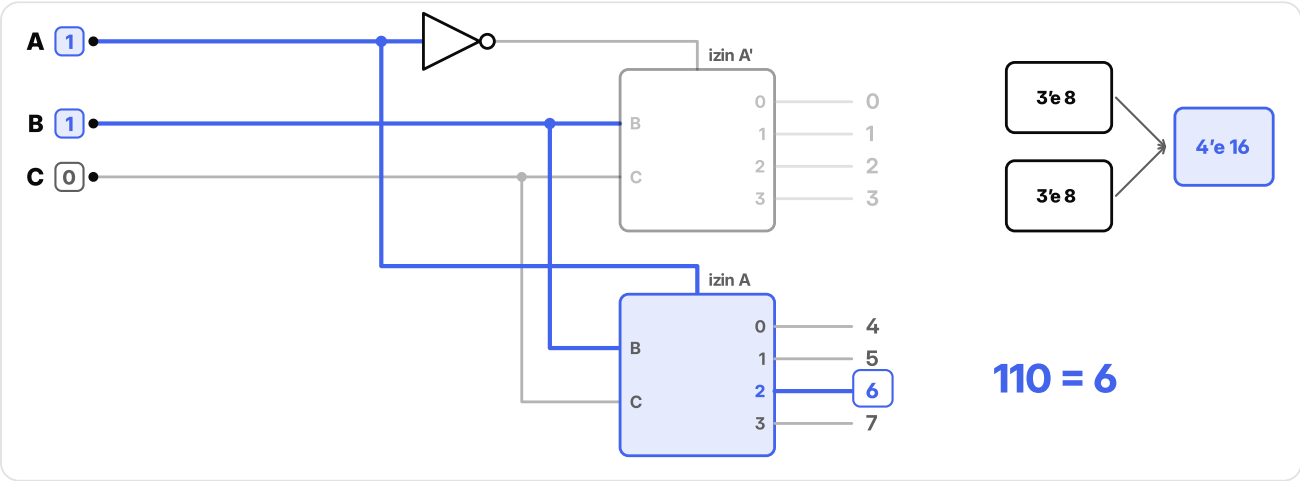
Adres 10 gelince yalnız $A \cdot \overline{B}$ kapısının iki girişi de 1 olur; 2 numaralı hat 1, öteki üçü 0 kalır. Her çıkış yalnız kendi satırında 1 veren bir VE terimidir, yani bir minterimdir: $Y0 = \overline{A}\,\overline{B}$, $Y1 = \overline{A}B$, $Y2 = A\overline{B}$, $Y3 = AB$. Her yeni girişle çıkış sayısı ikiye katlanır: n girişli çözücünün 2^n çıkışı vardır; 3 girişte 8, 4 girişte 16.



Şekil 6.1 2'ye 4 kod çözücü, adres 10: yalnız $A\overline{B}$ kapısı ve 2 numaralı hücre 1; yanda her satırında tek 1 olan tablo.

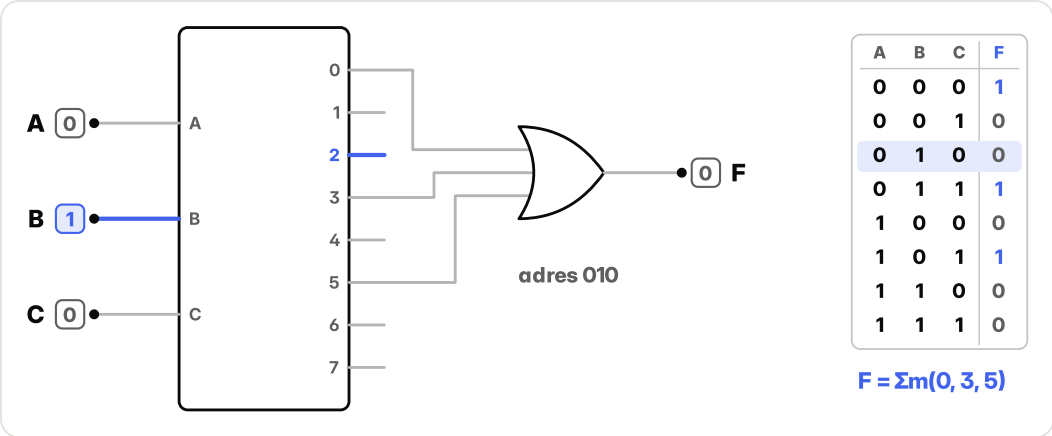
Bazen hiçbir hücre seçilmemelidir; örneğin işlemci o an belleğe hiç dokunmuyorsa. Bunun için her VE kapısına bir giriş daha eklenir: **izin** girişi (E). İzin 0 iken adres ne olursa olsun bütün çıkışlar 0'dır; izin 1 iken çözücü her zamanki gibi çalışır.

İzin girişi büyük çözücüyü küçüklerden kurmayı sağlar. 8 hücre için 3 adres teli gerekir: A, B, C. İki 2'ye 4 çözücü alınır; B ile C ikisine birden girer, A ise izinlere: üstteki çözücünün izni $\overline{A}$, alttakinin izni A. Adres 110 gelince A = 1 olduğu için yalnız alttaki çözücü çalışır; BC = 10 olduğu için onun 2 numaralı hattı 1 olur. Alttaki çözücünün hatları 4-7 olduğundan bu, sekiz hattın 6 numaralısıdır; adres de 6'dır. Aynı yolla iki 3'e 8 çözücü bir 4'e 16 çözücü yapar.



Şekil 6.2 İki 2'ye 4 çözücüden 3'e 8: A izinlere, B ve C ikisine birden; adres 110 alttaki çözücünün 2 numaralı hattını, yani 6. hattı seçer.

Çözücünün her çıkışı bir minterim olduğuna göre, ona bir VEYA kapısı eklemek her fonksiyonu kurar: fonksiyonun 1 olduğu satırların hatları VEYA kapısında toplanır. 3'e 8 çözücünün 0, 3 ve 5 numaralı hatları bir VEYA kapısına bağlanırsa çıkış tam bu üç adreste 1 olur: $F = \Sigma m(0, 3, 5) = \overline{A}\overline{B}\overline{C} + \overline{A}BC + A\overline{B}C$.



Şekil 6.3 3'e 8 çözücü ve VEYA: 0, 3 ve 5 numaralı hatlar VEYA'ya bağlı; yanda F = Σm(0, 3, 5) tablosu.

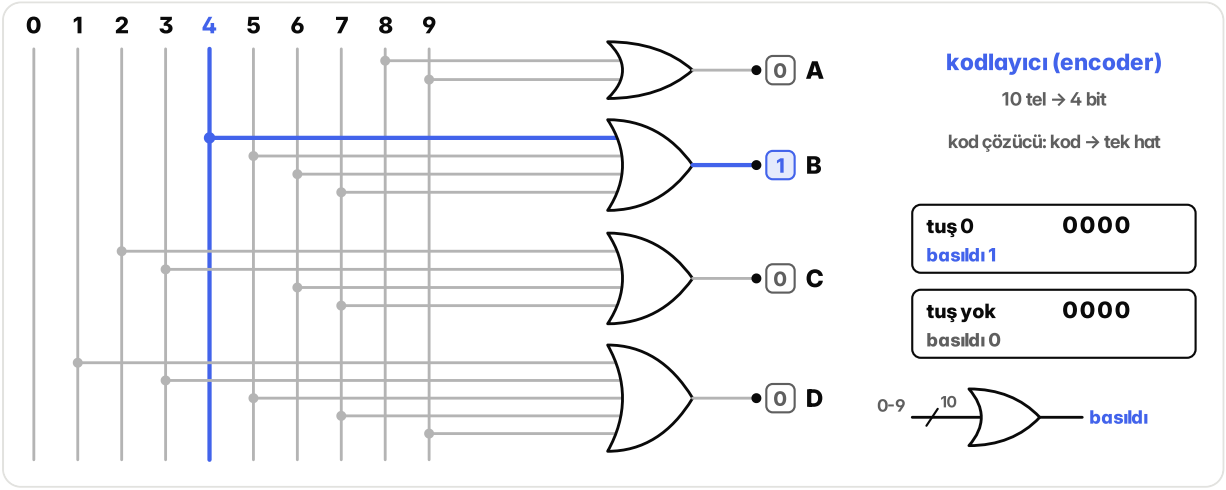
Her adres teli seçilebilen hücre sayısını ikiye katlar: 10 tel 2^{10} = 1.024 hücre seçer, 30 tel 2^{30} = 1.073.741.824, yani bir milyarı aşkın hücre. Bellekteki çözücüler de böyle küçük çözücülerden kurulur. İşlemci hücreleri tek tek aramaz: adresi tellere koyar, çözücüler milyarlarca hücre arasından yalnız birine giden yolu açar.

Tuştan ışığa

videoda 33:20

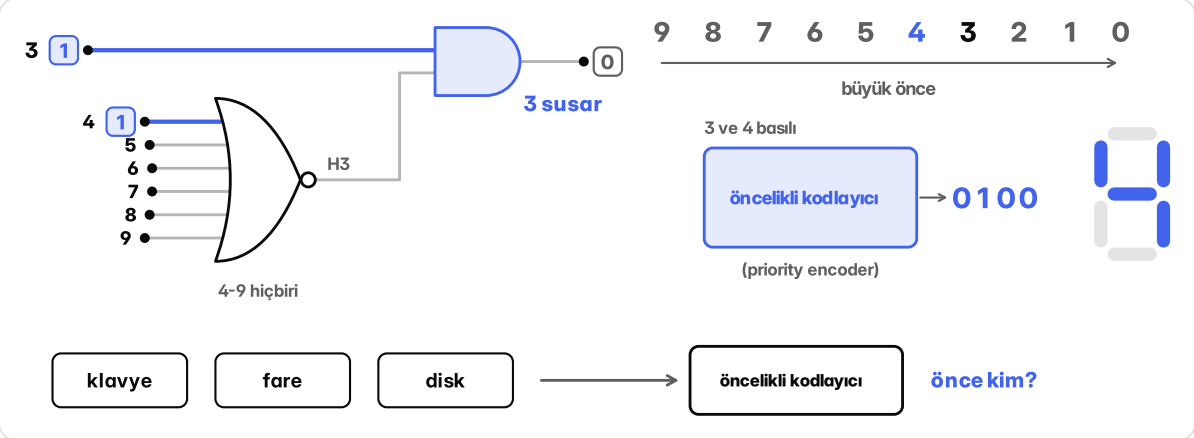
Mikrodalga fırının tuş takımında 4'e basınca ekranda 4 belirir. Arada iki çeviri vardır: basılan tuştan 4 bitlik koda, koddan da rakamı yazan 7 küçük çubuğa.

Her tuşun kendi teli olsun: basılan tuşun teli 1, öteki 9'u 0. Bu 10 telden rakamın 4 bitlik ikilik kodu çıkarılır; bitler soldan sağa A, B, C, D (A en büyük basamak, 8). D biti tek rakamlarda 1'dir: 1, 3, 5, 7, 9; öyleyse bu beş tuşun telleri bir VEYA kapısına girer. C biti 2, 3, 6 ve 7'de, B biti 4-7'de, A biti 8 ve 9'da 1'dir. Dört VEYA kapısından oluşan bu devreye **kodlayıcı** (encoder) denir. Kod çözücü kodu tek bir hatta açar; kodlayıcı tersini yapar: birçok hattan biri 1 olduğunda onu kısa bir koda çevirir. 0 tuşu ise hiç basılmamakla aynı kodu verir, 0000. Bu yüzden bir çıkış daha eklenir: 10 telin VEYA'sı, yani "bir tuşa basıldı" bilgisi.



Şekil 6.4 Kodlayıcı: 10 tuş teli, dört VEYA kapısı; 4 tuşu basılıyken yalnız B çıkışı 1, kod 0100.

3'e ve 4'e aynı anda basılırsa ne olur? 3'ün kodu 0011, 4'ün kodu 0100'dür. VEYA kapıları ikisinin 1'lerini birleştirir: 0111, yani 7. Ekranda kimsenin basmadığı bir rakam belirir. Çözüm, tuşlara sıra vermektir: büyük rakam önce gelir. Her tuş, kendisinden büyük tuşların hiçbiri basılı değilse sayılır. 3'ün teli ancak 4-9 tuşlarının hepsi 0 iken geçer; bu koşulu 4-9 tuşlarını alan bir VEYA DEĞİL kapısı verir, 3'ün teli onunla bir VE kapısına girer. 4 basılı olduğu için 3 susar ve kod 0100, yani 4 olur. Buna **öncelikli kodlayıcı** (priority encoder) denir. Bilgisayarda da aynı anda dikkat isteyen aygıtlar olur (klavye, fare, disk); hangisine önce bakılacağına bu fikirle çalışan bir devre karar verir.

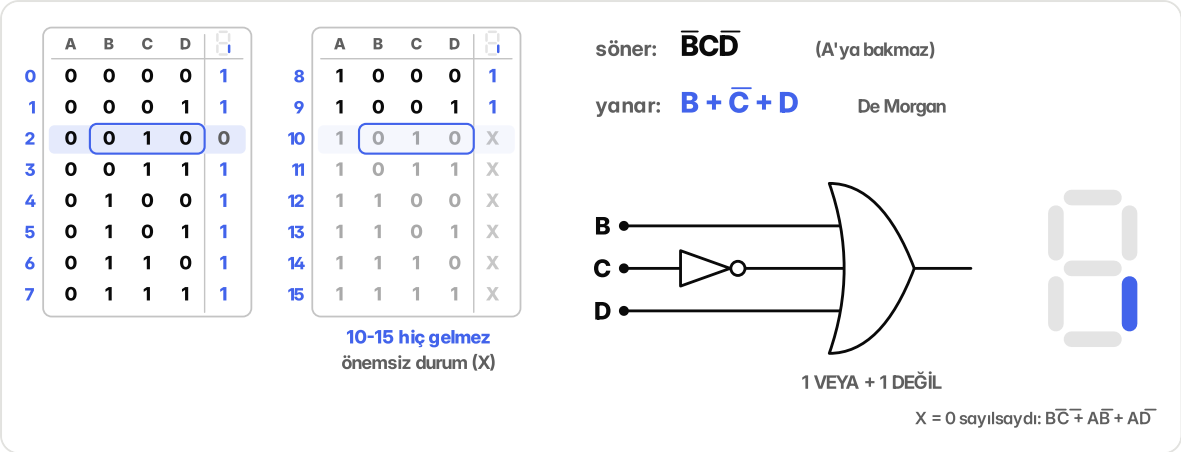


Şekil 6.5 Öncelikli kodlayıcı: 3 ve 4 basılı; 4-9 tuşlarının VEYA DEĞİL'i 0 olduğu için 3'ün VE kapısı 0 verir, kod 0100.

Her onluk rakamın 4 bitle yazıldığı bu koda **BCD** denir. Yedi parçalı göstergenin çubukları a (üst), b (sağ üst), c (sağ alt), d (alt), e (sol alt), f (sol üst), g (orta) diye adlandırılır; her çubuk ayrı bir çıkıştır:

```
rakam  A B C D | a b c d e f g
  0    0 0 0 0 | 1 1 1 1 1 1 0
  1    0 0 0 1 | 0 1 1 0 0 0 0
  2    0 0 1 0 | 1 1 0 1 1 0 1
  3    0 0 1 1 | 1 1 1 1 0 0 1
  4    0 1 0 0 | 0 1 1 0 0 1 1
  5    0 1 0 1 | 1 0 1 1 0 1 1
  6    0 1 1 0 | 1 0 1 1 1 1 1
  7    0 1 1 1 | 1 1 1 0 0 0 0
  8    1 0 0 0 | 1 1 1 1 1 1 1
  9    1 0 0 1 | 1 1 1 1 0 1 1
```

Sağ alt çubuk (c) dokuz rakamda yanar, yalnız 2'de söner. 2'nin kodu 0010'dur: B = 0, C = 1, D = 0. Aynı üç bit yalnız 1010'da, yani 10'da da vardır. Ama rakam 9'da biter; 10-15 arası altı kod bu girişe hiç gelmez. Gelmeyen girişte çıkışın ne olduğu fark etmez; buna **önemsiz durum** (X) denir. Öyleyse A'ya bakmaya gerek yoktur: $\overline{B}C\overline{D}$ ise çubuk söner. De Morgan kuralıyla tersi alınınca yanma koşulu çıkar: $c = B + \overline{C} + D$. Tek bir VEYA ve bir DEĞİL kapısı yeter. Önemsiz durumlar 0 sayılsaydı formül $\overline{BC} + \overline{A}B + \overline{A}D$ olurdu.



Şekil 6.6 Gelmeyen kodlar: 10-15 satırları önemsiz (X); 2 ile 10'un B, C, D bitleri aynı; sağ alt çubuk = B + $\overline{C}$ + D devresi.

Öteki altı çubuk da önemsiz durumlardan yararlanan kendi formülünü alır; yedisi birlikte **gösterge sürücüsünü** oluşturur:

```
a = A + C + BD + B̄D̄
b = B̄ + C̄D̄ + CD
c = B + C̄ + D
d = A + B̄D̄ + CD̄ + BC̄D + B̄C
e = B̄D̄ + CD̄
f = A + C̄D̄ + BC̄ + BD̄
g = A + BC̄ + B̄C + CD̄
```

4 tuşunda kod 0100'dür; B = 1 olduğu için sağ alt çubuk yanar. Fırındaki süre de asansördeki kat numarası da böyle yedi formülden geçerek ekrana çıkar.

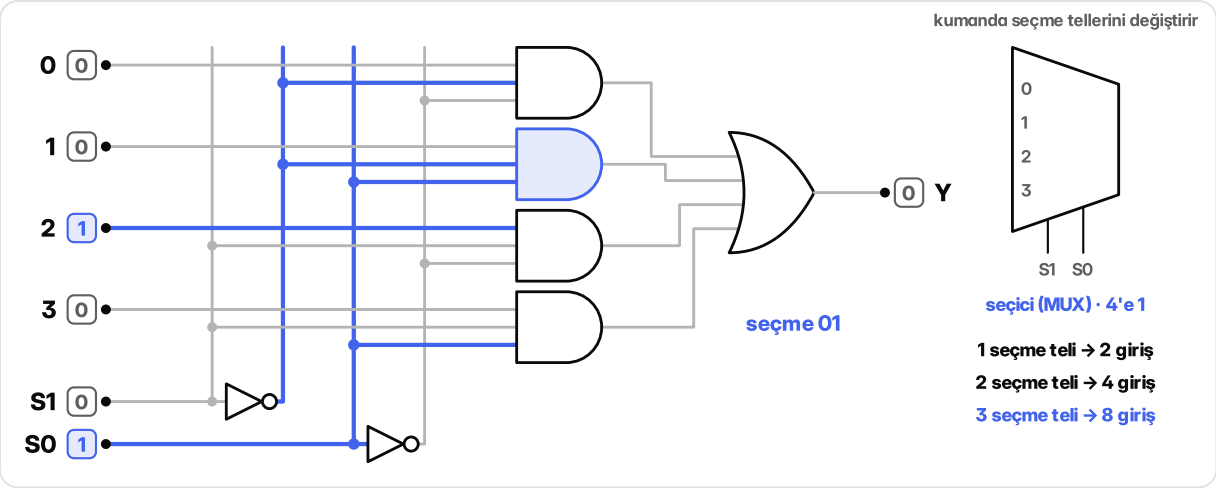
Dört giriş, bir ekran

videoda 35:25

Televizyonun arkasında 4 giriş vardır ama ekran birdir. Kumandadan kaynak değiştirilince 4 kablodan yalnız birinin görüntüsü ekrana geçer. Bu seçimi kapılarla yapan devreye **seçici** (multiplexer, MUX) denir.

Önce iki veri teli (0 ve 1 numaralı) ve bir seçme teli S alalım. S = 0 iken 0 numaralı tel, S = 1 iken 1 numaralı tel çıkışa geçmelidir. İki VE kapısı ve bir VEYA yeter: 0 numaralı tel $\overline{S}$ ile, 1 numaralı tel S ile VE kapısına girer; iki kapının çıkışı VEYA'da buluşur. S hangi kapıyı açarsa yalnız onun teli geçer, öteki kapı 0 verir. Veri tellerine G0 ve G1 dersek Y = $\overline{S}$·G0 + S·G1; DEĞİL ile birlikte 4 kapı.

4 giriş için 2 seçme teli gerekir (S1 S0): 00, 0 numaralı girişi; 11, 3 numaralıyı seçer. Her veri teli kendi üç girişli VE kapısına girer; kapının öteki iki girişi, seçme telleri o numarayı gösterdiği an birlikte 1 olur. Dört kapı bir VEYA'da buluşur: 2 DEĞİL, 4 VE ve 1 VEYA, toplam 7 kapı. Örneğin yalnız 2 numaralı girişte 1 varsa seçme 10 iken çıkış 1, seçme 01 iken 0'dır. Her yeni seçme teli giriş sayısını ikiye katlar: n seçme teliyle 2^n'den 1'e, 3 telle 8'den 1'e. Kumanda da yalnız seçme tellerini değiştirir.

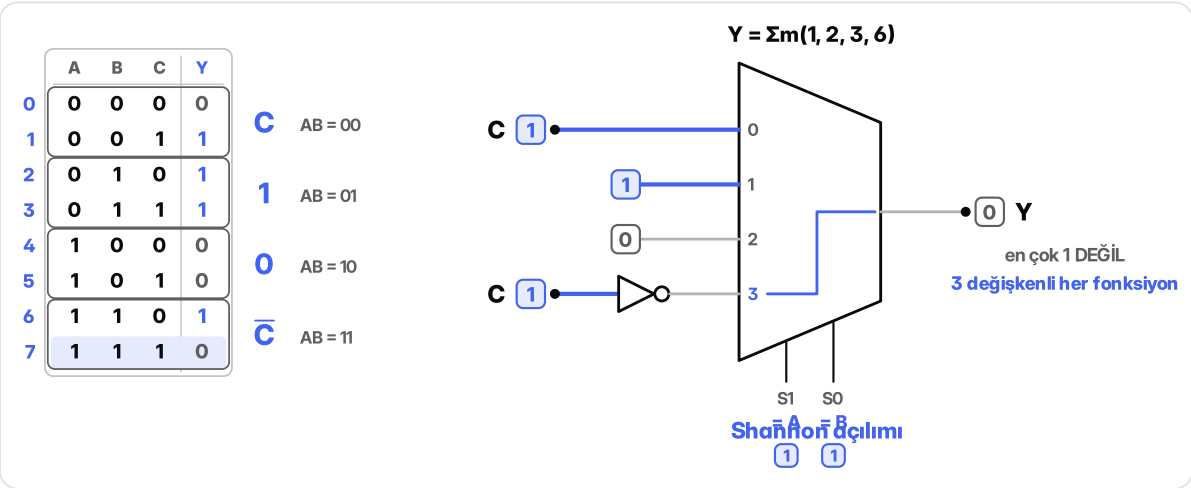


Şekil 6.7 Kapı düzeyinde 4'e 1 seçici: veri 0010, seçme 01; yanda seçici simgesi ve seçme teli sayısıyla giriş sayısı.

Seçici yalnız seçmez, hesap da yapar. Üç girişli bir fonksiyon alalım: çıkış 1, 2, 3 ve 6. satırlarda 1, yani Y = Σm(1, 2, 3, 6). A ile B seçme uçlarına bağlanır. Tablo ikişer satırlık dört parçaya ayrılır; her parçada A ile B sabittir, yalnız C değişir:

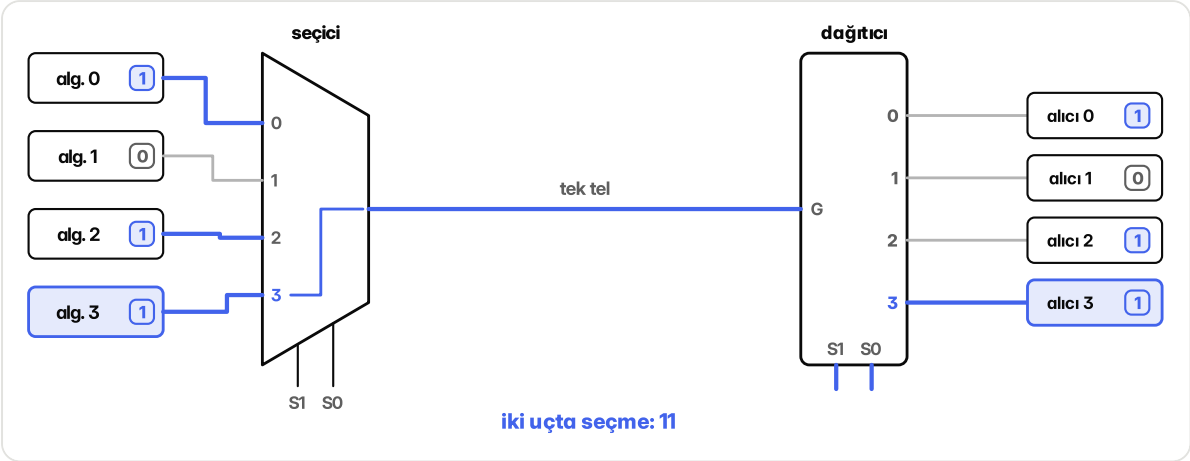
```
A  B  C | Y
0  0  0 | 0    AB = 00: Y = C
0  0  1 | 1
0  1  0 | 1    AB = 01: Y = 1
0  1  1 | 1
1  0  0 | 0    AB = 10: Y = 0
1  0  1 | 0
1  1  0 | 1    AB = 11: Y = C̄
1  1  1 | 0
```

Seçicinin dört girişine sırayla C, 1, 0 ve $\overline{C}$ bağlanır; çıkış sekiz satırın hepsinde tabloyla aynıdır. Her parça bu dört davranıştan birini taşımak zorunda olduğu için 4 girişli tek bir seçici, en çok bir DEĞİL kapısıyla, 3 değişkenli her fonksiyonu kurar (256 fonksiyonun hepsi). Fonksiyonu değişkenlerin değerlerine göre parçalara ayırmanın adı **Shannon açılımı**dır.



Şekil 6.8 Seçiciyle Σm(1, 2, 3, 6): ikişer satırlık parçalar C, 1, 0, $\overline{C}$; A ve B seçme uçlarında.

Seçicinin tersi **dağıtıcıdır** (demultiplexer, DEMUX): tek bir girişi seçme tellerinin gösterdiği çıkışa verir, öteki çıkışlar 0'da kalır. 1'den 4'e dağıtıcı 2 DEĞİL ve 4 VE kapısıyla kurulur. İkisi birlikte uzun bir teli paylaştırır: 4 algılayıcının verisi tek telden gider. Gönderen uçta seçici, alan uçta dağıtıcı vardır; iki uçtaki seçme telleri aynı anda aynı numarayı gösterir (00, 01, 10, 11) ve her algılayıcıya sırası gelince yol açılır. Algılayıcılar 1, 0, 1, 1 ise alıcılara sırayla 1, 0, 1, 1 ulaşır.



Şekil 6.9 Tek telden dört algılayıcı: gönderen uçta seçici, alan uçta dağıtıcı; iki uçtaki seçme bitleri aynı, alıcılara sırayla 1, 0, 1, 1 ulaşıyor.

Bilgisayarın içinde kaynak çoksa başka bir yol kullanılır: **ortak veri yolu**. Her kaynak bu tele **üç durumlu tamponla** bağlanır. Tampon izinliyken girişini tele verir; izinsizken telden kopar, çıkışı ne 0 ne 1 olur. Bu üçüncü duruma **yüksek empedans** (Z) denir. Aynı anda yalnız bir tampona izin verilir: ikisi birden konuşsa biri teli 0'a, öteki 1'e çeker ve teldeki değer belirsiz kalır.

Özet

- Kod çözücü n bitlik adresi 2^n hattan yalnız birine açar; her çıkışı bir minterimdir, izin girişi 0 iken bütün çıkışlar 0'dır.
- Büyük çözücü küçüklerden kurulur: üstteki bitler izin girişlerine gider; kod çözücüye VEYA eklemek her fonksiyonu kurar.
- Kodlayıcı çözücünün tersidir; iki tuş birden basılınca yanlış kod çıkmasın diye öncelikli kodlayıcı büyük numarayı seçer.
- BCD'den yedi parçaya sürücüde gelmeyen 10-15 kodları önemsiz durumdur; sağ alt çubuk B + $\overline{C}$ + D'ye iner.
- Seçici, seçme bitlerinin gösterdiği girişi çıkışa geçirir; 4'e 1 seçici Shannon açılımıyla 3 değişkenli her fonksiyonu kurar.
- Dağıtıcı tek girişi seçilen çıkışa verir; ortak veri yolunda üç durumlu tamponlardan aynı anda yalnız birine izin verilir.

Sorular

1 3 girişli bir kod çözücünün kaç çıkışı vardır?

A 3
B 6
C 8
D 9

2 2'ye 4 kod çözücüde A = 0, B = 1 iken hangi çıkış 1 olur ve o çıkışın ifadesi nedir?

A Y2 = A $\overline{B}$
B Y1 = $\overline{A}$ B
C Y1 = A $\overline{B}$
D Y3 = AB

3 İki 2'ye 4 çözücüyle kurulan 3'e 8 çözücüde B ve C iki çözücüye birden, A izinlere gidiyor (üsttekinin izni $\overline{A}$, alttakinin izni A; alttakinin hatları 4-7). Adres 101 gelince 8 hattan kaç numaralısı 1 olur?

Cevap:

4 Öncelikli olmayan, dört VEYA kapısından kurulu on tuşlu kodlayıcıda 1 ve 2 tuşlarına aynı anda basılıyor. Dört bitlik çıkış kodu (ABCD) ne olur?

Cevap:

5 Aynı durumda (1 ve 2 tuşları birlikte basılı) büyük rakama öncelik veren öncelikli kodlayıcı hangi kodu verir?

A 0011
B 0001
C 0000
D 0010

6 BCD'den yedi parçalı göstergeye sürücüde sağ alt çubuğun formülü B + $\overline{C}$ + D'dir. Bu formül neden A bitine bakmaz?

A A biti hiçbir rakamda 1 olmaz
B Çubuğun söndüğü tek rakam 2'dir (BCD = 010); aynı üç biti taşıyan öteki kod 10'dur ve bu kod girişe hiç gelmez
C Sağ alt çubuk yalnız tek rakamlarda yanar
D De Morgan kuralı A değişkenini her formülden siler

7 4'e 1 seçiciyle Y = Σm(1, 2, 3, 6) kuruluyor; A ve B seçme uçlarına bağlı. Seçicinin 3 numaralı girişine (AB = 11) ne bağlanır?

Cevap:

8 1'den 4'e dağıtıcıda veri girişi 1, seçme bitleri S1 S0 = 10. Çıkışlar 0, 1, 2, 3 sırasıyla ne olur?

A 0010

B 0100

C 1101

D 1111

SAYISAL TASARIM

7 ALU tasarımı

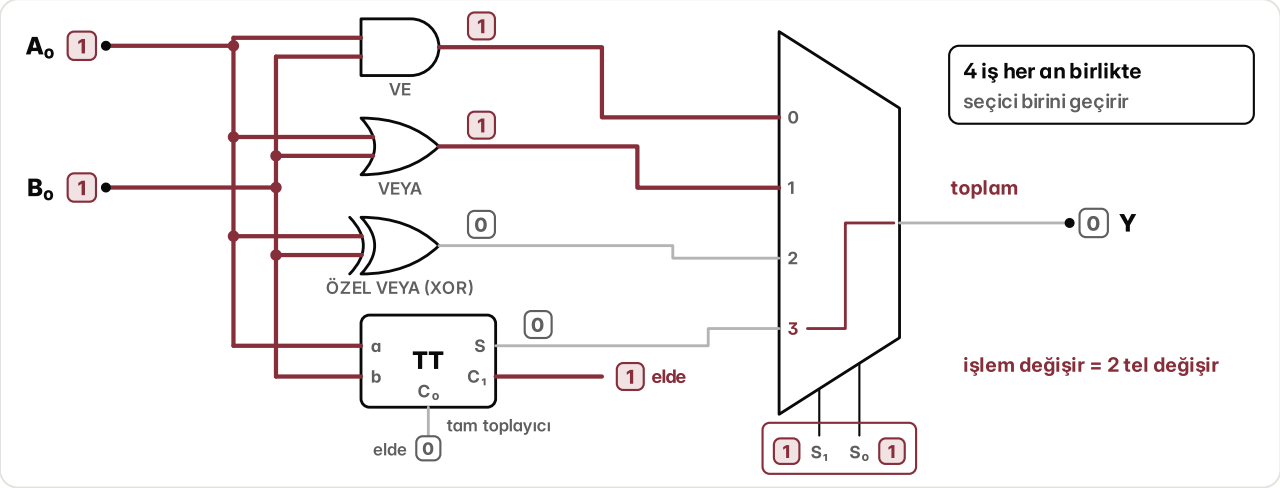
Bu bölümde işlemcinin hesap yapan bloğunu, ALU'yu kapı kapı kuruyoruz: tek bir bitlik dilimden dört bitlik ALU'ya, aynı toplayıcıyla dört aritmetik işe, işlem koduna ve sonucun yanında gelen bayraklara. Sonra işlemcinin iki sayıyı nasıl karşılaştırdığını görüyoruz: bir çıkarma yapıp bayraklara bakarak.

Ayrı devre mi?

videoda 37:26

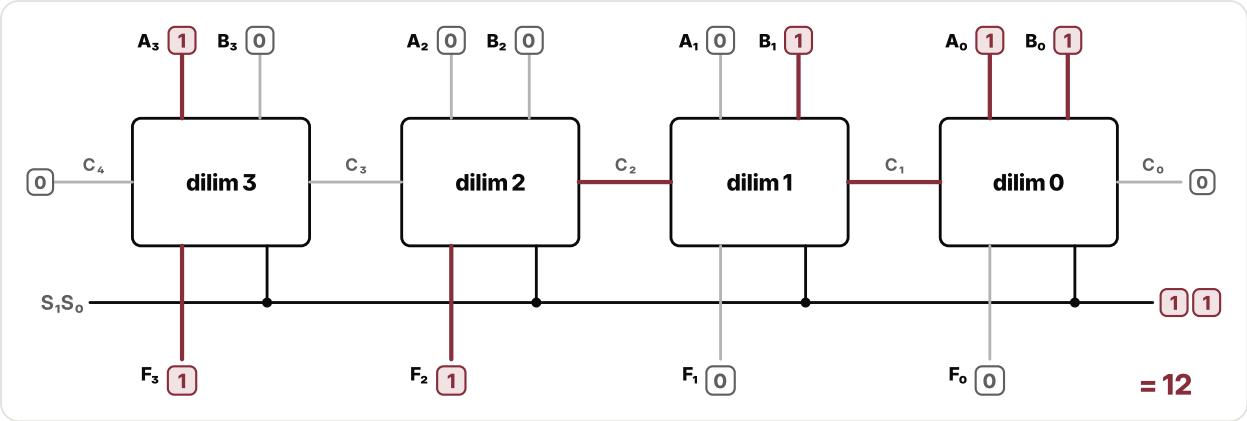
Bir komut iki sayıyı topluyor, sonraki ikisinin VE'sini alıyor, bir sonraki birini ötekinden çıkarıyor. İşlemci bunların her biri için ayrı devre taşımaz. Hepsini tek bir blok yapar: **ALU (aritmetik mantık birimi)**. İki sayı girer, bir sonuç çıkar; yanında birkaç bitlik bir giriş daha vardır, **işlem kodu**. ALU'nun hafızası yoktur, çıkışı yalnız o anki girişlere bağlıdır: kombinezonsal bir devredir.

Tasarım tek bitle başlar. A ile B'nin en sağdaki bitleri aynı anda dört yere gider: bir VE kapısına, bir VEYA kapısına, bir ÖZEL VEYA kapısına ve bir tam toplayıcıya. İkisi de 1 olsun: VE 1, VEYA 1, ÖZEL VEYA 0 verir; tam toplayıcının toplamı 0, eldesi 1 olur. Dört sonuç 4'e 1 bir seçiciye girer, iki seçme teli (S_1S_0) hangisinin çıkacağını belirler: 00 VE, 01 VEYA, 10 ÖZEL VEYA, 11 toplam. Dört iş her an birlikte yapılır, seçici yalnız birini geçirir. Bu yüzden işlemi değiştirmek için iki teli değiştirmek yeter.



Şekil 7.1 Bir bitlik dilim: VE, VEYA, ÖZEL VEYA ve tam toplayıcının sonuçları 4'e 1 seçiciye giriyor; S_1S_0 = 11 toplamı seçiyor.

Dört bitlik ALU için dört dilim yan yana dizilir. Seçme telleri dördüne birden gider, hepsi aynı işi yapar. Dilimler arasında yalnız elde yürür: her tam toplayıcı eldesini solundakine verir. A = 1001 (9) ve B = 0011 (3) ile seçme 00 olunca bitlerin VE'si çıkar: 0001. Seçme 11 olunca toplam çıkar: 1100, yani 12.



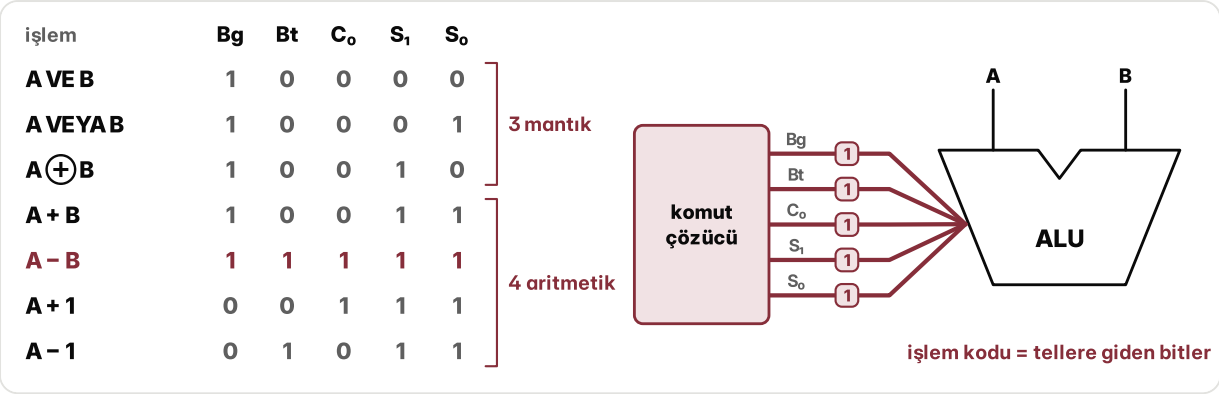
Şekil 7.2 Dört dilimli ALU, A = 1001 ve B = 0011: seçme 11, sonuç 1100 (12); eldeler $C_1 = 1$, $C_2 = 1$, $C_3 = 0$, $C_4 = 0$.

Toplayıcı yalnız toplar, ama girişine ne verdiğimizi değiştirebiliriz. B'nin her bitinin önüne iki kapı konur: bir VE kapısı B'yi geçirir ya da sıfırlar (öteki girişi Bg), bir ÖZEL VEYA kapısı olduğu gibi bırakır ya da ters çevirir (öteki girişi Bt). Bir tel de en sağdaki ilk eldeye (C_0) gider. Çıkarma bilinen yolla yapılır: B'nin bitleri ters çevrilir, 1 eklenir. B hem sıfırlanıp hem ters çevrilirse dört biti de 1 olur; 1111 ikinin tümleyeninde −1'dir.

```
Bg Bt C₀ | toplayıcıya giren B | işlem | sonuç (A = 1001)
1  0  0  | 0011               | A + B | 1100 = 12
1  1  1  | 1100               | A − B | 0110 = 6
0  0  1  | 0000               | A + 1 | 1010 = 10
0  1  0  | 1111               | A − 1 | 1000 = 8
```

Böylece işlem kodu beş bittir: Bg, Bt, C_0 ve seçicinin iki teli. Tablonun her satırı bir iştir: üç mantık, dört aritmetik işlem. İşlem kodu, ALU'nun içindeki tellere giden bitlerin listesidir; komut çözücü her komut için bu beş biti üretir.

```
işlem      Bg Bt C₀ S₁ S₀
A VE B     1  0  0  0  0
A VEYA B   1  0  0  0  1
A ⊕ B      1  0  0  1  0
A + B      1  0  0  1  1
A − B      1  1  1  1  1
A + 1      0  0  1  1  1
A − 1      0  1  0  1  1
```



Şekil 7.3 İşlem kodu tablosu ve komut çözücü: A − B satırının beş biti (11111) ALU'nun tellerine gidiyor.

Kaydırma için de yeni devre gerekmez. Bir sayıyı bir bit sola kaydırmak, yani iki katına çıkarmak için iki girişe aynı sayı verilir: 0011 + 0011 = 0110, 3'ün iki katı 6. Sağa kaydırmada kapı yoktur, teller bir basamak kayık bağlanır: 1001 (9) sağa kayınca 0100 (4) olur, en sağdaki bit düşer.

Dört bitlik böyle bir ALU'yu tek çipte toplayan ilk ürün 1970'te çıktı (Texas Instruments 74181): 16 aritmetik ve 16 mantık işlemi; mantık tarafı iki girişin 16 fonksiyonunun hepsini verir. Bugün her işlemci çekirdeğinde birkaç ALU yan yana çalışır; fikir aynıdır: işler birlikte yapılır, işlem kodu birini seçer.

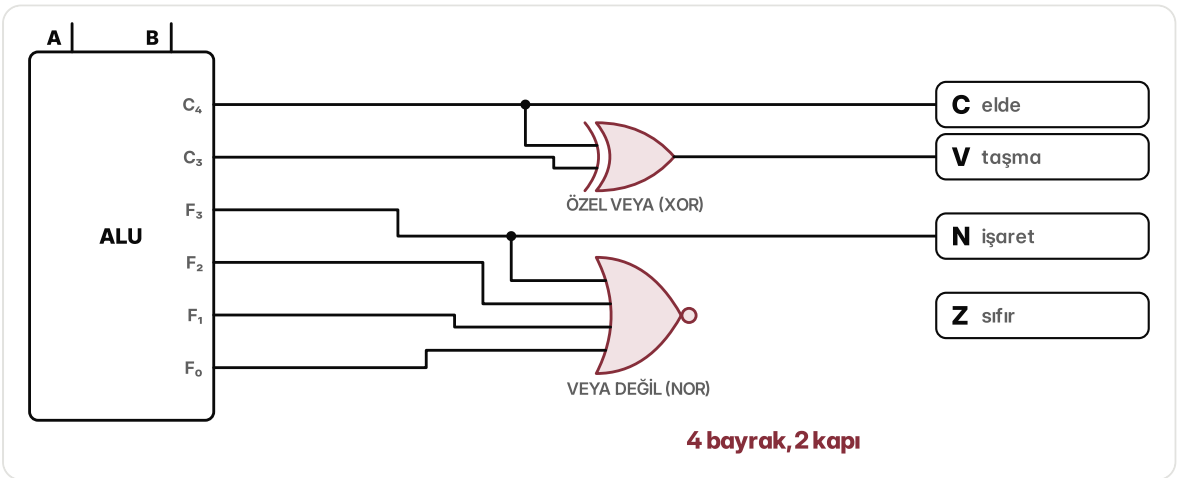
Eşit mi?

videoda 39:37

İşlemci "bu iki sayı eşit mi?" sorusunu ayrı bir karşılaştırıcıyla cevaplamaz: birini ötekinden çıkarır, sonucun yanında gelen dört bite bakar. Bu bitlere **bayrak** denir:

- Sıfır bayrağı (Z): sonucun dört biti de 0'sa 1'dir; dört çıkış bir VEYA DEĞİL (NOR) kapısına girer.
- Elde bayrağı (C): en soldaki basamaktan çıkan eldedir (C_4).
- İşaret bayrağı (N): sonucun en soldaki bitidir; sayı işaretli okunursa 1, negatif demektir.
- Taşma bayrağı (V): en sol basamağa giren ve çıkan eldenin ÖZEL VEYA'sıdır ($C_3 \oplus C_4$); işaretli sonuç dört bite sığmadıysa 1'dir.

Elde ve işaret doğrudan teldir; dört bayrak için iki kapı yeter.

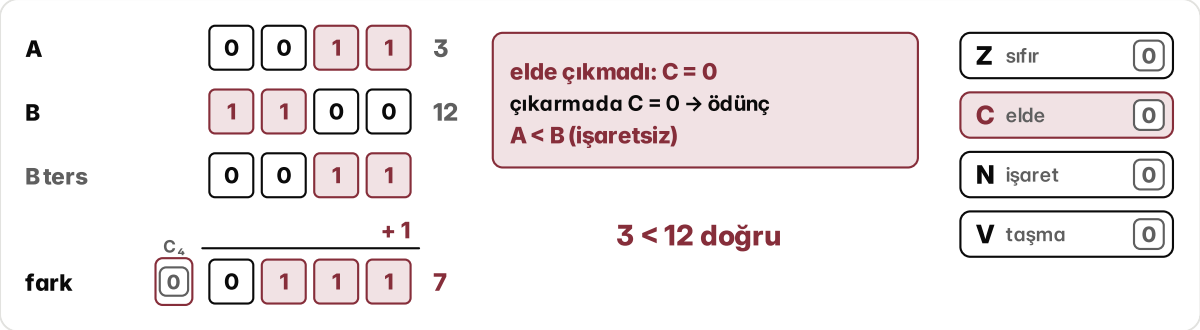


Şekil 7.4 Bayrak devresi: C_4 elde bayrağına, $C_3 \oplus C_4$ taşma bayrağına, F_3 işaret bayrağına, dört çıkışın VEYA DEĞİL'i sıfır bayrağına gidiyor.

Karşılaştırmada A − B hesaplanır, sonuç atılır, yalnız bayraklar kalır. 12 ile 12: B ters çevrilip 1 eklenir, 1100 + 0011 + 1 = 0000; sıfır bayrağı 1, sayılar eşit. Fark 0 değilse sıfır bayrağı da 0'dır.

Küçüklük için de aynı çıkarma yeter. 3 ile 12'yi karşılaştıralım: 0011 + 0011 + 1 = 0111. Sıfır bayrağı 0, eşit değiller. En soldan elde çıkmadı; çıkarmada elde çıkmaması ödünç alındığı demektir: A, B'den küçük. 3 gerçekten 12'den küçük.

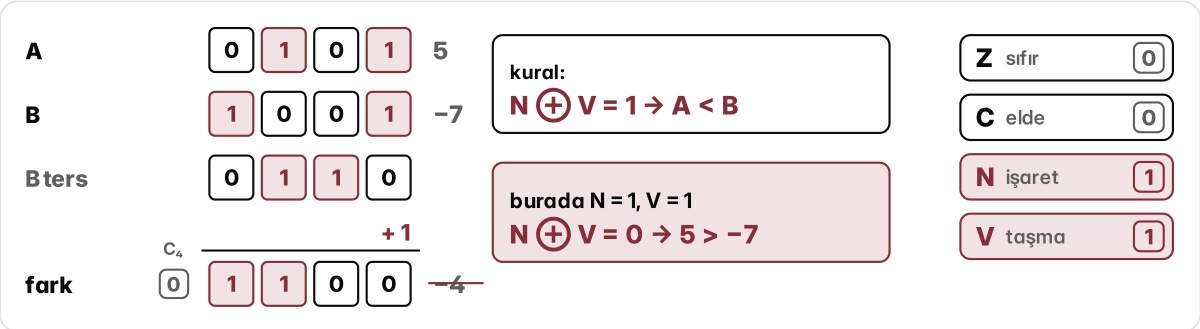
```
A − B          fark  Z  C  N  V
1100 − 1100    0000  1  1  0  0
0011 − 1100    0111  0  0  0  0
0101 − 1001    1100  0  0  1  1
```



Şekil 7.5 3 ile 12: 0011 + 0011 + 1 = 0111, elde çıkmadı (C = 0), işaretsiz okumada A < B.

Aynı bitleri işaretli okuyalım: 1100 artık −4'tür ve 3, −4'ten büyüktür. Elde bayrağı yine 0'dır, çünkü bitler değişmedi. İşaretli sayılarda elde değil işaret bayrağına bakılır: fark 0111, yani 7; işaret 0, fark pozitif, A büyük. Devre ve çıkarma aynıdır; hangi bayrağa bakılacağını sayının nasıl okunduğu belirler.

İşaret bayrağı tek başına her zaman doğruyu söylemez. 5 (0101) ile −7 (1001): 5 − (−7) = 12 eder, ama dört bitle işaretli sayı en çok 7'dir. Sonuç 1100 çıkar; işaret 1, sanki fark negatif. Taşma bayrağı da 1'dir: sonuç sığmamış, işaret ters dönmüştür. Kural bu yüzden şöyledir: işaret ile taşma farklıysa (N ⊕ V = 1) A küçüktür, aynıysa değildir. Burada ikisi de 1, N ⊕ V = 0: 5, −7'den büyük.



Şekil 7.6 5 ile −7: fark 1100, işaret ve taşma bayrakları 1; N ⊕ V = 0 olduğu için 5 > −7.

Tek çıkarma üç soruyu birden cevaplar: eşitliği sıfır bayrağı (Z = 1), işaretsiz küçüklüğü elde bayrağı (C = 0), işaretli küçüklüğü işaret ile taşmanın ÖZEL VEYA'sı (N ⊕ V = 1). Dört bitin 256 sayı çiftinin hepsinde bu üç kural doğrudur. Bir programdaki "eğer A, B'den küçükse" satırı işlemcide böyle çalışır: karşılaştırma komutu çıkarma yapar, sonucu atar, bayrakları bırakır; ardından gelen komut bayraklara bakar ve program iki yoldan birine gider.

Özet

- ALU, işlem koduna göre aritmetik ve mantık işlemlerinden birini yapan kombinezonsal bir bloktur.
- Bir bitlik dilimde VE, VEYA, ÖZEL VEYA ve tam toplayıcı aynı anda çalışır; 4'e 1 seçici birini geçirir.
- B'yi geçiren ya da sıfırlayan VE, ters çeviren ÖZEL VEYA ve ilk elde ile aynı toplayıcı A + B, A − B, A + 1 ve A − 1'i hesaplar.
- İşlem kodu, ALU'nun tellerine giden bitlerin listesidir; bu tasarımda beş bittir.
- Bayraklar: Z sonucun sıfır olduğunu, C en soldan çıkan eldeyi, N sonucun işaretini, V işaretli taşmayı söyler.
- Karşılaştırma bir çıkarmadır: Z = 1 ise eşit, C = 0 ise işaretsiz küçük, N $\oplus$ V = 1 ise işaretli küçük.

Sorular

1 **Bir bitlik ALU diliminde 4'e 1 seçicinin görevi nedir?**

A Yalnız seçilen işin kapısını çalıştırır, öteki kapılar bekler.

B VE, VEYA, ÖZEL VEYA ve toplamdan, seçme tellerinin gösterdiğini çıkışa geçirir.

C Eldeyi bir soldaki dilime taşır.

D Önceki işlemin sonucunu bir sonraki işleme kadar saklar.

2 **Bu bölümdeki ALU'da S_1S_o = 00 ve A = 1001, B = 0011 iken çıkış nedir?**

A 1011

B 1010

C 0001

D 1100

3 **B'nin önündeki kapılarda Bg = 0 (B sıfırlanır), Bt = 1 (ters çevrilir) ve ilk elde C_o = 0 ise toplayıcı ne hesaplar?**

A A + 1

B A − B

C A + B

D A − 1

4 **A = 0110, B = 0101. ALU'ya A − B işlem kodu (Bg Bt C_o S_1 S_o = 11111) verilince 4 bitlik sonuç ne olur?**

Cevap: _____

5 **Dört bitlik ALU'da A = 0101 iken iki girişe de aynı sayı verilip toplanıyor (A + A). 4 bitlik sonuç ne olur?**

Cevap: _____

6 **Dört bitlik ALU'da sıfır bayrağı (Z) nasıl üretilir?**

A Dört çıkış bitinin VEYA DEĞİL'i alınır.

B Dört çıkış bitinin VE'si alınır.

C En sol basamağa giren ve çıkan eldenin ÖZEL VEYA'sı alınır.

D En soldaki çıkış biti olduğu gibi alınır.

7 **İşaretsiz A = 0010 ve B = 0111 için ALU A − B hesaplıyor. Elde bayrağı ve karşılaştırmanın sonucu nedir?**

A C = 1, yani A < B

B C = 0, yani A > B

C C = 0, yani A < B

D C = 1, yani A = B

8 **İşaretli A = 0110 (6) ve B = 1010 (−6) için ALU A − B hesaplıyor; fark 1100 çıkıyor. İşaret (N) ve taşma (V) bayrakları ile karar nedir?**

A N = 1, V = 0; A < B

B N = 1, V = 1; A < B değil

C N = 0, V = 1; A < B

D N = 1, V = 1; A < B

SAYISAL TASARIM

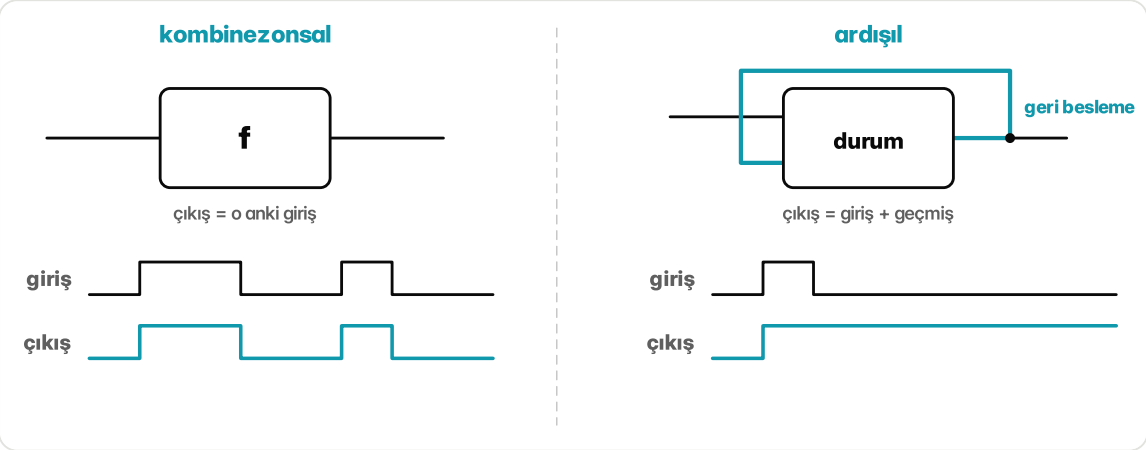
8 Mandallar ve flip-floplar

Şimdiye kadarki devrelerin çıkışı yalnız o anki girişe bağlıydı. Bu bölümde geçmişi hatırlayan devreler kurulur: çapraz bağlı iki kapıdan oluşan SR mandalı, izin girişli D mandalı, saat kenarında yazan D flip-flop ve SR, JK, T, D flip-floplarının karakteristik tabloları. Sonda flip-flopun doğru çalışması için gereken zamanlama koşulları ve açılışta sıfırlama ele alınır.

Duracak ışığı

videoda 41:40

Otobüste durak düğmesine basılıp el çekildiğinde öndeki "duracak" ışığı yanık kalır. Düğme artık 0 veriyor, ışık ise hâlâ 1. Toplayıcı ya da seçici gibi devrelerde böyle bir şey olmaz: çıkış yalnız o anki girişe bağlıdır, giriş geri dönünce çıkış da geri döner. Bu devrelere **kombinezonsal devre** denir. Işık ise düğmenin şimdiki hâline değil, geçmişine bağlıdır. Çıkışı girişle birlikte geçmişe, yani devrenin içinde tuttuğu duruma bağlı olan devreye **ardışıl devre** denir. Durumu tutmanın yolu **geri besleme**: bir kapının çıkışı dönüp kendi girişine bağlanır.



Şekil 8.1 Solda kombinezonsal devrede çıkış girişi izler; sağda geri beslemeli ardışıl devrede giriş kısa bir 1 verip 0'a dönse de çıkış 1'de kalır.

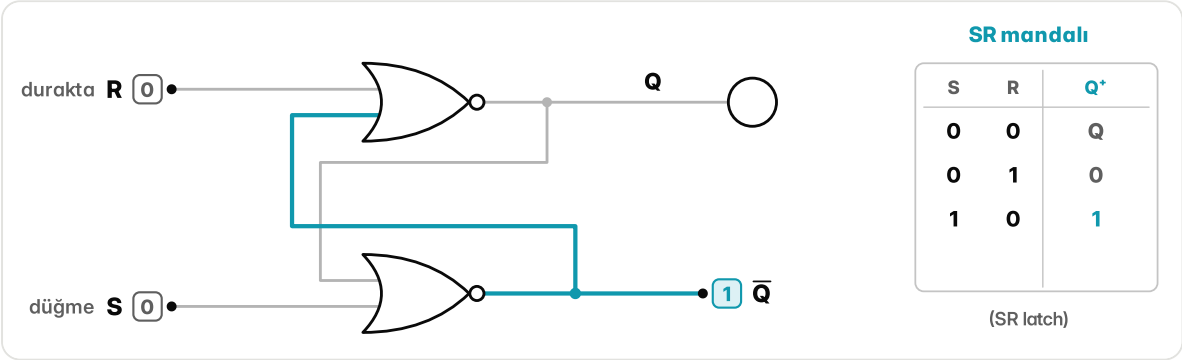
Bir biti tutan en küçük devre iki VEYA DEĞİL (NOR) kapısıyla kurulur. VEYA DEĞİL, girişlerinden biri 1 olunca 0 verir; yalnız ikisi de 0 iken 1 verir. İki kapının her birinin çıkışı ötekinin girişlerinden birine bağlanır. Üstteki kapının çıkışı Q ışığa gider, alttakinin çıkışı $\overline{Q}$ onun tersidir. Boşta kalan girişlerden alttakine S (düğme), üsttekine R (otobüs durakta durunca 1) denir. Başta S = R = 0, Q = 0 ve $\overline{Q}$ = 1'dir.

Düğmeye basılınca S 1 olur. Alttaki kapının bir girişi 1 olduğu için çıkışı $\overline{Q}$ 0'a iner. Bu 0 üstteki kapıya döner; üstteki kapının iki girişi de artık 0'dır, çıkışı Q 1 olur ve ışık yanar. Değişim kapıdan kapıya, her adımda bir kapı gecikmesiyle yürür:

```
adım | Q  Q̄
  0  | 0  1
  1  | 0  0
  2  | 1  0
```

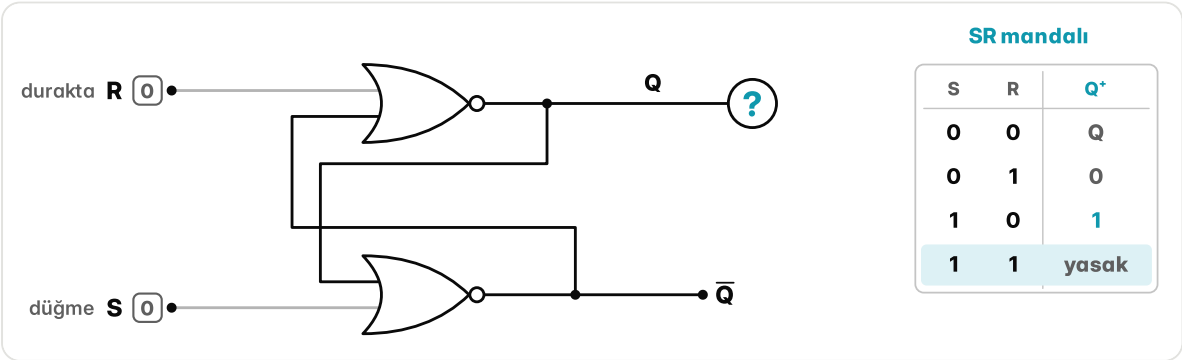
El çekilince S 0 olur. Alttaki kapıya üstten hâlâ 1 geldiği için çıkışı 0 kalır; üstteki kapıya o 0 ile R girişinin 0'ı girer, çıkışı 1 kalır. İki kapı birbirini yerinde tutar; düğmeye yeniden basmak bir şey değiştirmez. Otobüs durakta durunca R 1 olur: bu kez önce Q 0'a iner ve ışık söner, sonra bu 0 alttaki kapıya döner ve $\overline{Q}$ 1 olur. R 0'a dönünce devre 0'ı tutar. S kurar, R sıfırlar, ikisi de 0 iken devre son değeri tutar. Bir bit tutan bu düzene **SR mandalı** (SR latch) denir. Tablodaki Q^+, girişler uygulandıktan sonraki çıkıştır:

```
S  R | Q⁺
0  0 | Q (tutar)
0  1 | 0
1  0 | 1
1  1 | yasak
```



Şekil 8.2 Çapraz bağlı iki VEYA DEĞİL kapısıyla SR mandalı: R 1 olup 0'a döndükten sonra Q = 0, $\overline{Q}$ = 1; yanda mandalın tablosu.

Otobüs durakta dururken biri düğmeye basarsa S ve R birlikte 1 olur: iki kapının da bir girişi 1'dir, ikisinin çıkışı da 0. İki çıkış artık birbirinin tersi değildir. Asıl sorun ikisi aynı anda 0'a dönünce çıkar: iki kapı birden 1'e çıkmaya çalışır ve önce davranan kazanır. Gecikmeler tam eşit olsaydı iki çıkış her adımda birlikte 1 ile 0 arasında gidip gelirdi; gerçekte kapıların küçük gecikme farkı sonucu seçer, ışık yanık da kalabilir, sönük de. Bu yüzden S ile R'nin birlikte 1 olmasına **yasak durum** denir ve tasarımda bu birleşime izin verilmez.



Şekil 8.3 S = R = 1'den sonra ikisi birden 0'a dönünce ışığın durumu belirsiz kalır; tabloda bu satır yasaktır.

Bu bitin konduğu bir kutu yoktur; bit, iki kapının birbirini durmadan doğrulamasıyla yaşar. Döngü sürdükçe ışık yanık kalır; elektrik kesilirse döngü dağılır ve bit kaybolur.

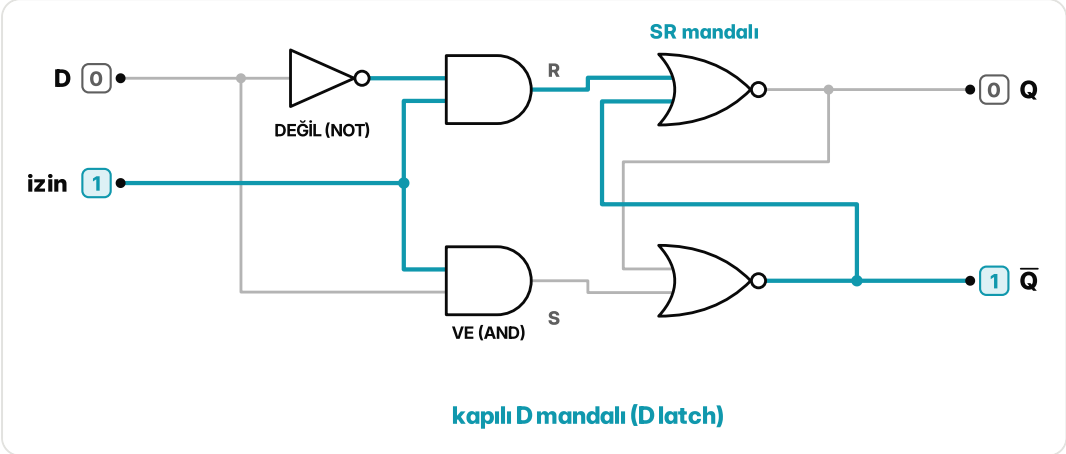
Her tıkta bir kez

videoda 43:35

Tutulan bir biti her saat tıkında bir kez tersine çevirmek isteyelim: bit okunur, tersi geri yazılır. Bunu mandalla kurmanın neden işe yaramadığını görmek için önce mandala bir izin girişi eklenir.

SR mandalının önüne iki VE kapısı ve bir DEĞİL kapısı konur. Veri girişi D, S tarafındaki VE kapısına doğrudan, R tarafındakine DEĞİL kapısından geçerek girer. İkinci giriş olan **izin**, iki VE kapısına birden girer. İzin 0 iken iki VE kapısı da 0 verir ve mandal son değeri tutar. İzin 1 iken D 1 ise mandal kurulur, 0 ise sıfırlanır. S ile R hiçbir giriş birleşiminde birlikte 1 olmadığı için yasak durum da kalkar. Bu devreye **kapılı D mandalı** (D latch) denir:

```
izin  D | Q⁺
  0   0 | Q
  0   1 | Q
  1   0 | 0
  1   1 | 1
```



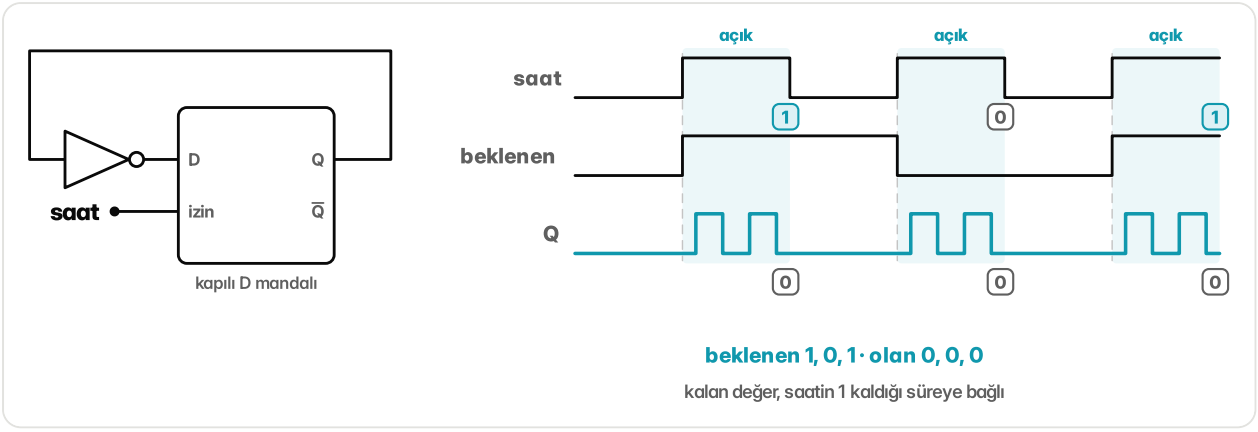
Şekil 8.4 Kapılı D mandalı: DEĞİL kapısı, izin girişli iki VE kapısı ve çapraz bağlı iki VEYA DEĞİL; izin 1 iken D = 0 yazılmış, Q = 0.

İzin 1 iken D'deki her değişiklik hemen çıkışta görünür; mandal açık bir pencere gibidir, buna **saydamlık** denir. İzin 0'a inince pencere kapanır: o anki değer içeride kalır, D ne yaparsa yapsın çıkış değişmez.

Bütün bitlerin aynı ritimle değişmesi için izin ucuna **saat işareti** bağlanır: düzenli aralıkla 0 ile 1 arasında gidip gelen bir sinyal. Bir tam turun süresine **periyot**, saniyedeki tur sayısına **frekans** denir; frekans = 1 / periyot. Periyot 1 ns ise frekans 1 GHz, yani saniyede 1.000.000.000 turdur. 0'dan 1'e geçişe **yükselen kenar**, 1'den 0'a geçişe **düşen kenar** denir.

Şimdi çıkış bir DEĞİL kapısından geçirilip D'ye verilir. Saat 1'e çıkınca mandal saydam olur: çıkış değişir, tersi girişe döner, çıkış yeniden değişir. Her tur yalnız birkaç kapı gecikmesi sürdüğü için saat 1 kaldığı sürece bit durmadan değişir. Saat inince hangi değerde kalacağı, saatin 1 kaldığı

süreye bağlıdır; şekildeki benzetimde her tıkta 0'da kalır, oysa beklenen 1, 0, 1'dir. Sorun, mandalın bir süre boyunca açık kalmasıdır.

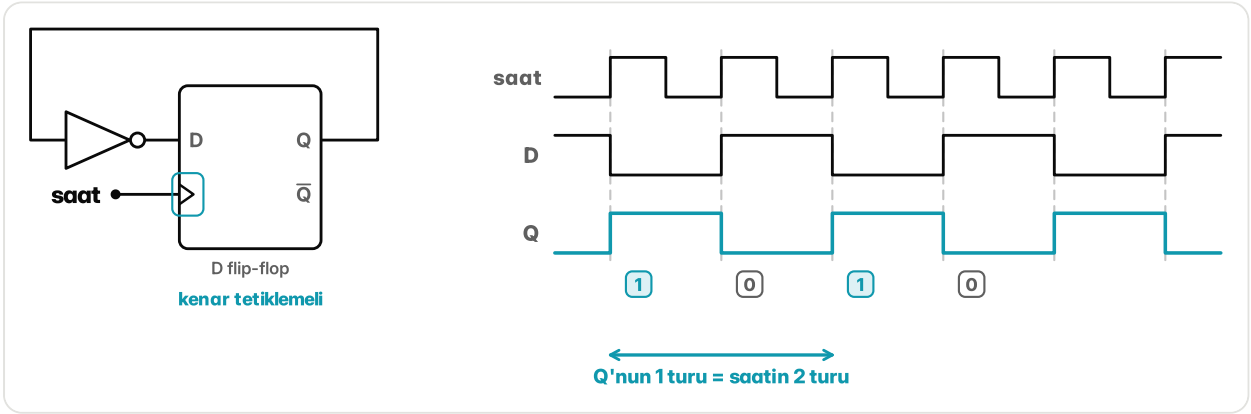


Şekil 8.5 Mandal ve DEĞİL kapısından kurulan döngü: saat 1 iken Q durmadan değişir; beklenen 1, 0, 1 iken her tıkın sonunda 0 kalır.

Çözüm, iki mandalı arka arkaya bağlamaktır. Birincisine **efendi**, ikincisine **köle** denir (master-slave). Efendinin izin ucuna saatin tersi, kölenin izin ucuna saatin kendisi bağlanır: efendi saat 0 iken, köle saat 1 iken açıktır; ikisi hiçbir zaman birlikte açık değildir. Saat 0 iken efendi girişi alır, köle eski değeri tutar. Saat yükselince efendi kapanır ve kenardan hemen önceki değeri saklar; köle açılır ve onu çıkışa verir. Girişteki yeni değer artık efendinin kapalı kapısında bekler:

```
saat  D | efendi  köle
  1   0 |   0      0
  0   1 |   1      0
  1   1 |   1      1
  1   0 |   1      1
```

Böylece değer yalnız yükselen kenar anında yazılır. Buna **kenar tetikleme**, bu yapıya **D flip-flop** denir; simgesinde saat ucu küçük bir üçgendir. Aynı döngü D flip-flopla kurulduğunda çıkış her yükselen kenarda tam bir kez döner: 1, 0, 1, 0. Çıkış, saatin iki turunda bir tur atar.



Şekil 8.6 D flip-flopla kurulan döngü: Q her yükselen kenarda bir kez döner; Q'nun 1 turu saatin 2 turudur.

Döngüdeki DEĞİL kapısı gibi, flip-floplar arasındaki devreler hesabını iki kenar arasında bitirir; sonuç bir sonraki kenarda yazılır. Aynı saate bağlı bütün flip-floplar tıkta bir kez ve birlikte yazılır, hesap adım adım ilerler.

Çakışan iki emir

videoda 45:37

Bir flip-flop saat kenarında girişlerine bakıp sonraki değerini seçer. Aynı kenarda hem "1'e kur" hem "sıfırla" emri gelirse ne yapmalı?

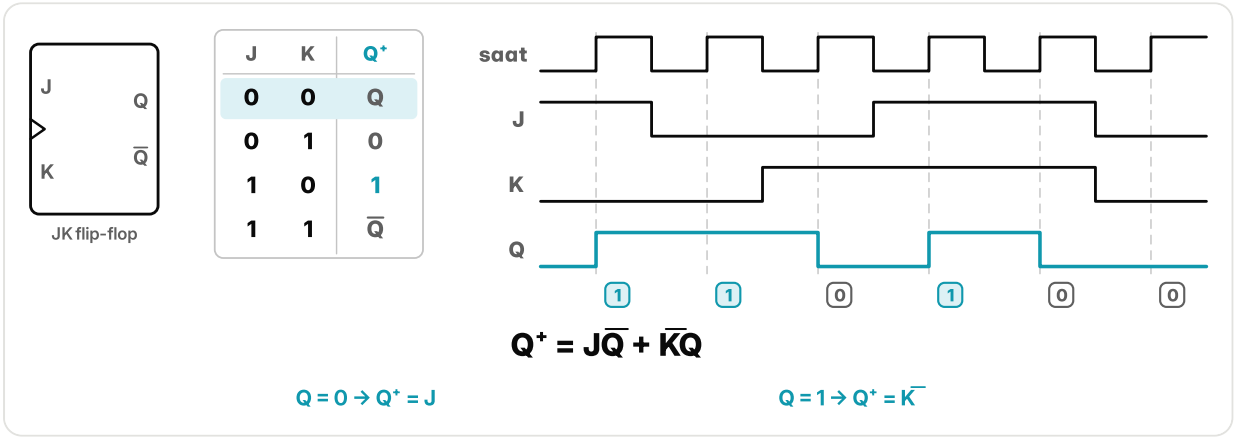
SR flip-flop, mandalın kenar tetiklemeli hâlidir: yalnız yükselen kenarda girişlerine bakar. S 1 ise çıkış 1, R 1 ise 0 olur; ikisi de 0 ise çıkış aynı kalır. Girişlerden ve şimdiki çıkıştan kenardan sonraki çıkışı (Q^+) veren tabloya **karakteristik tablo** denir. SR flip-flopun tablosu SR mandalınınkiyle aynıdır ($Q^+ = S + \overline{R}Q$); S ile R birlikte 1 olunca sonuç tanımsızdır, bu satır yasaktır.

JK flip-flop bu boşluğu doldurur. J kurar, K sıfırlar, ikisi de 0 ise çıkış aynı kalır; ikisi birden 1 olunca çıkış tersine döner. Açılıştaki sorunun bir cevabı budur: emirler çakışırsa flip-flop tersine döner. Solda karakteristik tablo, sağda Q başta 0 iken bir örnek dizi:

J	K	Q^+
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

kenar	J	K	Q^+
1	1	0	1
2	0	0	1
3	0	1	0
4	1	1	1
5	1	1	0

Denklem tablodan okunur: Q = 0 iken J 1 ise çıkış 1'e geçer ($Q^+ = J$); Q = 1 iken K 0 olduğu sürece 1'de kalır ($Q^+ = \overline{K}$). İki durum birlikte yazılınca $Q^+ = J\overline{Q} + \overline{K}Q$ olur.

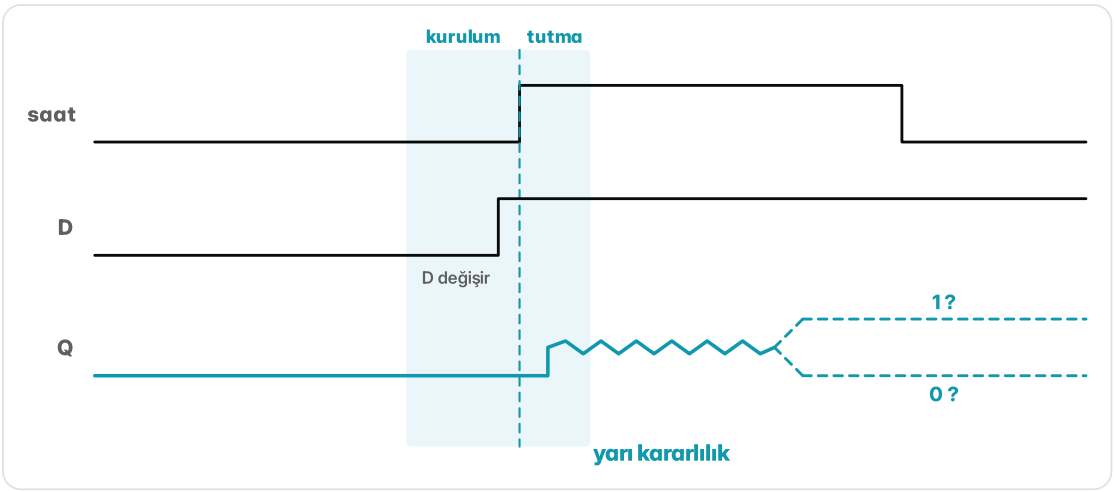


Şekil 8.7 JK flip-flopun karakteristik tablosu, örnek zaman çizelgesi ve denklemi $Q^+ = J\overline{Q} + \overline{K}Q$.

J ile K birbirine bağlanırsa tek girişli **T flip-flop** çıkar. T 0 ise çıkış aynı kalır, 1 ise her kenarda tersine döner: $Q^+ = T \oplus Q$. ÖZEL VEYA iki girişi farklıysa 1 verdiği için T = 1 iken $Q^+ = \overline{Q}$ olur. Q başta 0 iken T dizisi 1, 1, 0, 1 için çıkış 1, 0, 0, 1 olur.

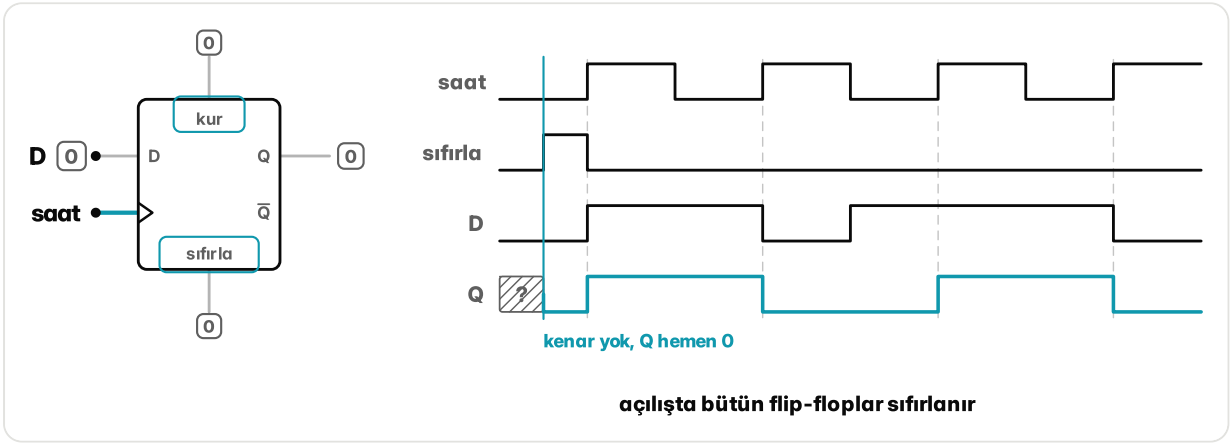
D flip-flop en sadesidir: $Q^+ = D$. Tasarımda en çok bu kullanılır, çünkü ötekiler ondan kurulabilir: D flip-flopun önüne $J\overline{Q} + \overline{K}Q$'yu hesaplayan küçük bir devre konunca flip-flop JK gibi çalışır.

Kenarda yazmak bir anı yakalamaktır, ama flip-flop o anı kısa bir aralıkta görür. Giriş, kenardan biraz önce sabitlenmiş olmalıdır; bu süreye **kurulum süresi** (setup time) denir. Kenardan sonra da biraz daha sabit kalmalıdır; buna **tutma süresi** (hold time) denir. Giriş tam bu aralıkta değişirse çapraz bağlı kapılar 1 ile 0 arasında asılı kalabilir. Bir süre sonra birine düşer, ama hangisine ve ne zaman düşeceği belli değildir; buna **yarı kararlılık** (metastability) denir.



Şekil 8.8 Yükselen kenarın çevresinde kurulum ve tutma aralıkları; D bu aralıkta değişince Q bir süre ara değerde kalır, sonra 1'e ya da 0'a düşer.

Cihaz açıldığında flip-flopların içindeki değerler rastgeledir. Bunun için birçok flip-flopta saati beklemeyen iki uç daha vardır: kur ve sıfırla. Sıfırla ucuna 1 verilince çıkış, kenar beklenmeden hemen 0 olur. Açılışta bütün flip-floplara sıfırla ucuna kısa bir 1 verilir; hepsi bilinen bir değerden başlar ve ilk kenardan itibaren her şey karakteristik tablolardaki gibi yürür.



Şekil 8.9 Kur ve sıfırla uçlu D flip-flop: başta belirsiz olan Q, sıfırla 1 olur olmaz kenar beklemeden 0 olur, sonra her yükselen kenarda D'yi alır.

Özet

- Kombinezonsal devrenin çıkışı yalnız o anki girişe, ardışıl devrenin çıkışı ayrıca tuttuğu duruma bağlıdır; durum geri beslemeyle tutulur.
- Çapraz bağlı iki VEYA DEĞİL kapısı SR mandalıdır: S kurar, R sıfırlar, ikisi 0 iken tutar; S = R = 1 yasaktır.
- Kapılı D mandalı izin 1 iken saydamdır, izin 0 iken tutar; geri beslemeli devrede bu saydamlık bitin durmadan değişmesine yol açar.
- Efendi-köle yapısıyla kurulan D flip-flop değeri yalnız yükselen kenarda yazar; buna kenar tetikleme denir.
- Karakteristik denklemler: D için $Q^+ = D$, JK için $Q^+ = J\overline{Q} + \overline{K}Q$, T için $Q^+ = T \oplus Q$.
- Giriş kenarın çevresinde kurulum ve tutma süresi boyunca sabit kalmalıdır, yoksa yarı kararlılık olur; açılışta flip-floplar saati beklemeyen sıfırla ucuyla sıfırlanır.

Sorular

1 **Çapraz bağlı iki VEYA DEĞİL kapısından oluşan SR mandalında başta Q = 0'dır. R hep 0 iken S kısa süre 1 yapılıp yeniden 0'a çekiliyor. Q kaç olur?**

A 0, çünkü S yeniden 0 oldu

B 1, çünkü iki kapı birbirini bu değerde tutar

C Q ile $\overline{Q}$ ikisi de 0 olur

D S'ye bağlı olarak 0 ile 1 arasında gidip gelir

2 **SR mandalında hangi giriş birleşimine yasak durum denir?**

A S = 0, R = 0

B S = 1, R = 0

C S = 0, R = 1

D S = 1, R = 1

3 **Kapılı D mandalında izin 1 iken D girişi birkaç kez değişirse çıkış ne yapar?**

A Her değişikliği hemen izler

B Yalnız ilk yükselen kenarda bir kez değişir

C İzin 0'a inene kadar hiç değişmez

D D'nin tersini gösterir

4 **Periyodu 2 ns olan bir saat işaretinin frekansı kaç MHz'dir?**

Cevap: _____

5 **Efendi-köle yapısında efendi saat 0 iken, köle saat 1 iken açıktır. Bu D flip-flopun çıkışı ne zaman değişebilir?**

A Saat 1 olduğu sürece her an

B Yalnız düşen kenarda

C Yalnız yükselen kenarda

D D her değiştiğinde

6 **Flip-flopun kurulum süresi nedir?**

A Kenardan sonra girişin sabit kalması gereken süre

B Kenardan önce girişin sabitlenmiş olması gereken süre

C Saatin bir tam turunun süresi

D Açılışta sıfırla ucuna verilen 1'in süresi

7 **Bir JK flip-flopta Q başta 0'dır. Dört yükselen kenarda sırayla (J, K) = (1, 0), (1, 1), (0, 0), (1, 1) uygulanıyor. Her kenardan sonraki Q değerlerini sırayla, bitişik yaz.**

Cevap: _____

8 **Bir T flip-flopta Q başta 1'dir. Dört yükselen kenarda T sırayla 1, 0, 1, 1 oluyor. Her kenardan sonraki Q değerlerini sırayla, bitişik yaz.**

Cevap: _____

SAYISAL TASARIM

9 Yazmaçlar ve sayıcılar

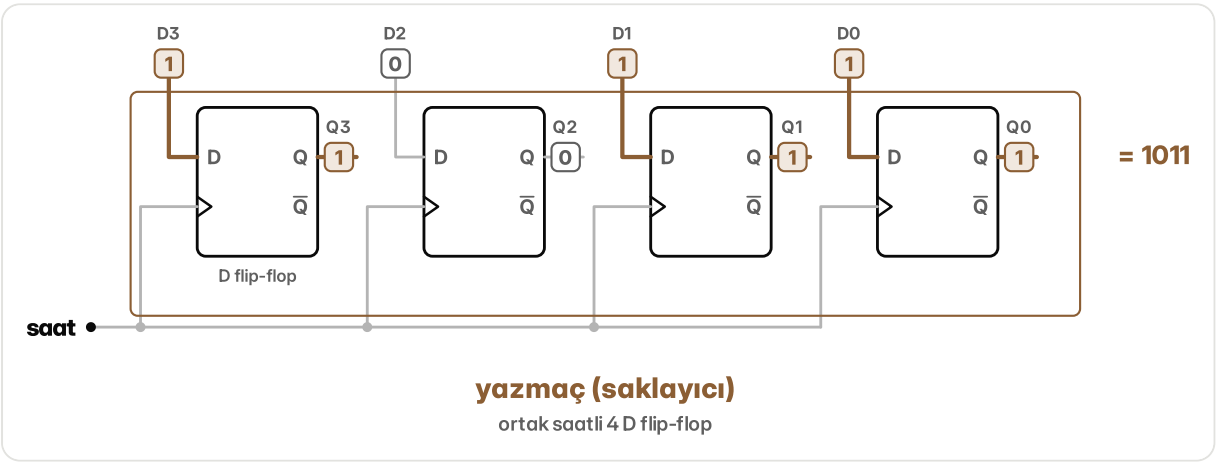
Bu bölüm, flip-flopları yan yana dizerek bir sayıyı tutan yazmacı, bitleri tek hattan gönderen kaydırmalı yazmacı ve saat tıklarını sayan sayıcıları anlatır. Sonunda kol saatindeki kristalin neden saniyede tam 32.768 kez titreştiği bu devrelerle açıklanır.

Tek hat

videoda 47:33

İşlemcinin içinde bir baytın 8 biti 8 ayrı telden aynı anda gider. USB kablosunda ise bitler yan yana değil, aynı hattan birbiri ardına geçer. 8 bitin tek hatta nasıl sıraya girdiğini ve öbür uçta nasıl yeniden bayt olduğunu anlamak için önce bitleri tutacak bir yer gerekir.

D flip-flop, saatin yükselen kenarında girişindeki biti yazar ve bir sonraki kenara kadar tutar. 4 D flip-flop yan yana konup saat girişleri aynı tele bağlanırsa her kenarda 4 bit birlikte yazılır. Saat girişleri ortak bu flip-flop dizisine **yazmaç** (register) denir; bazı üniversitelerde adı saklayıcıdır. Girişlere 1011 verilirse kenar gelince 4 çıkışta da 1011 belirir.



Şekil 9.1 Saat girişleri ortak tele bağlı 4 D flip-flop; girişlerdeki 1011 kenarla çıkışlara yazılmış.

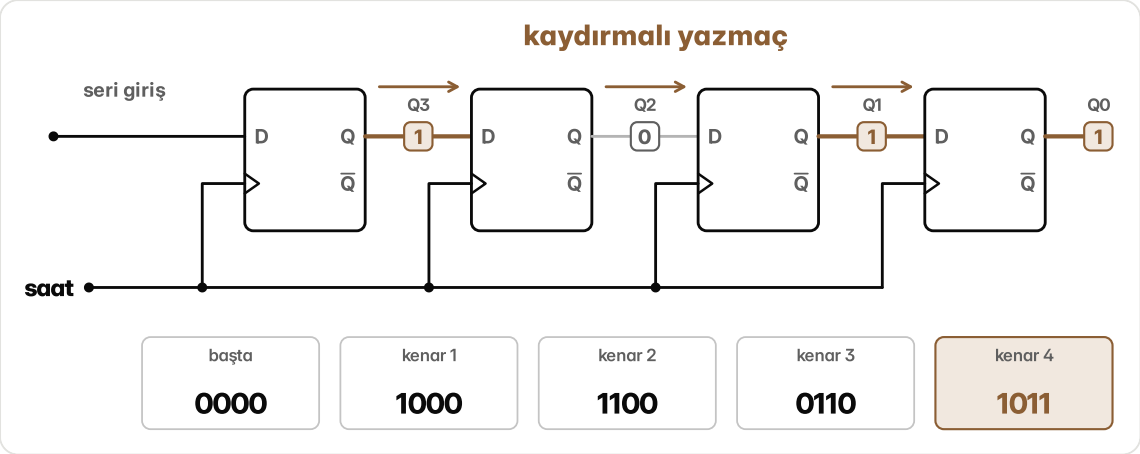
Saat ise hiç durmaz: yazmaç her kenarda girişte ne varsa onu alır ve tuttuğu sayı gider. Saati durdurmak akla gelir, ama saat teline konan kapının da bir gecikmesi vardır; o flip-flopun kenarı ötekilerden geç gelir. Bu yüzden seçim girişte yapılır: her flip-flopun önüne 2 girişli bir seçici konur. Seçicinin bir girişi dışarıdan gelen bit, öteki flip-flopun kendi çıkışıdır; hangisinin geçeceğini **yükleme izni** seçer. İzin 1 ise yeni bit girer; 0 ise flip-flop kendi değerini yeniden yazar ve değer değişmez. Yazmaç 1011 tutarken girişe 1100 gelsin:

```
kenar  yükle  giriş X  Q (kenardan sonra)
  1      1     1011      1011
  2      0     1100      1011
  3      0     1100      1011
  4      1     1100      1100
```

Bağlantı değiştirilip her flip-flopun çıkışı bir sağdakinin girişine, en soldakinin girişine de dışarıdan tek bir tel verilirse her kenarda bütün bitler bir sağa kayar, boşalan yere telden yeni bit girer. Bu devreye **kaydırmalı yazmaç** (shift register) denir. Teldeki bitler sırayla 1, 1, 0, 1 olsun:

```
kenar  Q3 Q2 Q1 Q0
başta   0  0  0  0
  1     1  0  0  0
  2     1  1  0  0
  3     0  1  1  0
  4     1  0  1  1
```

İlk gönderilen bit 4 kenarda en sağa yürür; 4. kenarın sonunda 1011 tamamdır.



Şekil 9.2 Zincir hâlinde bağlı 4 D flip-flop ve seri girişten giren 1, 1, 0, 1 bitlerinin kenar kenar yazmaçtaki durumu.

Tek telden 4 kenarda giren 4 bit şimdi 4 çıkışta birden durur. Bu, **seri girişli, paralel çıkışlı** yazmaçtır: tek hattan gelen bitleri yeniden sayıya toplar. Tersi de kurulur: sayı yazmaca 4 bit birden yüklenir, sonra her kenarda en sağdaki bit hatta çıkar. Yüklemekle kaydırmak arasında yine her flip-flopun önündeki seçici seçer. Gönderen taraf paralelden seriye, alan taraf seriden paralele çevirir. 1011'i yükleyen gönderen hatta sırayla 1, 1, 0, 1 verir; alan yazmaç 4 kenardan sonra 1011'i tutar. Bir baytın 8 biti de 8 kenarda geçer.



Şekil 9.3 Bitlerini hatta vermiş gönderen yazmaç (0000) ve 1011'i toplamış alan yazmaç; alanın 4 paralel çıkışı.

USB'nin açılımı Universal Serial Bus'tır; S, seri demektir: bitler sırayla gider. Kablonun bir ucunda baytı bitlere ayıran, öteki ucunda yeniden bayt yapan iş, özünde bu iki kaydırmalı yazmaçtır.

Yanlış sayı

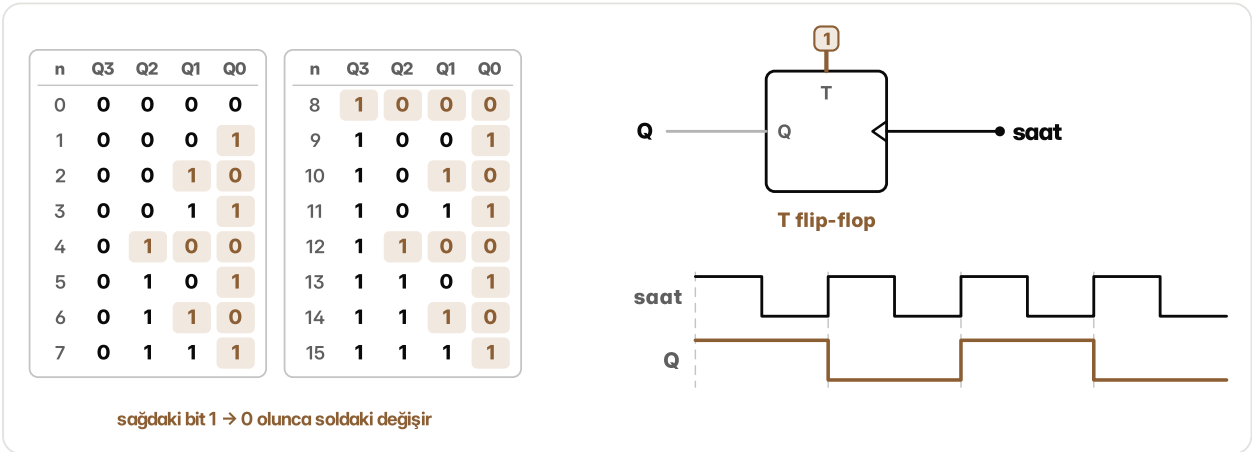
videoda 49:22

Saat tıklarını sayan 4 bitlik bir devre 7'den 8'e geçerken çıkışlarına çok hızlı bakan biri arada 6'yı, 4'ü, hatta 0'ı görebilir. Sayan bir devrenin yanlış sayı göstermesi, sayıcının nasıl kurulduğundan gelir.

Cevap ikilik sayının kendisinde. 0'dan 15'e kadar sayılar alt alta yazılınca bir düzen görünür:

```
n   Q3 Q2 Q1 Q0
0    0  0  0  0
1    0  0  0  1
2    0  0  1  0
3    0  0  1  1
4    0  1  0  0
5    0  1  0  1
6    0  1  1  0
7    0  1  1  1
8    1  0  0  0
9    1  0  0  1
10   1  0  1  0
11   1  0  1  1
12   1  1  0  0
13   1  1  0  1
14   1  1  1  0
15   1  1  1  1
```

En sağdaki bit her adımda değişir. Öteki bitlerin her biri, hemen sağındaki bit 1'den 0'a düştüğünde değişir; bu, sağındaki bütün bitlerin 1 olduğu adımdır. 7'den 8'e geçerken dördü birden değişir. Demek ki her bit için kendini ters çeviren bir parça gerekir: **T flip-flop**, T girişi 1 iken her saat kenarında çıkışını tersine çevirir, 0 iken tutar.



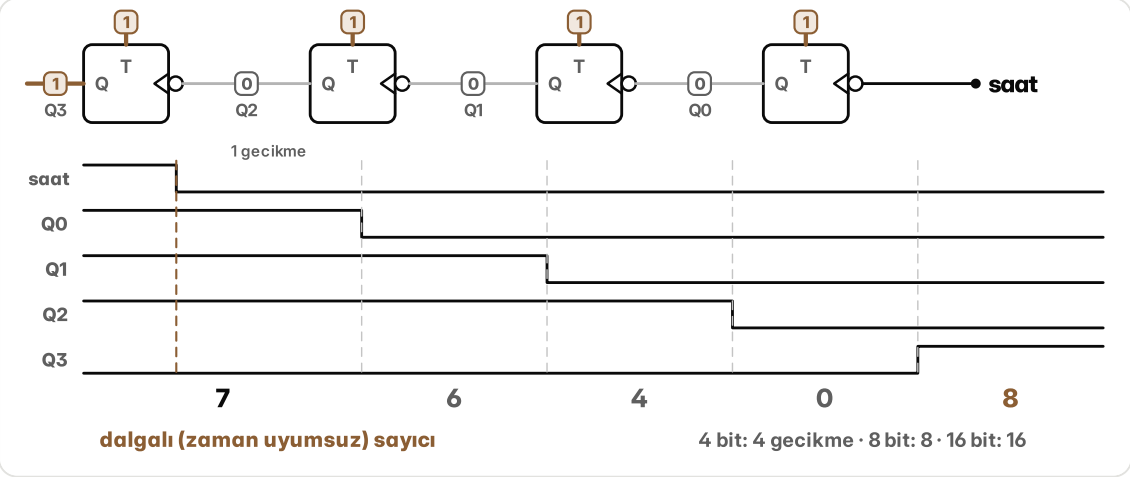
Şekil 9.4 0-15 sayma tablosu, bir önceki satıra göre değişen bitler işaretli; yanında T = 1 olan T flip-flop ve her kenarda dönen çıkışı.

En kolay sayıcı bu gözleme dayanır. 4 T flip-flopun T girişleri hep 1'dir ve hepsi düşen kenarda tetiklenir. En sağdakinin saat girişine saat gelir, öbürlerininkine sağdaki komşunun çıkışı. Böylece bir bit 1'den 0'a düşünce bu düşüş solundakinin kenarı olur ve solundaki döner. Sayıcı 0, 1, 2 ... 15 sayar ve 15'ten sonra 0'a döner; tek bir ek kapı bile gerekmez.

Ama her flip-flop sağındakinin değişmesini bekler ve kendisi de küçük bir gecikmeyle değişir. Her flip-flopun gecikmesi 1 birim sayılırsa 7'den 8'e geçiş şöyledir:

```
zaman              Q3 Q2 Q1 Q0  okunan
kenardan önce       0  1  1  1    7
1 gecikme sonra     0  1  1  0    6
2 gecikme sonra     0  1  0  0    4
3 gecikme sonra     0  0  0  0    0
4 gecikme sonra     1  0  0  0    8
```

Değişim bitten bite dalga gibi yayılır; bu yüzden böyle sayıcıya **dalgalı sayıcı** ya da **zaman uyumsuz sayıcı** (ripple counter) denir. Sayı ancak son bit oturunca doğrudur. En kötü geçişte n bitlik sayıcı n gecikme bekler: 4 bitte 4, 8 bitte 8, 16 bitte 16.

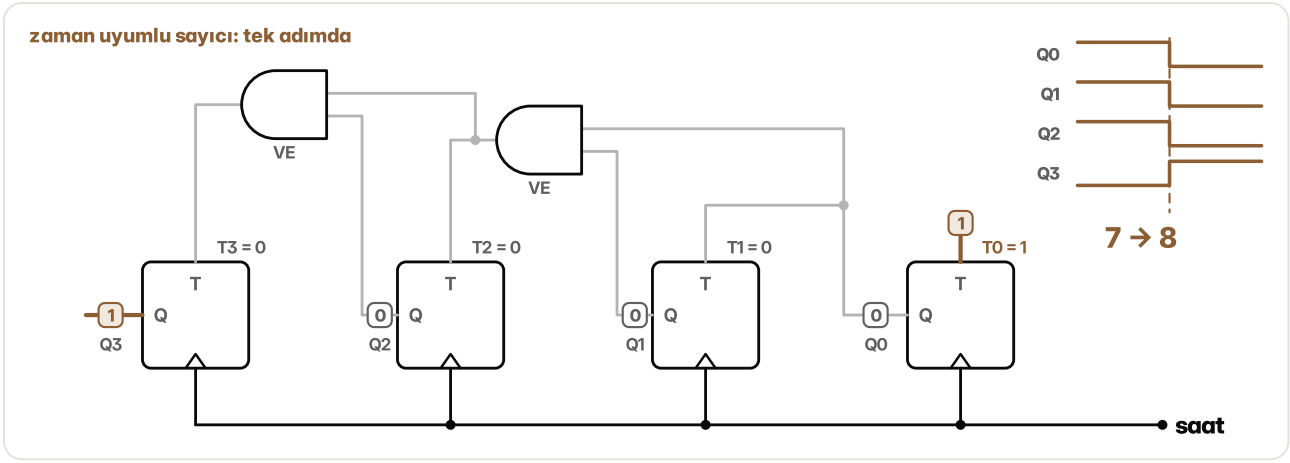


Şekil 9.5 7'den 8'e geçişin zaman çizelgesi: bitler birer gecikmeyle sırayla değişiyor, arada 6, 4 ve 0 okunuyor.

Çare, 4 flip-flopa aynı saati vermektir: hepsi aynı kenarda değişir. Bu kez her flip-flopa değişip değişmeyeceğini T girişi söyler. Kural yine sayma tablosundadır: bir bit, sağındaki bütün bitler 1 iken değişir. Buna göre T0 = 1, T1 = Q0, T2 = Q0·Q1, T3 = Q0·Q1·Q2. T3 için Q0·Q1 yeniden kullanılır ve Q2 ile VE'lenir; 2 VE kapısı yeter. Böyle kurulan sayıcıya **zaman uyumlu sayıcı** (synchronous counter) denir.

```
durum  Q3 Q2 Q1 Q0  T3 T2 T1 T0  sonraki
  7     0  1  1  1   1  1  1  1   1000
  8     1  0  0  0   0  0  0  1   1001
```

Sayı 7 iken sağdaki 3 bit 1'dir ve 4 T girişinin dördü de 1 olur. Kenar gelince 4 bit aynı anda döner; çıkışta doğrudan 1000, yani 8 belirir, arada başka sayı görünmez. Hızlı devrelerdeki sayıcılar bu yüzden zaman uyumludur.



Şekil 9.6 Ortak saatli zaman uyumlu sayıcı, T girişlerini üreten 2 VE kapısı ve dört çıkışın aynı kenarda 7'den 8'e geçtiği çizelge.

Tuhaf sayı

videoda 51:00

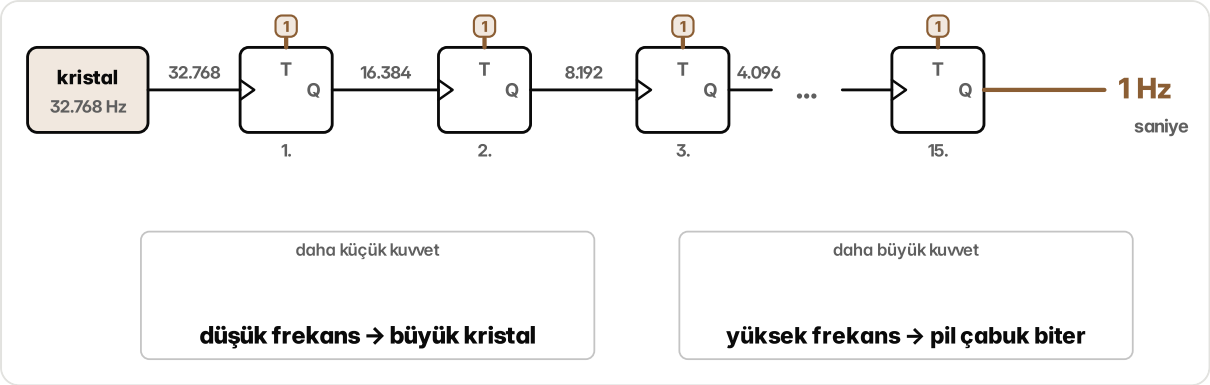
Kol saatinin içindeki küçük kuvars kristali saniyede 32.768 kez titreşir. Neden 30.000 gibi yuvarlak bir sayı değil de tam bu?

Cevap bir T flip-flopta. T girişi hep 1 olan flip-flop saatin her yükselen kenarında döner: bir kenarda 1'e çıkar, sonrakinde 0'a iner. Yani 2 tıkta bir tur atar; girişe 8 tık gelirse çıkış 4 kez 1'e çıkar. Bir flip-flop frekansı 2'ye böler.

Flip-floplar arka arkaya bağlanınca her biri öncekinin frekansını yarıya indirir:

```
çıkış            frekans (Hz)
kristal            32.768
1. flip-flop       16.384
2. flip-flop        8.192
3. flip-flop        4.096
...
15. flip-flop           1
```

15 flip-flop 2^{15}'e böler ve 2^{15} = 32.768, kristalin sayısının ta kendisidir. Kristalin titreşimi 15 kez 2'ye bölününce geriye saniyede tam 1 tık kalır. Sayı ikinin kuvveti olduğu için seçilmiştir. Daha küçük bir kuvvet de seçilebilirdi, ama frekans düştükçe kristal büyür; daha büyüğü de daha çok akım çeker ve pili çabuk bitirir.

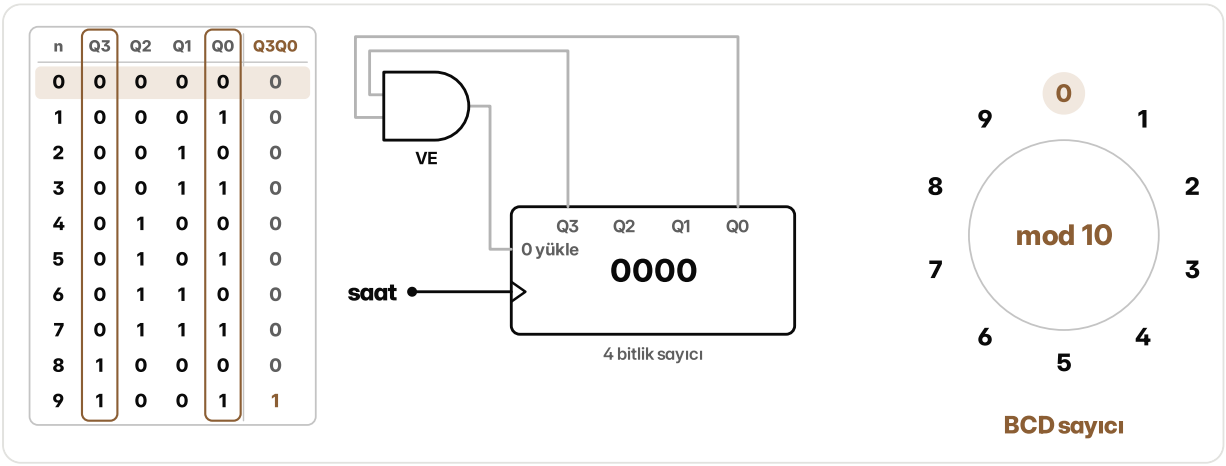


Şekil 9.7 Kristalden başlayan T flip-flop zinciri, her aralıktaki frekans ve 15. flip-flopun çıkışında 1 Hz.

Saniye elde edildi; şimdi saniyeleri saymak gerekir. 4 bitlik sayıcı 0'dan 15'e sayar, saniyenin birler basamağı ise 9'dan sonra 0'a dönmelidir. 0'dan 9'a kadar en soldaki ve en sağdaki bit yalnız 9'da (1001) birlikte 1'dir:

```
n  Q3 Q2 Q1 Q0  Q3·Q0
0   0  0  0  0    0
1   0  0  0  1    0
2   0  0  1  0    0
3   0  0  1  1    0
4   0  1  0  0    0
5   0  1  0  1    0
6   0  1  1  0    0
7   0  1  1  1    0
8   1  0  0  0    0
9   1  0  0  1    1
```

Bu durumu tek bir VE kapısı yakalar. Kapı 1 olunca sayıcı sonraki kenarda artmaz, 0'ı yükler. 0'dan 9'a 10 durumdan geçen bu sayıcıya **mod 10 sayıcı** denir. Her onluk basamağı 4 bitle tuttuğu için **BCD sayıcı** da denir.



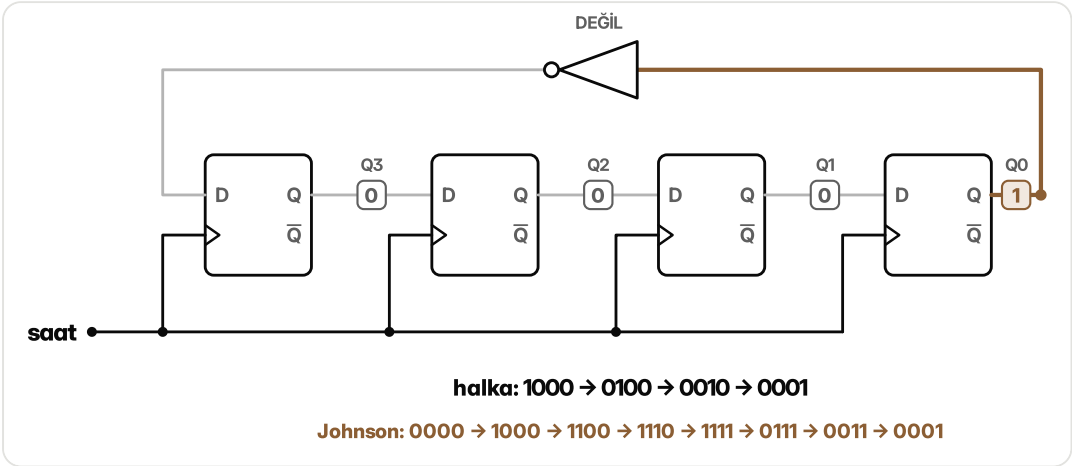
Şekil 9.8 4 bitlik sayıcının Q3 ve Q0 çıkışlarına bağlı VE kapısı, 0-9 tablosu ve 10 durumlu sayma halkası.

Aynı yolla 5'i (101) yakalayan Q2·Q0 kapısı 3 bitlik sayıcıyı 0'dan 5'e sayan **mod 6 sayıcı** yapar. Saniyenin onlar basamağı budur: birler 9'dan 0'a dönerken onlar 1 artar. İkisi birlikte 00'dan 59'a sayar, sonra dakikaya 1 tık verir. Sayıcıya bir giriş daha eklenirse geriye de sayar; geri sayarken

bir bit, sağındaki bütün bitler 0 iken değişir. 8'den (1000) 7'ye (0111) geçerken bu yüzden 4 bitin dördü de değişir. Böyle sayıcıya **yukarı-aşağı sayıcı** denir.

Kaydırmalı yazmaç da sayıcı olur: en sağdaki çıkış en soldaki girişe bağlanır. Yazmaçta yalnız bir bit 1 ise o 1, 4 flip-flop arasında dolaşır; 4 durumlu bu devreye **halka sayıcı** denir. Geri dönen bit DEĞİL kapısından geçirilirse aynı 4 flip-flop 8 durumdan geçer; buna **Johnson sayıcısı** denir:

```
halka:    1000  0100  0010  0001
Johnson:  0000  1000  1100  1110  1111  0111  0011  0001
```



Şekil 9.9 Geri dönen yoluna DEĞİL kapısı konmuş 4 flip-floplu Johnson sayıcısı, altında halka ve Johnson sayıcısının durum dizileri.

Ekranlı bir kol saatinde hepsi birlikte çalışır: 15 flip-flop kristalin titreşimini saniyeye böler, mod 10 ve mod 6 sayıcılar saniyeyi ve dakikayı sayar. Kristalin tuhaf görünen sayısı da bu zincirin başına göre seçilmiştir.

Özet

- Yazmaç, saat girişleri ortak D flip-floplardır; yükleme izni her flip-flopun önündeki seçiciyle "yaz ya da tut" seçimini yapar.
- Kaydırmalı yazmaç her kenarda bitleri bir yana kaydırır; seriden paralele ve paralelden seriye çevirerek bir baytı tek hattan gönderir.
- Dalgalı sayıcıda her flip-flop solundakini tetikler; kurması kolaydır ama gecikme bitten bite birikir ve geçişte ara değerler görünür.
- Zaman uyumlu sayıcıda bütün flip-floplar aynı saatle değişir; bir bit, sağındaki bütün bitler 1 iken döner ve T girişleri VE kapılarıyla üretilir.
- Her T flip-flop frekansı ikiye böler; $32.768 = 2^{15}$ olduğu için 15 flip-flop kristali tam 1 Hz'e indirir.
- Bir durumu yakalayıp 0'ı yükleyen kapı mod n sayıcı kurar: mod 10 (BCD) ve mod 6 birlikte saniyeyi sayar; halka ve Johnson sayıcıları kaydırmalı yazmaçtan çıkar.

Sorular

1 **4 bitlik bir yazmaçtaki 4 D flip-flopun ortak noktası nedir?**

- **A** D girişleri aynı tele bağlıdır
- **B** Saat girişleri aynı tele bağlıdır
- **C** Her biri bir öncekinin çıkışıyla tetiklenir
- **D** Aralarında VE kapıları vardır

2 **Yükleme izinli bir yazmaçta izin 0 iken saat kenarı gelirse ne olur?**

- **A** Girişteki yeni bit yazılır
- **B** Yazmaç 0000 olur
- **C** Her flip-flop kendi çıkışını yeniden yazar, değer değişmez
- **D** Saat durur, kenar flip-floplara ulaşmaz

3 **32.768 Hz'lik kristalden saniyede 1 tık elde etmek için zincire kaç T flip-flop gerekir?**

- **A** 15
- **B** 16
- **C** 14
- **D** 32

4 **4 bitlik kaydırmalı yazmaç 0000'dan başlıyor. Her kenarda bitler bir sağa kayıyor ve seri giriş en soldaki flip-flopa (Q3) giriyor. Seri girişe sırayla 1, 0, 1, 1 verilirse 4 kenardan sonra Q3 Q2 Q1 Q0 ne olur?**

Cevap:

5 **4 bitlik dalgalı sayıcı 15'ten (1111) 0'a geçerken, bitler sırayla değişirken çıkışta hangi ara değerler okunur?**

- **A** 8, 4, 2
- **B** Ara değer okunmaz, doğrudan 0 olur
- **C** 14, 12, 8
- **D** 7, 3, 1

6 **3 bitlik sayıcıyı 0'dan 5'e sayan mod 6 sayıcı yapmak için 5'i yakalayıp sonraki kenarda 0'ı yükleten kapı hangisidir?**

- **A** Q2·Q1
- **B** Q1·Q0
- **C** Q2·Q1·Q0
- **D** Q2·Q0

7 **4 bitlik zaman uyumlu sayıcıda T0 = 1, T1 = Q0, T2 = Q0·Q1, T3 = Q0·Q1·Q2. Sayıcı 1011 (11) durumundayken T3 T2 T1 T0 sırasıyla ne olur?**

Cevap:

8 **4 flip-floplu Johnson sayıcısı 1110 durumunda. Geri dönen bit DEĞİL kapısından geçip en soldaki flip-flopa giriyor. Bir sonraki durum nedir?**

Cevap:

SAYISAL TASARIM

10 Sonlu durum makineleri

Bu bölüm, geçmişin yalnız gereken kısmını hatırlayan devrelerin nasıl tasarlandığını anlatır: durum ve durum diyagramı, durumlara kod verip devreyi flip-flop ve kapıyla kurmak, Moore ve Mealy makineleri ve hazır bir devrenin ne yaptığını çözümlemek.

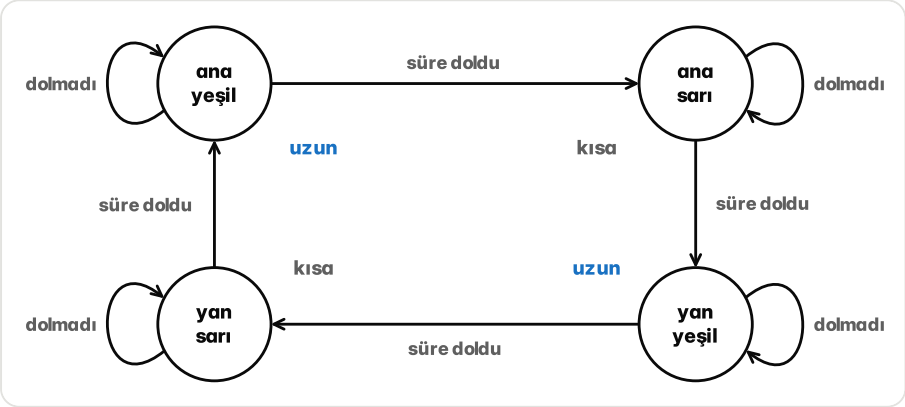
Sıradaki renk

videoda 52:44

Kavşakta ana yolun lambası sarıya döndü; birazdan kırmızı yanacak, yan yol yeşile geçecek. Sıradaki rengi neye bakarak seçtiğinin cevabı, şu an hangi renkte olduğudur. Sarıdan sonra hep kırmızı gelir; lambanın bütün geçmişi değil, yalnız şimdi nerede olduğunu bilmesi yeter.

İki yol birlikte düşünülünce 4 aşama vardır: ana yol yeşil, ana yol sarı, yan yol yeşil, yan yol sarı; bir yol yeşil ya da sarıyken öteki kırmızıdır. Bu aşamaların her birine **durum** denir. Geçmişten yalnız geleceği etkileyen kısmı tutan böyle bir devreye **sonlu durum makinesi** denir; durumları sayılı olduğu için sonludur.

Makine **durum diyagramı** ile çizilir: her durum bir daire, her geçiş bir ok. 4 daire halka gibi dizilir: ana yeşilden ana sarıya, oradan yan yeşile, yan sarıya ve yeniden ana yeşile. Ama lamba her an geçmez; yeşil uzun, sarı kısa sürer. Bu yüzden okların üstüne bir koşul yazılır: süre doldu (T = 1). Süre dolmadıysa (T = 0) makine yerinde kalır; bunu daireden çıkıp kendine dönen bir ok gösterir.



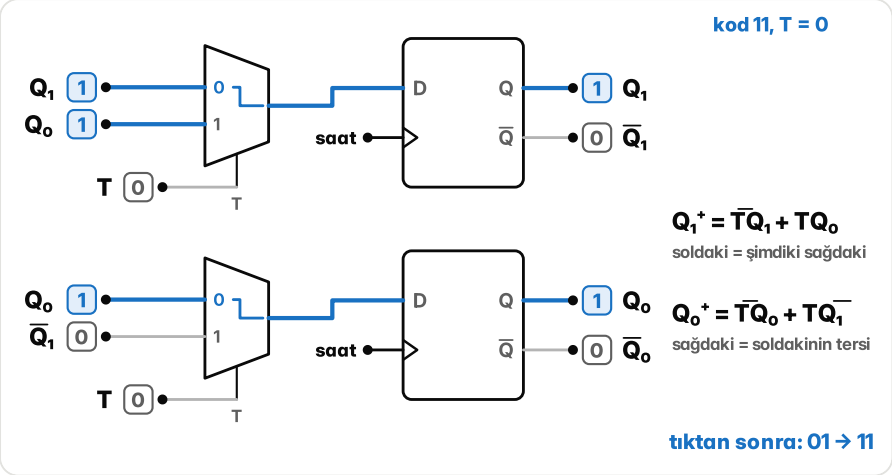
Şekil 10.1 Trafik lambasının durum diyagramı: 4 durum halka gibi dizili, sıradaki oklarda "süre doldu", kendine dönen oklarda "dolmadı".

Saniyeleri saymak makinenin işi değildir; yanına bir sayıcı konur. Saat saniyede bir tık atar; sayıcı her tıkta 1 artar, o durumun süresine ulaşınca T = 1 verir ve sıfırlanır. Bu örnekte ana yol 20, yan yol 10 saniye yeşil yanar, sarılar 3'er saniye. Makine yalnız 4 aşamayı hatırlar, saniyeleri sayıcı sayar.

4 durum 2 bitle yazılır, yani 2 flip-flop yeter. Kodlar halka boyunca 00, 01, 11, 10 diye verilir; her adımda yalnız bir bit değişir. Kural tek cümledir: süre dolduysa sıradaki kod, dolmadıysa aynısı.

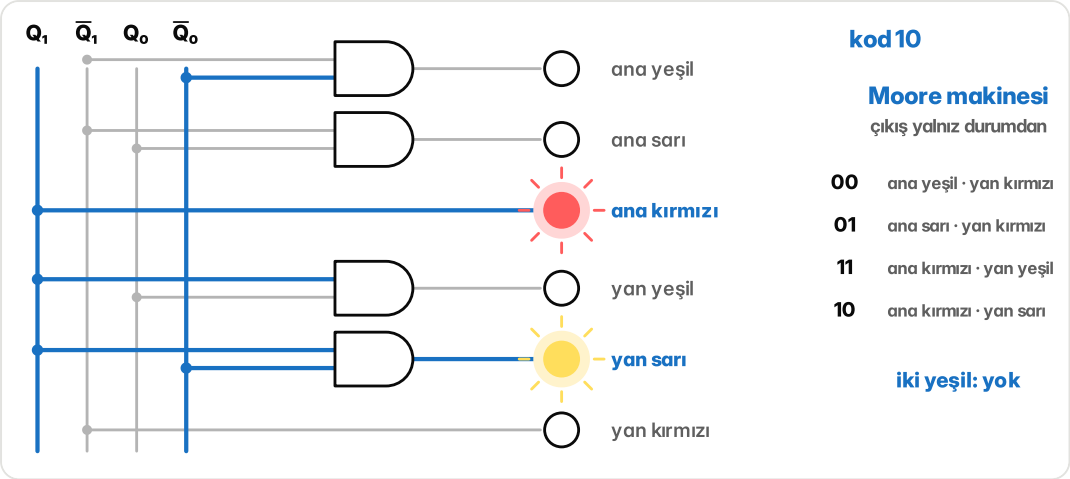
```
T  Q1 Q0 | Q1⁺ Q0⁺
0  0  0  |  0   0
0  0  1  |  0   1
0  1  1  |  1   1
0  1  0  |  1   0
1  0  0  |  0   1
1  0  1  |  1   1
1  1  1  |  1   0
1  1  0  |  0   0
```

Bu, her flip-flopun önüne 2 girişli bir seçici koymak demektir: T, eski bitle yeni bit arasından birini seçer, D flip-flop da onu saatin kenarında alır. Yeni bitler kodlardan okunur: sıradaki kodun soldaki biti şimdikinin sağdaki biti, sağdaki biti de şimdiki soldakinin tersidir. Yani $Q1^+ = \overline{T}\cdot Q1 + T\cdot Q0$ ve $Q0^+ = \overline{T}\cdot Q0 + T\cdot\overline{Q1}$.



Şekil 10.2 Önlerinde T ile seçen birer seçici olan 2 D flip-flop; kod 01 iken T = 1 gelmiş, kenardan sonra kod 11, T yeniden 0.

6 lambanın hepsi bu iki bitten okunur. Soldaki bit 0 iken ana yol akar: iki bit de 0'sa yeşil ($\overline{Q1}\cdot\overline{Q0}$), sağdaki 1'se sarı ($\overline{Q1}\cdot Q0$); yan yolda kırmızı yanar ($\overline{Q1}$). Soldaki bit 1 iken roller değişir: ana yol kırmızı ($Q1$), yan yol önce yeşil ($Q1\cdot Q0$), sonra sarı ($Q1\cdot\overline{Q0}$). Çıkışı yalnız duruma bağlı bu tür makineye **Moore makinesi** denir. İki yolun birden yeşil yandığı bir durum çizilmediği için devre o duruma hiç giremez.



Şekil 10.3 Q1, Q̄1, Q0, Q̄0 raylarından 4 VE kapısı ve 2 doğrudan telle sürülen 6 lamba; kod 10'da ana kırmızı ve yan sarı yanıyor.

Saat çalıştırılınca 00'da ana yol yeşildir. 20 tık sonra süre dolar, kod 01 olur: ana yol sarı. 3 tık sonra 11: ana yol kırmızı, yan yol yeşil. 10 tık sonra 10, 3 tık sonra yine 00. 36 saniyelik halka böyle döner; makinenin hatırladığı her şey 2 bittir. Çamaşır makinesinin programı da böyle ilerler: yıkama, durulama, sıkma; her aşamada süre dolunca sıradakine geçer.

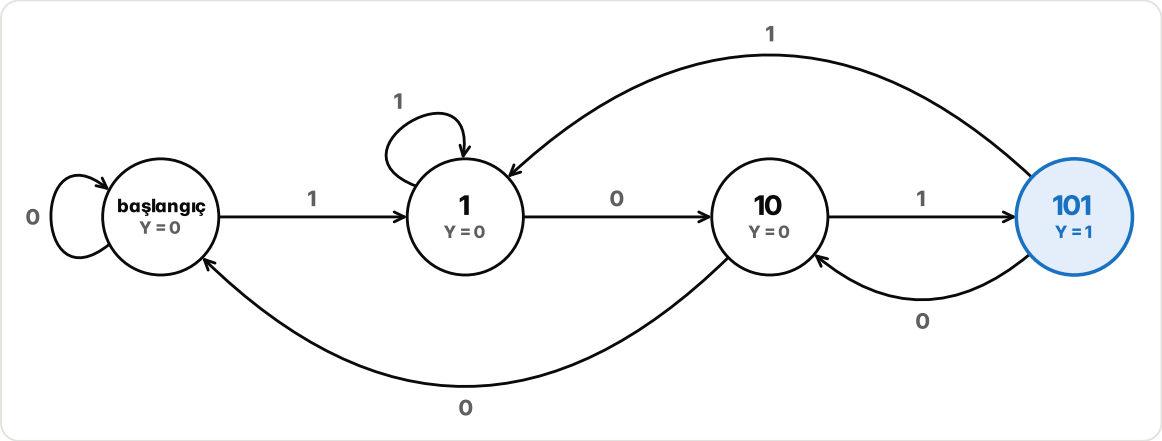
Telden gelen bitler

videoda 54:53

Bir telden her saat tıkında bir bit geliyor: 1, 1, 0, 1, sonra yine 0, 1. Art arda 1, 0, 1 geldiğinde lamba yanmalı. Oysa devre her tıkta yalnız o anki biti görür.

Tasarım sözel tariften başlar: girişte 1, 0, 1 görülünce lamba yansın; diziler iç içe geçebilir, 10101 dizisinde lamba 2 kez yanar. Geçmişten hatırlanması gereken yalnız dizinin ne kadarının geldiğidir. 4 durum yeter: başlangıç, "1", "10" ve lambanın yandığı "101".

Her durumdan 2 ok çıkar: giriş 1 ve giriş 0 için. Başlangıçta 1 gelirse "1" durumuna geçilir, 0 gelirse yerinde kalınır. "1" durumunda 0 gelirse "10"a geçilir; 1 gelirse yerinde kalınır, çünkü son bit yine 1'dir. "10"da 1 gelirse dizi tamamdır ve lamba yanar; 0 gelirse başa dönülür. Lambalı durumdan 1 gelirse "1"e, 0 gelirse "10"a geçilir: son bitler yeni dizinin başı olabilir. Lamba dairenin içine yazılır; bu bir Moore makinesidir.

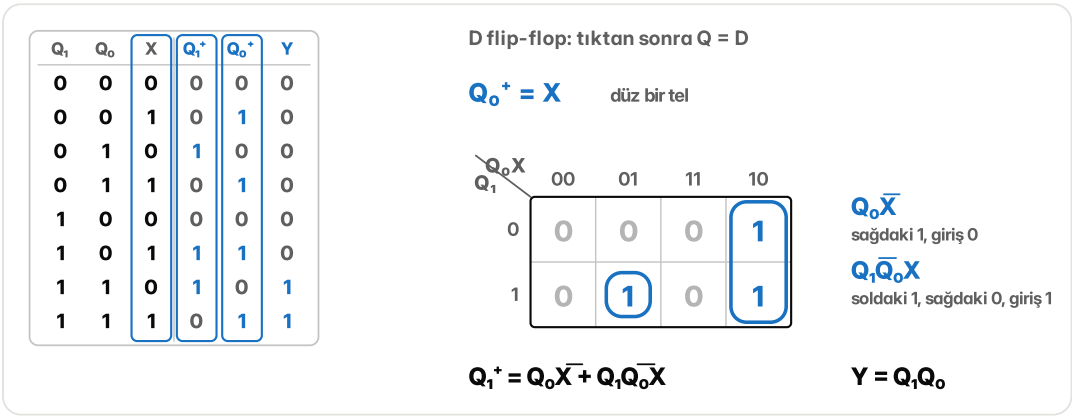


Şekil 10.4 1, 0, 1 algılayıcısının durum diyagramı: 4 durum, her durumdan giriş 0 ve 1 için birer ok, lamba yalnız "101" durumunda Y = 1.

Lambalı durumun okları "1" durumununkilerle aynıdır, yalnız lambası farklıdır. Lambası da aynı olsaydı ikisi tek durumda birleşirdi; durum sayısı böyle azaltılır. Kodlar sırayla verilir: başlangıç 00, "1" 01, "10" 10, "101" 11. Durum adlarının yerine kodları yazınca 8 satırlık bir doğruluk tablosu çıkar:

```
Q1 Q0 X | Q1⁺ Q0⁺ | Y
0  0  0 |  0   0  | 0
0  0  1 |  0   1  | 0
0  1  0 |  1   0  | 0
0  1  1 |  0   1  | 0
1  0  0 |  0   0  | 0
1  0  1 |  1   1  | 0
1  1  0 |  1   0  | 1
1  1  1 |  0   1  | 1
```

D flip-flop kenarda girişindekini alır; bu yüzden her flip-flopun girişine tablodaki sonraki bit bağlanır. $Q0^+$ sütunu 8 satırın hepsinde X ile aynıdır: sağdaki flip-flop yalnız son gelen biti tutar, girişi düz bir teldir. $Q1^+$ sütununda 3 tane 1 vardır: 010, 101 ve 110 satırlarında. Karnaugh haritası 2 grup verir: $Q1^+ = Q0\cdot\overline{X} + Q1\cdot\overline{Q0}\cdot X$. Lamba $Y = Q1\cdot Q0$'dır.



Şekil 10.5 Kod tablosu ve $Q1^+$ için 3 değişkenli Karnaugh haritası: 2 ve 6 numaralı hücreler bir grup, 5 tek başına.

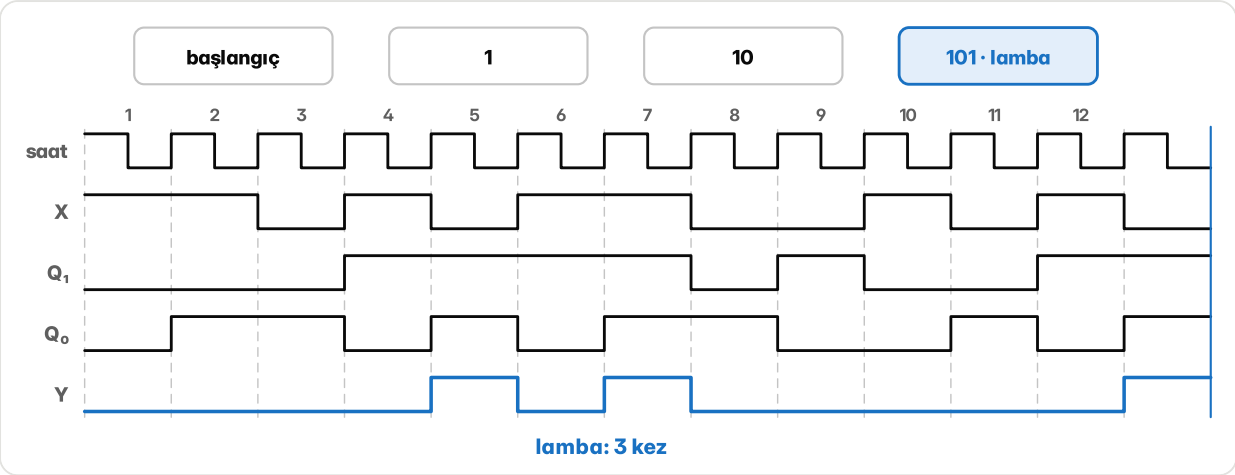
Devre 5 kapıyla kurulur: X'in tersi için bir DEĞİL, soldaki flip-flopun önünde 2 VE ile bir VEYA, lambanın önünde bir VE. Kodlar başka dağıtılınca sayı değişir: "1"e 11, "101"e 01 verilince $Q1^+ = \overline{Q1}\cdot X + Q0$, $Q0^+ = X$, $Y = \overline{Q1}\cdot Q0$ olur ve 3 kapı yeter; 00, 01, 11, 10 sırasıyla verilince 10 kapı gerekir. Bir yol daha vardır: her duruma ayrı bir flip-flop. 4 flip-flop gerekir ama denklemler diyagramdan doğrudan okunur.

12 bitlik bir giriş deneyelim: 1, 1, 0, 1; sonra 0, 1; sonra 1, 0, 0; en son 1, 0, 1.

```
bit       1   2   3    4   5    6   7   8   9   10  11  12
X         1   1   0    1   0    1   1   0   0   1   0   1
durum     1   1   10  101  10  101   1  10  baş  1  10 101
```

Son satır, o bit alındıktan sonraki durumdur (baş: başlangıç). 4. bit diziyi tamamlar ve bir sonraki tıkta lamba yanar. 5. ve 6. bitler 4. bitle yine 1, 0, 1 eder; diziler iç içedir, lamba yine yanar. 7. bit "1"e götürür, ardından gelen 2 sıfır başa döndürür, son 3 bit diziyi bir kez daha tamamlar. Lamba

3 kez yanar; tablo da devre de aynı sonucu verir. Ağ kablosundan gelen bitlerde verinin başladığı yer de böyle bulunur: alıcı, önceden bilinen bir bit dizisini yakalayınca okumaya başlar.



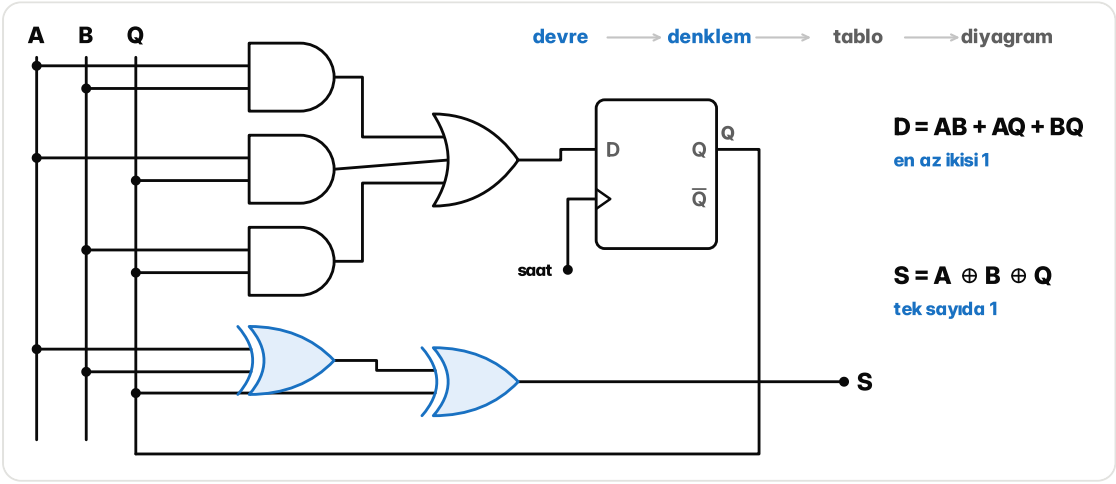
Şekil 10.6 12 bitlik denemenin zaman çizelgesi: saat, X, Q1, Q0 ve Y; lamba 4., 6. ve 12. bitten sonraki tıkta yanıyor.

Etiketsiz devre

videoda 57:10

Önümüzde A ve B diye 2 girişi, bir flip-flopu, 6 kapısı ve bir çıkışı olan küçük bir devre var; üstünde ne iş yaptığı yazmıyor. Yalnız çizimine bakarak ne yaptığı bulunabilir. Bu işe **ardışıl devre analizi** denir ve tasarımın tersine yürür: devreden denkleme, denklemden tabloya, tablodan diyagrama.

Önce flip-flopun girişine bakılır: 3 VE kapısı ve bir VEYA. VE kapıları A ile B'yi, A ile flip-flopun çıkışı Q'yu, B ile Q'yu birleştirir: D = A·B + A·Q + B·Q. Yani bu 3 değerden en az 2'si 1 olunca D = 1 olur. Sonra devrenin çıkışı: 2 ÖZEL VEYA kapısı, S = A ⊕ B ⊕ Q. Aynı 3 değerde tek sayıda 1 varsa S = 1 olur.



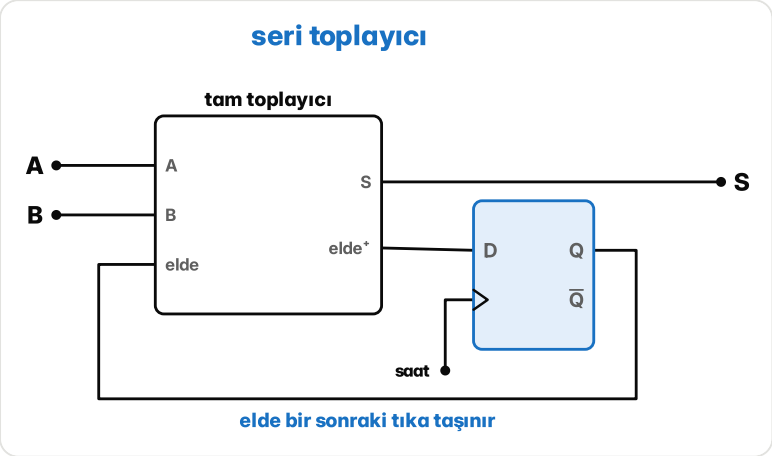
Şekil 10.7 Etiketsiz devre: A, B ve Q raylarından beslenen 3 VE ve bir VEYA kapısı D girişine, 2 ÖZEL VEYA kapısı S çıkışına; yanda D ve S denklemleri.

Flip-flopun çıkışı 0 ya da 1'dir; makinenin 2 durumu vardır. Her durum için 4 giriş birleşimi, toplam 8 satır:

```
Q  A  B | Q+  S
0  0  0 |  0  0
0  0  1 |  0  1
0  1  0 |  0  1
0  1  1 |  1  0
1  0  0 |  0  1
1  0  1 |  1  0
1  1  0 |  1  0
1  1  1 |  1  1
```

Durum 0'ken yalnız A ile B ikisi birden 1 gelince durum 1'e geçer. Durum 1'ken yalnız ikisi birden 0 gelince 0'a döner. Çıkış da her satıra yazılır, çünkü S yalnız duruma değil girişlere de bağlıdır. Diyagramda 2 daire vardır; her daireden kendine dönen 3 ok ve karşıya giden bir ok çıkar, okların üstünde girişler ve çıkış birlikte yazılır (ör. 11/0).

Bu tablo tanıdıktır: 2 bit ve bir elde toplanıyor; S toplam biti, durum da bir sonraki basamağa geçen eldedir. Devre, iki sayıyı en sağdaki basamaktan başlayıp her tıkta bir basamak toplayan bir **seri toplayıcıdır**: ortada bir tam toplayıcı vardır, eldeyi bir sonraki tıka flip-flop taşır.



Şekil 10.8 Seri toplayıcı: tam toplayıcının elde çıkışı D flip-flopa, flip-flopun çıkışı yeniden tam toplayıcının elde girişine bağlı.

5 (0101) ile 6 (0110) toplansın; bitler en sağdan gelir:

```
tık   A  B  elde | S  yeni elde
 1    1  0   0   | 1      0
 2    0  1   0   | 1      0
 3    1  1   0   | 0      1
 4    0  0   1   | 1      0
```

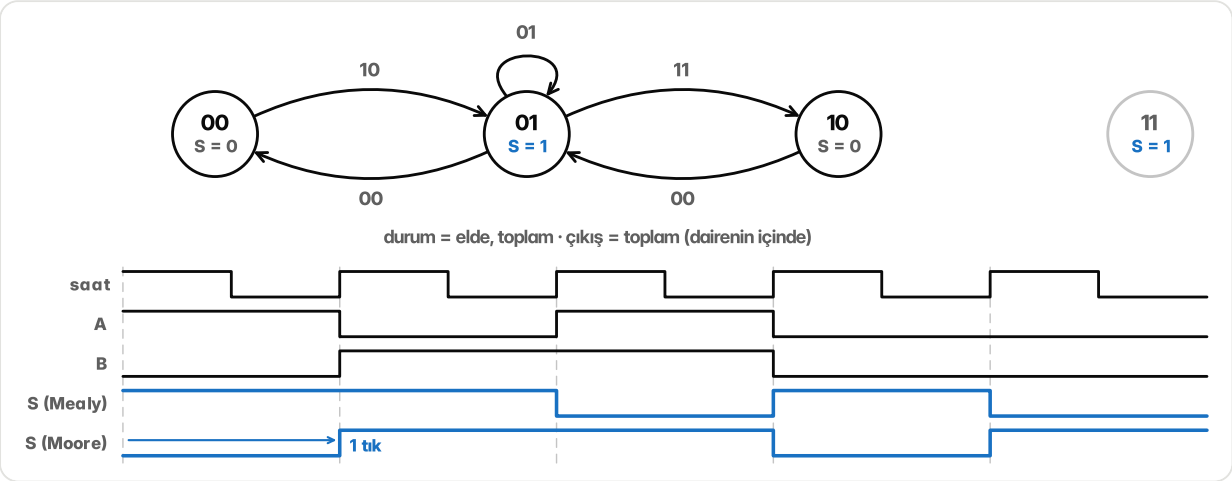
Çıkan bitler sondan başa okununca 1011, yani 11 olur.

Bu makinede çıkış yalnız duruma bağlı değildir: aynı durumda girişler değişince toplam biti de hemen değişir. Çıkışı duruma ve girişe birlikte bağlı makineye **Mealy makinesi** denir; yalnız duruma bağlı olana Moore makinesi. Ayrık matematikte ve hesaplama kuramında bu iki tür kâğıt üstünde, soyut makine olarak işlenir; burada ikisinin de flip-flop ve kapıdan devresi vardır.

Aynı toplayıcı Moore makinesi olarak da kurulabilir: toplam biti de durumun içine girer; elde ve toplam birlikte 4 durum eder, 2 flip-flop gerekir. Fark zaman çizelgesinde görünür. 5 + 6 için (5. tıkta girişler 0):

```
tık          1  2  3  4  5
S (Mealy)    1  1  0  1  0
S (Moore)    0  1  1  0  1
```

Mealy makinesinin toplam biti girişlerle aynı tıkta çıkar, Moore makinesininki bir tık sonra. Mealy daha az durumla hemen cevap verir; ama giriş tık ortasında oynarsa çıkış da oynar. Moore bir tık geç kalır; karşılığında çıkışı tık boyunca sabit durur.



Şekil 10.9 Moore toplayıcının 5 + 6 sırasında geçtiği durumlar ve zaman çizelgesi: Moore'un S çıkışı Mealy'ninkinden bir tık geride.

Hatırlanması gereken tek şey elde olduğu için 64 bitlik iki sayıyı toplamaya da bu tek flip-flop yeter; iş yalnız 64 tıkta biter.

Özet

- Sonlu durum makinesi geçmişten yalnız geleceği etkileyen kısmı tutar; her aşama bir durumdur ve durum diyagramında daireyle, geçişler okla çizilir.
- Tasarım adımları: sözel tarif, durum diyagramı, durum tablosu, durumlara kod, flip-flop giriş denklemleri (Karnaugh ile), devre.
- 4 durum 2 bitle, yani 2 flip-flopla kodlanır; kodların dağıtılışı kapı sayısını değiştirir (1, 0, 1 algılayıcısında 3, 5 ya da 10 kapı), her duruma bir flip-flop da bir seçenektir.
- Aynı okları ve aynı çıkışı olan iki durum tek durumda birleşir; durum sayısı böyle azaltılır.
- Moore makinesinde çıkış yalnız duruma, Mealy makinesinde duruma ve girişe bağlıdır; Mealy daha az durumla hemen cevap verir, Moore bir tık geç ama sabit çıkış verir.
- Analiz tasarımın tersidir: devreden denklem, denklemden tablo, tablodan diyagram; etiketsiz devre bu yolla tek flip-floplu seri toplayıcı çıktı.

Sorular

1 **Bir sonlu durum makinesinde durum neyi tutar?**

A Girişlerin bugüne kadarki bütün değerlerini

B Geçmişten yalnız geleceği etkileyen kısmı

C Yalnız son saat kenarındaki girişi

D Saat tıklarının sayısını

2 **4 durumlu trafik lambası makinesinin durumları ikilik kodlanıyor. En az kaç flip-flop gerekir?**

A 4 **B** 3

C 2 **D** 1

3 **Trafik lambası makinesinde kodlar halka boyunca 00, 01, 11, 10 sırasıyla veriliyor. Kod 11 iken süre doluyor (T = 1). Kenardan sonraki kodu yaz.**

Cevap:

4 **1, 0, 1 algılayıcısında "1" ve "101" durumlarının okları aynı yerlere gider. Bu iki durum neden tek durumda birleşmez?**

A Kodları farklıdır

B Okları farklı durumlardan gelir

C Birinden 3 ok çıkar

D Lambaları farklıdır

5 **1, 0, 1 algılayıcısında kodlar başlangıç 00, "1" 01, "10" 10, "101" 11 diye veriliyor. Sağdaki flip-flopun girişi $Q0^{+}$ hangisidir?**

A X **B** $\overline{X}$

C $Q1 \cdot Q0$ **D** $Q0 \cdot \overline{X}$

6 **1, 0, 1 algılayıcısı başlangıç durumundan, iç içe diziler de sayılarak 1, 0, 1, 0, 1, 0, 1 girişini alıyor. Lamba kaç kez yanar?**

Cevap:

7 **Seri toplayıcı bir kez Mealy, bir kez Moore makinesi olarak kuruluyor. Moore hâlinin toplam biti Mealy hâlininkine göre ne zaman çıkar?**

A Aynı tıkta **B** İki tık sonra

C Bir tık önce **D** Bir tık sonra

8 Seri toplayıcıda $D = A{\cdot}B + A{\cdot}Q + B{\cdot}Q$ ve $S = A \oplus B \oplus Q$. Flip-flop $Q = 1$ iken $A = 1$, $B = 0$ geliyor. S ile Q^{+}'yı bu sırayla iki bit olarak yaz.

Cevap: _____

SAYISAL TASARIM

11 Bellekler ve programlanabilir lojik

Bu bölüm belleğin dışarıdan nasıl göründüğünü, küçük çiplerden nasıl büyük bellek kurulduğunu ve bir bitin içeride nasıl saklandığını anlatır. Ardından belleğin bir doğruluk tablosu olarak kullanılmasından yola çıkarak ROM'a, programlanabilir dizilere ve aynı çipin her yüklemede başka bir devre olduğu FPGA'ya varır.

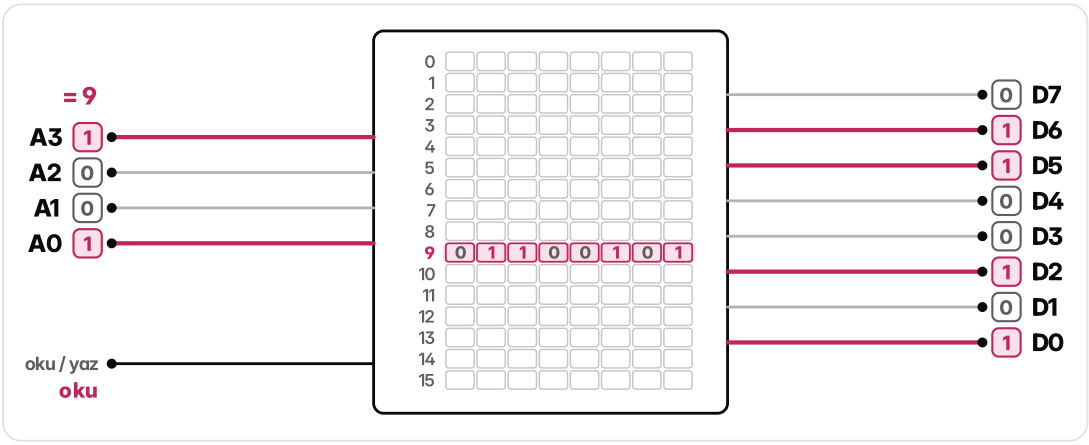
Hangi kutu?

videoda 59:19

Bir bellek çipinin içinde binlerce kutu vardır ve her kutuda bir sayı durur. İşlemci bunlardan yalnız birini ister. Hangisini istediğini çipe nasıl söyler, sayı ona hangi yoldan gelir?

Küçük bir çiple başlanabilir: 16 kutu, her kutuda 8 bit. Bir kutunun tuttuğu bitlere **kelime** denir; bu çipin kelimesi 8 bittir. Kutular 0'dan 15'e numaralanır ve kutunun numarasına **adres** denir. 16 numara 4 bitle yazılır, bu yüzden çipin 4 **adres teli** vardır. 8 bitlik kelime 8 **veri telinden** girip çıkar. Bir tel daha işin yönünü söyler: okuma mı, yazma mı. Böyle bir çipe "16 × 8" denir ve 16 · 8 = 128 bit tutar.

Yazmak için adres tellerine 1001, yani 9 konur; veri tellerine 0110 0101 konur ve yön teli yazmaya çevrilir. Kelime 9 numaralı kutuya girer. Aynı adres verilip yön okumaya çevrilince aynı 8 bit veri tellerinden geri gelir.

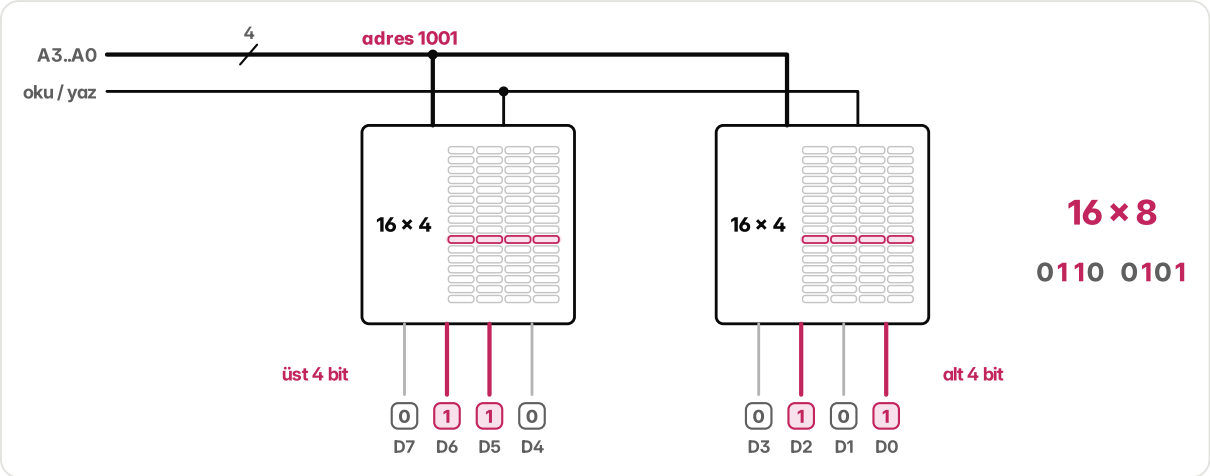


Şekil 11.1 16 × 8 belleğe 1001 adresi verilmiş; 9 numaralı satırda 0110 0101 duruyor ve okuma yönünde aynı bitler veri tellerinden çıkıyor.

Çip doğru kutuyu kendi içindeki bir kod çözücüyle bulur: 4 adres biti girer, 16 çıkıştan yalnız biri 1 olur ve o kutuyu açar. Kural geneldir: her yeni adres teli kutu sayısını iki katına çıkarır, k adres teli 2^k kutu seçer.

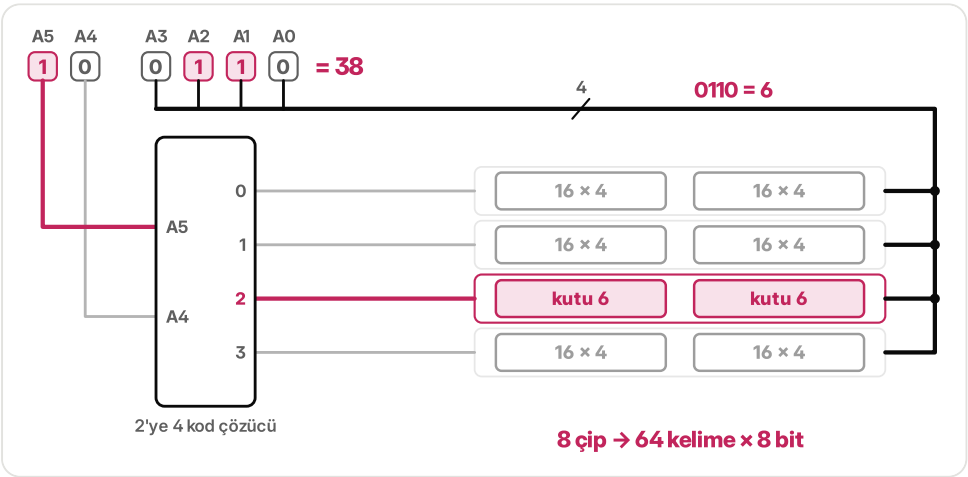
```
adres teli   kutu
    4         16
    5         32
    6         64
```

Şimdi elde yalnız 4 bitlik kelime tutan 16 × 4 çipler olsun; işlemci ise 8 bit istiyor. İki çip yan yana konur: adres telleri ikisine birden gider, yön teli de. Soldaki çip kelimenin üst 4 bitini, sağdaki alt 4 bitini tutar. İkisi her zaman aynı numaralı kutuyu açar ve 8 bit birlikte çıkar; 1001 adresinde soldan 0110, sağdan 0101 gelir. Böylece **kelime genişletilmiş** olur: 16 × 8.



Şekil 11.2 İki 16 × 4 çip yan yana; ortak adres ve yön telleri, soldan D7-D4, sağdan D3-D0 çıkıyor ve birlikte 0110 0101 kelimesini veriyor.

Kutu yetmezse bu ikililer üst üste dizilir: 4 sıra, 8 çip, 64 kutu. 64 numara 6 bit ister. Alttaki 4 bit bütün çiplere gider ve kutuyu seçer; üstteki 2 bit ise 2'ye 4 bir kod çözücüye girer ve 4 sıradan yalnız birini çalıştırır. Sıralar da 0'dan numaralanır. Adres 100110, yani 38 olsun: üstteki 10 sıra 2'yi seçer, alttaki 0110 o sıradaki 6 numaralı kutuyu. Sağlama: her sırada 16 kutu olduğuna göre 2 · 16 + 6 = 38.



Şekil 11.3 4 sıra × 2 çipten 64 × 8 bellek; 2'ye 4 çözücü 100110 adresinin üst iki bitiyle sıra 2'yi seçmiş, o sıradaki iki çipte 6 numaralı kutu açık.

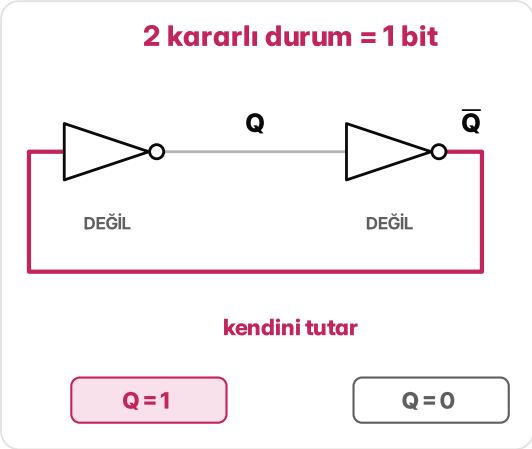
Bilgisayardaki RAM modülünde yan yana duran çipler de böyle çalışır: birçok modülde 8 bitlik 8 çip aynı adresi alır ve işlemciye 64 bitlik kelimeyi birlikte verir.

Kendini tutan bit

videoda 1:00:56

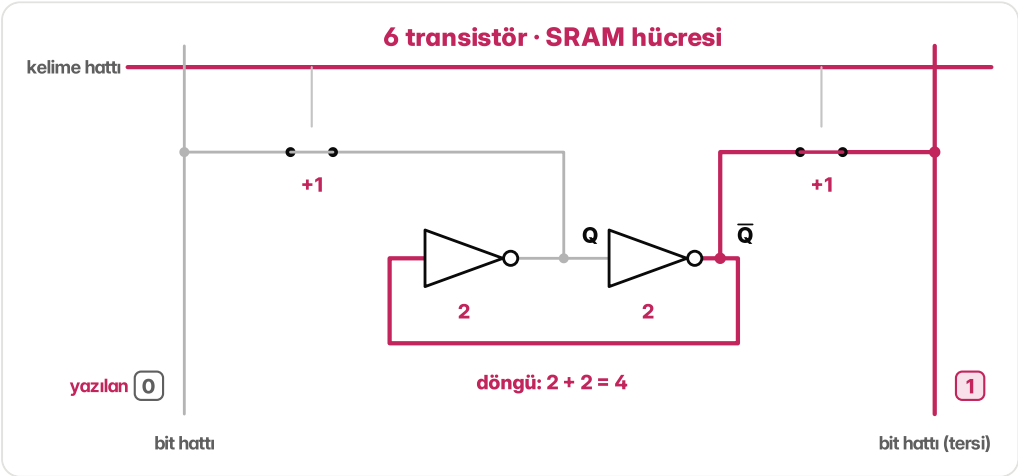
Bir bit nasıl saklanır: elektrik varken nasıl yerinde durur, elektrik gidince nasıl unutulmaz?

İki DEĞİL kapısı birbirine bağlanır: soldakinin çıkışı sağdakinin girişine, sağdakinin çıkışı soldakinin girişine. Soldakinin çıkışı 1 olsun. Sağdaki bunu tersine çevirip 0 verir, 0 soldakine döner, soldaki onu yine 1'e çevirir. Döngü kendi kendini besler ve bit yerinde kalır. Soldaki 0'la başlasa o da kalır. Devrenin iki **kararlı durumu** vardır, yani bir bit tutar. Soldaki çıkışa Q, sağdakine $\overline{Q}$ denir.



Şekil 11.4 Çapraz bağlı iki DEĞİL kapısı Q = 0 durumunda; alttaki geri besleme teli 1 taşıyor, altta iki kararlı durum Q = 1 ve Q = 0.

Her DEĞİL kapısı 2 transistörle kurulur; döngü 4 transistör eder. 2 transistör daha hücreyi dışarıya bağlar. Hücrenin satırı seçilince, yani **kelime hattı** 1 olunca, bu ikisi açılır ve hücrenin iki ucu iki **bit hattına** bağlanır. Okurken bit hatlarından hücrenin değeri ve tersi gelir. Yazarken bit hatları güçlü sürülür ve döngü yeni değere döner. Toplam 6 transistörlü bu hücre **SRAM** (statik RAM) hücresidir.



Şekil 11.5 Altı transistörlü SRAM hücresi: döngüdeki 4 transistör, kelime hattına bağlı 2 geçiş transistörü; bit hatlarından 0 ve 1 yazılınca hücre Q = 0'a dönmüş.

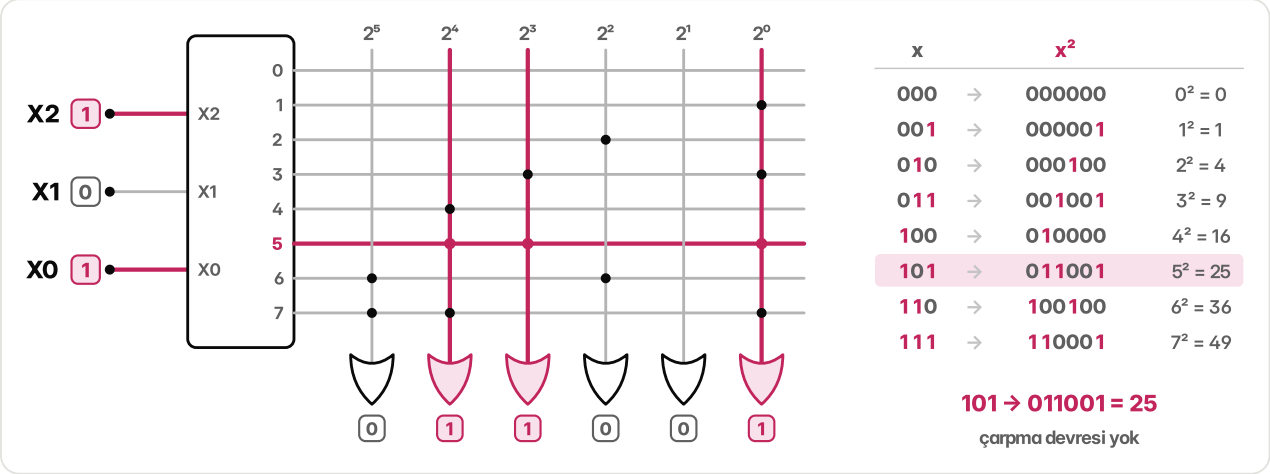
Bu döngü elektrik oldukça bekler, tazelenmez ve çok hızlıdır; işlemcinin önbelleği bu hücrelerden yapılır. Ana bellekteki **DRAM** (dinamik RAM) hücresi ise 1 transistör ve 1 kondansatörle yetinir: daha küçük ve ucuzdur, ama kondansatördeki yük sızdığı için sürekli tazelenir. İkisi de elektrik kesilince her şeyi unutur.

Bilgisayar açılırken çalışan ilk komutlar gibi hiç unutulmaması gereken bilgiler için **ROM** (salt okunur bellek) vardır: içeriği çip yapılırken bağlantılarla sabitlenir ve yalnız okunur.

Küçük bir ROM kurulabilir: 3 bitlik bir sayı girsin, karesi çıksın. En büyük kare $7^2 = 49$'dur, ikilikte 110001; 6 bit ister. ROM'un içinde önce bir kod çözücü durur: 3 adres biti 8 satırdan birini açar. Sonra her çıkış biti için bir VEYA kapısı gelir; o bitin 1 olduğu satırlar bu kapıya bağlanır, ötekiler bağlanmaz.

```
x     x²       ondalık
000   000000    0
001   000001    1
010   000100    4
011   001001    9
100   010000   16
101   011001   25
110   100100   36
111   110001   49
```

Örneğin 2^4 basamağının VEYA kapısına 4, 5 ve 7 numaralı satırlar bağlıdır, 2^1 basamağına hiçbir satır bağlı değildir: bir karenin 4'e bölümünden kalan hep 0 ya da 1 olduğu için o bit hiç 1 olmaz. 5, yani 101 verilince 5 numaralı satır açılır; ona bağlı üç sütun 1 verir, ötekiler 0: çıkış 011001, yani 25. Ortada çarpma devresi yoktur; sonucu tablo verir.



Şekil 11.6 3'e 8 çözücü ve altı VEYA sütunundan kare alan ROM; 101 girişinde 5 numaralı satır açık, 2^4, 2^3 ve 2^0 sütunları 1 veriyor ve yandaki tabloda 101 → 011001 satırı işaretli.

Çözücünün her çıkışı bir minterimdir, VEYA kapıları da minterimlerin toplamını kurar. Bu yüzden kaç girişli olursa olsun her fonksiyon bir ROM'a sığar: girişler adres olur, doğruluk tablosu içerik. Bedeli, her yeni girişin satır sayısını iki katına çıkarmasıdır; n girişli, m çıkışlı bir fonksiyon $2^n \times m$ bit ister. Kare alan ROM $8 \times 6 = 48$ bittir.

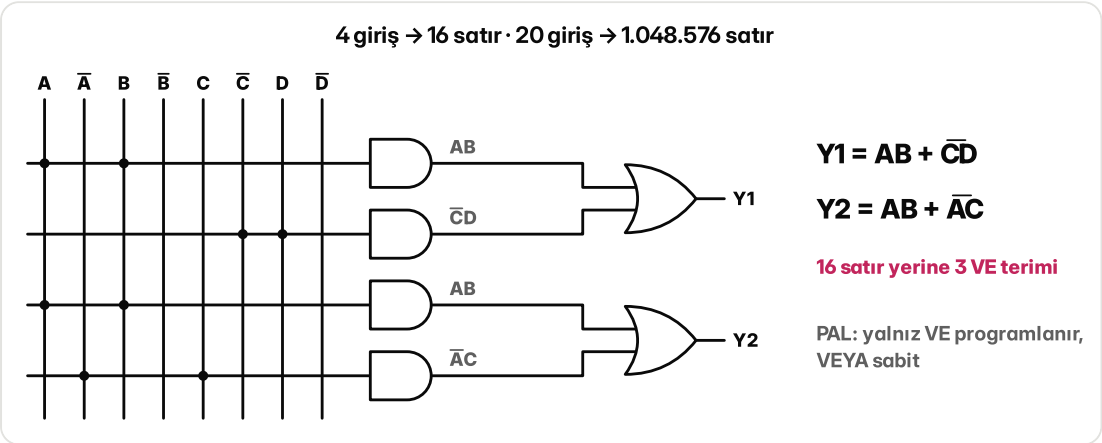
İçeriği fabrikada sabitlenen ROM'u zamanla yeni türler izledi: bir kez programlanabilen PROM, morötesi ışıkla silinip yeniden yazılabilen EPROM ve elektrikle silinebilen EEPROM. Telefonun ve bilgisayarın kalıcı belleği olan flaş da bu son aileden gelir.

Bir çip, birçok devre

videoda 1:02:47

Aynı çip bugün bir ses kartı, yarın bir işlemci olabilir mi? Çoğu çipin kapıları ve telleri fabrikada bir kez çizilir, bir daha değişmez. İpucu bellektedir: girişler adres, çıkışlar içerik yapılırsa bir bellek her doğruluk tablosunu tutar. İçerik değişince devre de değişir.

Ama bellekte her yeni giriş satır sayısını iki katına çıkarır: 4 giriş 16 satır, 20 giriş 1.048.576 satır ister. Oysa çoğu fonksiyon birkaç terimle yazılır. İki çıkışlı bir örnek: Y1 = AB + $\overline{C}$D ve Y2 = AB + $\overline{A}$C. 16 satır yerine 3 VE terimi yeter: AB, $\overline{C}$D ve $\overline{A}$C. **Programlanabilir VE-VEYA dizisi** (PLA) bunu yapar: girişler ve tersleri VE kapılarına, VE kapılarının çıkışları VEYA kapılarına ızgara gibi uzanır; hangi kesişmede bağlantı olacağına tasarımcı karar verir. AB terimi iki çıkışta ortak kullanılır. Daha ucuz türünde (PAL) yalnız VE tarafı programlanır, VEYA tarafı sabittir; her çıkışın kendi VE terimleri vardır, bu yüzden AB iki kez kurulur ve 4 terim gerekir.

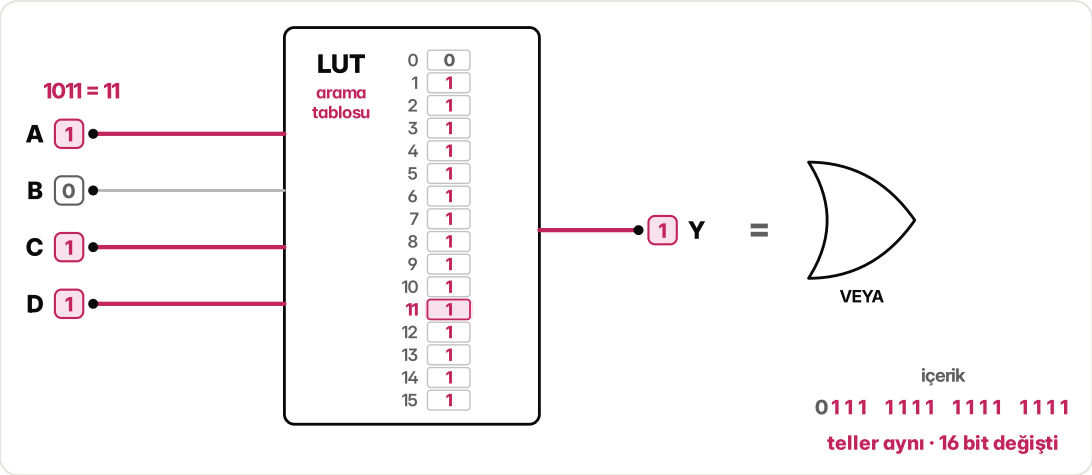


Şekil 11.7 VEYA tarafı sabit dizi (PAL): girişler ve tersleri ızgarada, dört VE terimi ve her çıkışın kendi VEYA kapısı; AB terimi iki kez kurulmuş.

FPGA çiplerinde ise ızgara yerine küçük bellekler vardır. 4 girişli bir kutu düşünülsün, içinde 16 bitlik bir bellek: 4 giriş bir adres seçer, o adresteki bit çıkışa geçer. Bu kutuya **arama tablosu** (LUT, lookup table) denir. İçine yalnız son biti 1 olan içerik yazılırsa kutu 4 girişli bir VE kapısı olur. Girişler 1011, yani 11 olsun: 11 numaralı bit 0, çıkış 0. İçerik "girişlerde tek sayıda 1 varsa 1" diye değiştirilirse aynı kutu ÖZEL VEYA olur; 1011'de 3 tane 1 vardır, çıkış 1. İlk bit 0, gerisi 1 yazılırsa kutu VEYA olur.

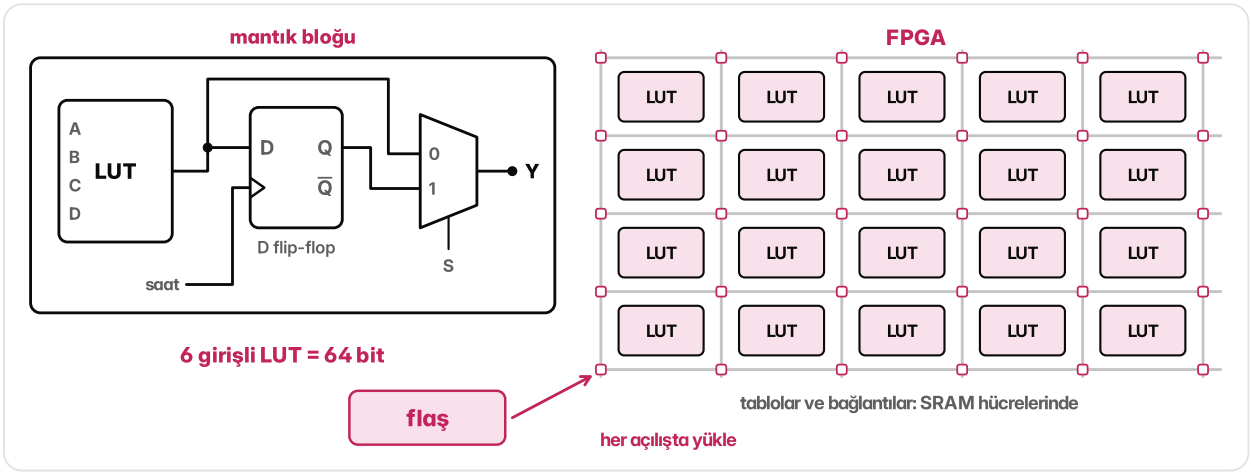
```
içerik (adres 0 → 15)    kapı        1011 için
0000 0000 0000 0001      VE              0
0110 1001 1001 0110      ÖZEL VEYA       1
0111 1111 1111 1111      VEYA            1
```

Hiçbir tel değişmez, yalnız 16 bit değişir.



Şekil 11.8 16 hücreli arama tablosu VEYA içeriğiyle; 1011 girişi 11 numaralı hücreyi seçiyor, çıkış 1 ve yanda eşdeğer VEYA kapısı.

Büyük bir FPGA çipinde bu kutulardan yüz binlercesi vardır; bugün çoğu 6 girişli, yani 64 bitliktir. Her kutunun yanında bir flip-flop durur; çıkış istenirse saatle saklanır. Kutu, flip-flop ve aralarından seçim yapan seçici birlikte bir **mantık bloğu** oluşturur. Blokların arasındaki bağlantılar da programlanır ve tablolarla birlikte hepsi SRAM hücrelerinde durur. SRAM elektrik kesilince unuttuğu için çip her açılışta içeriğini yanındaki flaş bellekten yükler ve o anda istenen devre olur.



Şekil 11.9 Solda tek mantık bloğu (LUT, D flip-flop, 2'ye 1 seçici); sağda programlanabilir bağlantılarla örülmüş blok ızgarası ve açılışta içeriği yükleyen flaş bellek.

Yüz binlerce tablo elle doldurulmaz: mühendisler devreyi bunun için yapılmış dillerde (Verilog, VHDL) yazıyla tarif eder, bir program da bu tarifi tablolara ve bağlantılara çevirir.

Bu yüzden aynı FPGA çipi bir gün ses işleyen bir devre, ertesi gün bir işlemci olabilir: çip aynı kalır, yüklenen içerik değişir. Yeni işlemci tasarımları da fabrikaya gitmeden önce çoğu zaman bu çiplerde kurulup denenir; bir hata çıkarsa yeni çip üretilmez, içerik düzeltilip yeniden yüklenir.

Özet

- Bellek kelimelerden oluşur; k adres teli 2^k kelime seçer, kelime kaç bitse o kadar veri teli vardır.
- Çipler yan yana konunca kelime genişler; üst üste dizilip üst adres bitleriyle bir kod çözücüden seçilince kelime sayısı artar.
- Çapraz bağlı iki DEĞİL kapısı bir bit tutar; 2 geçiş transistörüyle 6 transistörlü SRAM hücresi olur, DRAM hücresi 1 transistör ve 1 kondansatördür ve tazelenir.
- ROM bir doğruluk tablosudur: kod çözücü minterimleri, VEYA dizisi minterimlerin toplamını kurar; n girişli fonksiyon 2^n satır ister.
- PLA'da VE ve VEYA dizileri, PAL'da yalnız VE dizisi programlanır.
- FPGA, SRAM'de saklanan arama tabloları, flip-floplar ve programlanabilir bağlantılardan oluşur; aynı çip her yüklemede başka bir devre olur.

Sorular

1 **16 kelime × 8 bitlik bir bellek çipinde kaç adres teli vardır?**

A 3 **B** 4

C 8 **D** 16

2 **İki tane 16 × 4 çip yan yana konarak 16 × 8 bellek kuruluyor. Adres tellerine ne yapılır?**

A Üst iki bit bir kod çözücüye girer, hangi çipin çalışacağını seçer.

B Her çipe ayrı bir adres verilir.

C Adres telleri yalnız soldaki çipe gider, sağdaki onu izler.

D Aynı adres iki çipe birden gider.

3 **64 kelimelik bir bellek dört sıra 16 kelimelik çipten kurulmuş; adresin üst iki biti sırayı, alt dört biti kutuyu seçiyor. 011101 adresi, seçilen sırada kaç numaralı kutuyu açar? (Kutular 0'dan numaralı.)**

Cevap: _____

4 **DRAM hücresinin sürekli tazelenmesi neden gerekir?**

A Kondansatördeki yük zamanla sızar.

B İki DEĞİL kapısının döngüsü kendiliğinden ters döner.

C Altı transistör çok ısındığı için değer bozulur.

D Adres çözücü her okumadan sonra satırı unutur.

5 **3 bitlik sayının karesini veren ROM'a 110 verilirse 6 bitlik çıkış ne olur?**

Cevap:

6 **5 girişli, 3 çıkışlı bir fonksiyon ROM'a yazılacak. ROM'da kaç satır olur?**

A 5

B 15

C 32

D 243

7 **PAL ile PLA arasındaki fark nedir?**

A PAL'da VEYA dizisi programlanır, VE dizisi sabittir.

B PAL'da yalnız VE dizisi programlanır, VEYA dizisi sabittir.

C PAL bir bellektir, içinde arama tabloları vardır.

D PAL'da hiçbir bağlantı programlanmaz, PLA'da hepsi programlanır.

8 **4 girişli bir arama tablosu VE DEĞİL (NAND) kapısı gibi çalışacak. 0'dan 15'e adres sırasıyla 16 bitlik içerik ne olmalı?**

Cevap:

Cevaplar ve çözümler

0 Sayısal sistemler ve ikilik kodlar

1. **B** · Oluğun kıvrımı dalgayı kesintisiz izler; iki değer arasında sayısız ara değer vardır.
   Analog değer süreklidir: oluk dalgayı adım adım değil kesintisiz izler. Saniyede 44.100 ölçüm ve belirli adımlarla yazılan sayılar sayısal saklamanın özelliğidir; analog kopyalar ise her kopyada biraz daha cızırtı toplar.

2. **C** · Tanımsız bölgededir; kapı 0 da sayabilir, 1 de.
   1,2 V, 0,8 ile 2 V arasındaki tanımsız bölgededir. Kapının bu gerilimi hangi değer olarak okuyacağı garanti değildir; doğru çalışan bir devrede tel bu bölgede yalnız geçiş anında çok kısa kalır.

3. **0,4**
   Gürültü payı, girişin kabul ettiği sınır ile çıkışın en kötü değeri arasındaki farktır: 0,8 − 0,4 = 0,4 V. Tele binen gürültü 0,4 V'u aşmadıkça okunan bit değişmez.

4. **01010111**
   BCD'de her onluk basamak ayrı 4 bitle yazılır: 5 → 0101, 7 → 0111. Yan yana: 0101 0111. 57'nin ikilik değeri (111001) bununla karıştırılmamalı.

5. **D** · 111
   Yansıtmayla kurulan sıra 000, 001, 011, 010, 110, 111, 101, 100'dür; 5 numaralı konum 111'dir. 101 ikilik koddaki 5'tir, 110 4 numaralı konumun Gray kodudur.

6. **A** · Komşu konumlar tek bitle ayrıldığı için geçiş anında yanlış bir sayı okunmaz.
   İkilikte 3'ten 4'e (011 → 100) üç bit birden değişir; kontaklar aynı anda değişmediği için bir an 7 gibi yanlış bir değer okunabilir. Gray kodunda yalnız tek bit değiştiği için okunan değer ya eski ya yeni konumdur. Konum sayısı iki kodda aynıdır.

7. **1**
   Verideki birler sayılır: 3, tek sayı. Toplam çift olsun diye eşlik biti 1 olur; gönderilen çerçeve 11001001'dir ve içinde 4 tane 1 vardır.

8. **C** · Birlerin sayısı tek; bir hata yakalanır ve veri yeniden istenir.
   Çerçevede 5 tane 1 var, yani tek sayı; çift eşlikte bu bir hata demektir. Eşlik biti hatanın yerini söylemez. Birlerin sayısı tek çıktığına göre tek sayıda bit bozulmuştur; iki bit bozulsaydı sayı çift kalır ve hata görünmezdi.

1 Boole cebri

1. **B** · 1
   Boole cebrinde + VEYA işlemidir; girişlerden en az biri 1 ise sonuç 1'dir. 2 sıradan toplamanın, 10 ikilik toplamanın sonucudur.

2. **D** · 1 ve 0
   VE, VEYA'dan önce yapılır: BC = 0, A + 0 = 1. Parantezli ifadede önce A + B = 1 bulunur, sonra 1·C = 0.

3. **C** · $A + 1 = A$

   $A + 1$ her zaman 1'dir; $A = 0$ iken $A + 1 = 1$ olur ve A'ya eşit değildir. Öteki üçü $A = 0$ ve $A = 1$ için denenince doğru çıkar.

4. **A** · $A(B + C) = AB + AC$

   İkizde VE ile VEYA yer değiştirir: $A + BC$, $A(B + C)$ olur; $(A + B)(A + C)$ de $AB + AC$ olur. Bu, VE'nin VEYA üstüne dağılmasıdır.

5. **A+B**

   VEYA'nın VE üstüne dağılmasıyla $A + \overline{A}B = (A + \overline{A})(A + B)$ olur. $A + \overline{A} = 1$ olduğu için geriye $A + B$ kalır.

6. **D** · $\overline{A} + \overline{B}$

   De Morgan kuralı: çizgi kırılır, VE VEYA'ya döner. $\overline{A}\cdot\overline{B}$ yanlıştır: $A = 1$, $B = 0$ iken AB'nin tümleyeni 1, $\overline{A}\cdot\overline{B}$ ise 0'dır.

7. **C** · BC

   $BC = 1$ iken $A = 1$ ise AB, $A = 0$ ise $\overline{A}C$ zaten 1'dir; BC hiçbir satırı tek başına 1 yapmaz. Bu uzlaşma kuralıdır; AB ya da $\overline{A}C$ silinirse bazı satırlar 0'a düşer.

8. **00011111**

   $A = 0$ olan ilk dört satırda $Y = BC$: 0, 0, 0, 1. $A = 1$ olan son dört satırda $Y = 1$. Sütun 00011111.

2 Lojik fonksiyonlar

1. **C** · 16

   İki girişin tablosu 4 satırdır ve her satırda çıkış 0 ya da 1 olabilir: $2^4 = 16$ farklı tablo. 4 satır sayısıdır; 256 ise 3 girişin fonksiyon sayısıdır.

2. **D** · Tam bir satırda

   Maksterim bir VEYA terimidir ve VEYA yalnız bütün girişleri 0 olunca 0 verir. Terim, o satırda 1 olan değişkenin tersiyle kurulduğu için yalnız kendi satırında bütün parçaları 0 olur; öteki satırların hepsinde 1'dir.

3. **B** · $\overline{A}BC$

   Minterim bir VE terimidir; yalnız bütün parçaları 1 olunca 1 verir. Bu satırda 0 olan A'nın tersi alınır: $\overline{A}BC$. $A + \overline{B} + \overline{C}$ aynı satırın maksterimidir, minterimi değil.

4. **1,4,5,6**

   Maksterim listesi, 0'dan 7'ye numaralardan minterim listesinde olmayanlardır. 0, 2, 3 ve 7 çıkarılınca 1, 4, 5, 6 kalır: ΠM(1, 4, 5, 6).

5. **C** · BC

   İki satırda da B ve C 1'dir, yalnız A farklıdır. A ne olursa olsun terim 1 verdiği için A düşer: $\overline{A}BC + ABC = BC$.

6. **001**

   $AB = 0 \cdot 1 = 0$, $AC = 0 \cdot 1 = 0$, $BC = 1 \cdot 1 = 1$. Çıkışlar 0, 0, 1 olur; VEYA kapısına bir 1 geldiği için $Y = 1$, ışık yanar.

7. **A** · 4

   Her terim için 2 girişli bir VE, toplam 3 VE; bunların çıkışlarını bağlayan 3 girişli 1 VEYA. $3 + 1 = 4$ kapı. Yalnız VE'leri sayınca 3 bulunur; VEYA kapısını unutma.

8. D · $\overline{AB} + \overline{AC} + \overline{BC}$

Tümleyen, en az ikisi hayır (0) dediğinde 1 olur: $\overline{AB} + \overline{AC} + \overline{BC}$; minterimleri 0, 1, 2, 4. $\overline{ABC}$ yalnız üçü de hayır derse 1 olur. $(A + B)(A + C)(B + C)$ ise fonksiyonun kendisidir, tümleyeni değil.

3 Lojik kapılar

1. B · VE DEĞİL

VE DEĞİL, VE kapısının ucuna balon konmuş hâlidir: VE yalnız bütün girişler 1 iken 1 verir, balon bunu 0'a çevirir. VEYA DEĞİL ise en az bir giriş 1 olunca 0 verir.

2. C · Üç girişi de 0 iken

VEYA, girişlerden biri bile 1 olunca 1 verir; balon bunu tersine çevirir. Bu yüzden VEYA DEĞİL'in çıkışı yalnız hiçbir giriş 1 değilken, yani 000 birleşiminde 1 olur.

3. 011

A = 1 olduğu için $\overline{A} = 0$, B = 0 olduğu için $\overline{B} = 1$. VE DEĞİL'in girişlerinden biri 0 olduğu için çıkış 1 olur. Sonuç 011; tıpkı VEYA gibi, A veya B 1 iken çıkış 1.

4. D · 6

Tek katlı CMOS kapısı tersler, bu yüzden VE doğrudan kurulamaz; VE DEĞİL (4 transistör) ile DEĞİL (2 transistör) art arda bağlanır: 4 + 2 = 6. 4, VE DEĞİL'in kendi sayısıdır.

5. A · Altta, sırayla

1'de iletenler çıkışı 0'a (toprağa) bağlar ve altta durur. Çıkış ancak iki giriş de 1 olunca 0 olmalı; bu yüzden ikisi sırayla bağlanır. Üstteki 0'da iletenler ise yan yana durur, biri bile iletince çıkış 1 olur.

6. 101

Üstteki VE DEĞİL: A·B = 0, çıkış 1. Alttaki VE DEĞİL: C·D = 1, çıkış 0. Sondaki VE DEĞİL'in girişlerinden biri 0 olduğu için Y = 1. Sağlama: AB + CD = 0 + 1 = 1.

7. 1

Verideki 1'leri say: 1 + 0 + 0 + 1 + 0 + 1 + 1 + 1 = 5, yani tek. Çok girişli ÖZEL VEYA bu durumda 1 verir; bu 1 eklenince 1'lerin sayısı 6, çift olur.

8. B · 4

Ağaçta her katta kapı sayısı yarıya iner: 8, 4, 2, 1; toplam 15 kapı, 4 kat. Her bit her kattan bir kapı geçer, yani 4 kapı. 15, aynı kapıların zincir düzenindeki en uzun yoludur.

4 Karnaugh haritaları

1. B · 00, 01, 11, 10

Yan yana her iki sütun tek bitle ayrılmalıdır: 00 → 01 → 11 → 10, ve 10'dan 00'a dönüş de tek bit. Sayma sırası olan 00, 01, 10, 11'de 01'den 10'a iki bit birden değişir.

2. C · 1, 2, 4 ya da 8

Çerçevede bazı değişkenler sabit kalır, ötekiler bütün değerlerini alır; k değişken serbestse kutu sayısı 2'nin k. kuvveti olur. Bu yüzden 3 ya da 6 kutuluk çerçeve olmaz.

3. D · İki bitte ayrıldıkları için

İki minterim yalnız bir değişkende ayrılıyorsa birleşir ve o değişken düşer. 101 ile 110 hem B'de hem C'de ayrılır. Tabloda komşu olmaları yetmez; tablo bu yüzden komşuları saklar.

4. A · $\overline{BD}$

Köşeler 0 (0000), 2 (0010), 8 (1000) ve 10 (1010) numaralı kutulardır. Dördünde de B = 0 ve D = 0, A ile C değişir. Terim, değişmeyenlerin VE'si: $\overline{BD}$.

5. C · Fonksiyonun tersi

Sıfır grupları, fonksiyonun 0 olduğu satırları kapsar; bu yüzden fonksiyonun tersini verir. Tersi De Morgan kuralıyla çevrilince toplamların çarpımı biçimi çıkar.

6. B · Bir 1'i kapsayan tek asal içeren

Asal içeren büyütülemeyen çerçevedir. Bir 1'i başka hiçbir asal içeren kapsamıyorsa o çerçeve formüle mutlaka girer; buna zorunlu asal içeren denir. Büyüklüğü ya da kenardan sarılması bunu belirlemez.

7. AC+BC

Haritada 3 (011) ile 7 (111) yalnız A'da ayrılır: terim BC. 5 (101) ile 7 (111) yalnız B'de ayrılır: terim AC. 7 iki çerçeveye birden girer. Sonuç AC + BC.

8. A

1'ler 8 ve 9, X'ler 10-15. Haritanın alttaki iki satırı (8-15) 1 ya da X'lerden oluşur; X'ler 1 sayılınca 8 kutuluk tek çerçeve olur ve içinde yalnız A sabittir. Sonuç Y = A.

5 Toplama ve çıkarma devreleri

1. C · Çıkışı yalnız o anki girişlere bağlıdır; aynı girişe her zaman aynı çıkışı verir.

Kombinezonsal devrenin hafızası yoktur: çıkış yalnız o anki girişlerden hesaplanır, bu yüzden aynı iki sayı her zaman aynı toplamı verir. Önceki değerleri hatırlayan devreler başka bir sınıftır; kombinezonsal devre her türlü kapıyla (ör. ÖZEL VEYA) kurulabilir.

2. B · Sağdaki basamaktan gelen eldeyi alacak girişi olmadığı için.

Yarım toplayıcının yalnız iki girişi (A, B) ve iki çıkışı (S, C) vardır. 1 + 1'i doğru toplar (S = 0, C = 1) ama çok bitli toplamada sağdan gelen eldeyi alamaz; bunu tam toplayıcı yapar.

3. D · $S = 0$, $C_o = 1$

1 + 0 + 1 = 2, ikilikte 10: toplam basamağına 0, eldeye 1 yazılır. Kuralla da aynı: girişlerde iki tane 1 var; sayı çift olduğu için $S = 0$, en az iki giriş 1 olduğu için $C_o = 1$.

4. A · Eldeyi geçirir: sağdan elde gelirse bu basamaktan da elde çıkar.

$G_i = A_iB_i = 0$, $P_i = A_i \oplus B_i = 1$. Elde üretmek için iki bitin ikisi de 1 olmalıdır; yalnız biri 1 olan basamak eldeyi geçirir: 0 + 1 + 1 = 10 olduğundan gelen elde varsa giden elde de vardır.

5. 32

Elde 16 basamağın her birinde 2 kapıdan geçer: 16 × 2 = 32 kapı gecikmesi. Aynı hesap 4 bitte 8, 64 bitte 128 verir.

6. 0011

M = 1 olduğu için B'nin bitleri ÖZEL VEYA'lardan ters çevrilmiş çıkar (1101) ve en sağa $C_o = 1$ girer. 0101 + 1101 + 1 = 10011; dışarı çıkan elde atılınca 0011 kalır, yani 5 − 2 = 3.

7. C · 0101 + 0100

0101 + 0100 = 1001: iki pozitif sayının (5 ve 4) toplamı işaretli okununca −7 çıkar, çünkü 9 dört bite sığmaz. En sol basamağa elde girer ($C_3 = 1$) ama çıkmaz ($C_4 = 0$), $V = C_3 \oplus C_4 = 1$. Öteki toplamlar 5, 3 ve −2 eder; hepsi −8 … +7 aralığındadır.

8. **0101**

   1000 + 0111 = 1111, yani 15; 9'u geçtiği için 0110 eklenir: 1111 + 0110 = 10101. Soldaki 1 onluk eldedir, rakam 0101 = 5 kalır; sonuç 15.

6 Seçiciler, kod çözücüler, kodlayıcılar

1. **C** · 8

   n girişli kod çözücünün 2^n çıkışı vardır; her giriş birleşimine bir çıkış düşer. 3 giriş için $2^3 = 8$. 6 ve 9, girişle çıkış sayısının toplanması ya da karesinin alınmasıyla bulunan yanlış cevaplardır.

2. **B** · $Y1 = \overline{A}B$

   Adres 01, yani 1'dir; 1 olan çıkış Y1'dir. Her çıkış yalnız kendi satırında 1 veren bir minterimdir: A = 0 olduğu için A'nın tersi, B = 1 olduğu için B kendisi alınır: $\overline{A}B$.

3. **5**

   A = 1 olduğu için yalnız alttaki çözücü çalışır. BC = 01 olduğu için onun 1 numaralı hattı 1 olur. Alttaki çözücünün hatları 4'ten başladığı için bu, 4 + 1 = 5 numaralı hattır; adres de 101 = 5'tir.

4. **0011**

   1'in kodu 0001, 2'nin kodu 0010'dur. VEYA kapıları ikisinin 1'lerini birleştirir: 0011. Bu, kimsenin basmadığı 3 rakamının kodudur.

5. **D** · 0010

   Öncelikli kodlayıcıda her tuş, kendisinden büyük tuşların hiçbiri basılı değilse sayılır. 2 basılı olduğu için 1 susar ve yalnız 2'nin kodu çıkar: 0010. 0011, öncelik olmadan VEYA'nın verdiği yanlış koddur.

6. **B** · Çubuğun söndüğü tek rakam 2'dir (BCD = 010); aynı üç biti taşıyan öteki kod 10'dur ve bu kod girişe hiç gelmez

   Çubuk 0-9 arasında yalnız 2'de (0010) söner. B = 0, C = 1, D = 0 olan öteki kod 1010, yani 10'dur; rakam 9'da bittiği için 10-15 kodları önemsiz durumdur. Bu yüzden söndürme koşulu $\overline{B}C\overline{D}$'ye, yanma koşulu De Morgan ile $B + \overline{C} + D$'ye iner. A biti 8 ve 9'da 1'dir; ilk seçenek bu yüzden yanlıştır.

7. **C'**

   AB = 11 parçası 6. ve 7. satırlardır: C = 0 iken Y = 1, C = 1 iken Y = 0. Çıkış C'nin tersi olduğu için 3 numaralı girişe $\overline{C}$ (bir DEĞİL kapısından geçmiş C) bağlanır.

8. **A** · 0010

   Dağıtıcı tek girişi yalnız seçme bitlerinin gösterdiği çıkışa verir; 10, 2 numaralı çıkıştır. Öteki çıkışlar 0'da kalır: 0, 0, 1, 0. 0100 sıranın tersten okunmasıyla, 1101 ise seçilmeyen çıkışların 1 sanılmasıyla bulunur.

7 ALU tasarımı

1. **B** · VE, VEYA, ÖZEL VEYA ve toplamdan, seçme tellerinin gösterdiğini çıkışa geçirir.

   Dilimdeki dört iş her an birlikte yapılır; seçici yalnız S_1S_0'ın gösterdiği sonucu çıkışa geçirir. Hiçbir kapı beklemez ve ALU kombinezonsal olduğu için bir şey saklamaz; eldeyi taşıyan tam toplayıcıdır.

2. **C** · 0001

   $S_1S_0 = 00$ VE'yi seçer; bitler tek tek VE'lenir: 1001 VE 0011 = 0001. 1011 VEYA'nın, 1010 ÖZEL VEYA'nın, 1100 toplamın sonucudur.

3. D · A − 1

B önce 0000 olur, ters çevrilince 1111 olur; 1111 ikinin tümleyeninde −1'dir. İlk elde 0 olduğu için toplayıcı A + (−1), yani A − 1 hesaplar. A + 1 için B sıfırlanır ama ters çevrilmez ve C_o = 1 verilir.

4. 0001

B ters çevrilir: 1010; ilk elde 1 eklenir. 0110 + 1010 + 1 = 10001; dört bit 0001, en soldan elde çıkar. Sağlama: 6 − 5 = 1.

5. 1010

A + A, A'nın iki katıdır; bitler bir sola kayar: 0101 + 0101 = 1010. Sağlama: 5 + 5 = 10.

6. A · Dört çıkış bitinin VEYA DEĞİL'i alınır.

VEYA DEĞİL yalnız bütün girişleri 0 iken 1 verir; bu da sonucun sıfır olmasıdır. Dört bitin VE'si yalnız sonuç 1111 iken 1 olur; eldelerin ÖZEL VEYA'sı taşma bayrağı, en soldaki bit işaret bayrağıdır.

7. C · C = 0, yani A < B

0010 + 1000 + 1 = 1011; en soldan elde çıkmaz, C = 0. Çıkarmada elde çıkmaması ödünç alındığı demektir, yani A < B (2 < 7). Eşitliği elde bayrağı değil sıfır bayrağı söyler.

8. B · N = 1, V = 1; A < B değil

6 − (−6) = 12 dört bitlik işaretli sayıya sığmaz (en çok 7), bu yüzden V = 1; sonuç 1100'ün en soldaki biti 1, N = 1. N $\oplus$ V = 0 olduğu için A < B değildir: 6, −6'dan büyüktür. Yalnız işaret bayrağına bakmak burada yanlış karar verir.

8 Mandallar ve flip-floplar

1. B · 1, çünkü iki kapı birbirini bu değerde tutar

S = 1 iken $\overline{Q}$ 0'a, ardından Q 1'e geçer. S 0'a dönünce alttaki kapıya Q'dan 1 gelmeye devam ettiği için $\overline{Q}$ 0, üstteki kapının iki girişi 0 olduğu için Q 1 kalır. S ve R ikisi de 0 iken mandal son değeri tutar; çıkışın S ile birlikte geri dönmesi kombinezonsal devrenin davranışıdır.

2. D · S = 1, R = 1

S = R = 1 iken iki VEYA DEĞİL kapısının da çıkışı 0 olur, Q ile $\overline{Q}$ birbirinin tersi olmaktan çıkar. İkisi aynı anda 0'a dönünce hangi kapının önce davranacağı belli olmadığından son değer de belli olmaz. S = R = 0 ise mandalın son değeri tuttuğu olağan durumdur.

3. A · Her değişikliği hemen izler

İzin 1 iken D, VE kapılarından geçip mandalı kurar ya da sıfırlar; çıkış girişi olduğu gibi izler. Mandalın bu açık pencere gibi davranışına saydamlık denir. Yalnız kenarda yazmak flip-flopun özelliğidir, mandalın değil.

4. 500

Frekans = 1 / periyot = 1 / ($2 \cdot 10^{-9}$ s) = 500.000.000 tur/saniye. Bu da 500 MHz eder (0,5 GHz).

5. C · Yalnız yükselen kenarda

Saat 0 iken efendi girişi alır ama köle kapalıdır, çıkış değişmez. Saat yükselince efendi kapanır ve kenardan hemen önceki değeri saklar, köle açılıp bu değeri çıkışa verir. Saat 1 iken D değişse bile efendi kapalı olduğu için çıkışa ulaşmaz; düşen kenarda yalnız efendi yeniden açılır.

6. B · Kenardan önce girişin sabitlenmiş olması gereken süre

Flip-flop kenar anını kısa bir aralıkta görür: giriş kenardan biraz önce sabit olmalıdır (kurulum süresi), kenardan sonra da biraz daha sabit kalmalıdır (tutma süresi). Giriş bu aralıkta değişirse yarı kararlılık olabilir. Saatin bir tam turu periyottur.

7. 1001

J = 1, K = 0 kurar: Q = 1. İkisi birden 1 olunca tersine döner: Q = 0. İkisi 0 iken aynı kalır: Q = 0. Yeniden ikisi birden 1: Q = 1. Sonuç 1001; $Q^+ = J\overline{Q} + \overline{K}Q$ denklemiyle de aynı çıkar.

8. 0010

$Q^+ = T \oplus Q$: T = 1 iken çıkış tersine döner, T = 0 iken aynı kalır. $1 \oplus 1 = 0$, $0 \oplus 0 = 0$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$. Sonuç 0010.

9 Yazmaçlar ve sayıcılar

1. B · Saat girişleri aynı tele bağlıdır

Yazmaç, saat girişleri ortak D flip-floplardır; böylece her kenarda 4 bit birlikte yazılır. D girişleri ortak olsaydı 4 flip-flop hep aynı biti tutardı. Birinin ötekini tetiklemesi dalgalı sayıcının özelliğidir.

2. C · Her flip-flop kendi çıkışını yeniden yazar, değer değişmez

İzin 0 iken her flip-flopun önündeki seçici flip-flopun kendi çıkışını geçirir; kenarda aynı değer yeniden yazılır. Saat hiç durdurulmaz: saat teline kapı koymak kenarı geciktirdiği için seçim girişte yapılır.

3. A · 15

Her T flip-flop frekansı 2'ye böler. $32.768 = 2^{15}$ olduğu için 15 kez ikiye bölmek gerekir: 32.768, 16.384, 8.192 … ve 15. flip-flopun çıkışında 1 Hz.

4. 1101

Kenar kenar: 1000, 0100, 1010, 1101. Her kenarda bitler bir sağa kayar ve yeni bit Q3'e girer; ilk giren 1 dördüncü kenarda Q0'a ulaşır.

5. C · 14, 12, 8

Önce en sağdaki bit 0'a düşer: 1110 = 14. Bu düşüş ikinci biti çevirir: 1100 = 12, sonra üçüncüyü: 1000 = 8; en son en soldaki bit 0'a iner ve sayı 0 olur. Doğrudan geçiş zaman uyumlu sayıcıda olur.

6. D · Q2·Q0

5 ikilikte 101'dir. 0-5 arasında Q2 ile Q0'ın birlikte 1 olduğu tek sayı 5'tir. Q2·Q1 6'yı (110) yakalar ve sayıcı 6'yı da gösterirdi; Q2·Q1·Q0 ise 7'de 1 olur, 0-5 arasında hiç 1 olmaz.

7. 0111

Q3 Q2 Q1 Q0 = 1 0 1 1. T0 = 1, T1 = Q0 = 1, T2 = 1·1 = 1, T3 = 1·1·0 = 0. Kenarda yalnız sağdaki 3 bit döner ve sayıcı 1100 (12) olur.

8. 1111

En sağdaki bit 0'dır; DEĞİL kapısından 1 olarak en sola girer, öteki bitler bir sağa kayar: 1111. Dizi 0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001 diye 8 durumdan geçer.

10 Sonlu durum makineleri

1. B · Geçmişten yalnız geleceği etkileyen kısmı

Trafik lambasının bütün geçmişi bilmesi gerekmez; sıradaki rengi seçmek için hangi aşamada olduğunu bilmesi yeter. Geçmişin bu kısmına durum denir. Saniyeleri ise makinenin yanındaki sayıcı sayar.

2. C · 2

2 bit 00, 01, 10, 11 diye 4 farklı kod verir ve her bit bir flip-floptur. 4 flip-flop ancak her duruma ayrı bir flip-flop verilirse gerekir.

3. 10

Halkada 11'den sonra 10 gelir. Kuraldan da okunur: yeni soldaki bit şimdiki sağdaki bittir (1), yeni sağdaki bit şimdiki soldaki bitin tersidir (0).

4. D · Lambaları farklıdır

İki durumun birleşmesi için hem okları hem çıkışları aynı olmalıdır. "101" durumunda lamba yanar (Y = 1), "1" durumunda yanmaz (Y = 0). Kodlar birleştirmeden sonra verilir; kararı etkilemez.

5. A · X

Kod tablosunda $Q0^+$ sütunu 8 satırın hepsinde X ile aynıdır: sağdaki bit son gelen biti tutar ve girişi düz bir teldir. $Q1 \cdot Q0$ lambanın (Y) ifadesi, $Q0 \cdot \overline{X}$ ise $Q1^+$'nın terimlerinden biridir.

6. 3

1-3., 3-5. ve 5-7. bitler 1, 0, 1'dir; diziler iç içe sayıldığı için ortak bitler yeniden kullanılır. Durumlar sırayla 1, 10, 101, 10, 101, 10, 101 olur; lambalı duruma 3 kez gelinir.

7. D · Bir tık sonra

Moore makinesinde toplam biti durumun içindedir ve durum ancak saat kenarında değişir; çıkış bu yüzden bir tık geç görünür. Mealy'de çıkış girişlere de bağlı olduğu için aynı tıkta çıkar. 5 + 6'da Mealy 1, 1, 0, 1, 0; Moore 0, 1, 1, 0, 1 verir.

8. 01

S: 1, 0 ve 1 arasında iki tane 1 var; sayı çift, S = 0. D: $A \cdot Q = 1$ olduğu için en az ikisi 1, $Q^+ = 1$. Toplama olarak da aynı: 1 + 0 + elde 1 = ikilikte 10; toplam biti 0, yeni elde 1.

11 Bellekler ve programlanabilir lojik

1. B · 4

Adres telleri kutunun numarasını taşır. 16 numara (0-15) 4 bitle yazılır, çünkü $2^4 = 16$. 8, veri tellerinin sayısıdır; 16 ise kutu sayısıdır, tel sayısı değil.

2. D · Aynı adres iki çipe birden gider.

Kelime genişletilirken iki çip her zaman aynı numaralı kutuyu açmalıdır; bu yüzden adres ve yön telleri ikisine birden gider. Soldaki kelimenin üst 4 bitini, sağdaki alt 4 bitini verir. Kod çözücüyle seçme, kutu sayısını artırmak için çipler üst üste dizildiğinde gerekir.

3. 13

Alttaki dört bit 1101'dir: 8 + 4 + 1 = 13. Üstteki 01 ise 1 numaralı sırayı seçer. Sağlama: 011101 = 29 ve 1 · 16 + 13 = 29.

4. **A** · Kondansatördeki yük zamanla sızar.

   DRAM hücresi biti 1 transistör ve 1 kondansatörle tutar; kondansatördeki yük sızdığı için değer kaybolmadan yeniden yazılmalıdır. İki DEĞİL kapısının döngüsü ve altı transistör SRAM hücresine aittir; o döngü elektrik oldukça kendini tutar ve tazelenmez.

5. **100100**

   110 = 6 ve $6^2 = 36$. 36 = 32 + 4, 6 bitle 100100. ROM'da 6 numaralı satır 2^5 ve 2^2 sütunlarının VEYA kapılarına bağlıdır, öteki sütunlar 0 verir.

6. **C** · 32

   Girişler ROM'un adresi olur; 5 adres biti $2^5 = 32$ satır seçer. Çıkış sayısı satır sayısını değil, her satırın bit sayısını belirler: ROM 32 × 3 = 96 bittir.

7. **B** · PAL'da yalnız VE dizisi programlanır, VEYA dizisi sabittir.

   PLA'da hem VE hem VEYA dizisi programlanır; bu yüzden bir VE terimi (örneğin AB) birden çok çıkışta ortak kullanılabilir. PAL'da VEYA tarafı sabittir, her çıkışın kendi VE terimleri vardır. Arama tablosu FPGA'nın yapı taşıdır.

8. **1111111111111110**

   VE DEĞİL yalnız dört giriş de 1 olunca 0 verir; bu yalnız 1111, yani 15 numaralı adrestir. Öteki 15 adreste çıkış 1'dir: 1111 1111 1111 1110. Bu, VE içeriğinin (0000 0000 0000 0001) her bitinin tersidir.

Öğrendiğini pekiştir

Bu notlardaki bütün soruları ve daha fazlasını sitede, her cevabın hemen ardından çözümüyle birlikte çözebilirsin.

hekzon.com/dersler

HEKZON