Mantıksal Sistem Tasarımı – BLM 201

Hafta 11: Optimizasyonlar ve Ödünleşmeler

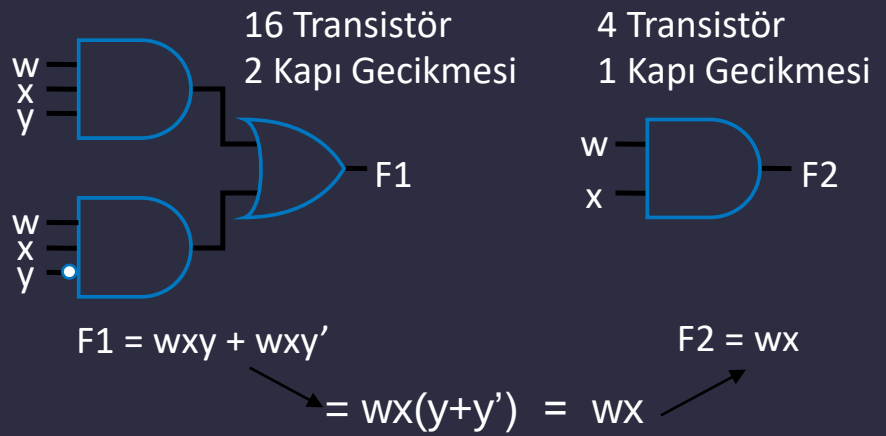
Fenerbahçe Üniversitesi

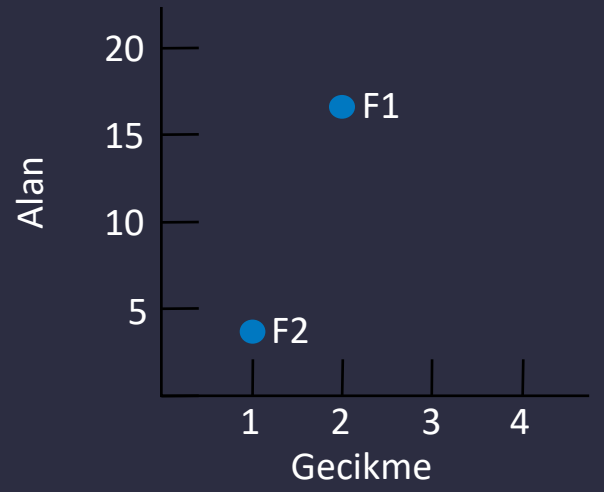
Ders Planı

- Optimizasyonlar ve Ödünleşmeler

Optimizasyonlar ve Ödünleşmeler

- Dijital devreleri nasıl kuracağımızı biliyoruz
  - Daha iyi devreler nasıl kurulur?
- İki önemli temel kısıt
  - ***Gecikme*** (Latency) – Girdi verildikten sonra çıktı elde edilene kadar geçen süre
  - ***Alan*** (Area) – Kullanılan transistör sayısı

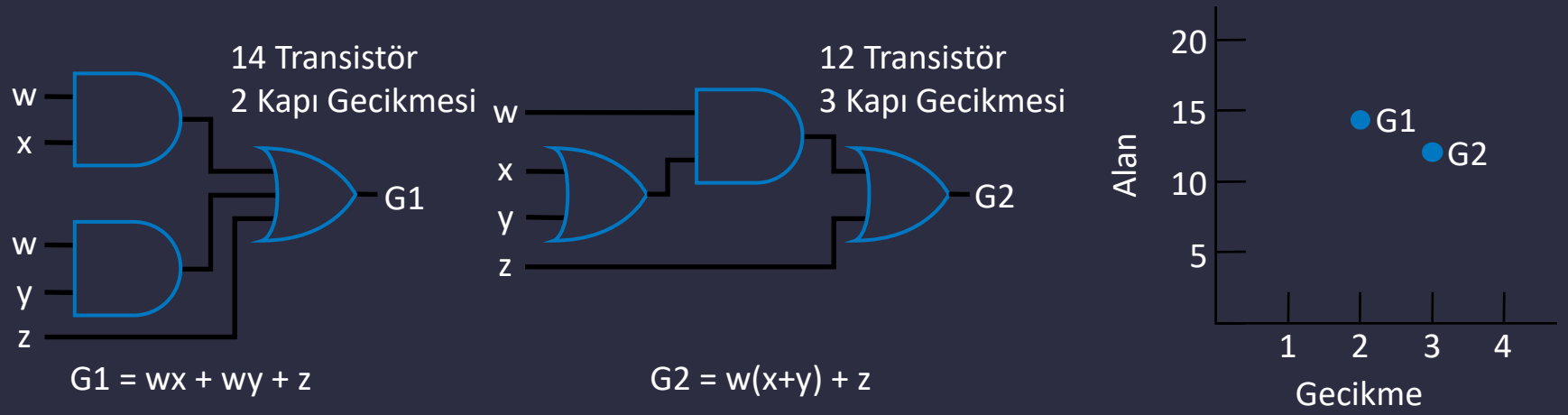




Not: Kapı gecikmeleri eşdeğer varsayınız

Optimizasyonlar ve Ödünleşmeler

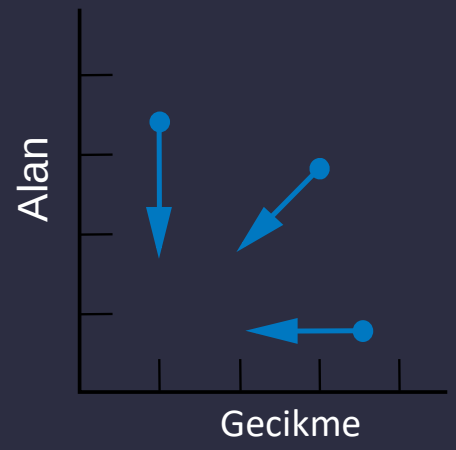
- Ödünleşme (Tradeoff)
  - Bir kriter iyileşirken diğer kriterin kötüleşmesidir
  - Bir kritere göre öncelik verilerek veya dengesini kurarak tasarım yapılır

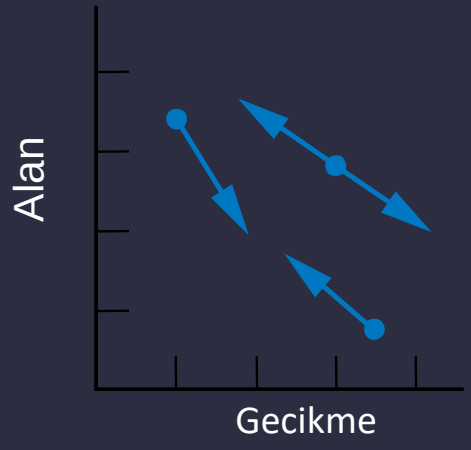


Not: Kapı gecikmeleri eşdeğer varsayınız

Optimizasyonlar ve Ödünleşmeler

Optimizasyon
Tüm kriterlerin iyileşmesidir





Ödünleşme
Bazı kriterlerin iyileşirken diğerlerinin kötüleşmesidir

- Mümkünse optimize etmek, değilse gereksinimlere göre bir tercih yapıp ödünleşerek tasarım yapılır.
  - En konforlu, en yüksek yakıt tasarruflu ve en hızlı araba nasıl olmadığı gibi hedefe göre tasarım yapılır.

Optimizasyonlar ve Ödünleşmeler

- Alan optimizasyonu
- Problemi fonksiyon olarak ifade ediniz
  - Moore kuralları ile denklemi sadeleştirin

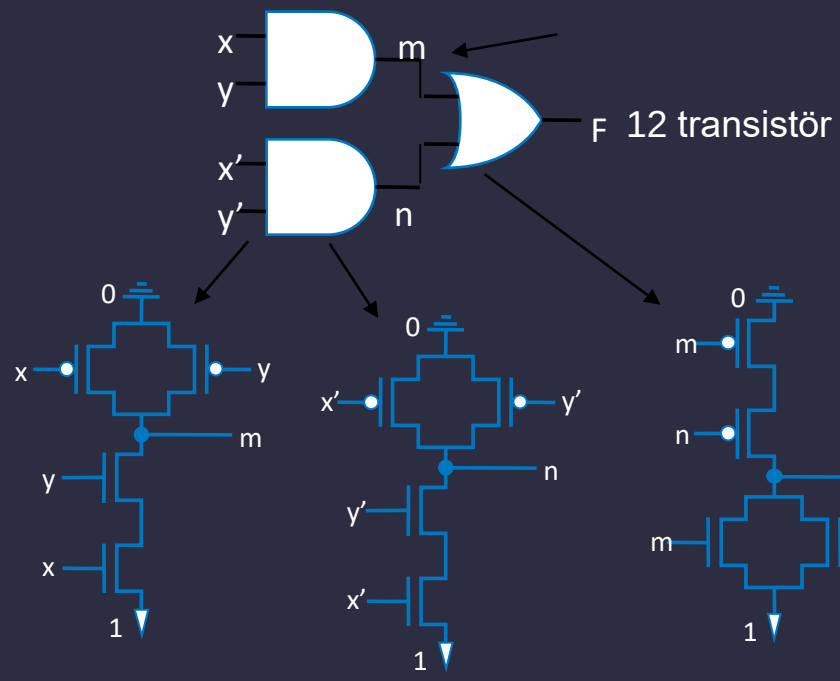
Örnek

F = xyz + xyz' + x'y'z' + x'y'z

F = xy(z + z') + x'y'(z + z')

F = xy*1 + x'y'*1

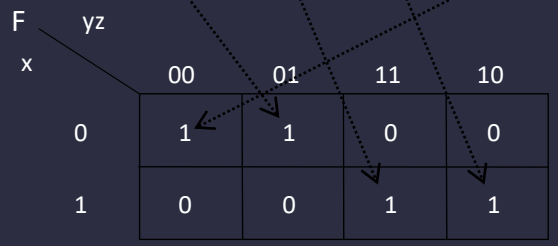
F = xy + x'y' → 6 gate

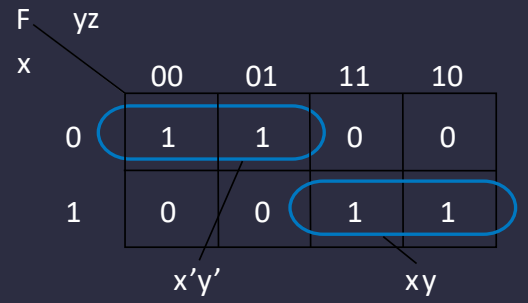


Karnaugh Maps

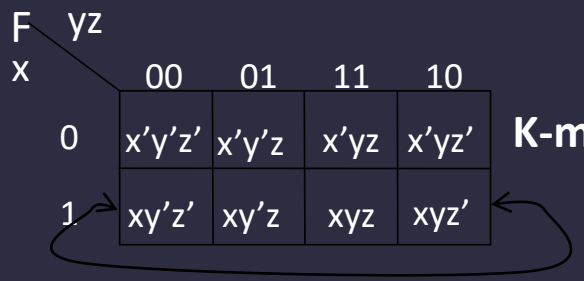
- Moore kuralları ile sadeleştirme yaparken, sadeleşebilecek ifadeleri gözden kaçırmak çok olasıdır.
- ***Karnaugh Map (K-map)***
  - Sadeleştirme için grafiksel bir bakış sağlar
  - İfadeler gruplanarak sadeleştirme yapılır
    - 2 ve katları'nın oluşturduğu dikdörtgen grupları içinde sadeleştirme yapılır

F = x'y'z + xyz + xyz' + x'y'z'





F = x'y' + xy

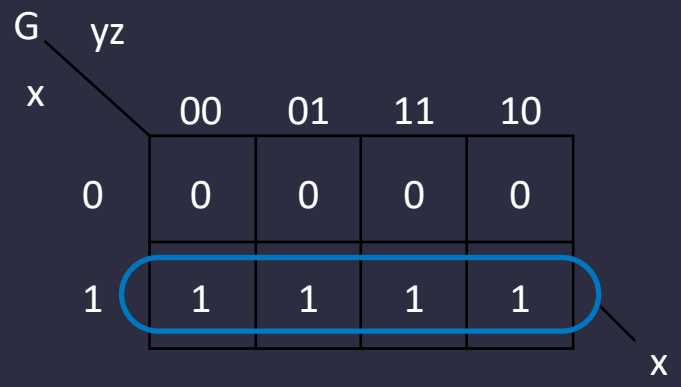


K-map

F = xyz + xyz' + x'y'z' + x'y'z
F = xy(z + z') + x'y'(z + z')
F = xy*1 + x'y'*1
F = xy + x'y'

K-maps

- 4 adet komşu 1 var ise, o değer elenebilir.
  - Yani her durumda çıktı 1 olmuş ise, demek ki çıktıya etkisi yoktur.



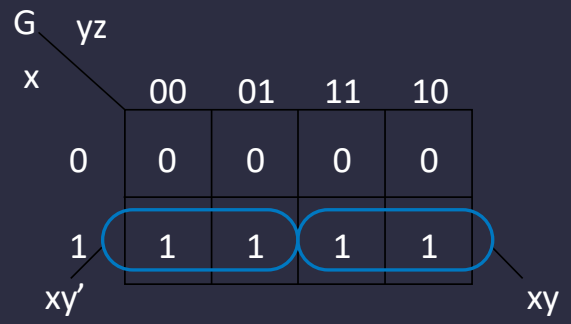
$G = xy'z' + xy'z + xyz + xyz'$

$G = x(y'z' + y'z + yz + yz')$

$G = x(y'(z'+z) + y(z+z'))$

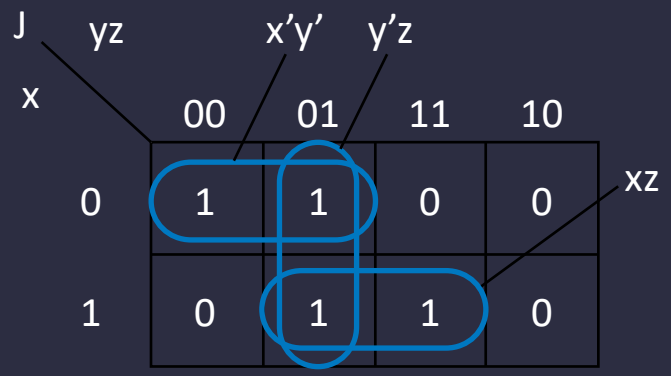
$G = x(y'+y)$

$G = x$



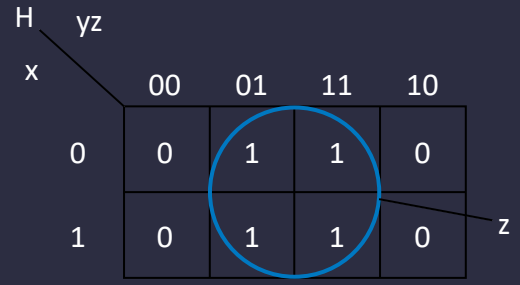
K-maps

- Örnekler

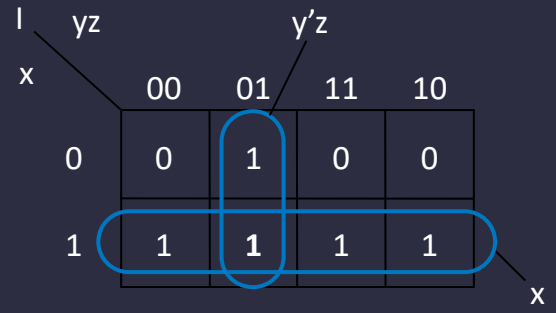


J =x'y' + y'z + xz

H = x'y'z + x'yz + xy'z + xyz



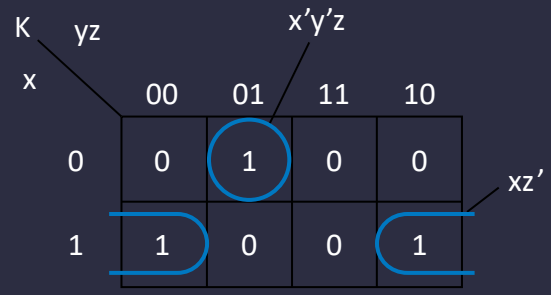
H = z



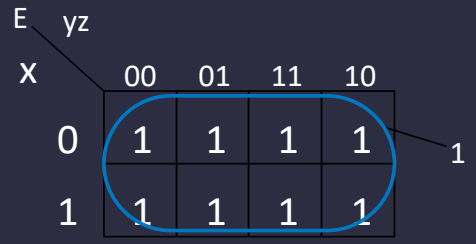
I = x'y'z + xy'z' + xy'z + xyz + xyz'
I = x'y'z + xy'z + xy'z' + xy'z + xyz + xyz'
I = (x'y'z + xy'z) + (xy'z' + xy'z + xyz + xyz')
I = (y'z) + (x)

K-maps

- Dikdörtgenler Kmap'in sınırlarından ters taraftan çıkabilir.

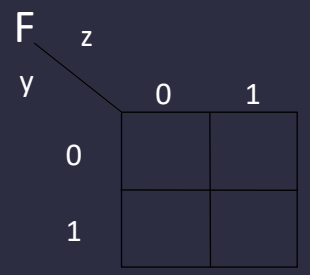


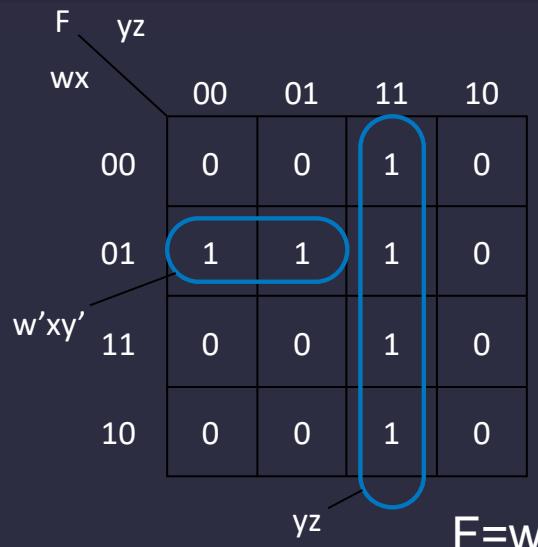
- Dikdörtgenler 1, 2, 4, 8 … katları sayıda elemanı olmalıdır



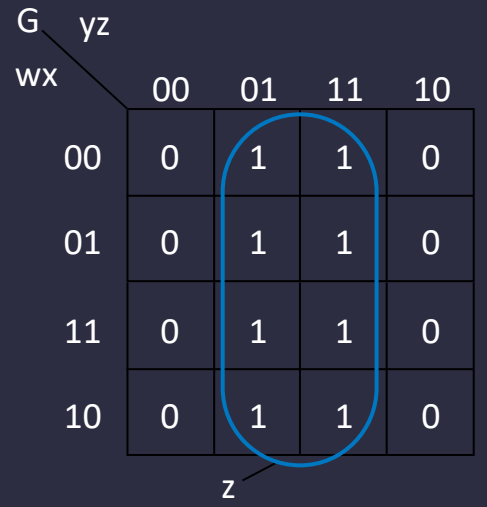
K-maps

- 4 değerli Kmap'lerde aynı prensiple hesaplanabilir
- 5 ve 6 değerli haritalarda mevcut
  - Ancak kullanması çok zor





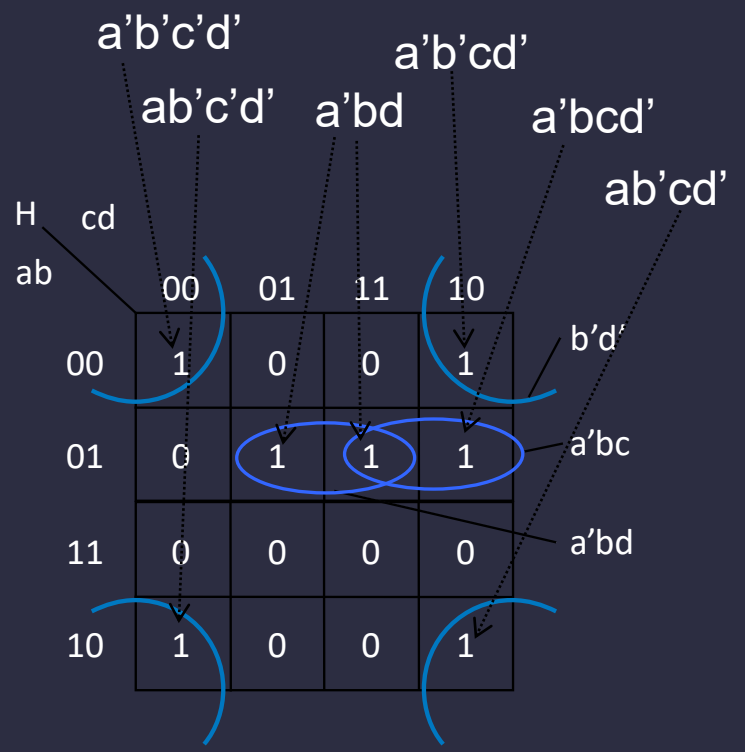
F=w'xy'+yz



G=z

K-maps

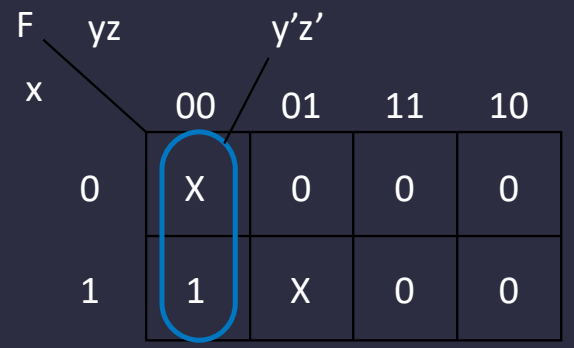
- Örnek:
  - H = a’b’(cd’ + c’d’) + ab’c’d’ + ab’cd’ + a’bd + a’bcd’

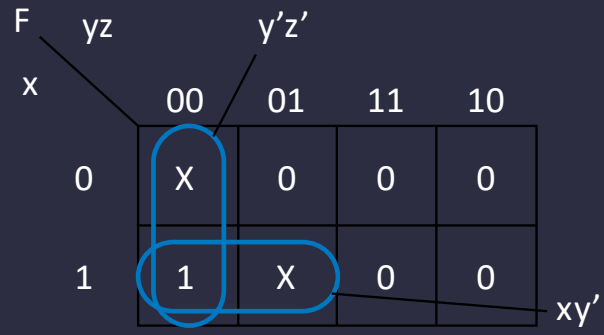


H = b’d’ + a’bc + a’bd

Önemsenmeyen (Don’t Care) Girişleri

- Bazı giriş kombinasyonları hiç girilmeyebilir
  - Örneğin, F = xy’z’ formülü için x, y ve z hiçbir zaman aynı anda 0 verilmeyecek ise ve x 1, y 0, z 1 aynı anda olmayacak ise, bunlar önemsenmeyen olarak yazılabilir.
  - Fonksiyonun sadeleştirilmesine yardımcı olacak şekilde 1 veya 0 olarak düşünülebilir
- Kmap’lerde
  - Önemsenmeyenlerin yerine X yazılır
  - Değişken X ile karıştırılmamalıdır

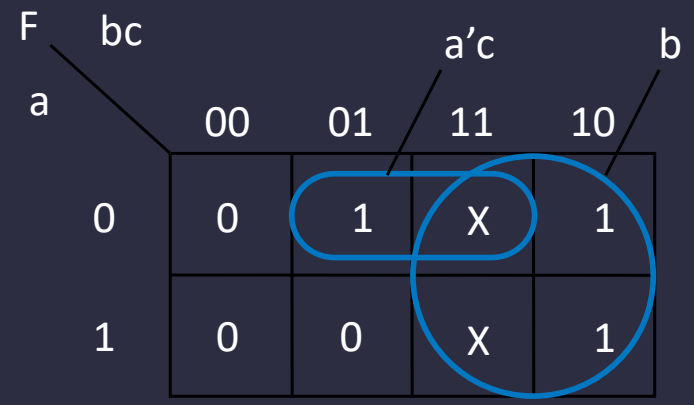




Gereksiz kullanımda fazla terim getirir

Dont Care Kullanarak Minimizasyon

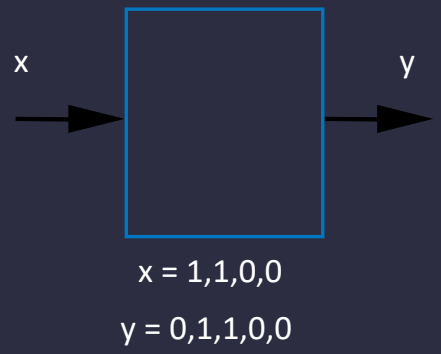
- Örnek:
  - F = a’bc’ + abc’ + a’b’c
  - Önemsenmeyecek girişler:
    - a’bc
    - abc

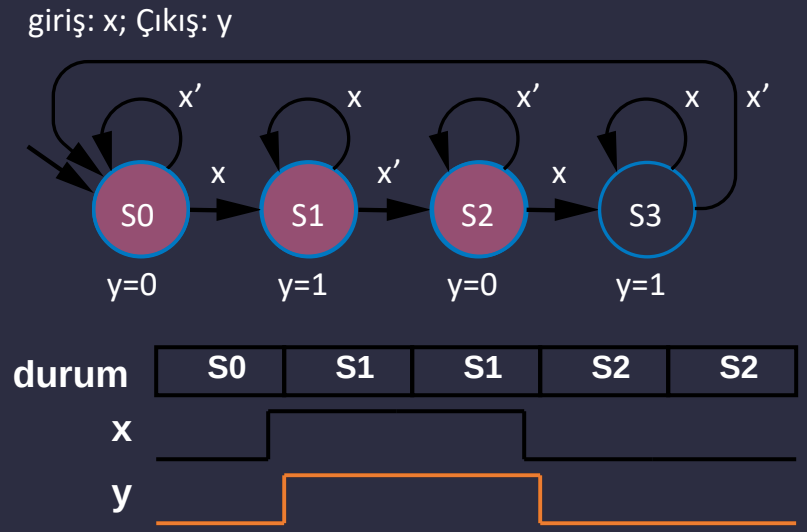


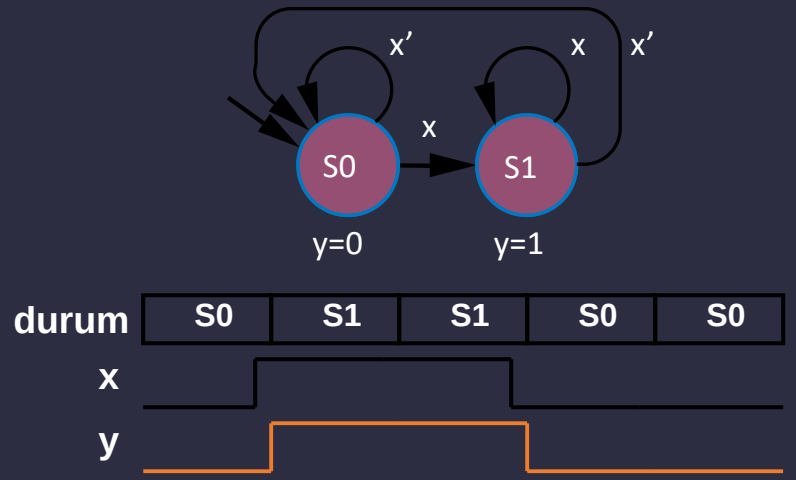
F = a’c + b

Durum Azaltma

- **Bir FSM'in davranışını değiştirmeden durum sayısının azaltılması**
  - Daha az durum muhtemelen daha az alan kaplamasına neden olacaktır
- X = 1, 1, 0, 0 girişleri için aşağıdaki FSM'leri değerlendiriniz







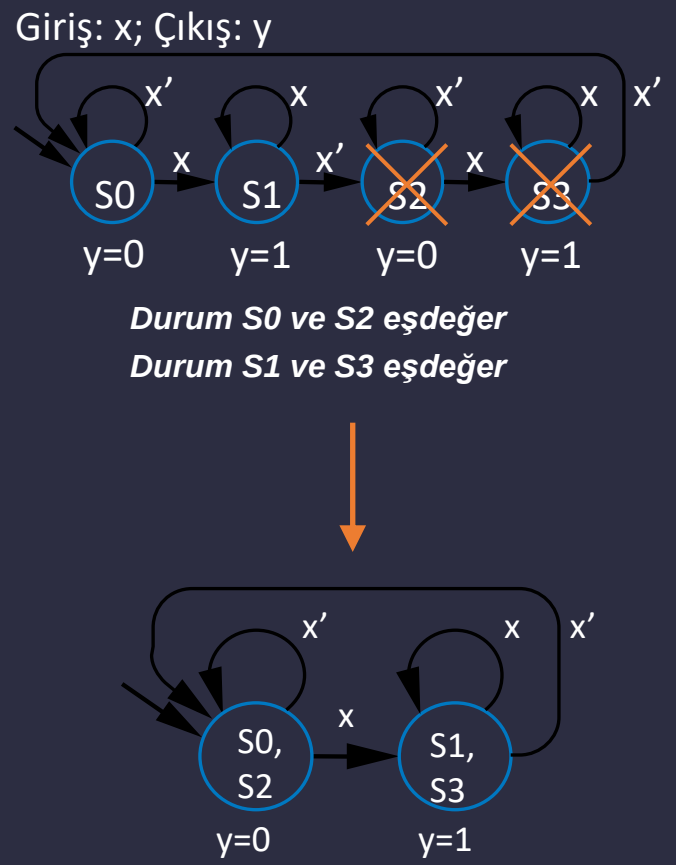
Durum Azaltma

Eğer iki durum eşdeğerse

Çıktıya aynı değeri atıyorlar ise
- Yani S0 ve S2 çıktıya 0 atıyorlar,
- **S1 ve S3 ise çıktıya 1 atıyorlar**

Eşdeğer durumlardan başlanılıp, aynı girişler geldiğinde aynı çıkışlar üretiliyorsa
- Örneğin *x*=1,1,0,0,…
  - **S1 başlangıç**, *y*=1,1,0,0,…
  - **S3 başlangıç**, *y*=1,1,0,0,…

Durumlar ortak kullanılabilir.

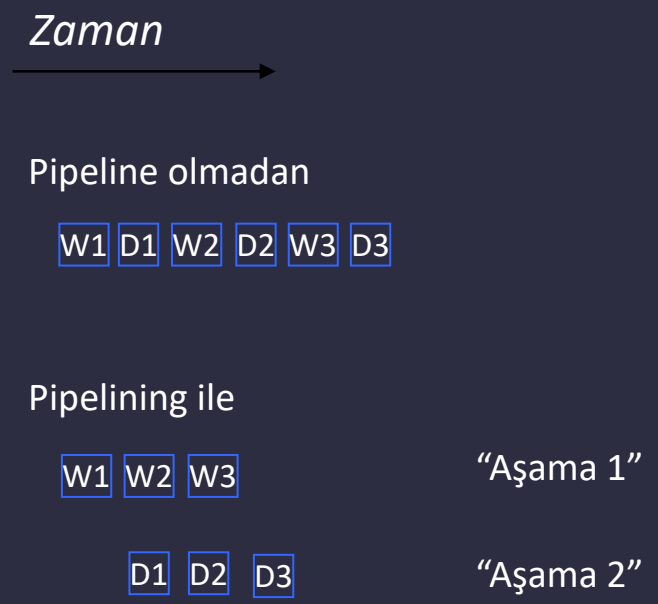


Pipelining

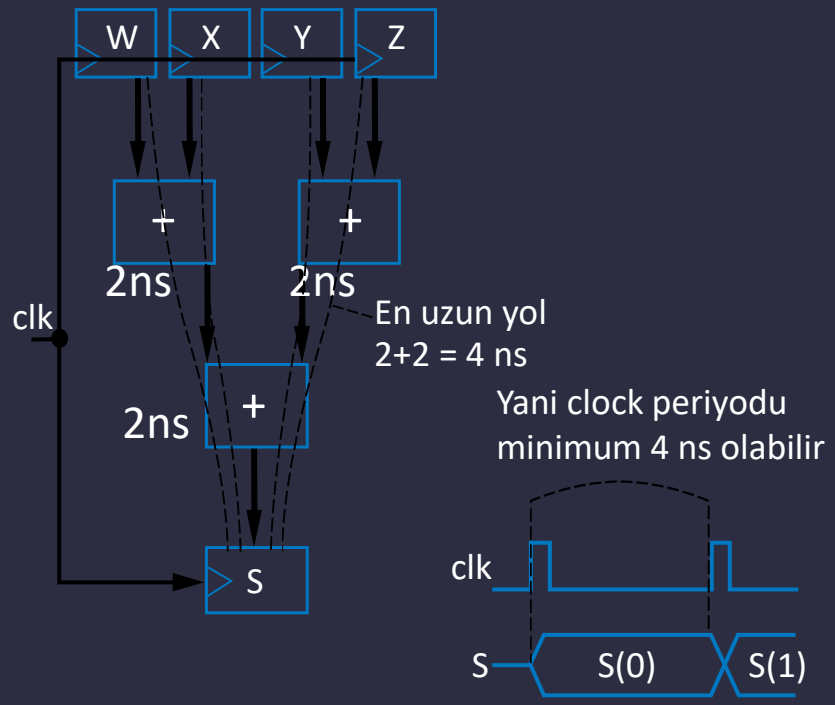
Pipelining: Bir işi aşamalara bölerek, aşamaların birbirlerine veri beslediği ve paralel çalışabildiği bir yapıdır

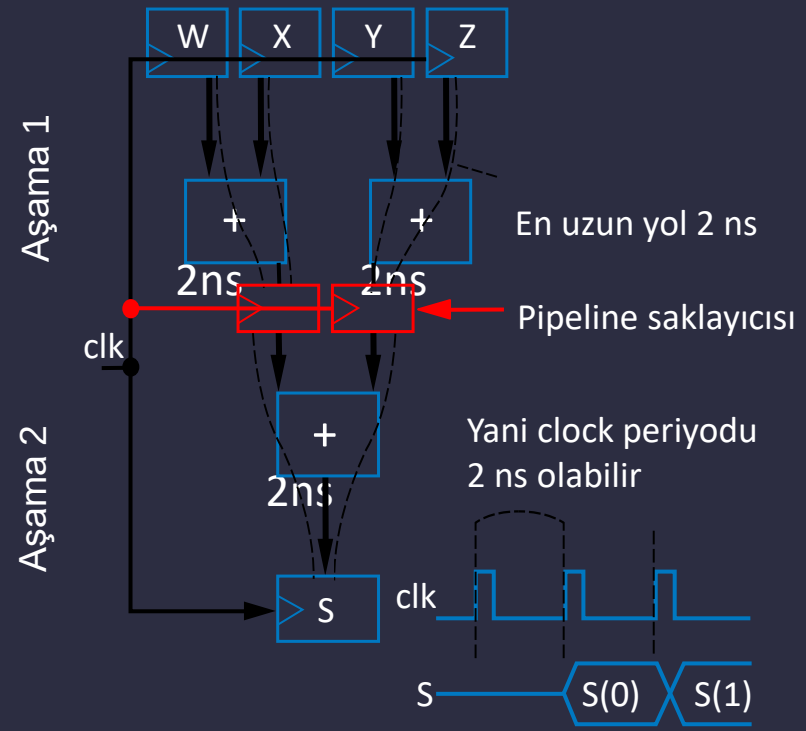
Örnek;

- Siz bulaşık yıkıyorsunuz, arkadaşınız kuruluyor.
  - 1 tabak yıkandı
  - 1 tabak kurulanırken, *2 tabak yıkanıyor*
  - 2 tabak kurulanırken, 3 tabak yıkanıyor…
  - Arkadaşınızın tabağı kurulamasını bitirmesini beklemiyorsunuz.



Pipelining Örneği



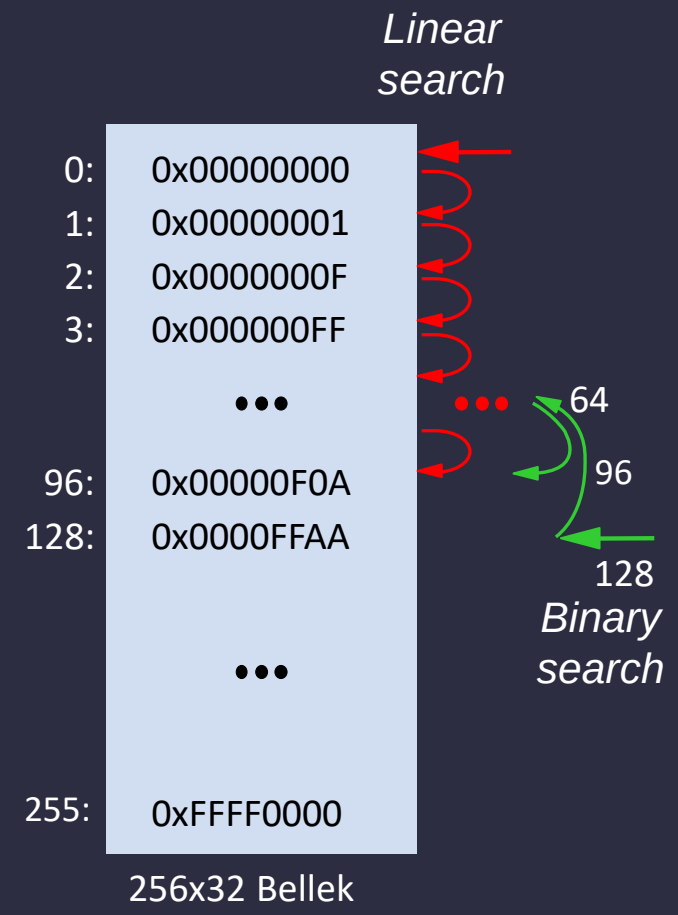


- S = W+X+Y+Z
- Sol taraftaki devre en hızlı olarak 4 ns'de giriş alabilir
- Sağ taraftaki devre ise en hızlı olarak 2 ns'de giriş alabilir. Hız 2'e katlandı.

Algoritma Seçimi

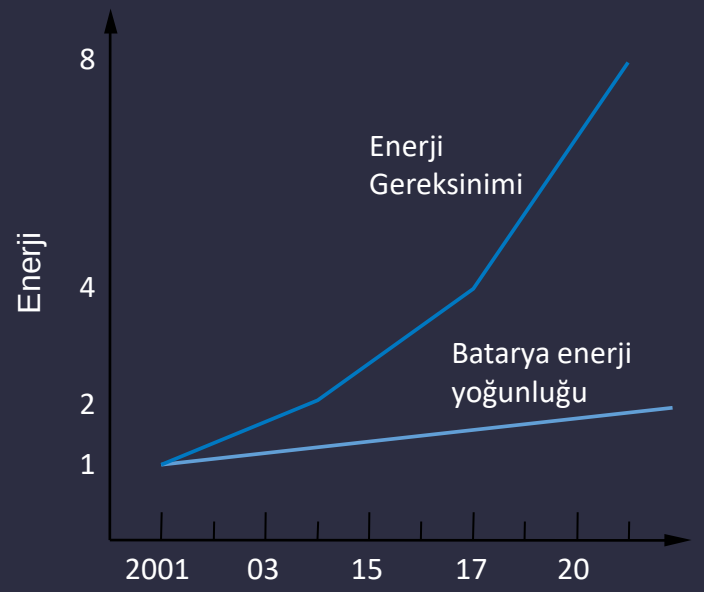
- Seçilen algoritma'nın büyük etkisi vardır

- Örnek: 256 adresli bir bellekte bir eleman aranacak
  - Algoritma 1: "Linear search"
    - Her elemanı tek tek kontrol edilir M[0] , M[1], M[2], ...
    - En kötü durumda 256 deneme yapılır
  - Algoritma 2: "Binary search"
    - Sadece 8 en kötü denemede bulunur



Güç Optimizasyonu

- **Güç** diğer bir önemli tasarım kriteridir
  - Watt (Volt * Akım) cinsinden ölçülür
- Alan tüketimi kadar önemli bir parametredir
  - Özellikle mobil cihazlarda çok önemlidir
  - Bataryalardaki gelişimler ile gerekli enerji ihtiyacı aynı seviyede artmamaktadır.
  - Dolayısıyla daha güç verimli tasarımlar yapılması gerekmektedir.
  - CMOS teknolojisinde, switching yani 0'dan 1'e dönüşüm (Dinamik Güç, Dynamic Power)
    - $P = k * CV^2f$
      - k: sabit;
      - C: kabloların kapasitansı;
      - V: voltaj;
      - f: Switchleme frekansı

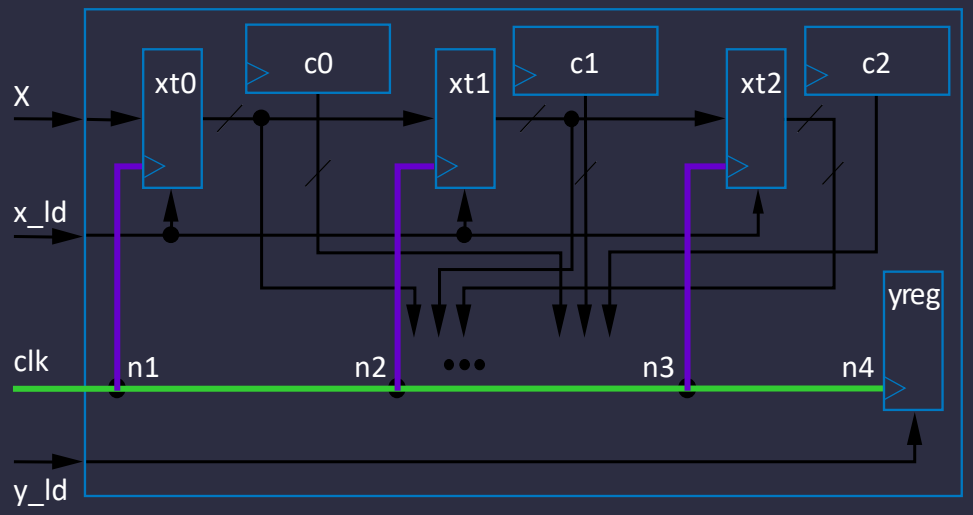


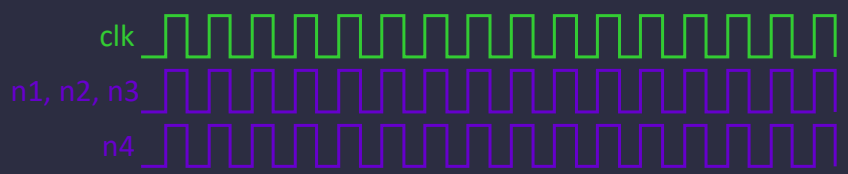
Güç Optimizasyonu (Clock Gating)

- Çip içerisinde sinyallerin değişimi güç tüketimini arttırır

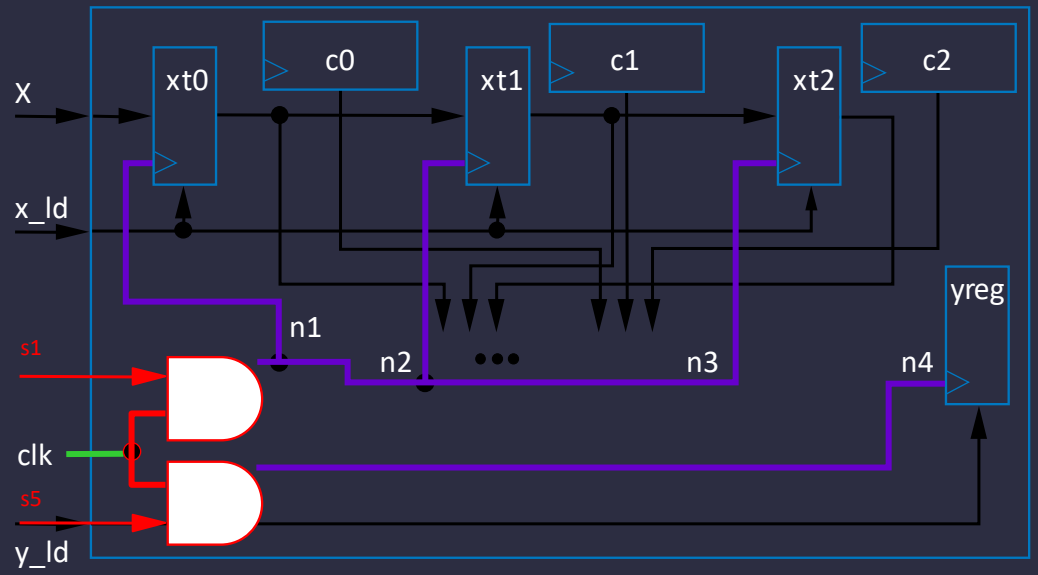
- Buna çözüm olarak, belirli durumlarda kullanılmayan saklayıcıları durdurmaktır.
  - And kapısı ile yapılabilir

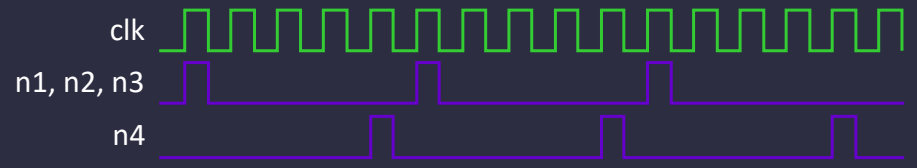
Güç Optimizasyonu (Clock Gating)





Yoğun bir şekilde Switch'in var





Switching azaldığı için güç tüketimi azalmıştır

Güç Optimizasyonu (Düşük Güç Tüketen Kapılar)

- Düşük Güç Tüketimli Kapılar
  - Aynı görevi yapan kapıların birden çok versiyonu olabilir
    - Hızlı/yüksek güç tüketimli, yavaş/düşük güç tüketimli
  - Kritik path'de olmayan kapıları yavaş/düşük güç tüketimli olarak seçip toplam gecikmeye etki etmeden güç düşürülebilir.