SAYISAL TASARIM

Ege Üniversitesi Ege MYO
Mekatronik Programı

BÖLÜM 4

Programlanabilir Mantık Elemanları

Programlanabilir Mantık

Programlanabilir mantık aygıtları (Programmable Logic Devices), PLD mantık geçitleri ve flip-floplar içeren ve bu elemanları yazılım ile programlayarak istenilen mantık işlemini gerçekleyen tümdevrelere verilen isimdir. Yaygın bilinen PLD çeşitleri;

SPLD: (Simple PLDs) ilk üretilen ve içerisinde kısıtlı sayıda mantık geçidi içeren PLD'lerdir. PAL ve GAL her ikisi SPLD'dir.

CPLD: (Complex PLDs) tümdevre içerisine birden fazla SPLD ve bunları birbirine bağlayacak düzenekler yerleştirillen PLD'lerdir.

FPLD: (Field Programmable Gate Array) bağlantı seçeneği ve kapasitesi daha fazla olan PLD'lerdir.

PLD'ler

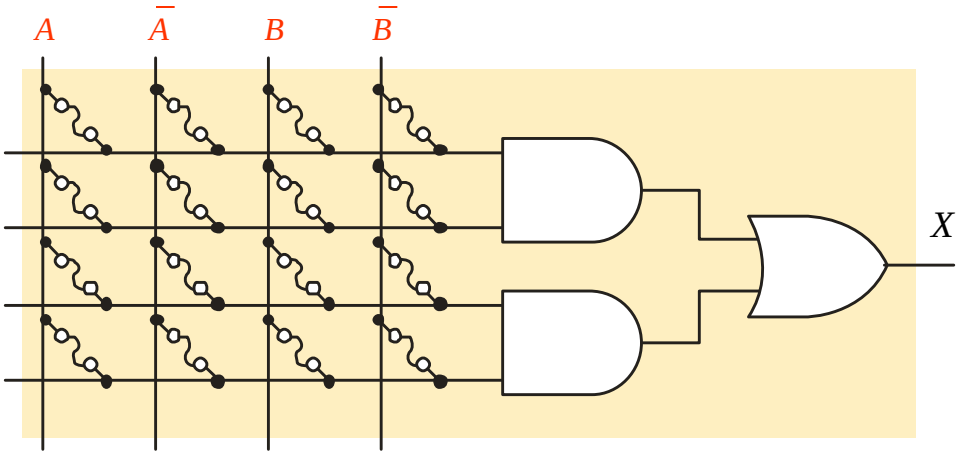
PLD Kullanmanın avantajları

- Baskıdevre kartını basitleştirir
  - Güç tüketimi azalır
  - Kart boyutu küçülür.
  - Üretim sonrası test sürecini basitleştirir
- Güvenirliği yükseltir.
- Tasarımda esneklik sağlar.

PAL ve GAL

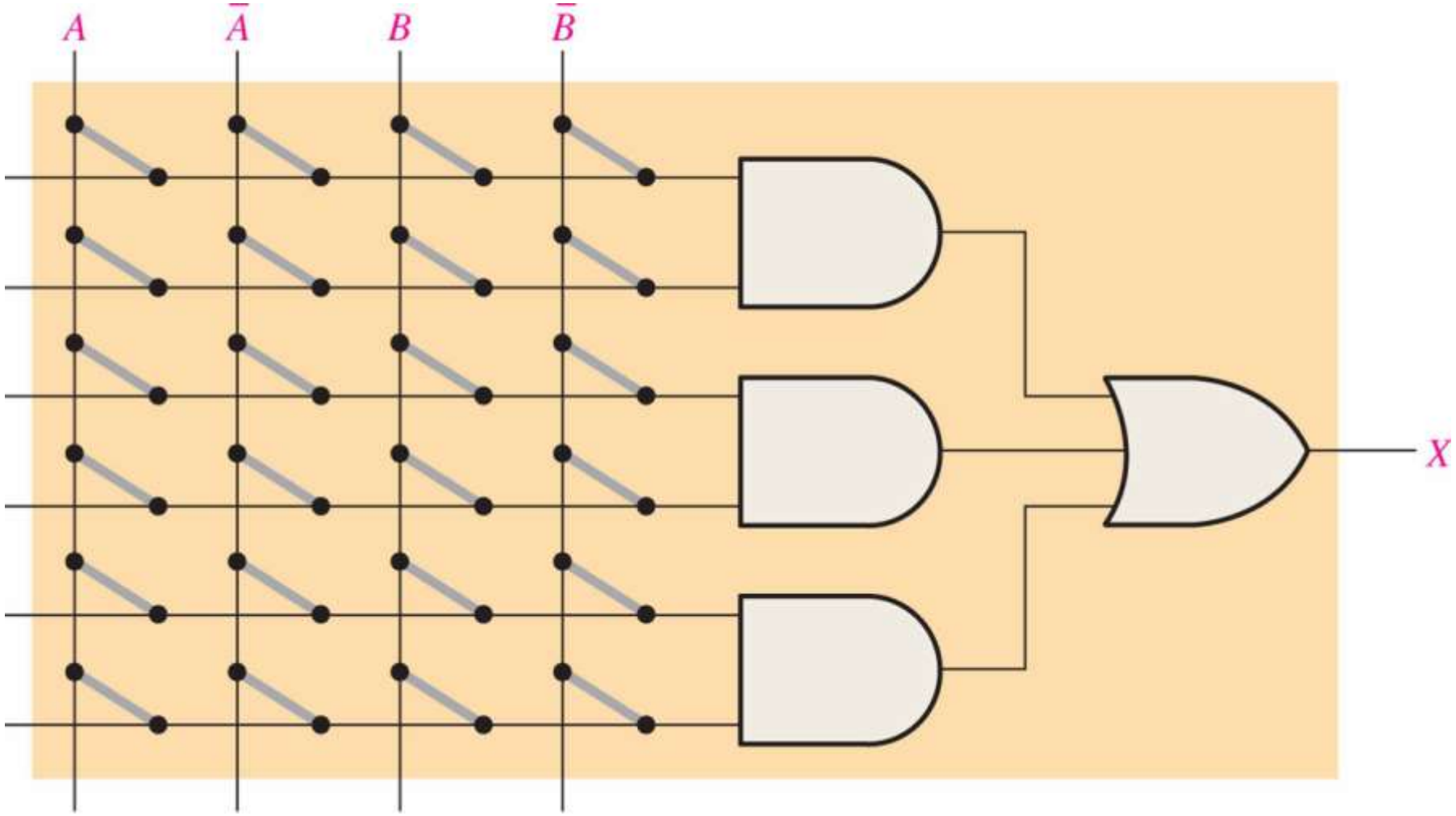
İki önemli SPLD çeşidi **PAL'ler** (Programmable Array Logic) ve **GAL'lerdir** (Generic Array Logic). Dizge (array) satır ve sütun şeklinde düzenlenmiş hatlardan oluşur, satırların ucu AND geçitlerine sütünların ucu ise girişlere bağlıdır.

PAL'ler sadece bir defa programlanabilen (OTP) dizgeye sahiptir**.** Satır ve sütün arasındaki sigortalar kalıcı olarak yakıldığında giriş terimleri AND'lenir.

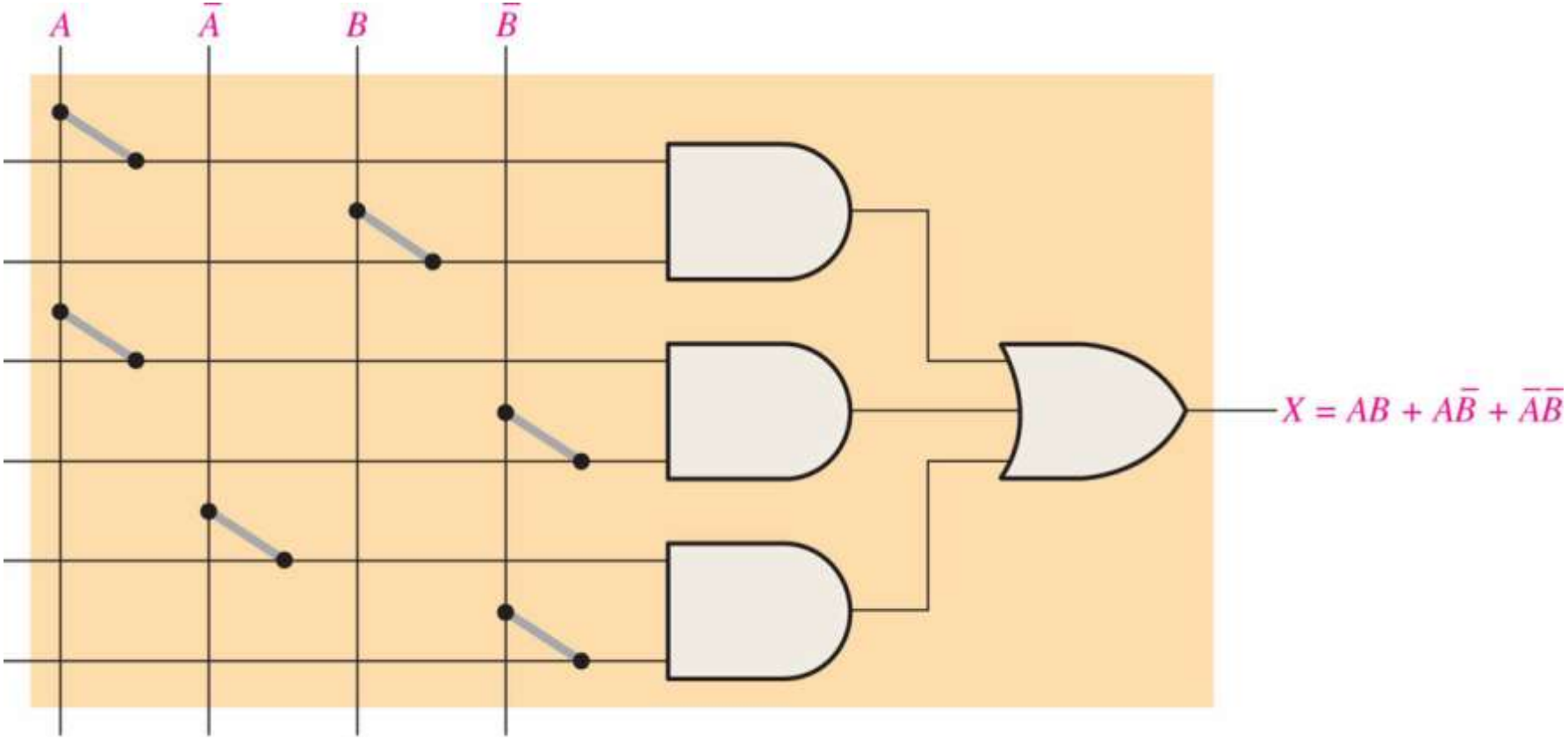


Basit AND-OR dizgesi

PAL'in basit AND/OR yapısı.



Çarpımların toplamı olarak PAL uygulama örneği.

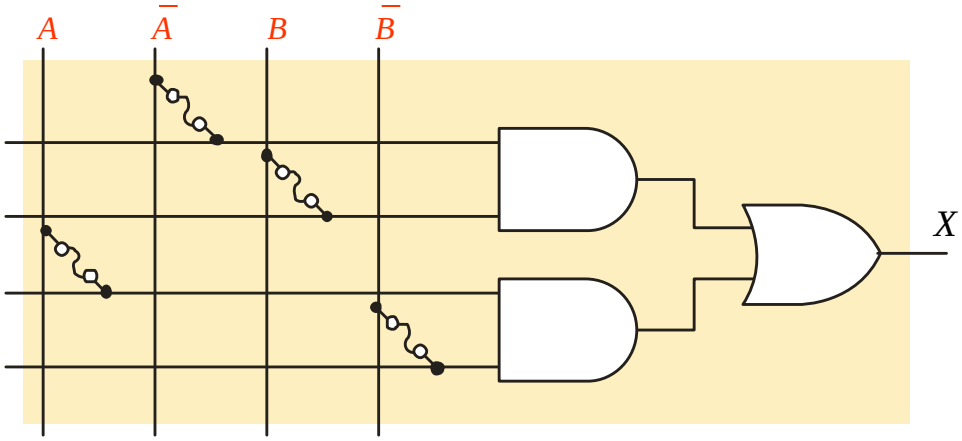


PAL'ler

PAL'ler özel programlayıcılar ile seçilen sigortaların yakılması ile programlanır.

Örnek

Dizgede gerçekleştirilen işlem nedir?



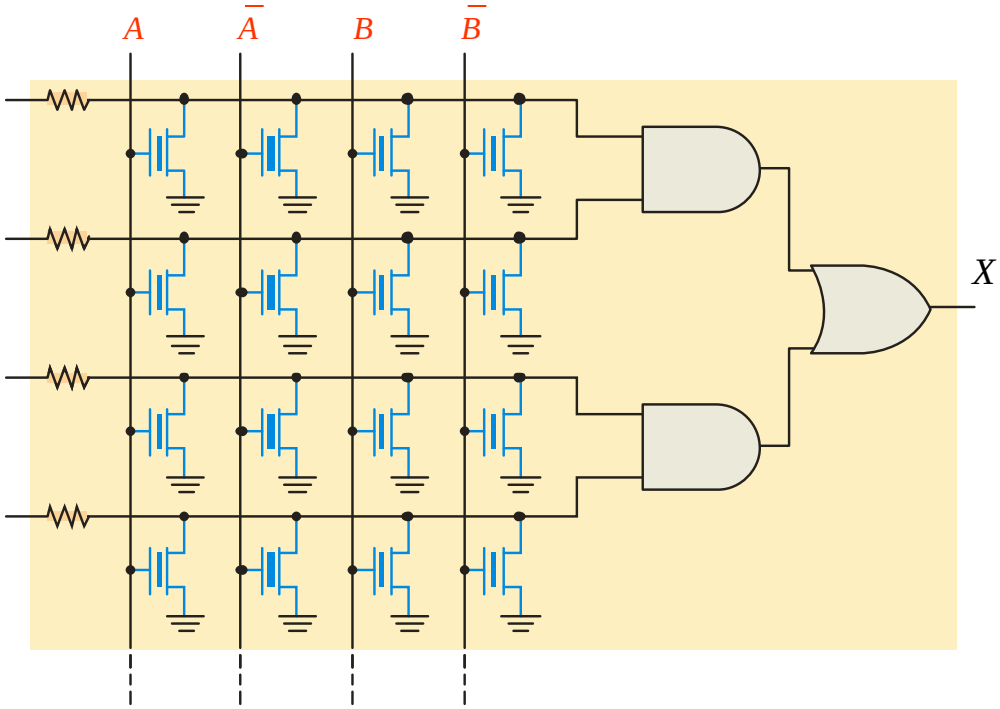
Çözüm

XOR işlemi yapar.

GAL'ler

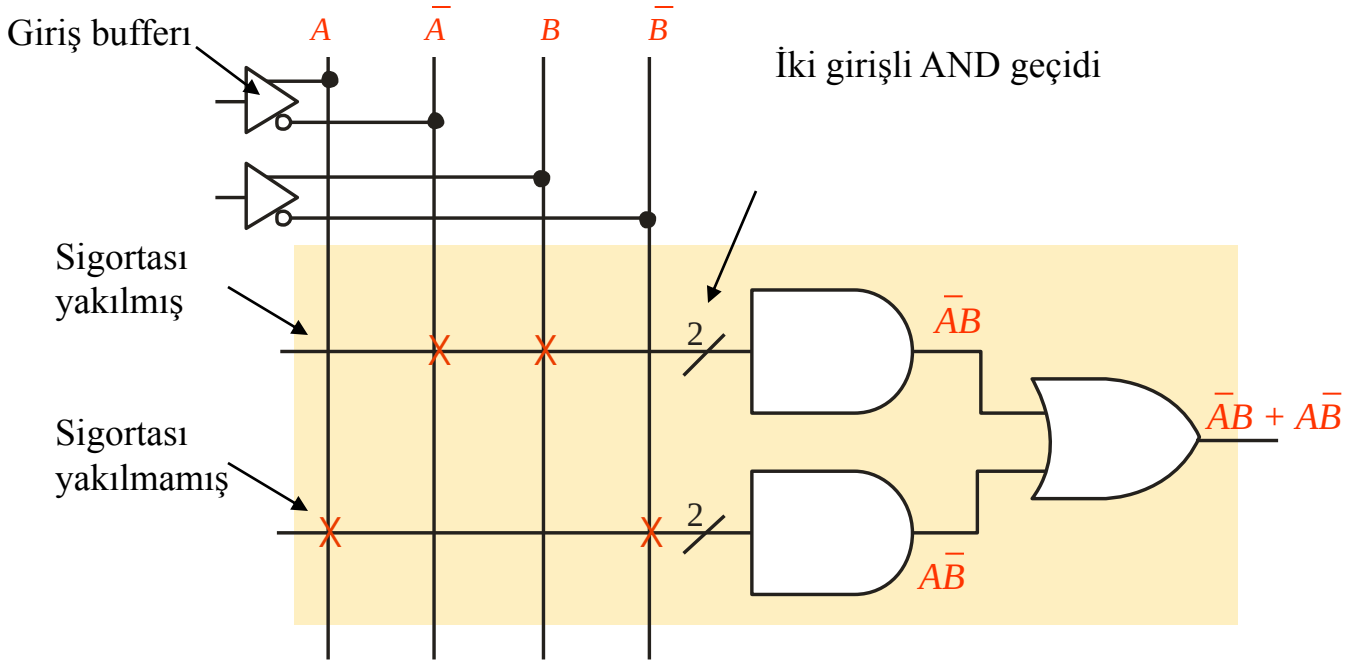
GAL (Generic Array Logic) yapı olarak PAL'lere benzer fakat tekrar programlanabilir bir yapıya sahiptir. Aslında PAL'lerden tek farkı satır ile sütün arasında yer alan sigortaların yapısıdır.

GAL'ler Lattice Semiconductor firması tarafından geliştirilmiştir. Bu elemanlar hızlıdır ve 3.3 V veya 5 V mantık devreleriyle çalışabilirler.

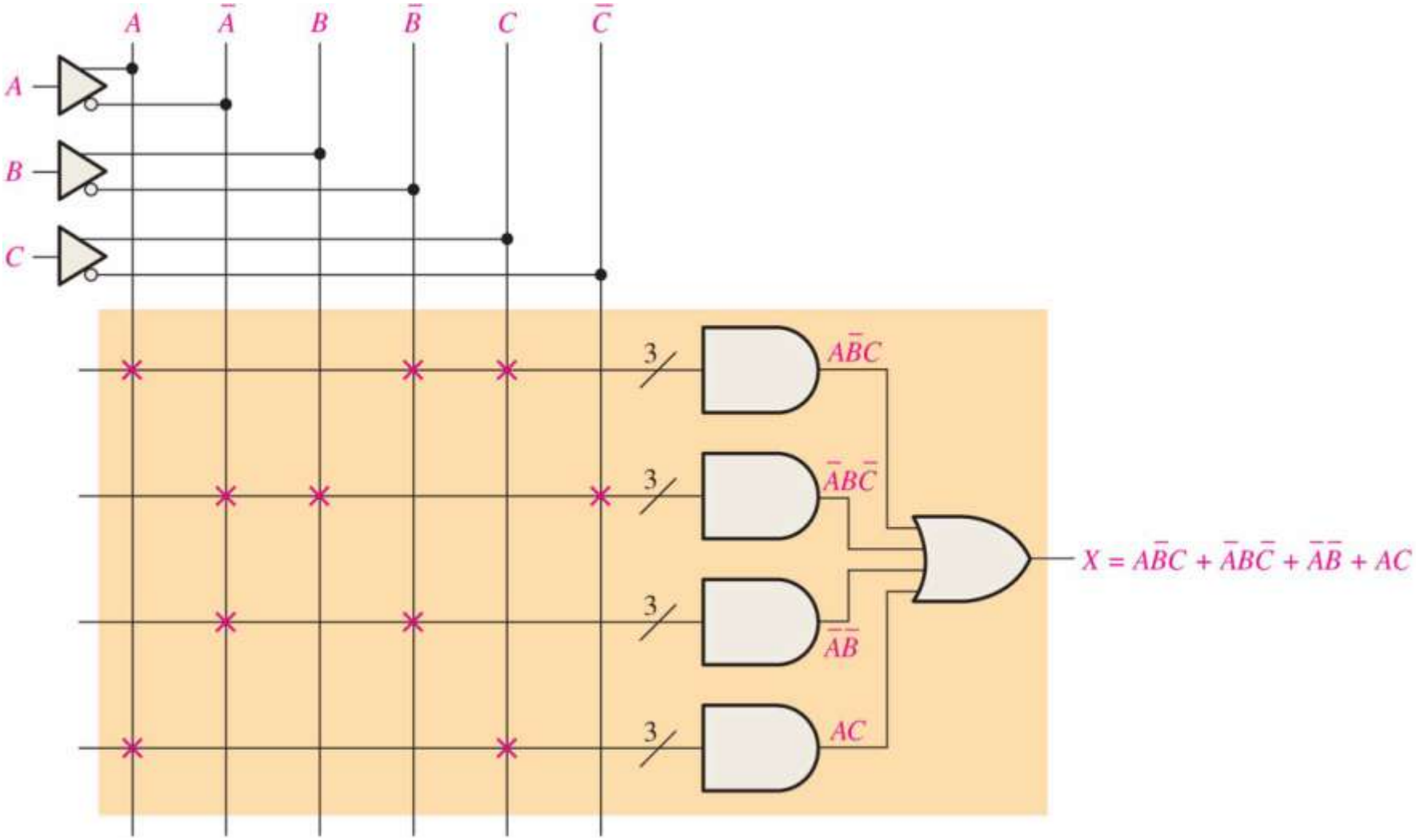


PAL'ler ve GAL'ler

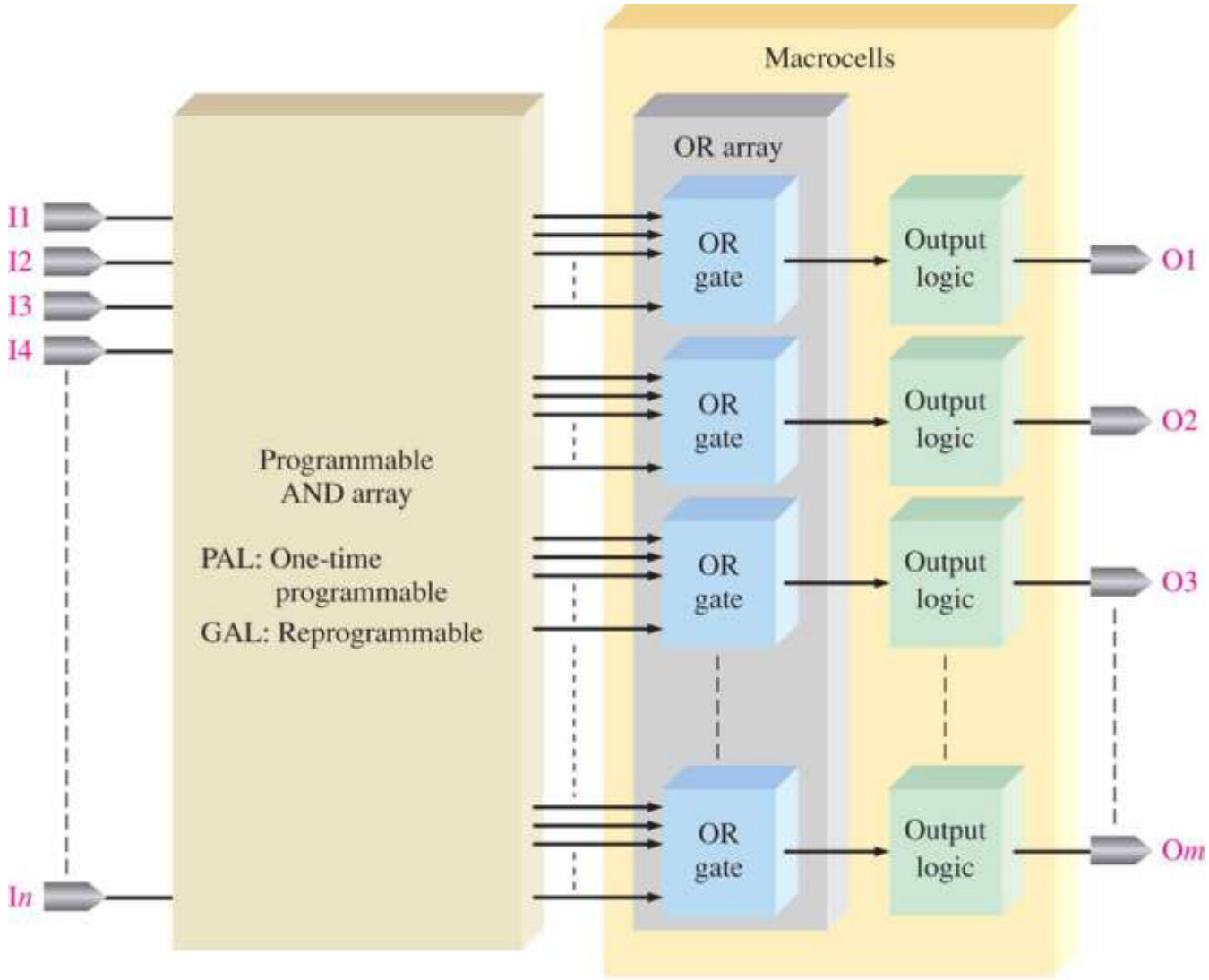
PAL'ler ve GAL'ler basitleştirilmiş devre ile gösterilebilir.tek bir hat üzerine çizilen slash ve sayı ile birden fazla hattı temsil edebilir. Aşağıdaki şekilde XOR geçidi olarak programlanmış PAL'in şekli verilmiştir.



Örnek bir PLD

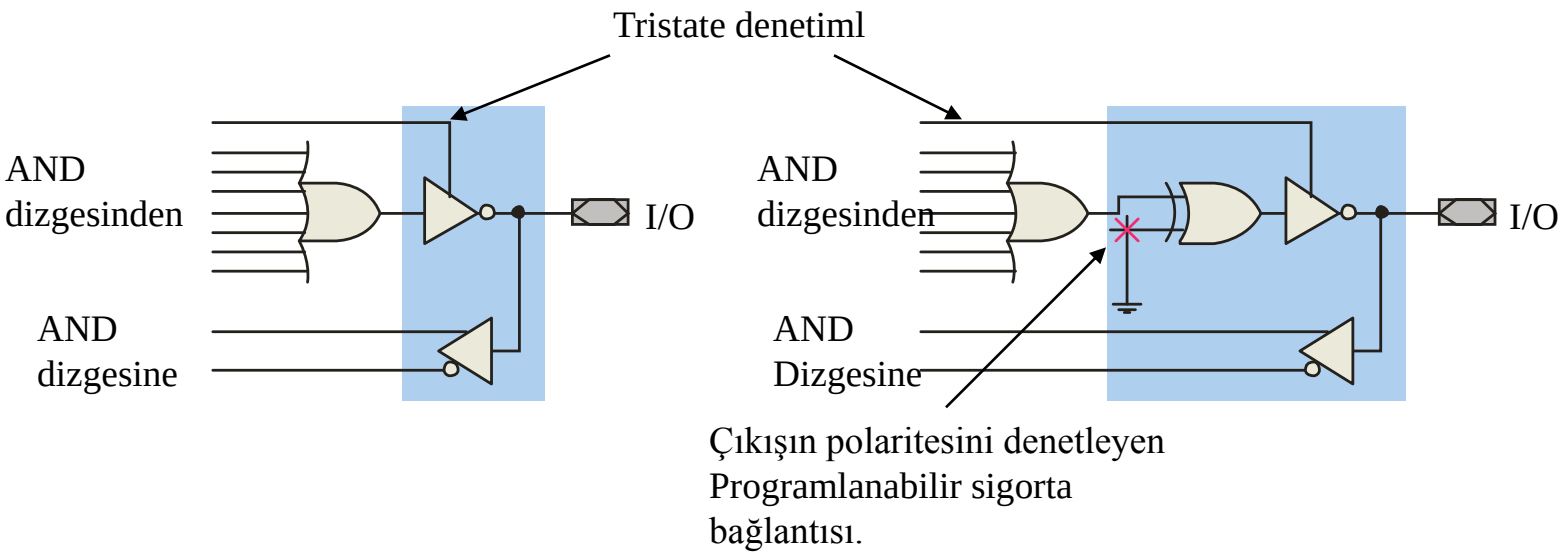


PAL veya GAL'in blok diyagramı

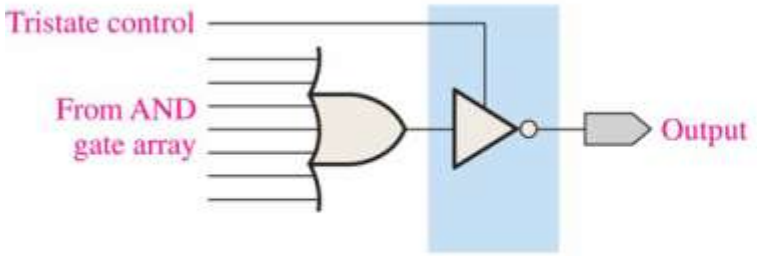


PAL'ler ve GAL'ler

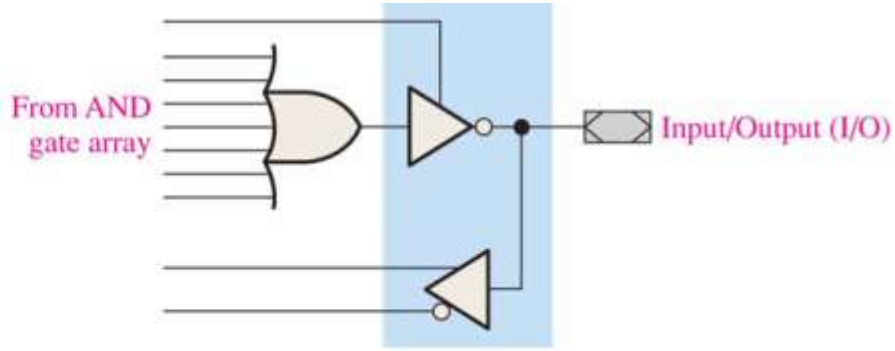
PAL'ler ve GAL'ler girişlerinde yapılarının karmaşıklığına göre büyük boyutlu dizge ve çıkışlarında kullanım alanlarını artıracak mantık devreleri içerirler. Her çıkış işareti güçlendirmek amacıyle OR geçidinden geçirilir, bu yapıya **macrocell** adı verilir. Aşağıda şekilde gösterildiği gibi iki tür PAL/GAL macrocell'i kullanılır, macrpcell'ler çıkış veya giriş hattı olarak kullanılabilir.



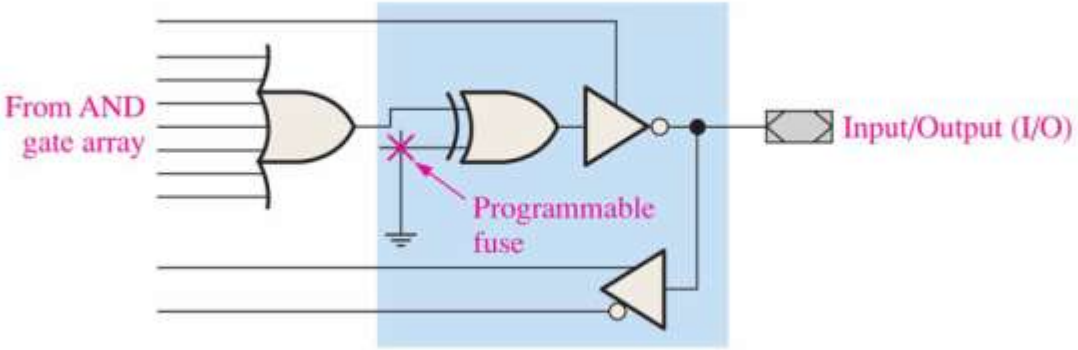
PAL/GAL Macrocell'lerinin Yapısı ve Kullanımı



(a) Combinational output (active-LOW). An active-HIGH output would be shown without the bubble on the tristate gate symbol.



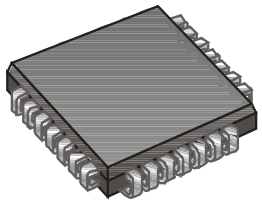
(b) Combinational input/output (active-LOW)



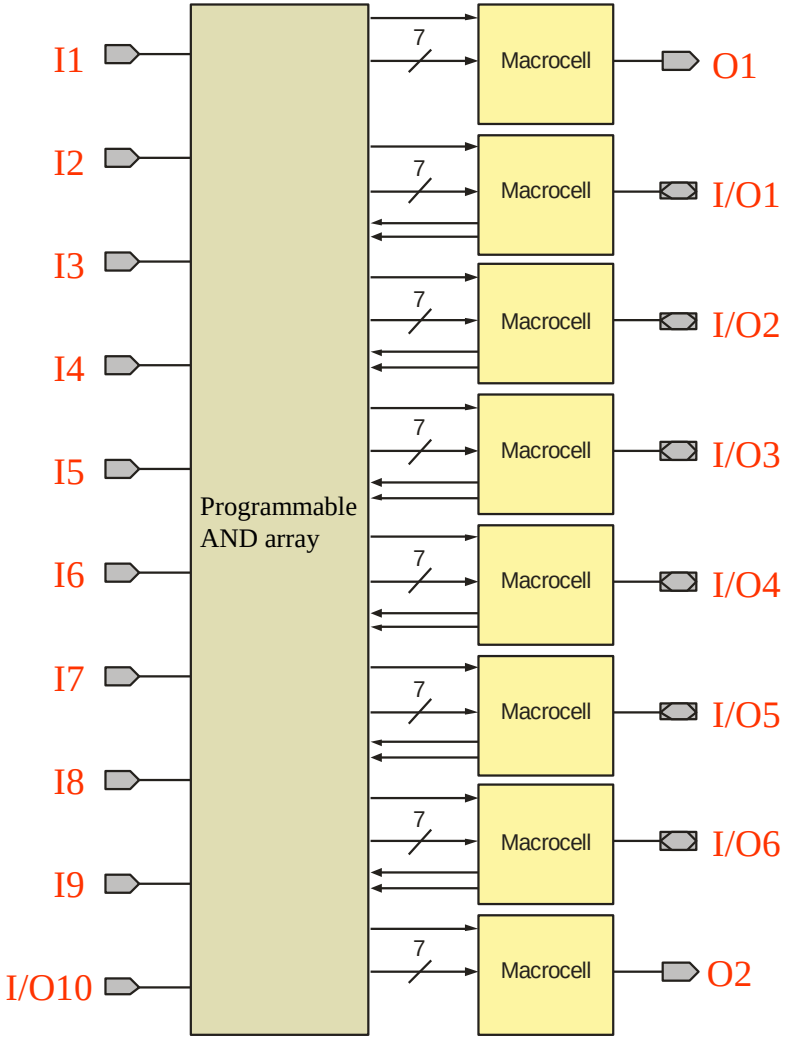
(c) Programmable polarity output

PAL'ler ve GAL'ler

PAL16V8 en yaygın satılan SPLD'dir. 16 giriş ucuna ve 8 çıkış ucuna sahiptir. I/O hatları ise hem çıkış hem de giriş olarak kullanılabilir.



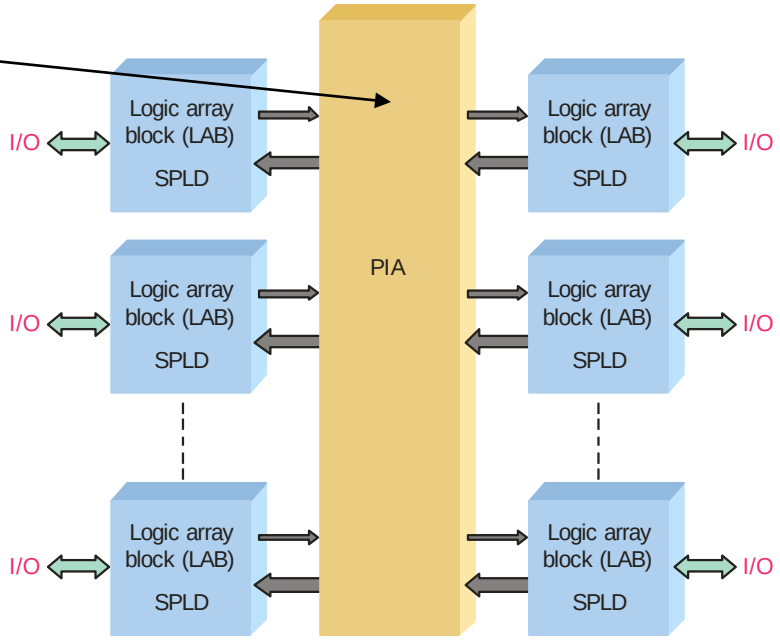
PLCC kılıfı



CPLD'ler

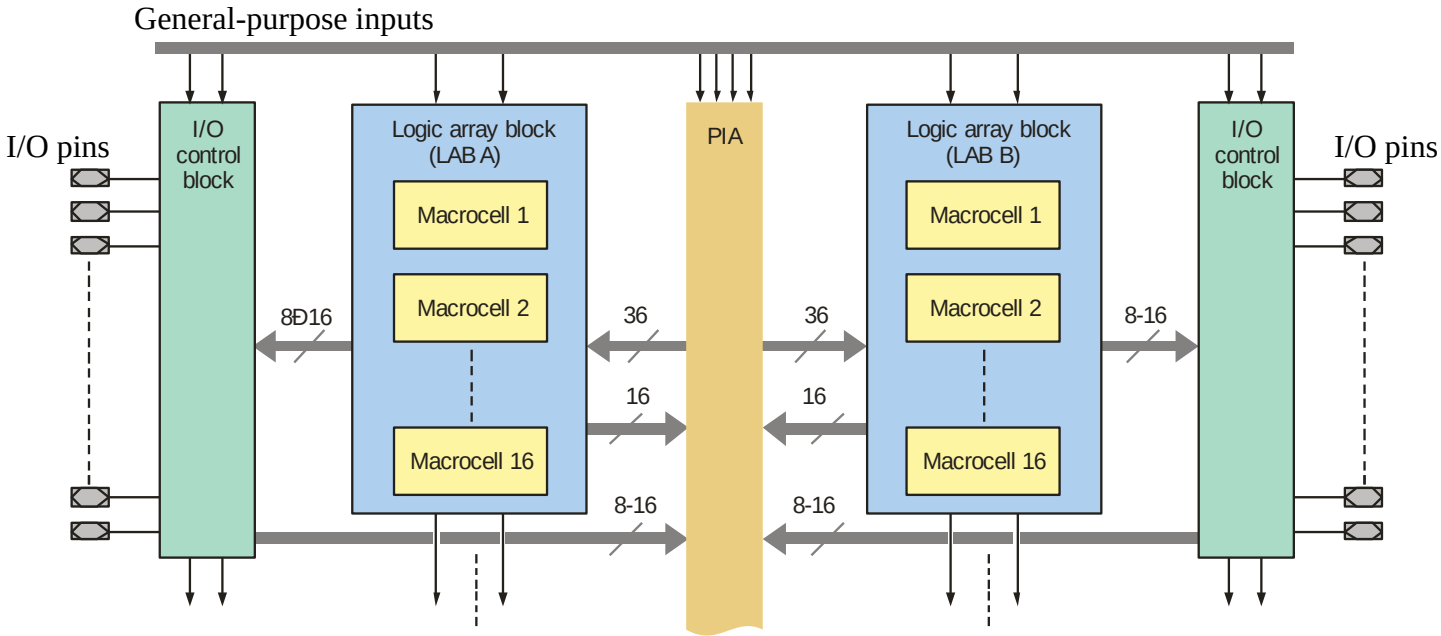
Karmaşık programlanabilir mantık elemanları (complex programmable logic device), **CPLD,** birden fazla mantık dizge bloğu (**LAB**) içermektedir. Aslında CPLD içerisinde birden fazla SPLD içeren ve bu SPLD bloklarını birbirine bağlayan programlanabilir iç bağlantı dizgesi (programmable interconnect array), **PIA,** içeren bir tümdevredir. Çok farklı yapıda CPLD üretilmektedir.

PIA LAB'ları entegre içinde birbirine bağlayan bir dizgedir. Bu bağlantılar yüksek seviyeli bir programlama dili (hardware description language), HDL, ile yapılabilir.

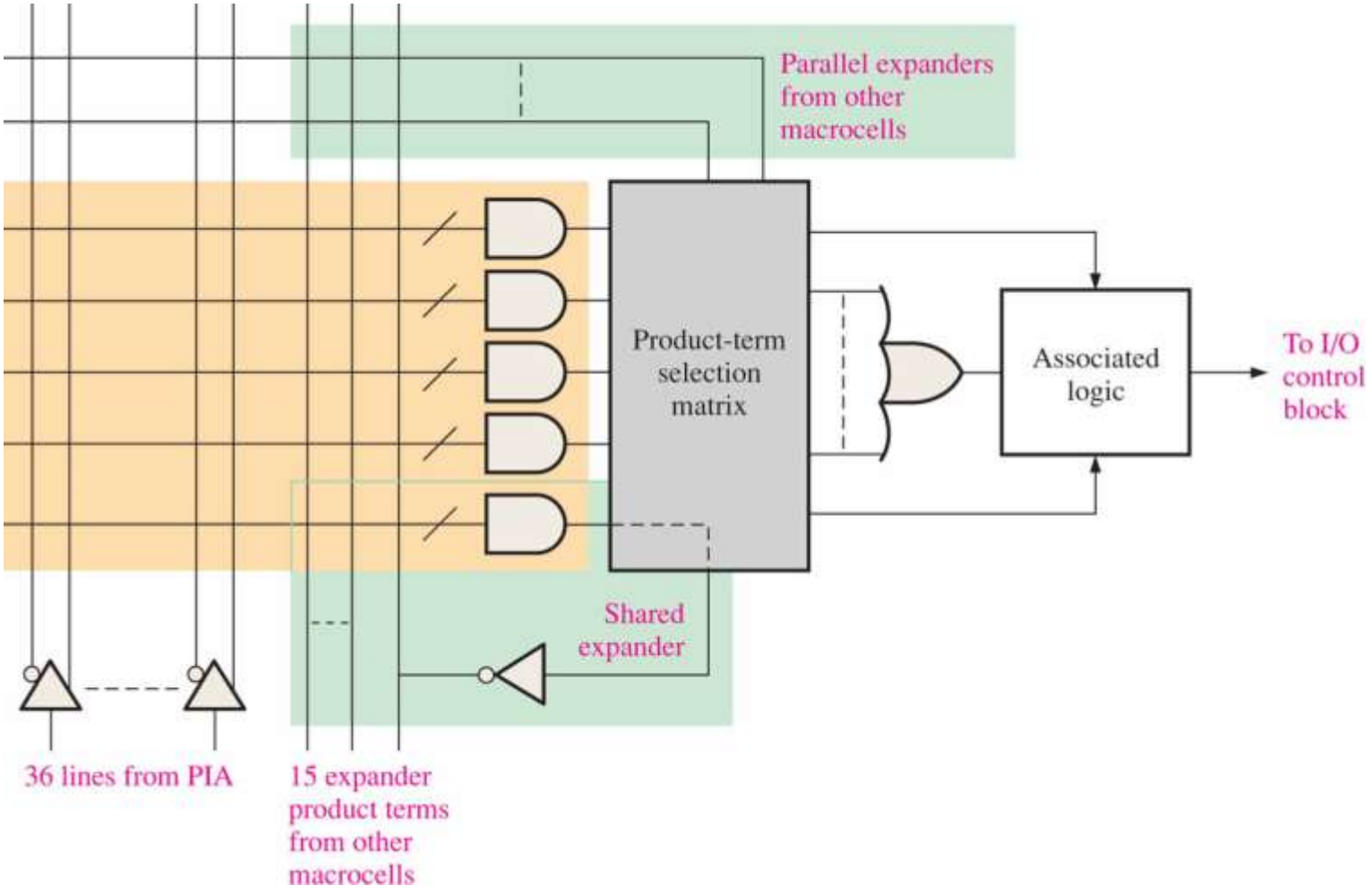


CPLD'ler

CPLD'lerin yapısı içerisine yerleştirilen elemanlara bağlıdır. Şağıdaki şekilde Altera MAX 7000 serisi CPLD'nin iç yapısı gösterilmiştir. CPLD iç yapısı üretici firmalara göre çok farklılık göstermektedir.

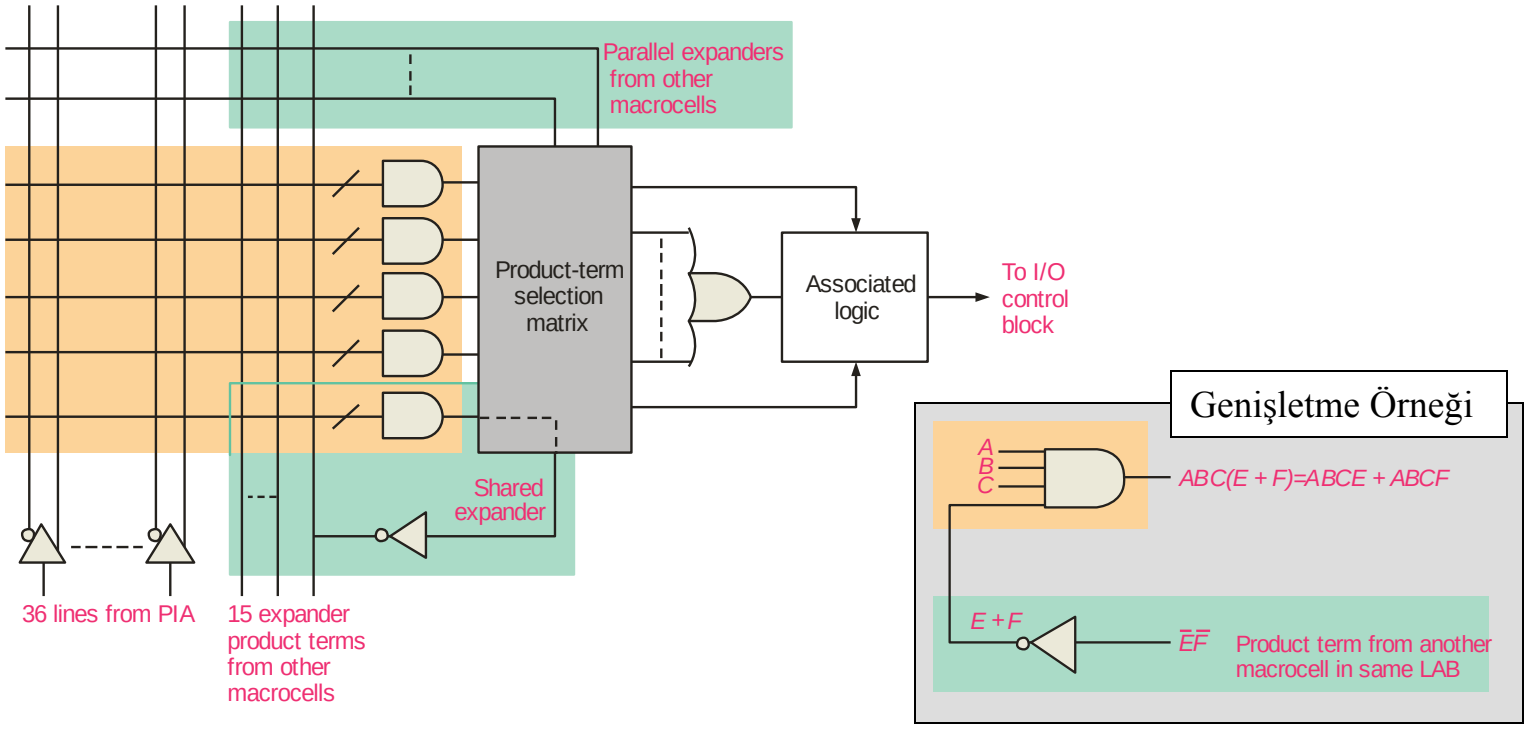


MAX 7000 serisi CPLD'lerin macrocell yapısı

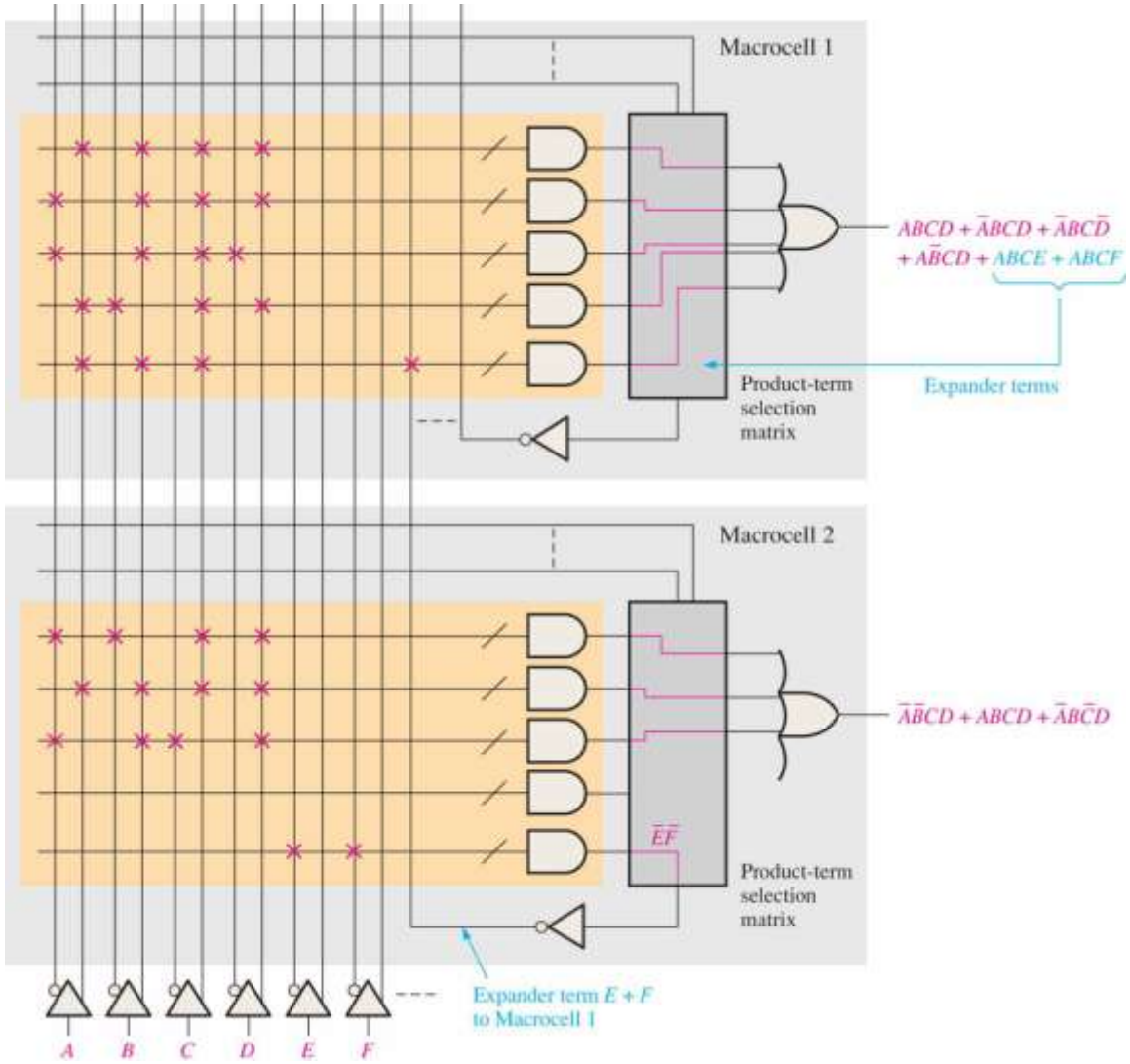


CPLD'ler

Altera MAX 7000 serisinde macrocell'lerinden 5 adet çarpımların toplamı terim elde edilir. Bu çıkışlar genişletme hatlarına bağlanarak daha geniş çarpım terimleri elde edilebilir.

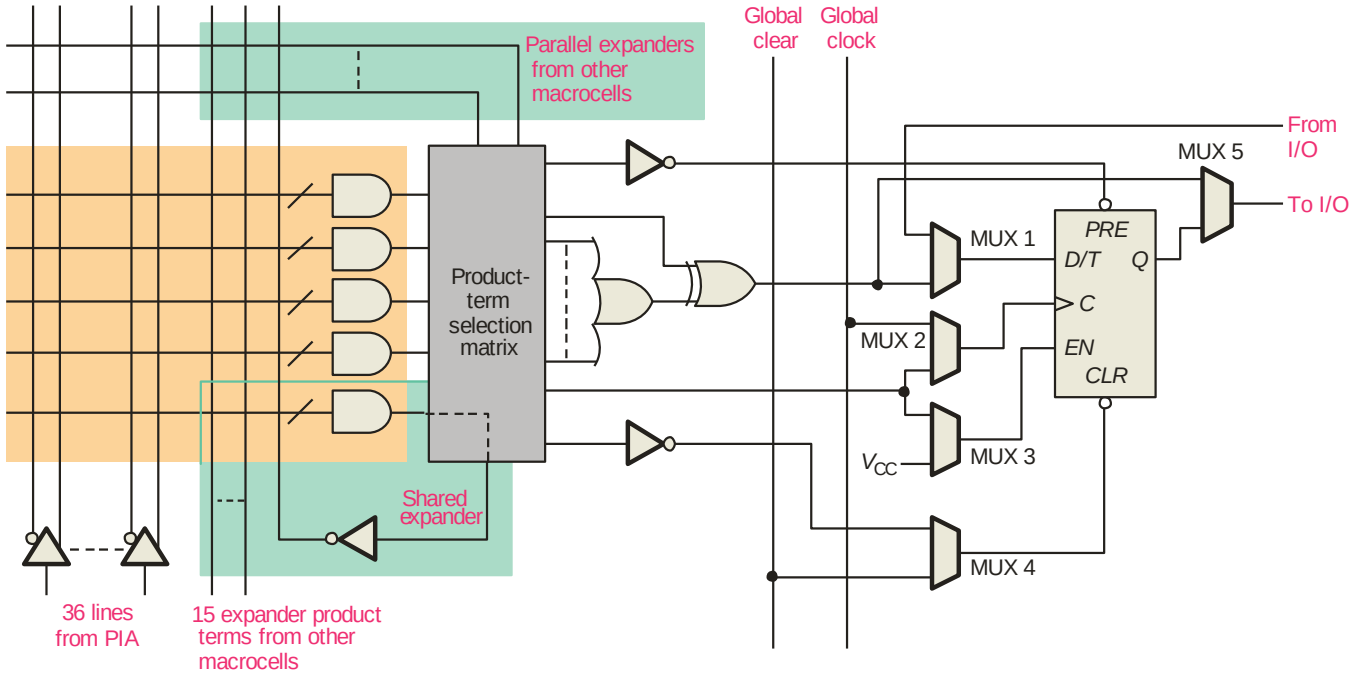


Macrocell kullanılarak terimleri genişletme

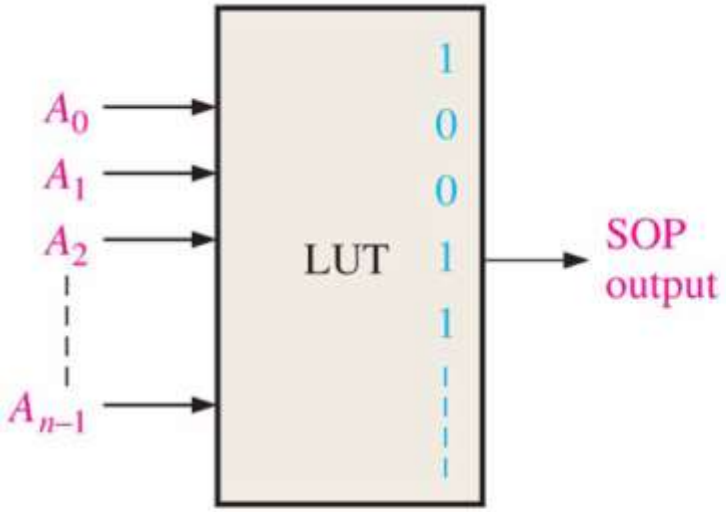


Macrocell

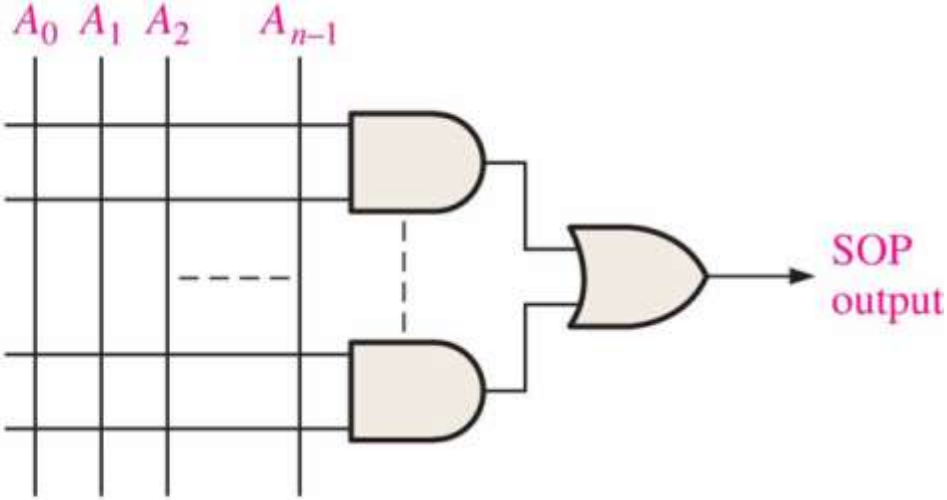
Kombinasyonel mantık çıkışlarına ek olarak macrocell'lerde kayıtlı çıkış almakta mümkündür. Bunun için kombinasyonel çıkışı programlanarak flip-flop girişine bağlanır ve çıkışından kayıtlı çıkış elde edilir. Bu işlem CPLD'lerin sıralı mantık devrelerinde de kullanılabileceğini gösterir.



MAX II CPLD'ler LUT mantık, Klasik CPLD'ler ise AND/OR Dizgesi içerir

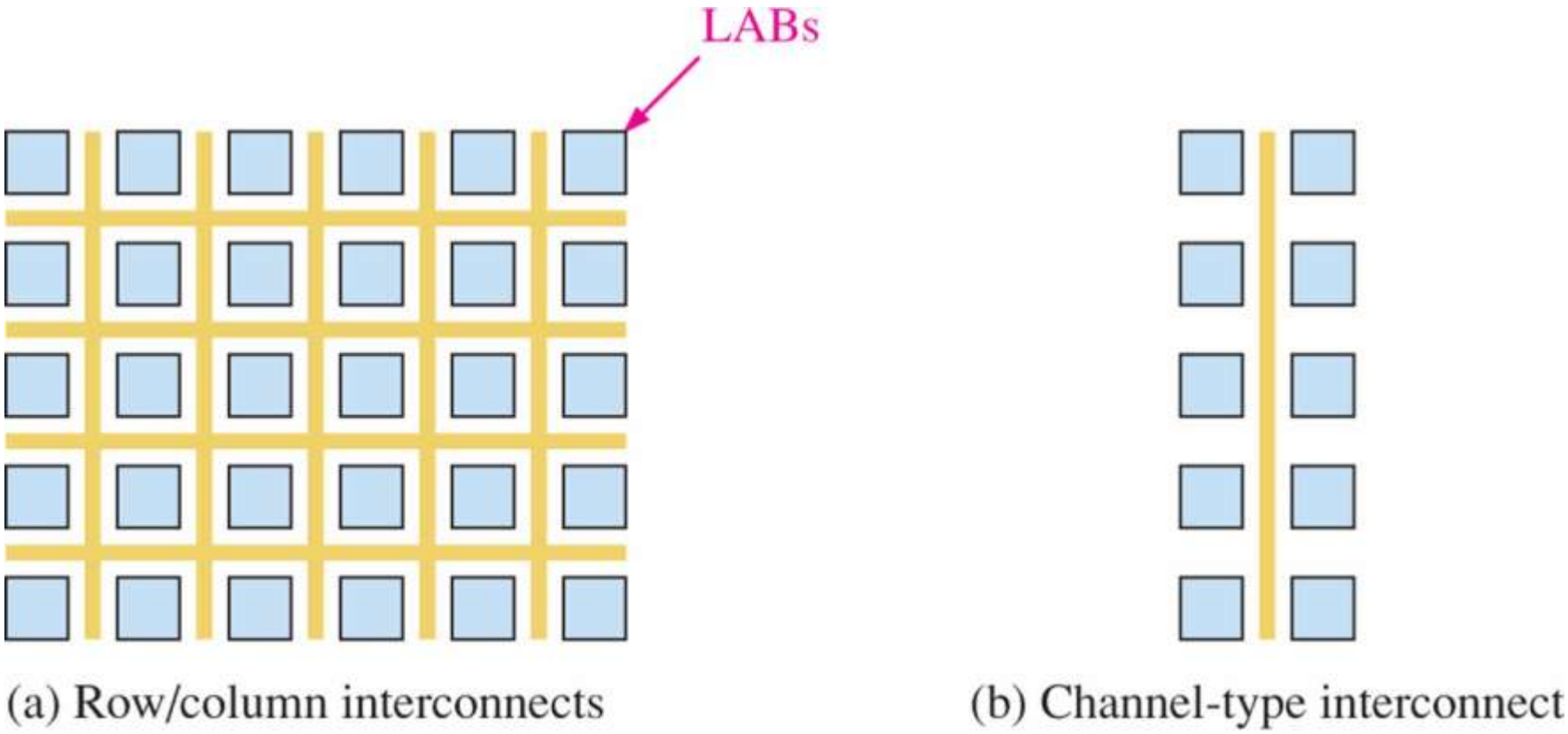


(a) Look-up table logic. A 1 is stored at each product term address.



(b) AND/OR array logic

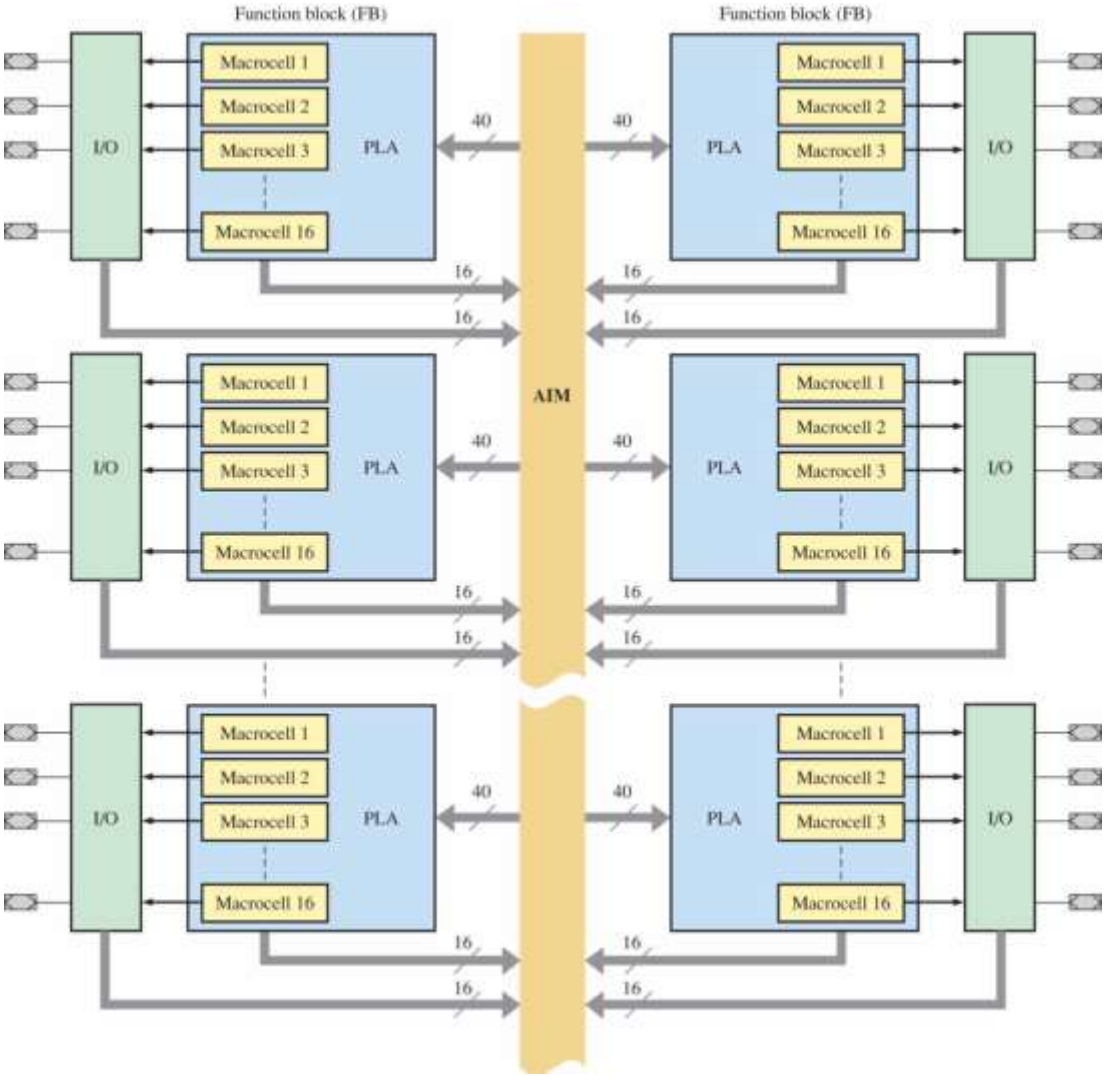
MAX II CPLD'lerin satır ve sütün iç bağlantı düzeneği klasik CPLD'ler ise kanal tipi iç bağlantı düzeneği içerir.



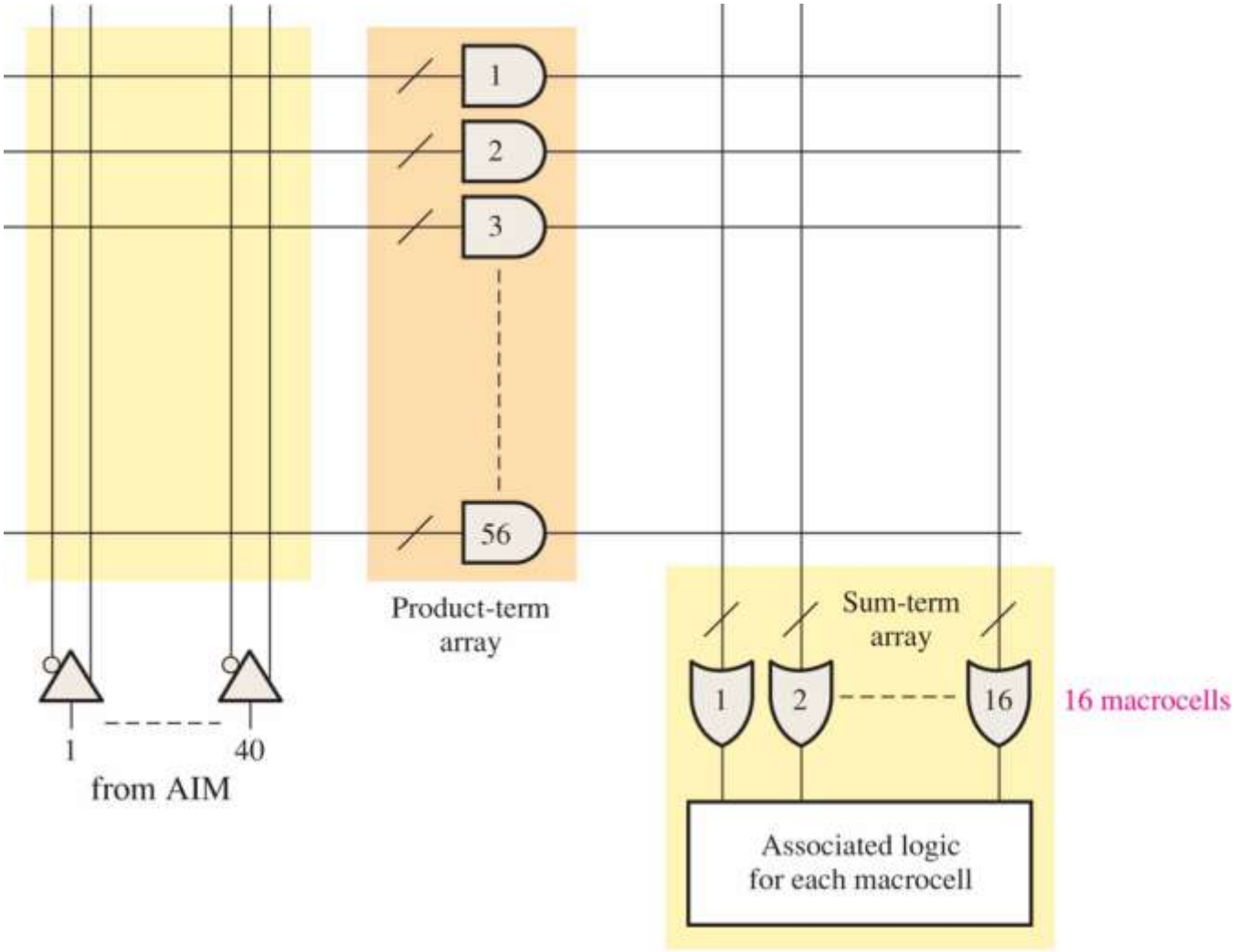
(a) Row/column interconnects

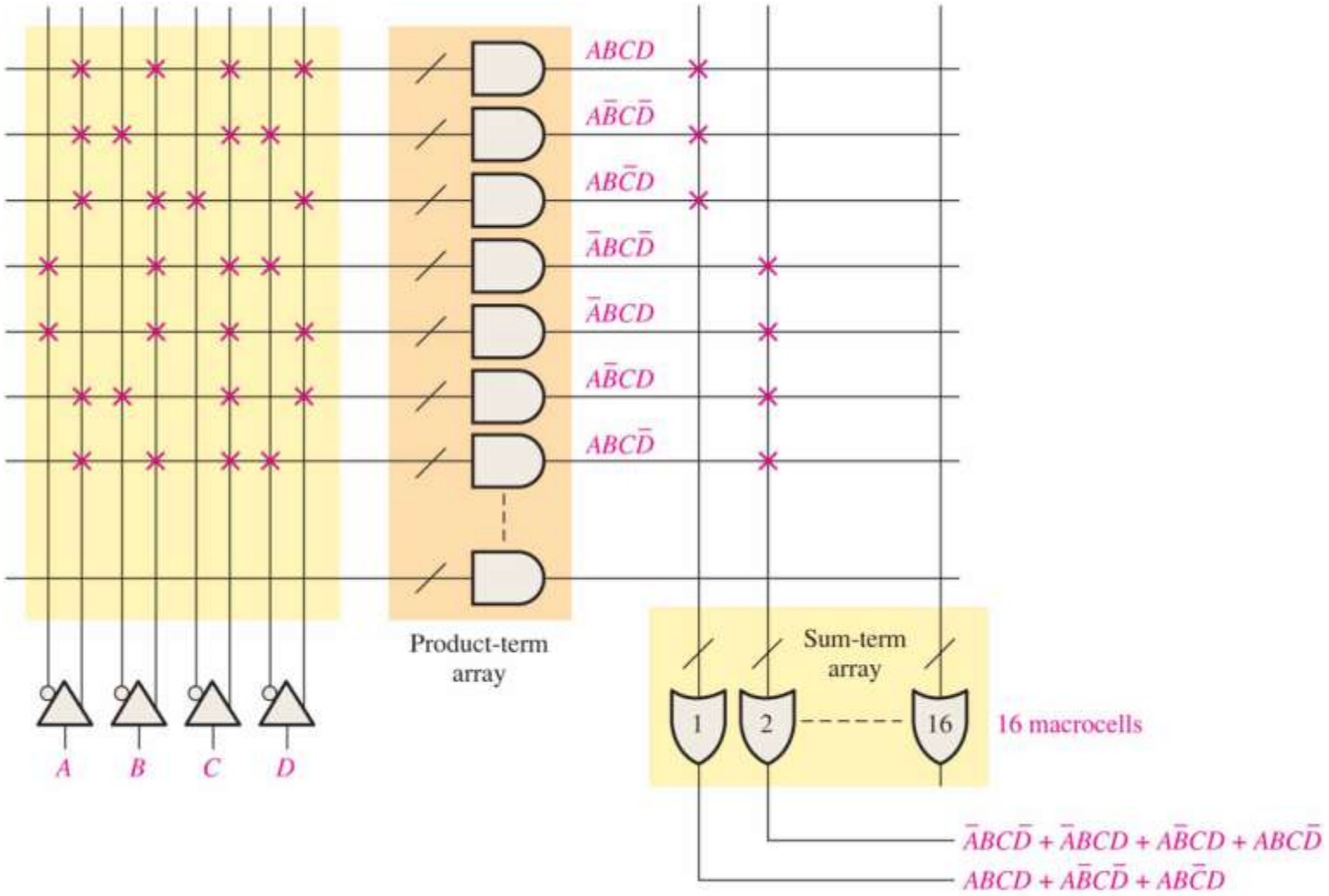
(b) Channel-type interconnect

Xilinx CoolRunner II CPLD'nin yapısı

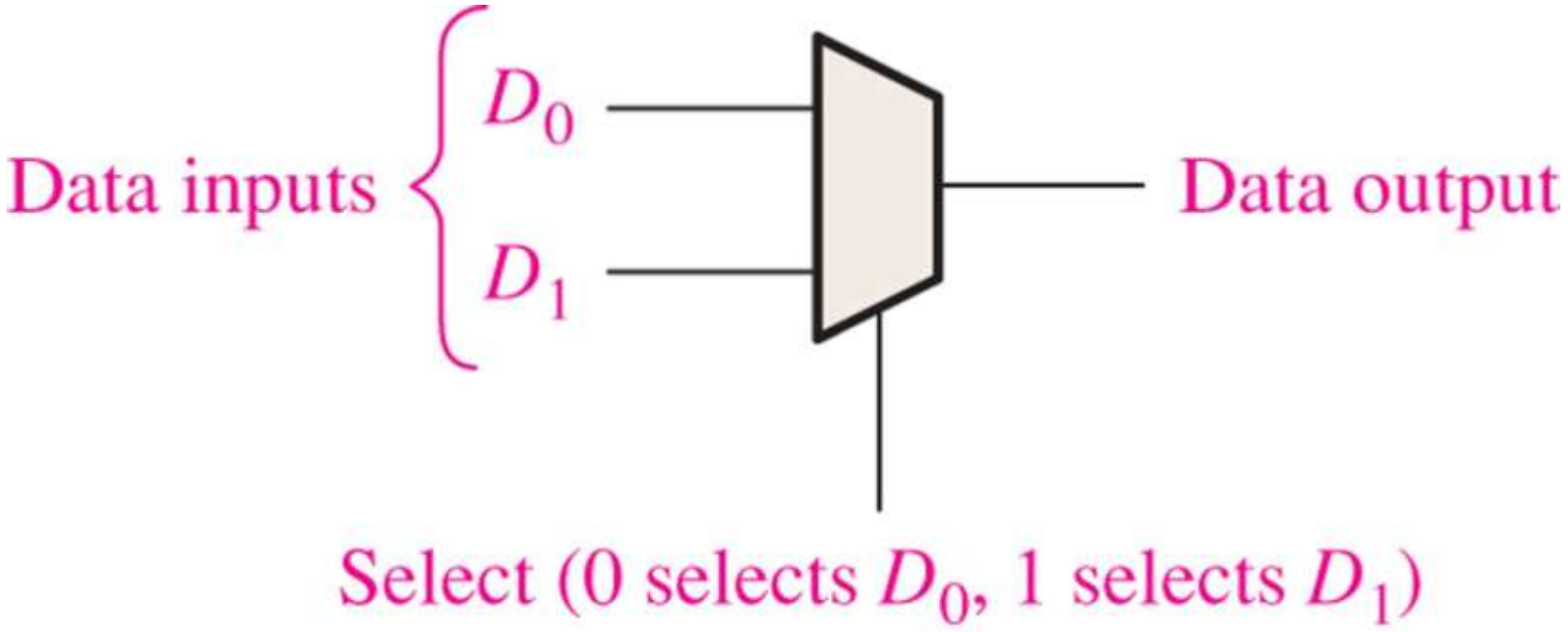


CoolRunner II'nin PLA yapılı fonksiyonel diyagramı

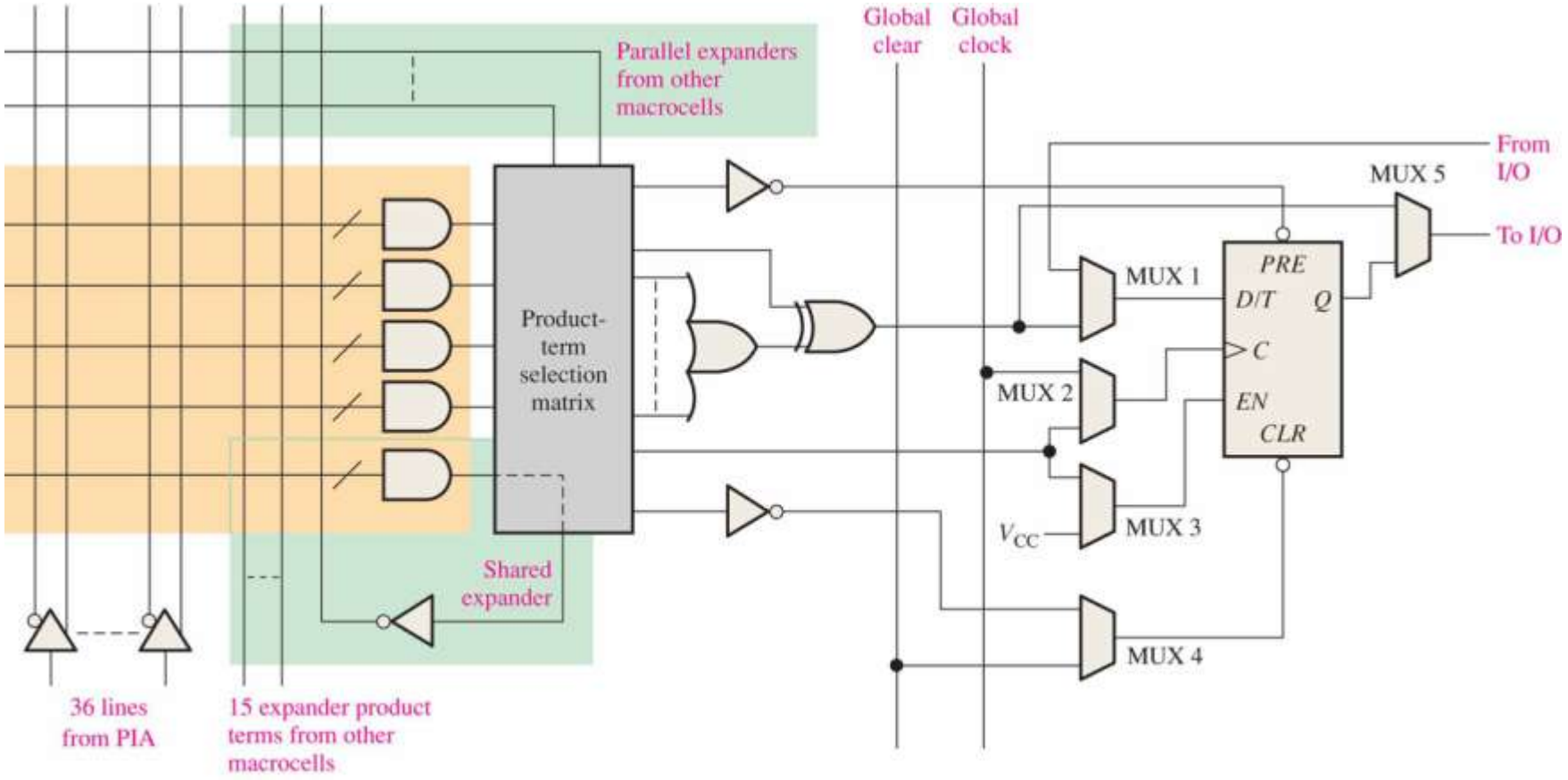



ABCD
$A\bar{B}C\bar{D}$
$AB\bar{C}D$
$\bar{A}BC\bar{D}$
$\bar{A}BCD$
$A\bar{B}CD$
$ABC\bar{D}$
Product-term array
Sum-term array
1
2
16
16 macrocells
A
B
C
D
$\bar{A}BC\bar{D} + \bar{A}BCD + A\bar{B}CD + ABC\bar{D}$
$ABCD + A\bar{B}C\bar{D} + AB\bar{C}D$

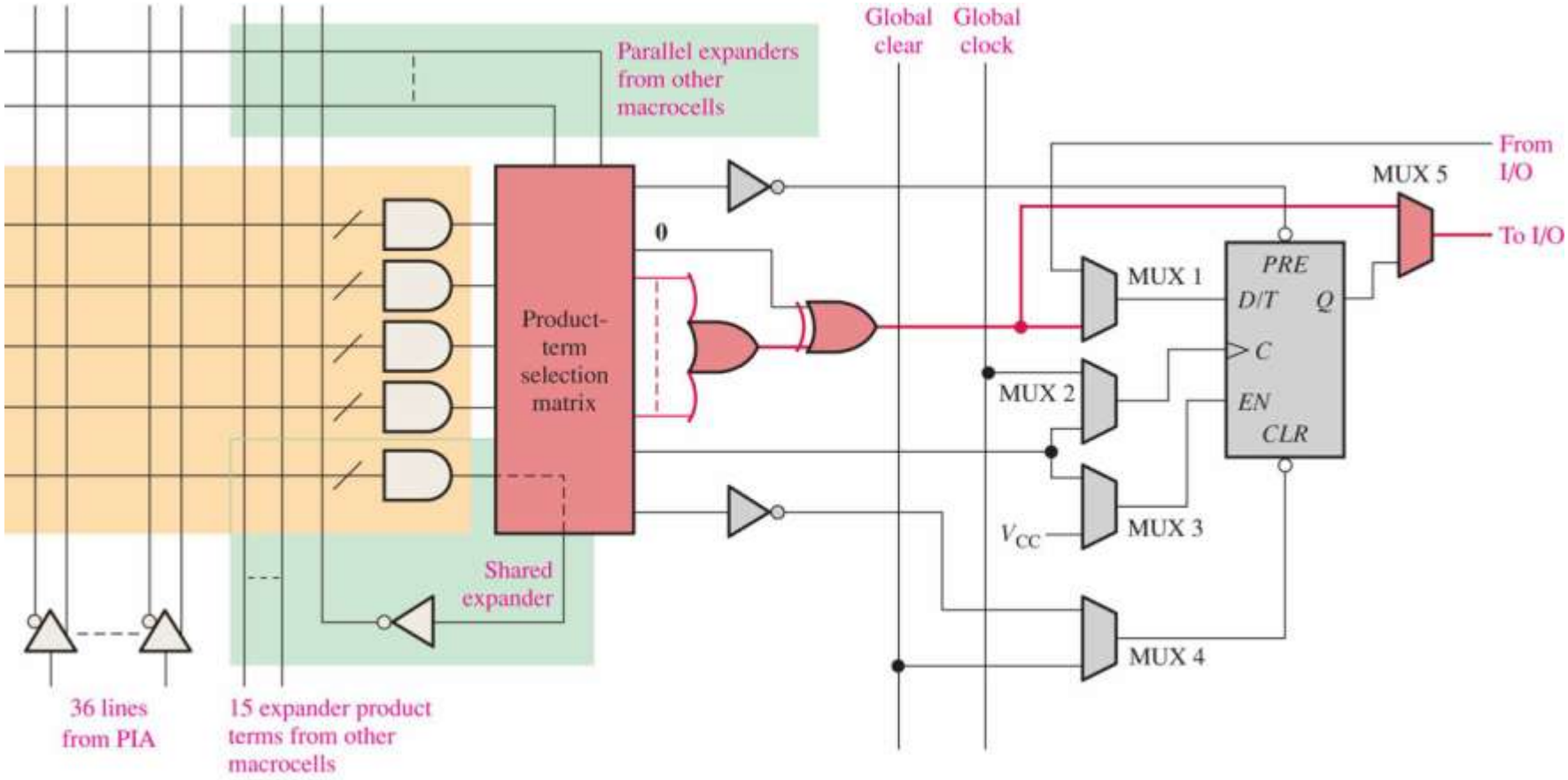
Multiplexer simgesi



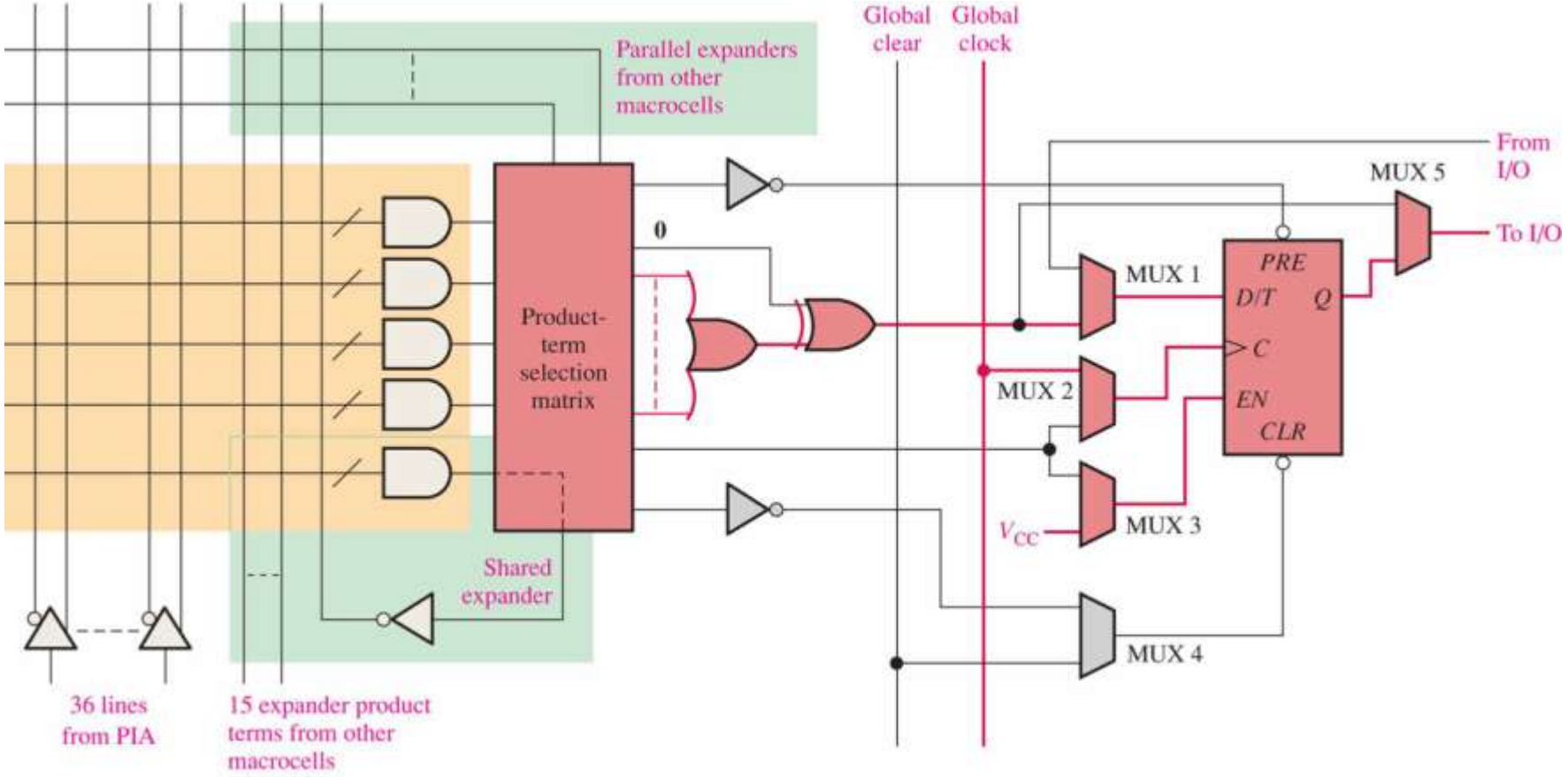
Altera MAX 7000 ailesi CPLD'ler



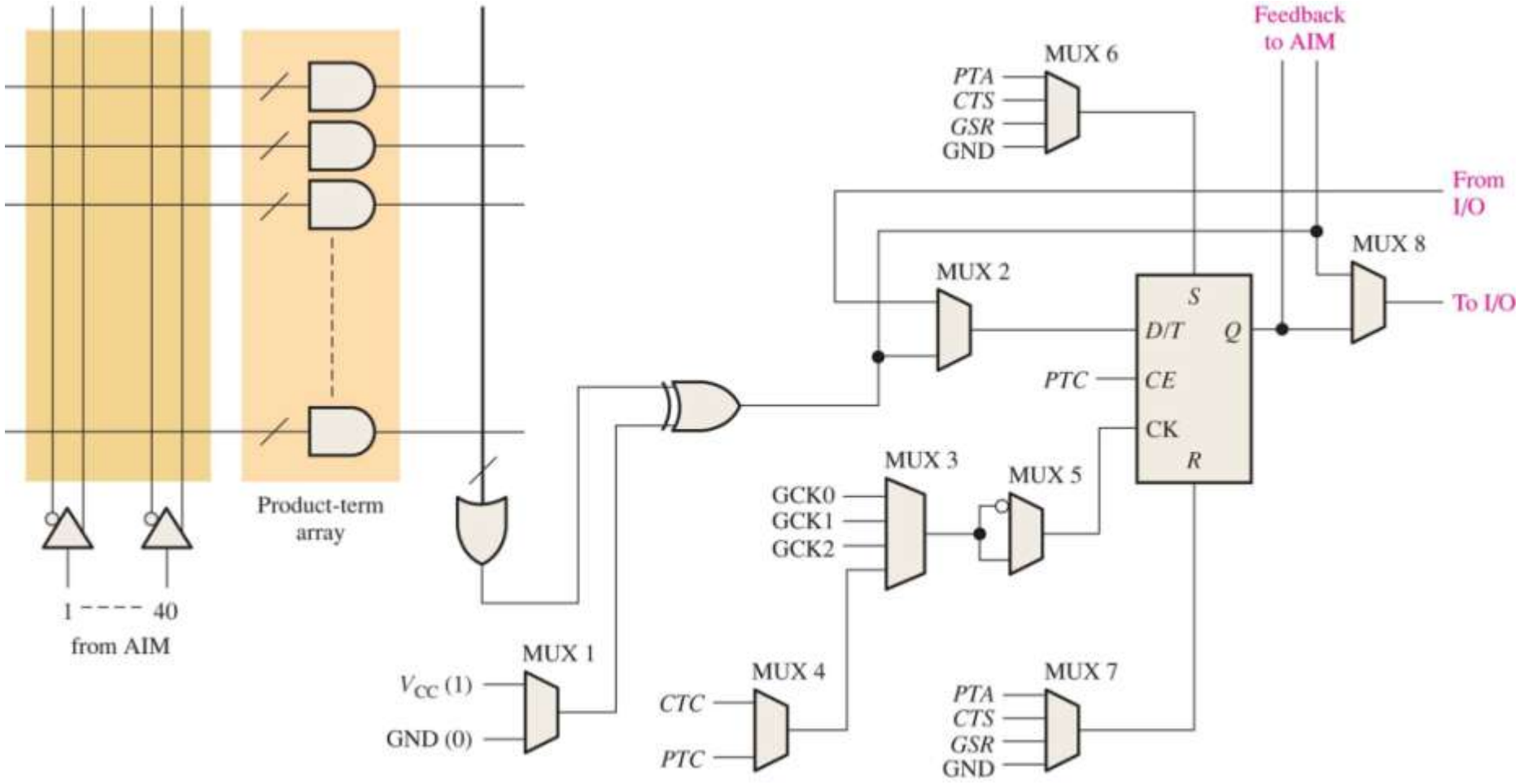
Toplamların çarpımı fonksiyonunu elde etmek için macrocell'in kullanımı



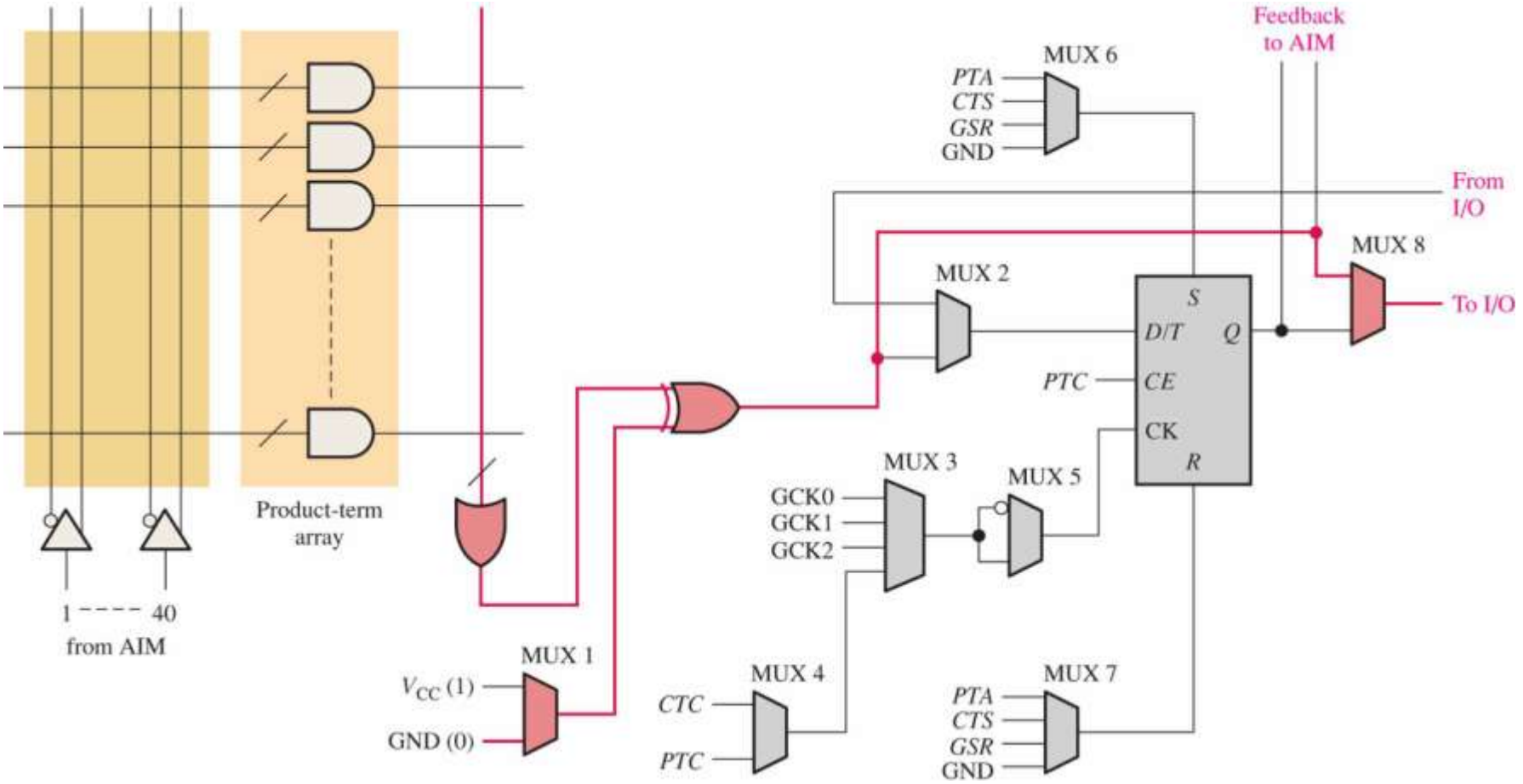
Kayıtlı çıkış elde etmek için macrocell'in kullanımı



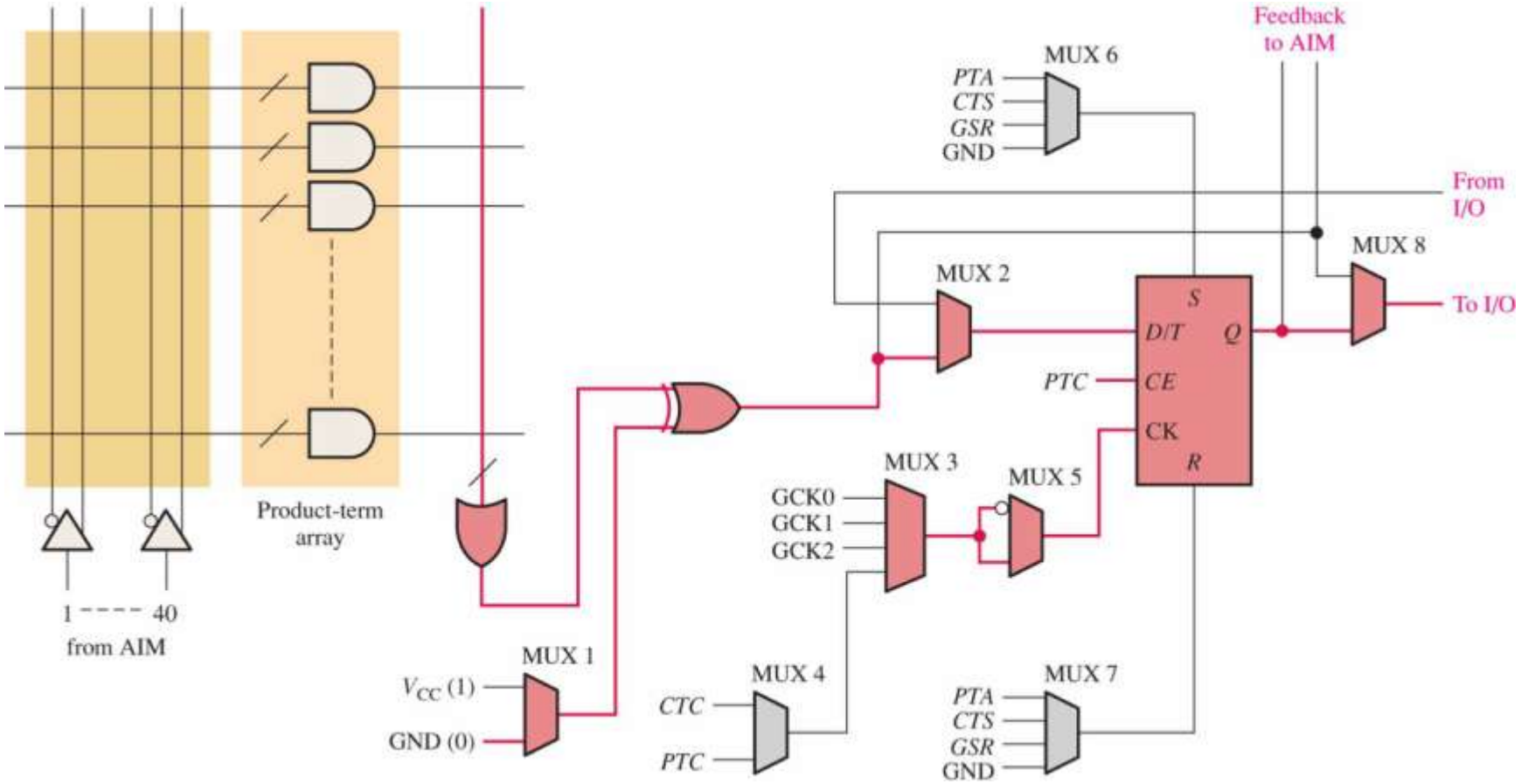
Xilinx CoolRunner II CPLD'nin macrocell yapısı.



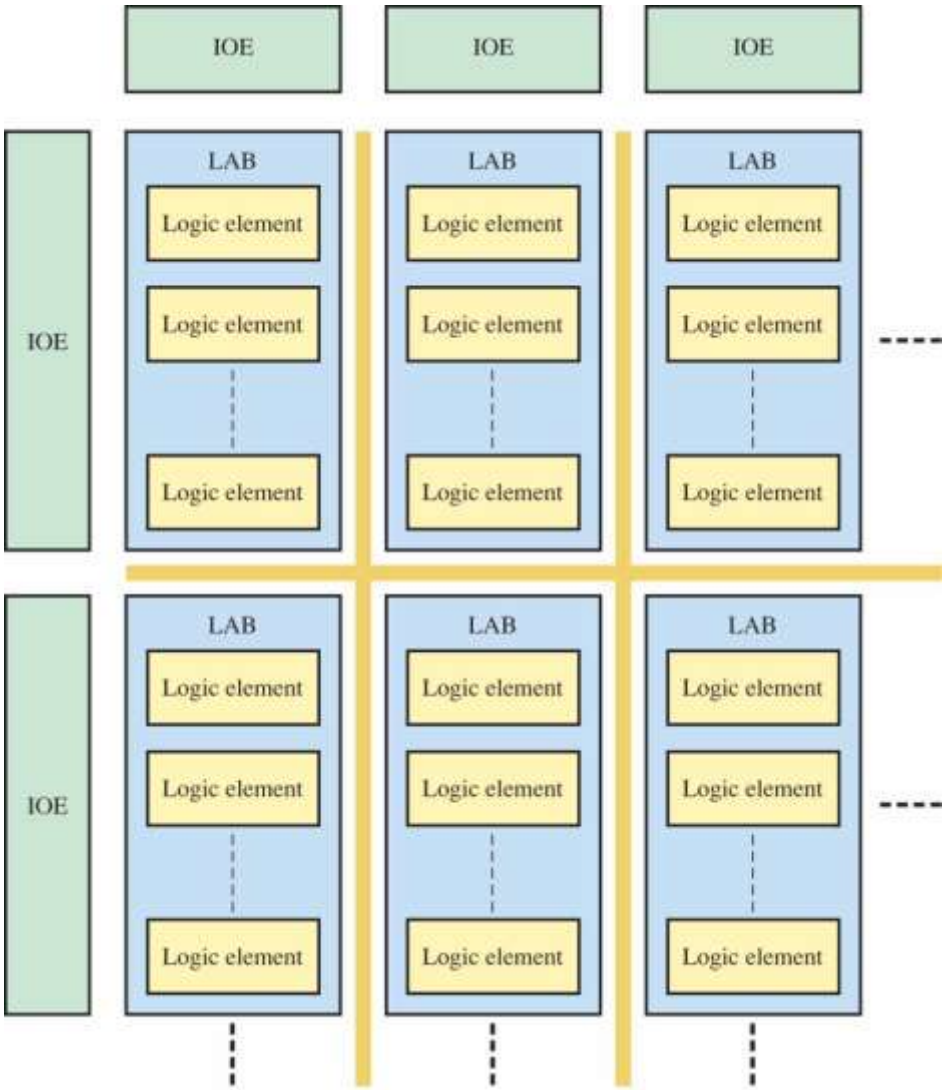
Kombinasyol mantık çıkışının kullanılması



Kayıtlı (sıralı) mantık çıkışının kullanımı



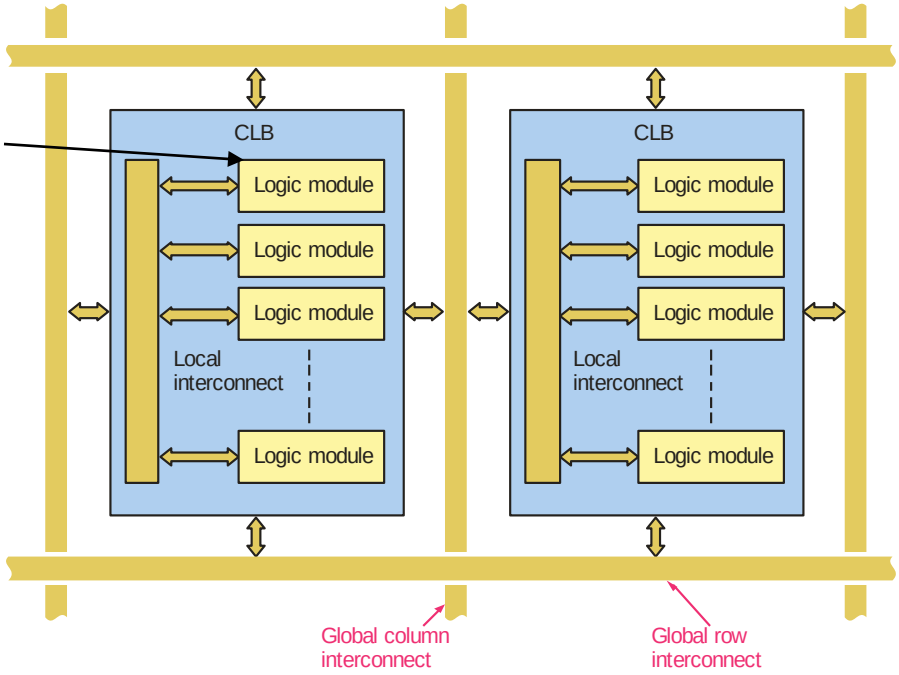
MAX II CPLD'nin blok diyagramı



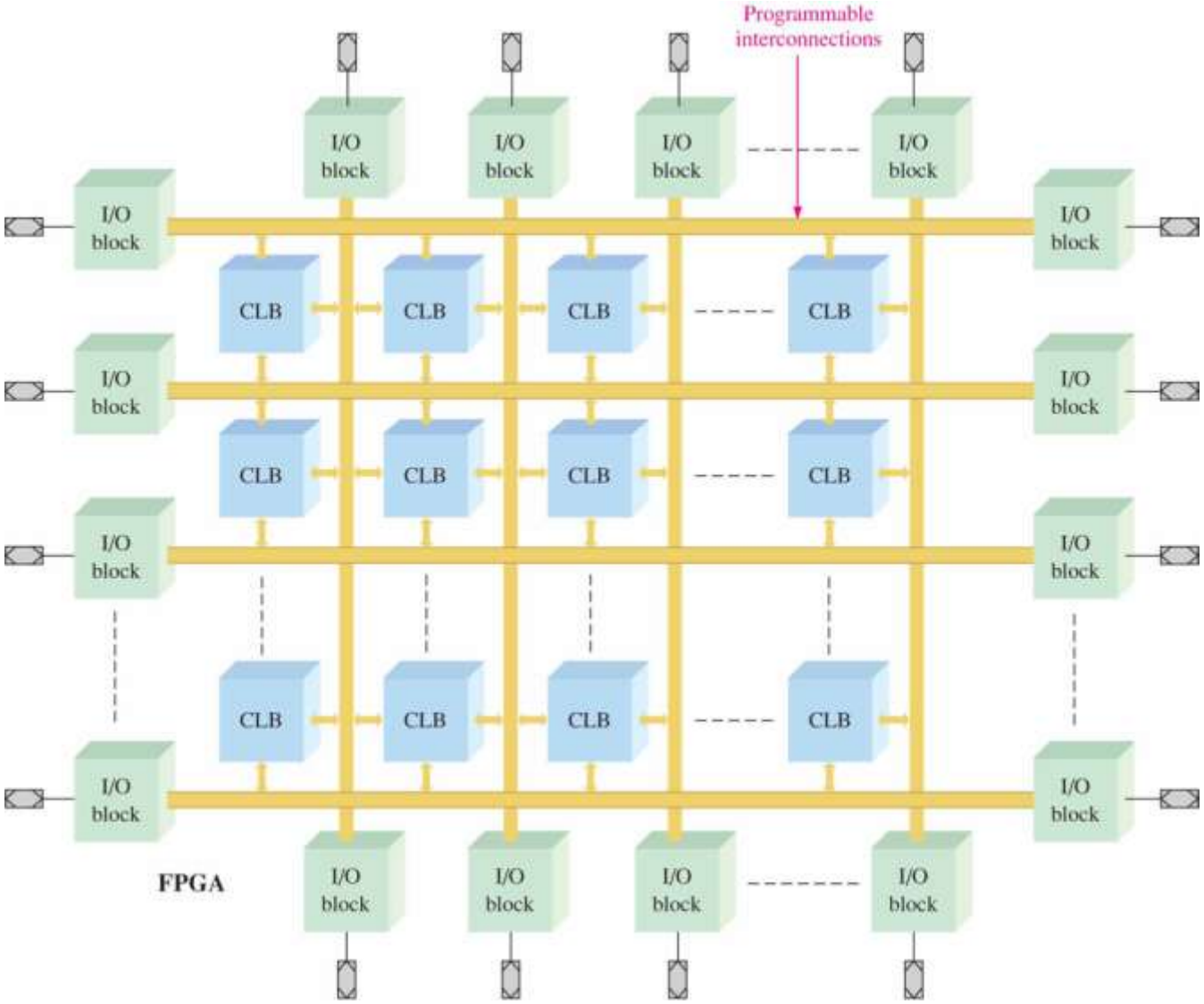
FPGA'ler

Alan programlanabilir geçit dizgesi (field programmable gate array), **FPGA**, CPLD'lerden farklı bir mimari kullanır. Tanımlanabilir, ayarlanabilir, lojik bloklar (configurable logic block), **CLB**, FPGA'lerin temel taşıdır. CLB'lerin sayısı ihtiyaca göre arttırılabilir.

CLB'ler satır ve sütün düzeneğine göre yapılandırılmıştır. CLB'ler içerisinde **logic moduller** birbiri ile iç bağlantıları ile bağlanabilir. Lojik moduller genellikle başvuru tabloları (LUT), ile desteklenir. flip-flop, ve MUX kullanılarak istenildiğinde kombinasyonel mantık çışı veya sıralı mantık çıkışı elde edilebilir.



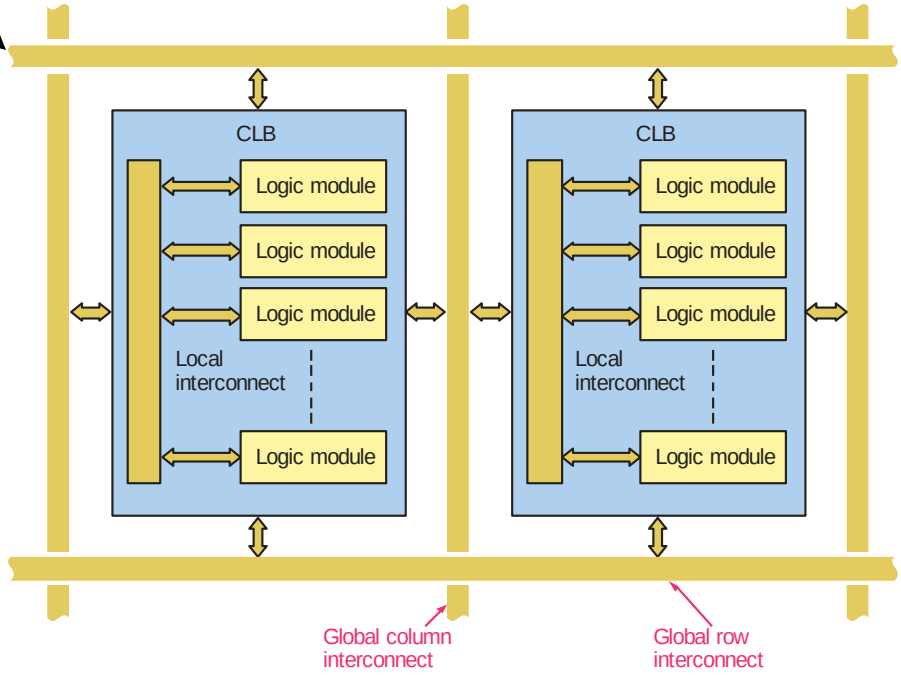
FPGA'in yapısı. CLB, (configurable logic block)



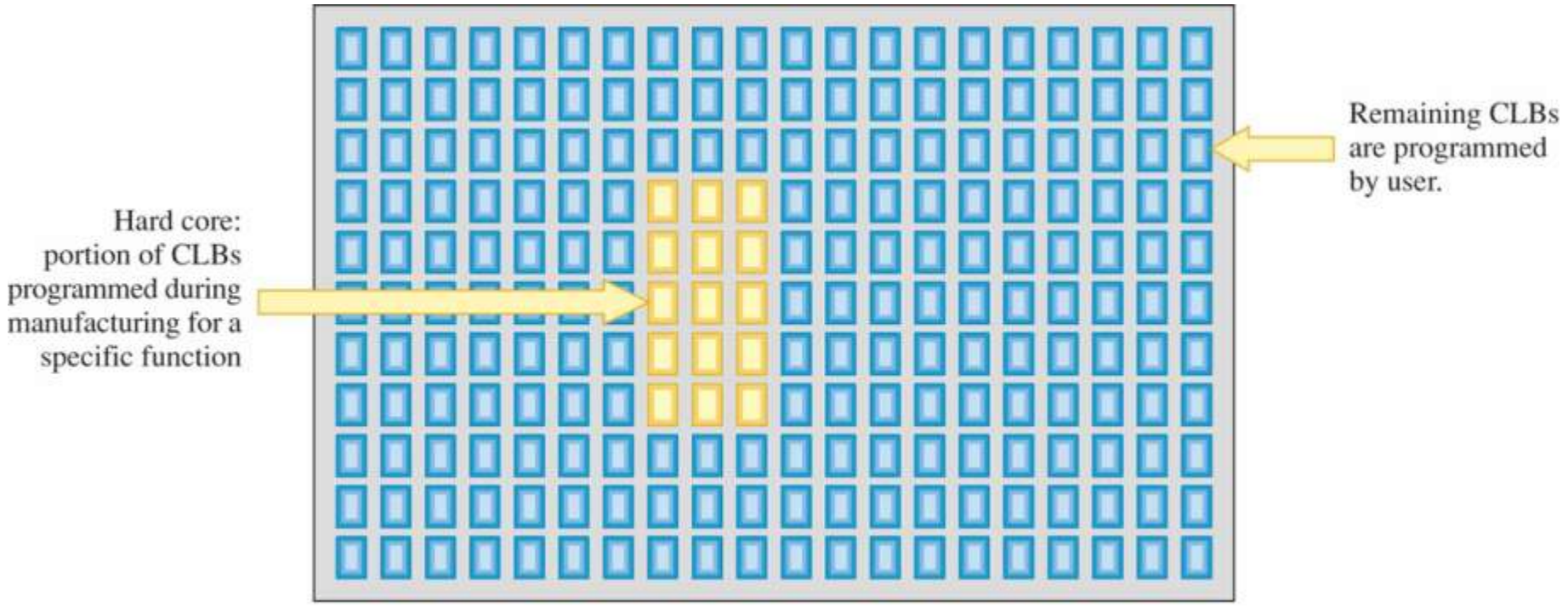
FPGA'ler

Mantık modüller kombinasyonel, kayıtlı veya karışık olarak kullanılabilir. Genel iç bağlantılar her CLB'ye ortak kullanılan işaretler ile giriş ve çıkış işaretlerini dağıtır (saat işaretide bunlara dahildir).

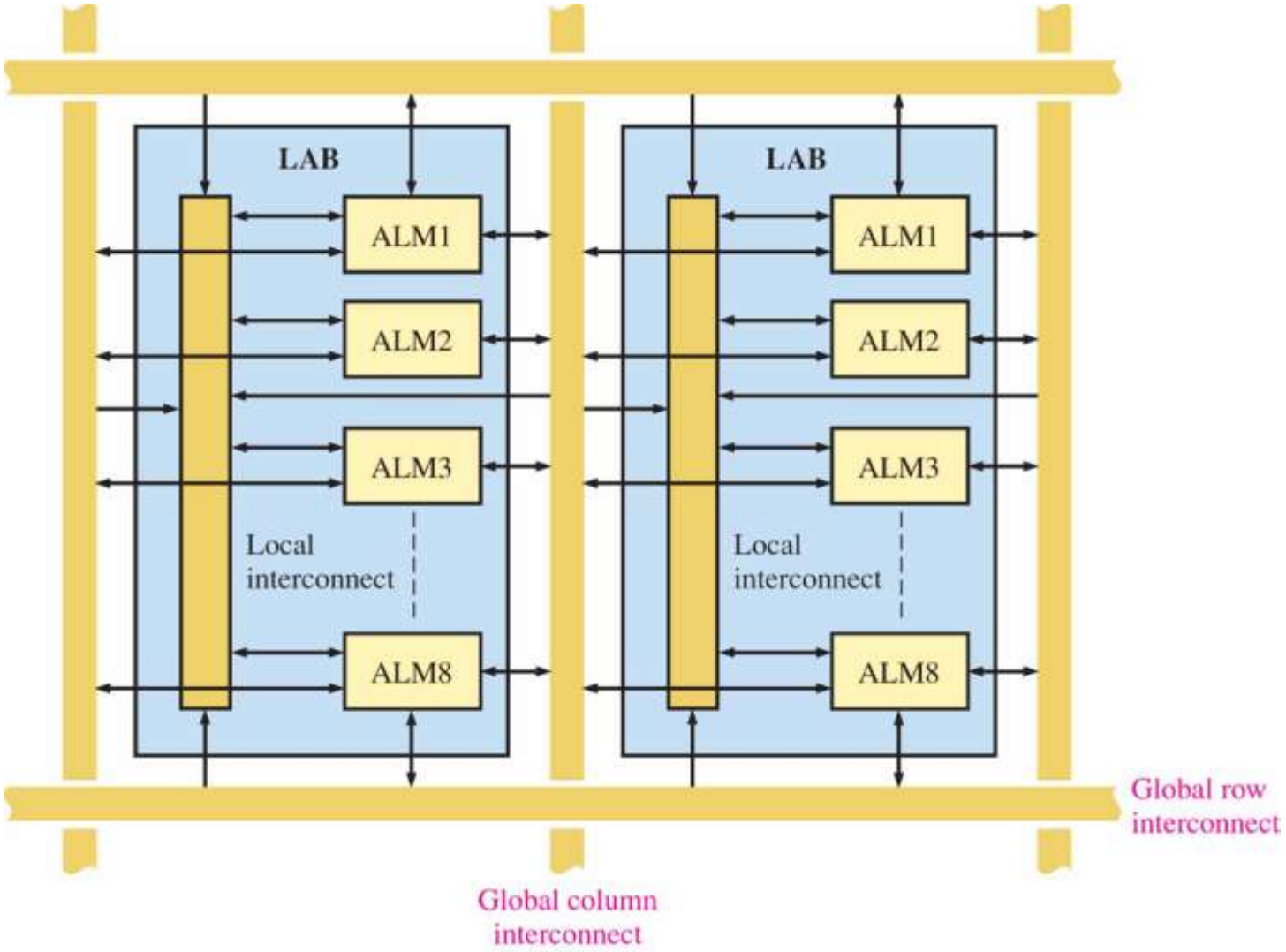
FPGA'ler üretici tarafından programlanan hard çekirdeğe sahip olabilirler. Bu tür FPGA'ler genel işlemler için kullanılır. I/O bağlantısı için kullanılan FPGA bu türe bir örnektir.



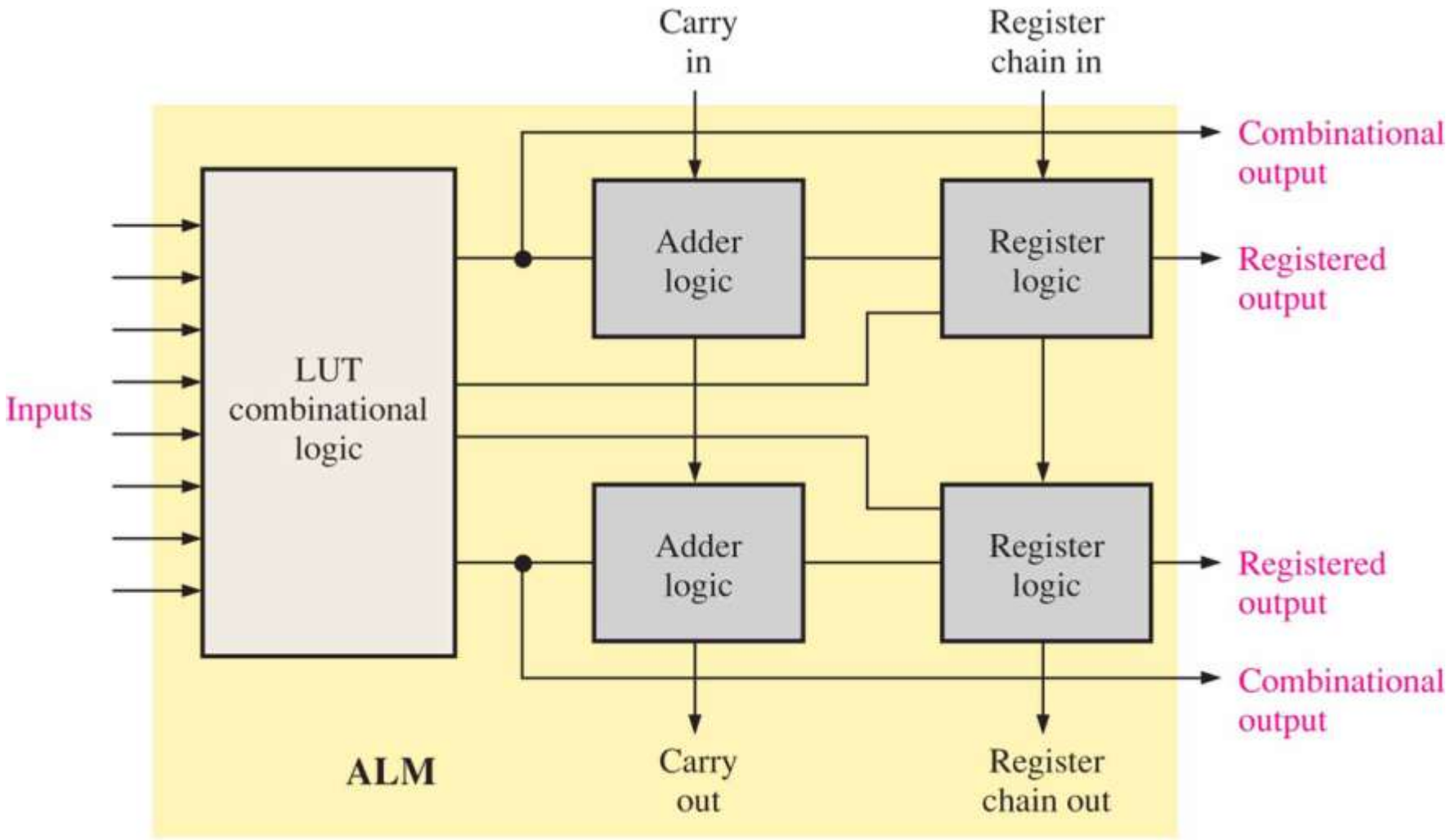
Hard çekirdek gömülü FPGA.



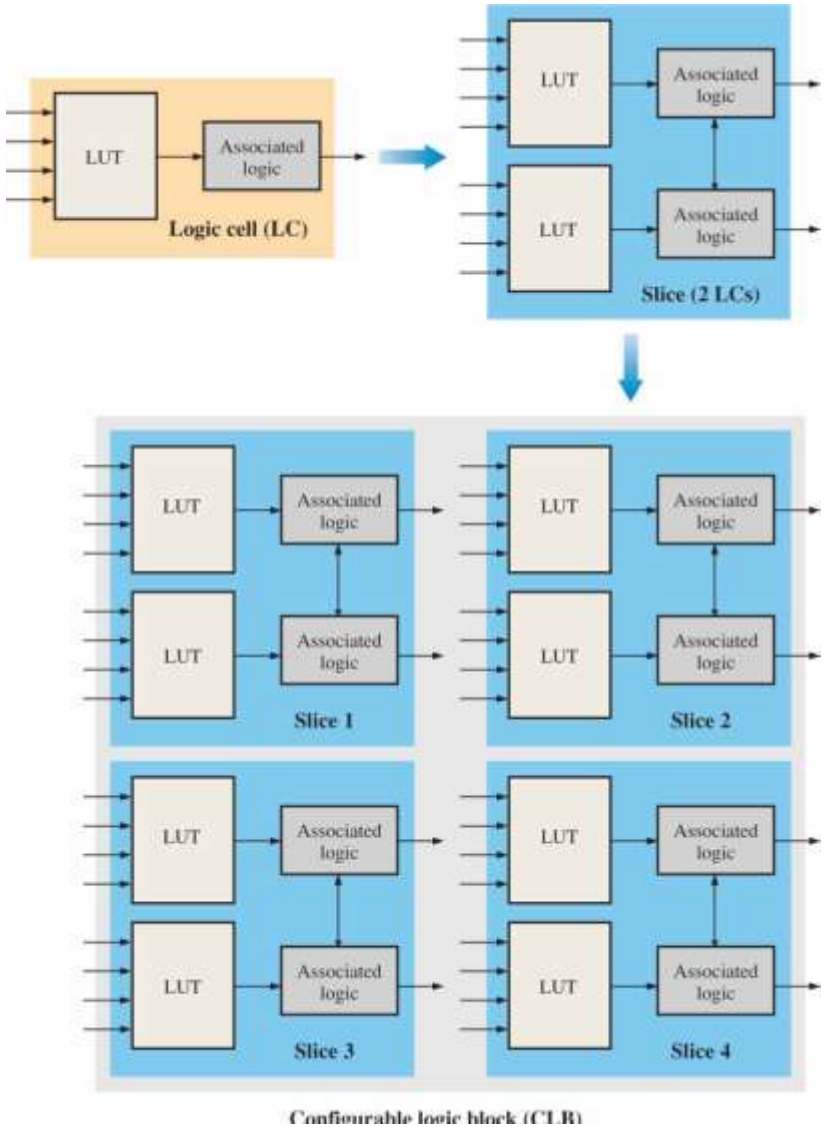
Stratix II FPGA LAB (logic array block) Yapısı, ALM (adaptive logic modules).



Stratix II 'nin ALM'lerin yapısı

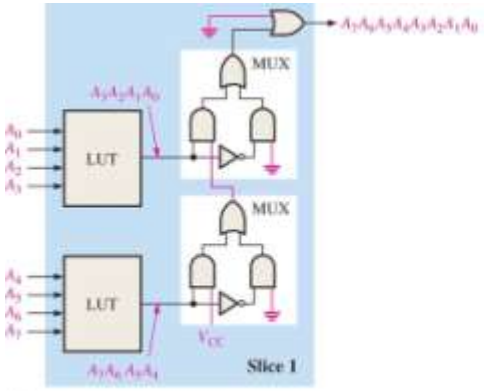


Virtex FPGA'in CLB yapısı

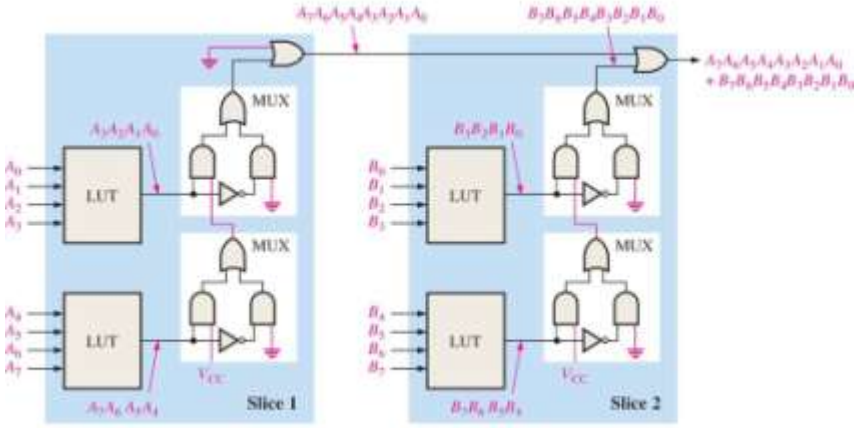


Configurable logic block (CLB)

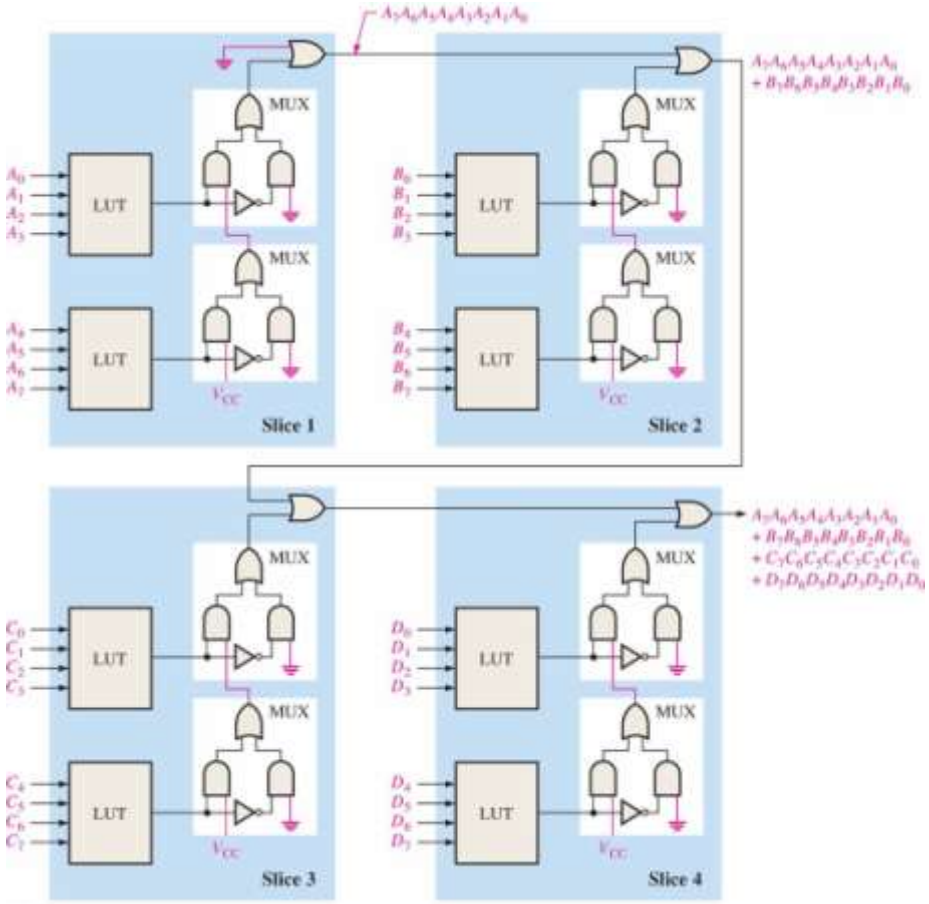
Toplamların çarpımı fonksiyonunun genişletilmesi



(a)



(b)



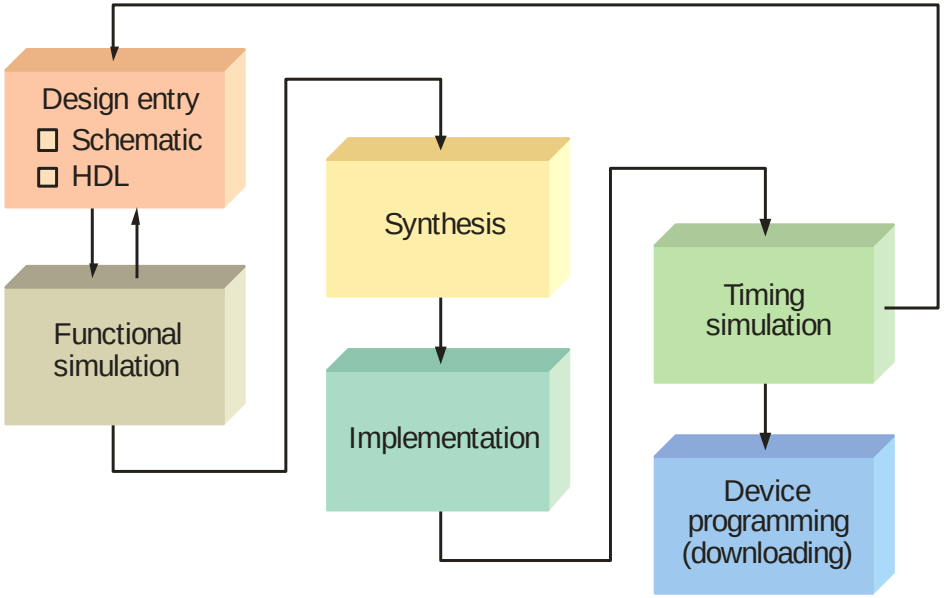
(c)

Programlanabilir Mantık Devre Yazılımları

PLD üreten her firma kendi ürününe uygun yazılımı sağlarlar. Yazılımın yazılması, test edilmesi ve PLD'ye yüklenmesi aşamalarını aşağıdaki gibi özetleyebiliriz..

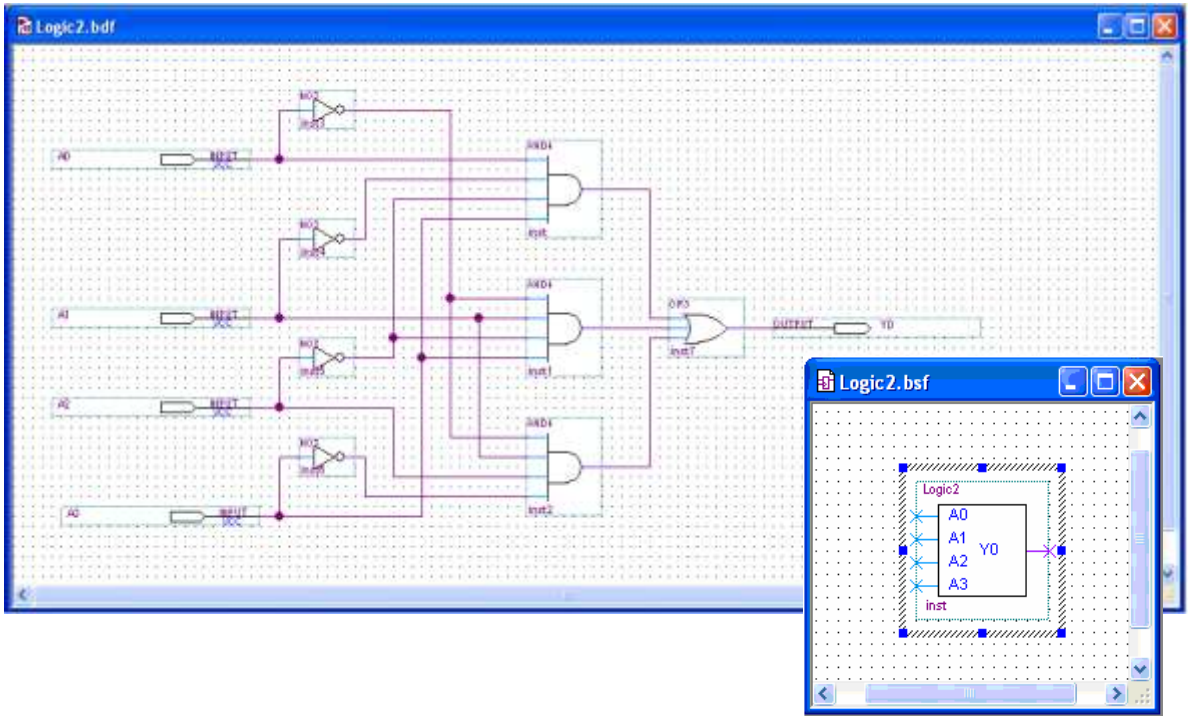
Birinci adım gerçekleştirmek istediğimiz mantık fonksiyonunun devresini tanımlamaktır. Bu işlem iki tür yapılır:

1. Şeması çizilerek
2. Donanım tanımlama yazılımı ile komut yazarak (Hardware description language), HDL.



Programlanabilir Mantık Devre Yazılımları

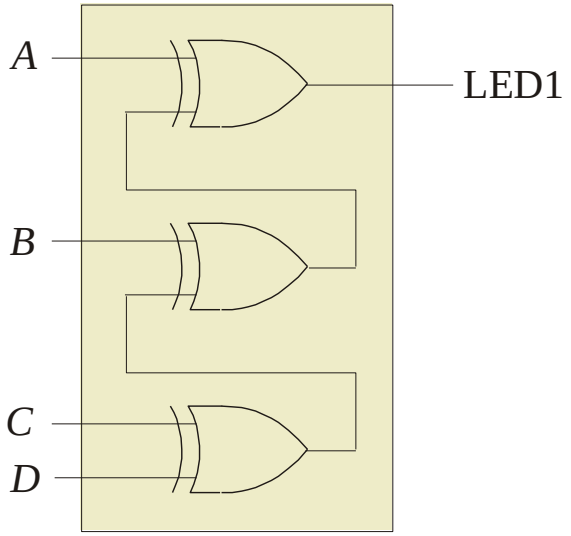
Mantık fonksiyonun şekil çizerek tanımlanması, bilgisayar ekranında temel mantık geçitlerini kullanarak mantık devresinin çizilmesi ile olacaktır. Daha sonra küçük şekilde görüldüğü gibi şekil blok haline dönüştürülecektir. Bu yöntemin avantajı HDL hakkında ayrıntı bilmek zorunda değilsiniz.



Programlanabilir Mantık Devre Yazılımları

Kod yazarak fonksiyonun tanımlanmasında bu iş için firmalar tarafından geliştirilen HDL (hardware description language) dili kullanılır. Önek; VHDL veya Verilog gibi.

VHDL programı iki kısımdan oluşur: Birinci kısım giriş (**entity)** ikinci kısım gövde (**architecture)** olarak adlandırılır. Giriş kısmı, girişleri, çıkışları ve değişkenleri tanımlanır. Gövde kısmında ise değişkenler arası boolean fonksiyonlarını tanımlanır. VHDL eşitliklerini bu programlama dilini bilmesenizde anlayabilirsiniz.



Örnek: LED1 çıkışı için VHDL fonksiyonunu yazalım;

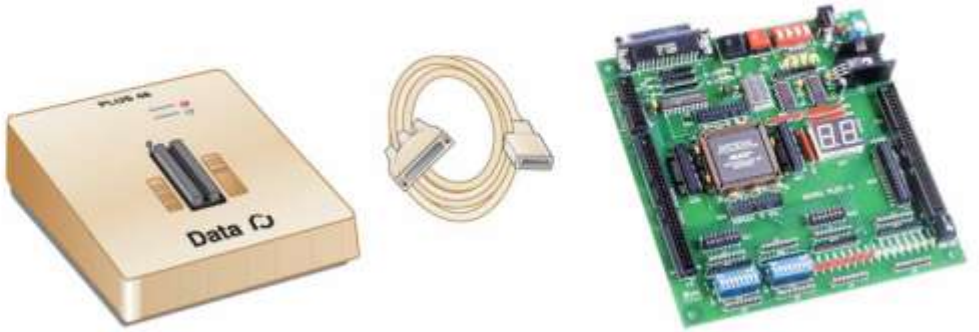
LED1 <= ((D **XOR** C) **XOR** B) **XOR** A;

SPLD, CPLD, veya FPGA Programlama Ortamı

(a) Computer

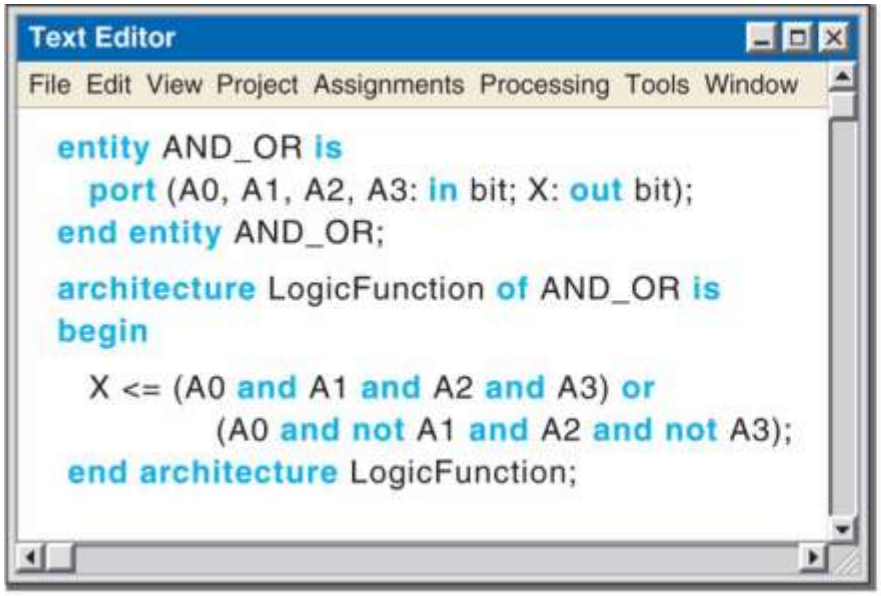
(b) Software (CD or Website download)

(c) Device

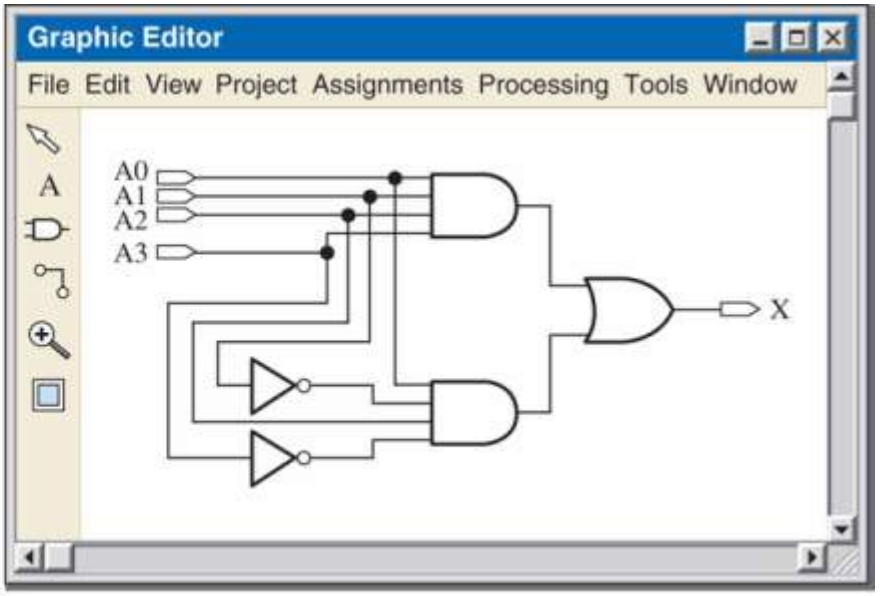


(d) Programming hardware (programming fixture or development board with cable for connection to computer port)

Şema çizme veya komut yazma ile fonksiyonun tanımlanması



(a) Text entry using VHDL to describe an AND-OR logic circuit

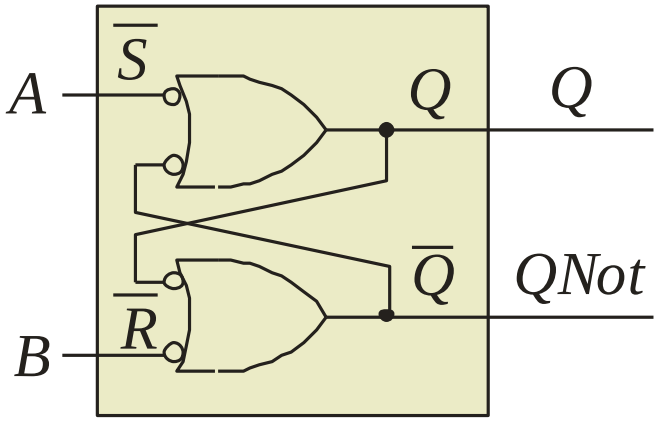


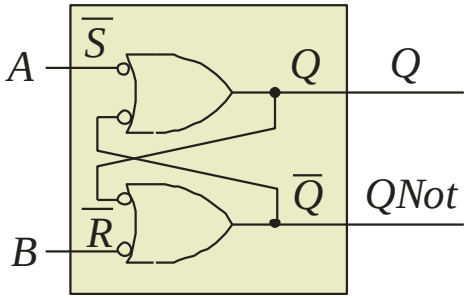
(b) Schematic entry of the same AND-OR logic circuit entered in (a)

Programlanabilir Mantık Devre Yazılımları

VHDL programı sık kullanılan fonksiyonları tanımlamaya ve daha sonra başka kod yazarken kullanmayı imkan sağlar.

Örnek; aktif düşük girişli $\overline{S}$-$\overline{R}$ tutucu devresi





Giriş Kısmı

```
entity S_RLatch is
 port (A, B: in bit; Q, QNot: inout bit);
end entity S_RLatch;
```

Giriş ve çıkış değişkenlerinin isimleri

Gövde kısmı

```
architecture Behavior of S_RLatch is
  begin
    Q  <=  not A or not QNot;
    QNot <=  not B or not Q;
end architecture Behavior;
```

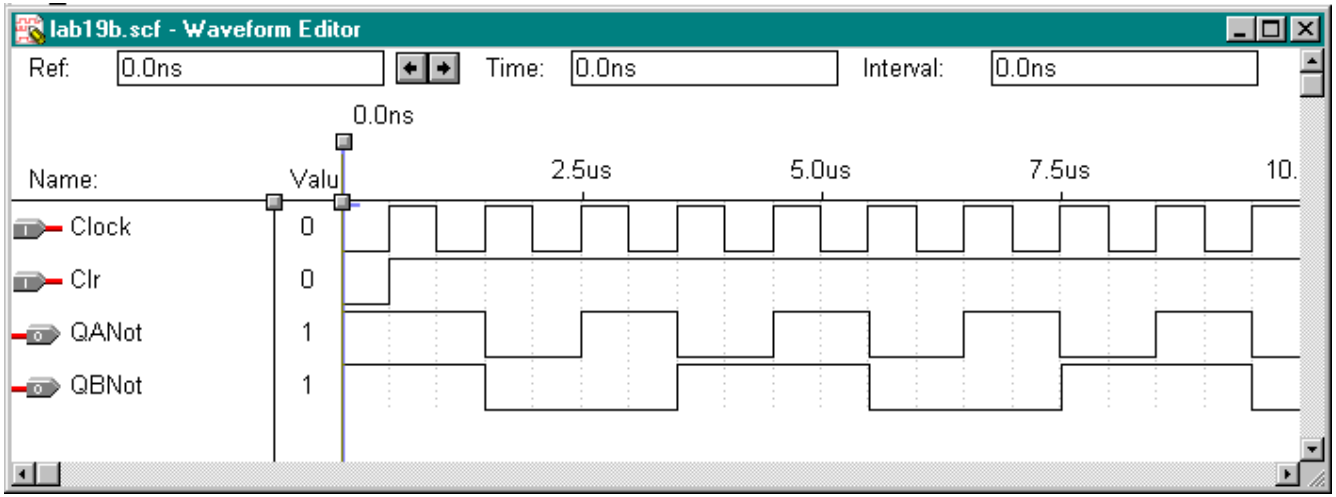
Devrenin Boolean tanımlaması

Sağdaki fonksiyonun çıkışı soldakine gönderilir.

Simulasyon

HDL kodları yazıldıktan sonra (VHDL gibi), fonksiyonel olarak devre simule edilir. Bu simulasyon HDL'in bir parçasıdır. Ve çıkış değişkenlerinin girişe göre değişim eğrilerini elde edebilirsiniz. Böylece hataları belirler ve kodlarda düzeltme yapabilirsiniz.

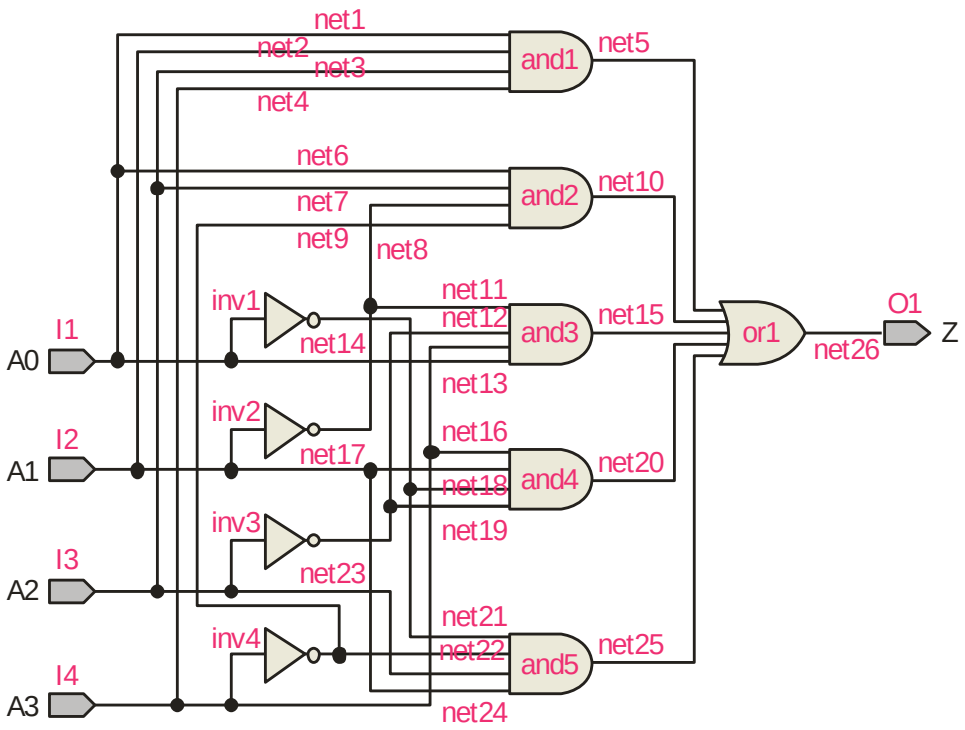
Örnek: İki bitlik sayıcının fonksiyonel testi sonucu elde edilen giriş işaretlerine göre çıkışların değişim diyagramı.



Sentezleme

Synthesis

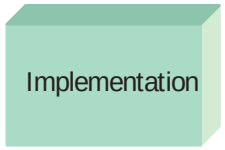
Simulasyon sonrası program fonksiyonu sadeleştirir ve gereksiz bağlantıları kaldırır ve yeni duruma uygun **netlist**, kodunu oluşturur. (netlist bağlantı listesidir.)



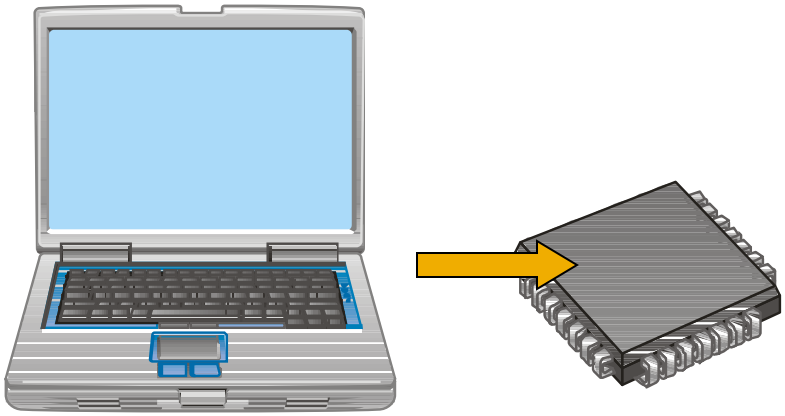
Netlist

Netlist (Logic3)
net<name>: instance<name>, <from>; <to>;
instances: and1, and2, and3, and4, and5, or1, inv2, inv3, inv4;
Input/outputs: I1, I2, I3, I4, O1;
net1: and1, inport1; I1;
net2: and1, inport2; I2;
net3: and1, inport3; I3;
net4: and1, inport4; I4;
net5: and1, outport1; or1, inport1;
net6: and2, inport1; I1;
net7: and2, inport2; I3;
net8: and2, inport3; inv2,outport1
net9: and2, inport4; inv4,outport1
net10: and2, outport1; or1,inport2;
net11: and3, inport1; inv2,outport1
net12: and3, inport2; inv3,outport1
net13: and3, inport3; I4;
net14: and3, in...

Uygulama



Program netlistten sonra seçilen elemana uygun kodları oluşturur.

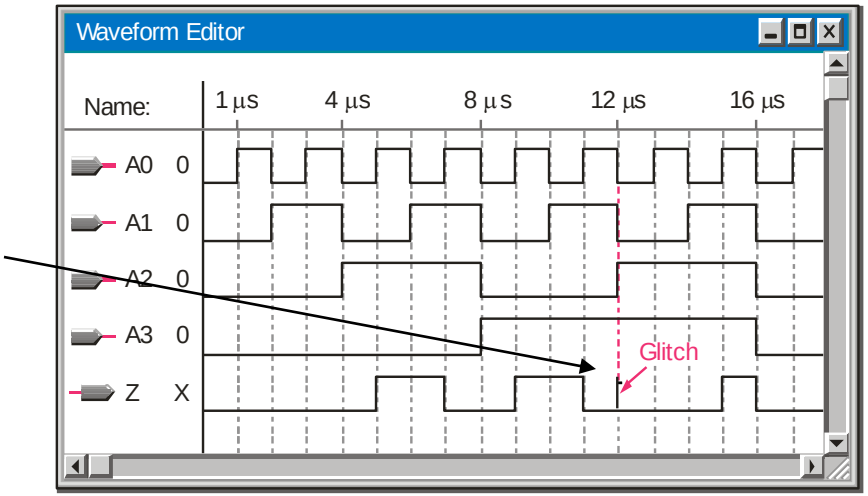


Zamanlama simulasyonu

Timing simulation

Seçilen elemana uygun kodlar oluşturulduktan sonra zamanlama simulaşyonu yapılır. Bunun yapılmasının sebebi her eleman kendi propagasyon gecikmesine shiptir. Seri bağlanan bloklar arası çıkışta istenmeyen uyumsuzluklar oluşabilir. Bu tür problemler hızlı olan işarete gecikme koyulması ile çözülür. Düzenleme sonrası tekrar zamanlama simulasyonu çalıştırılır.

Eğer zamanlama hatası oluştu ise hala düzeltme yapılabilir.

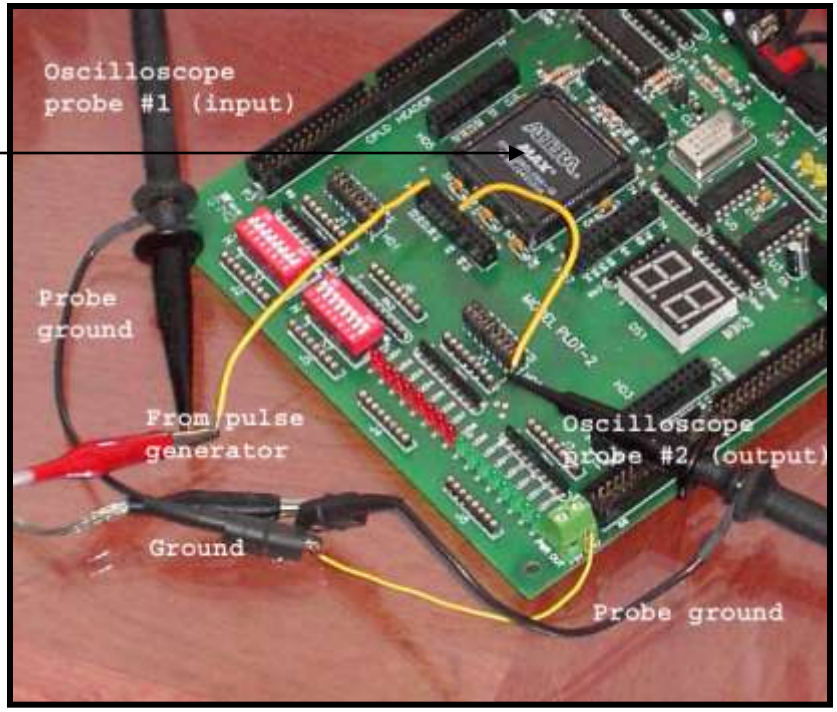


Aygıt Programlama

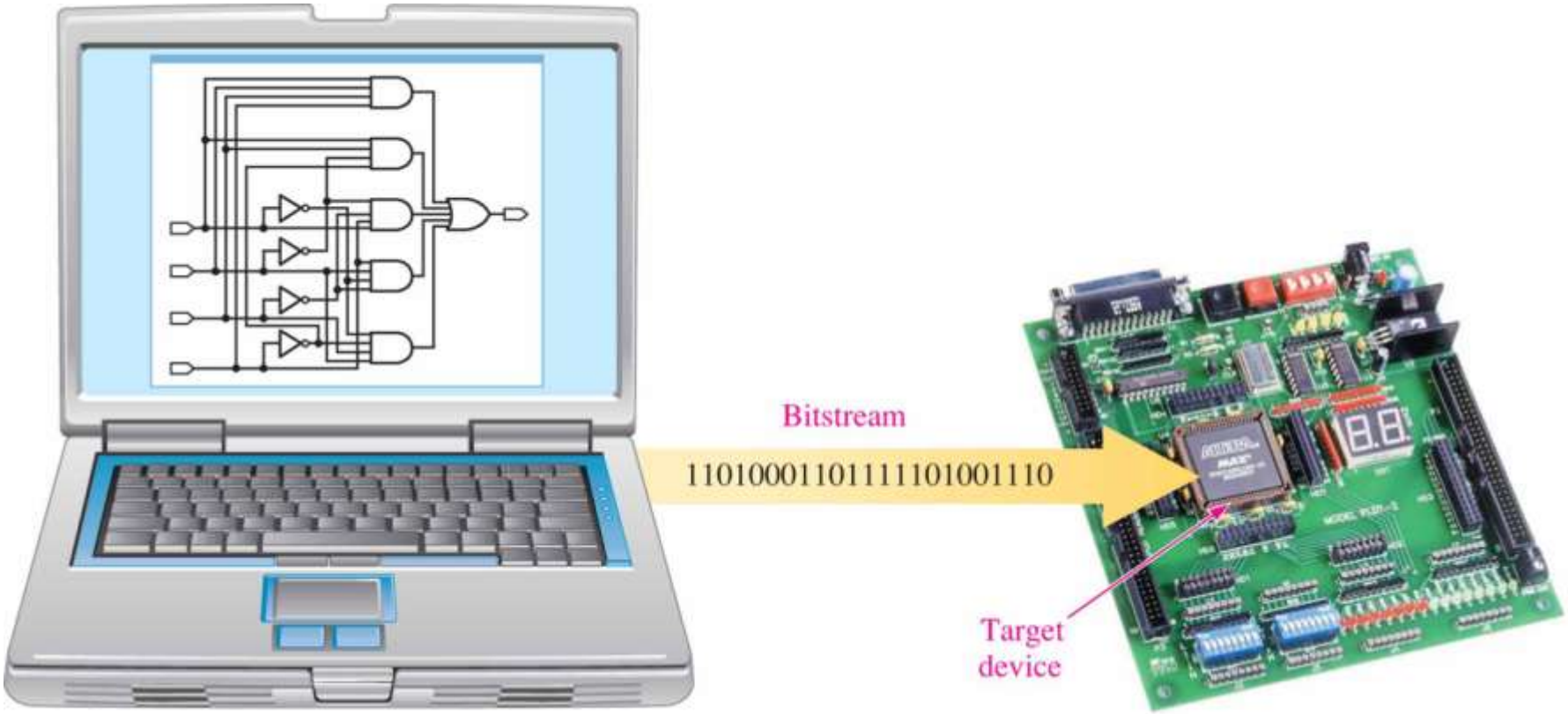
Device programming (downloading)

Son adım bilgisayarda elde edilen programı hedef PLA'ya göndermektir. Bu işlem için programlama kablosu ve arayüzüne gereksinim duyulur.

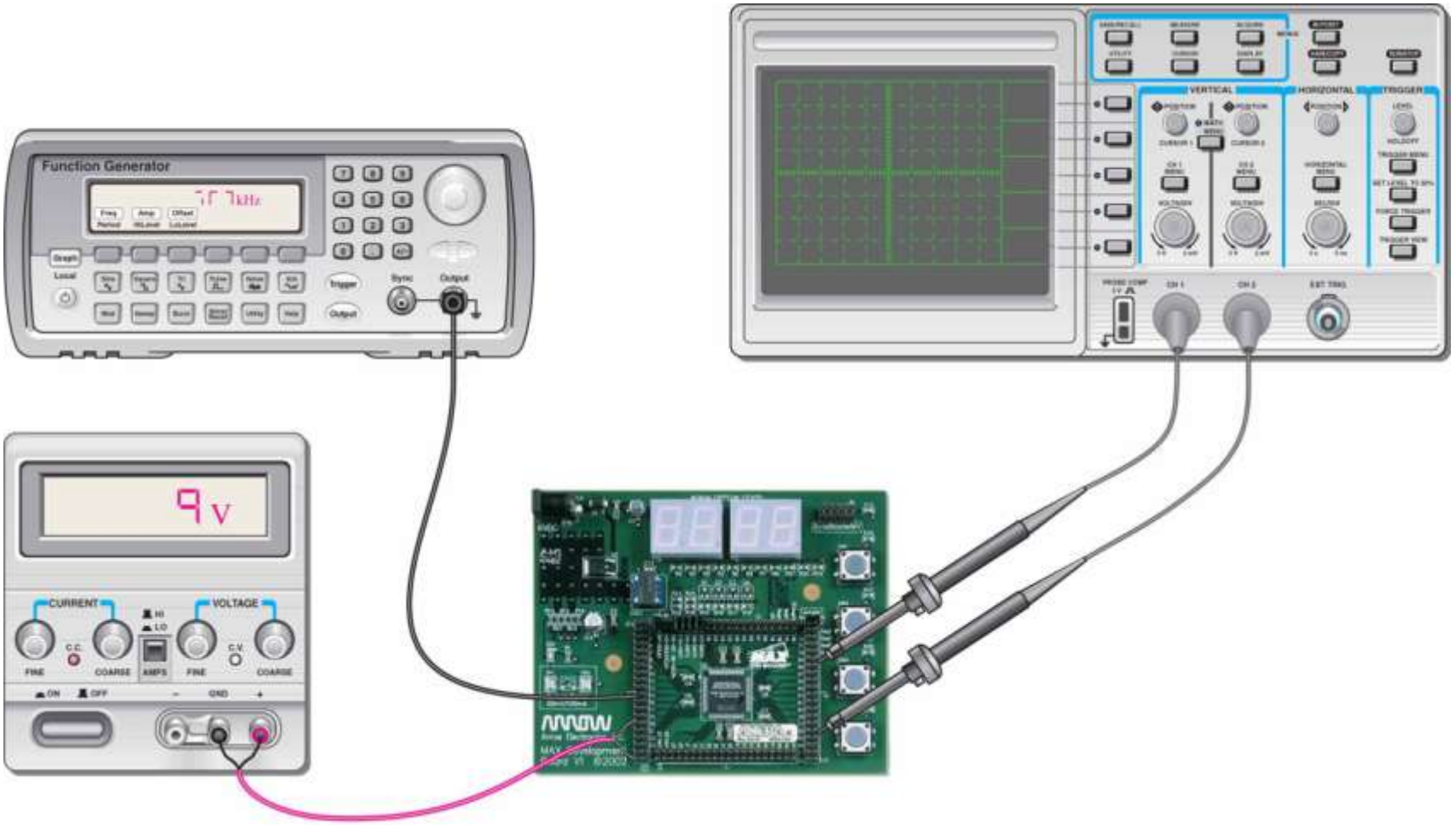
PLDT-2 Altera firması tarafından geliştirilmiş PLD programlama kartıdır. Programlama tamamlandıktan sonra pals üreteci, osilaskop kullanılarak yazılan program gerçek ortamda test edilir.



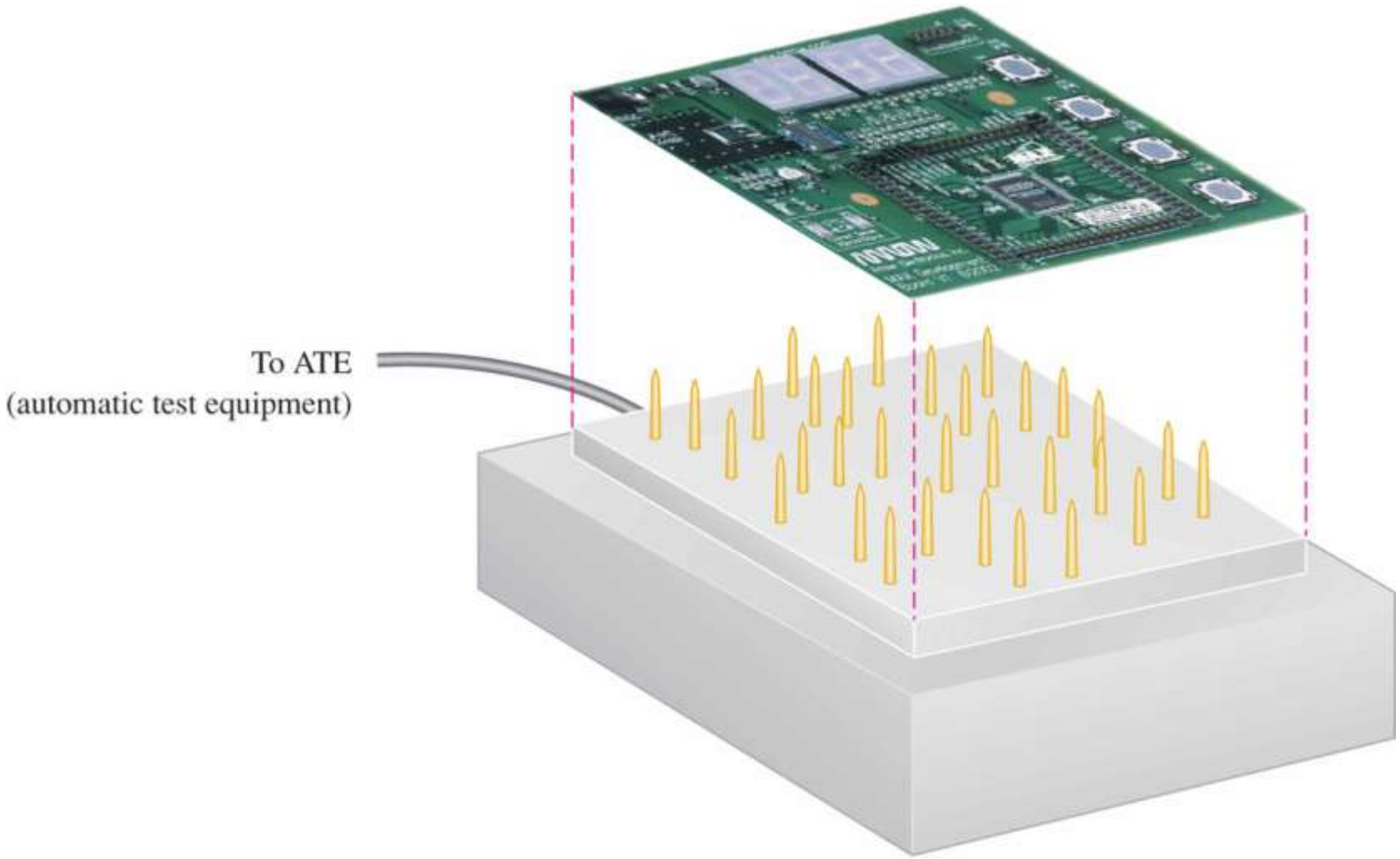
Programın deneme kartına Download edilmesi.



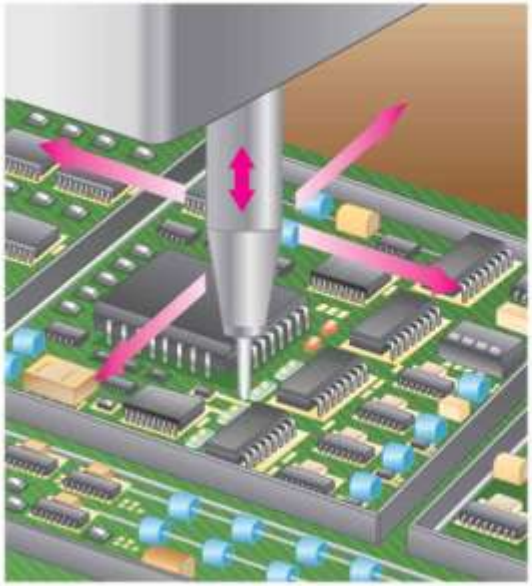
Test sistemi



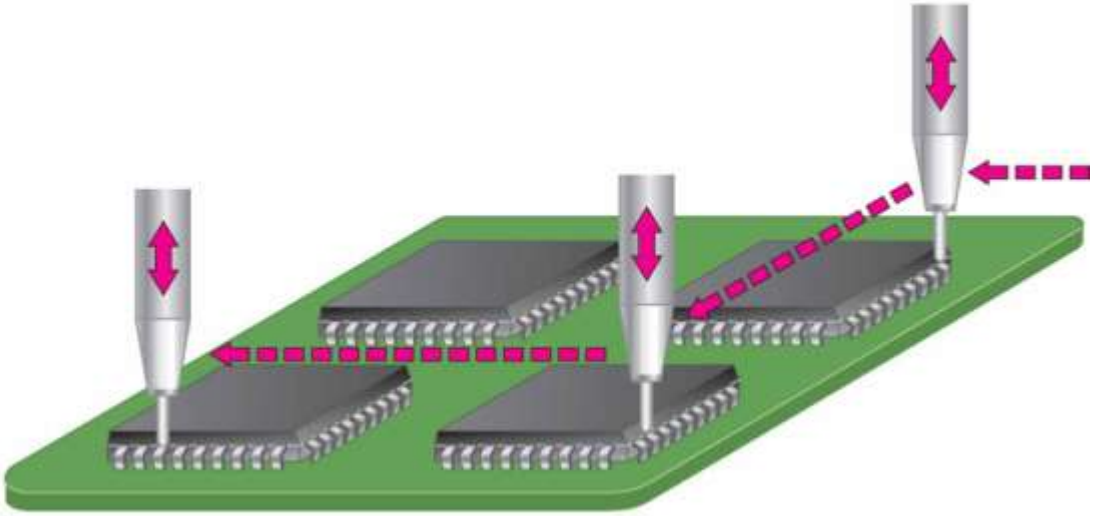
Seri Üretim Test Sistemi.



Manual test etme

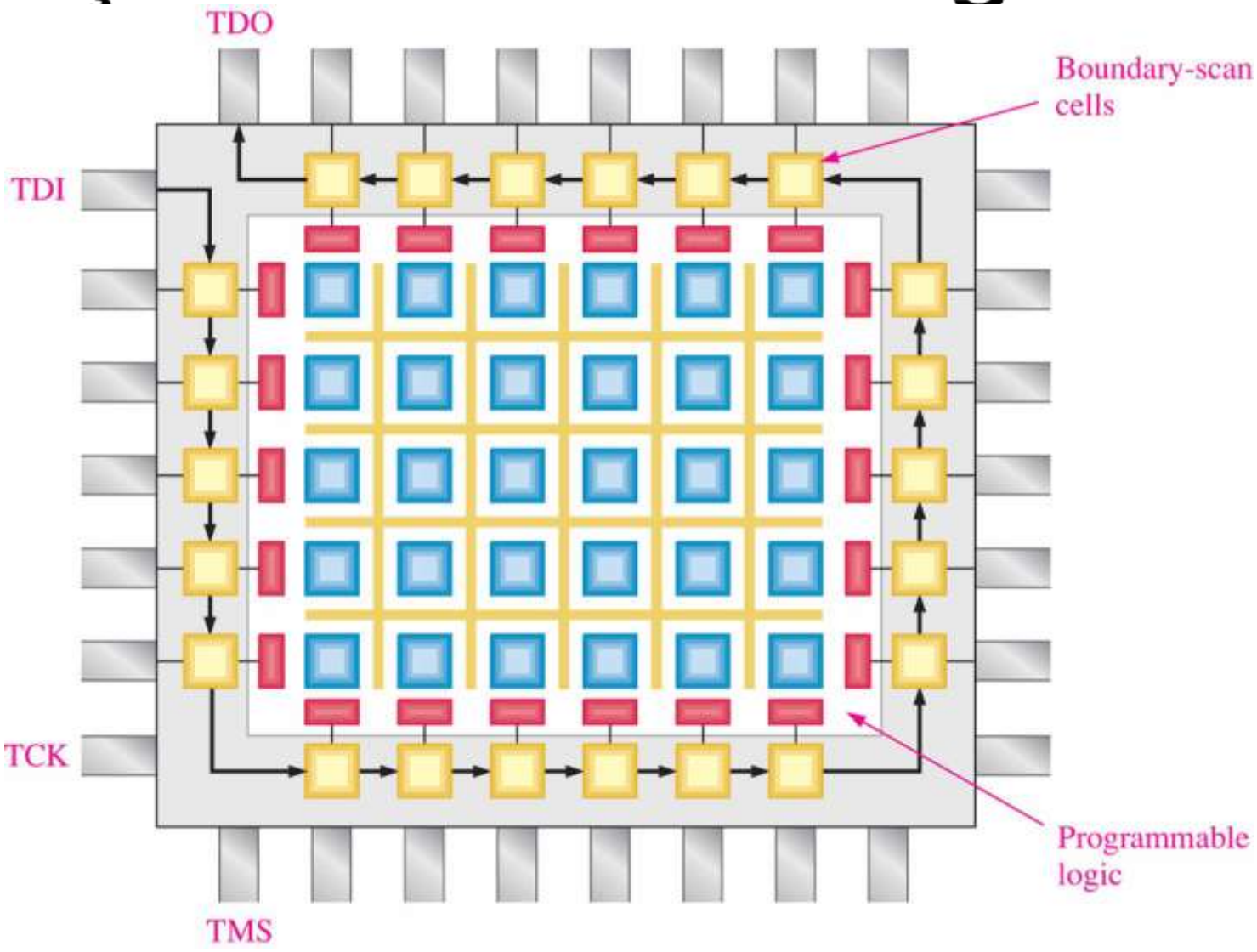


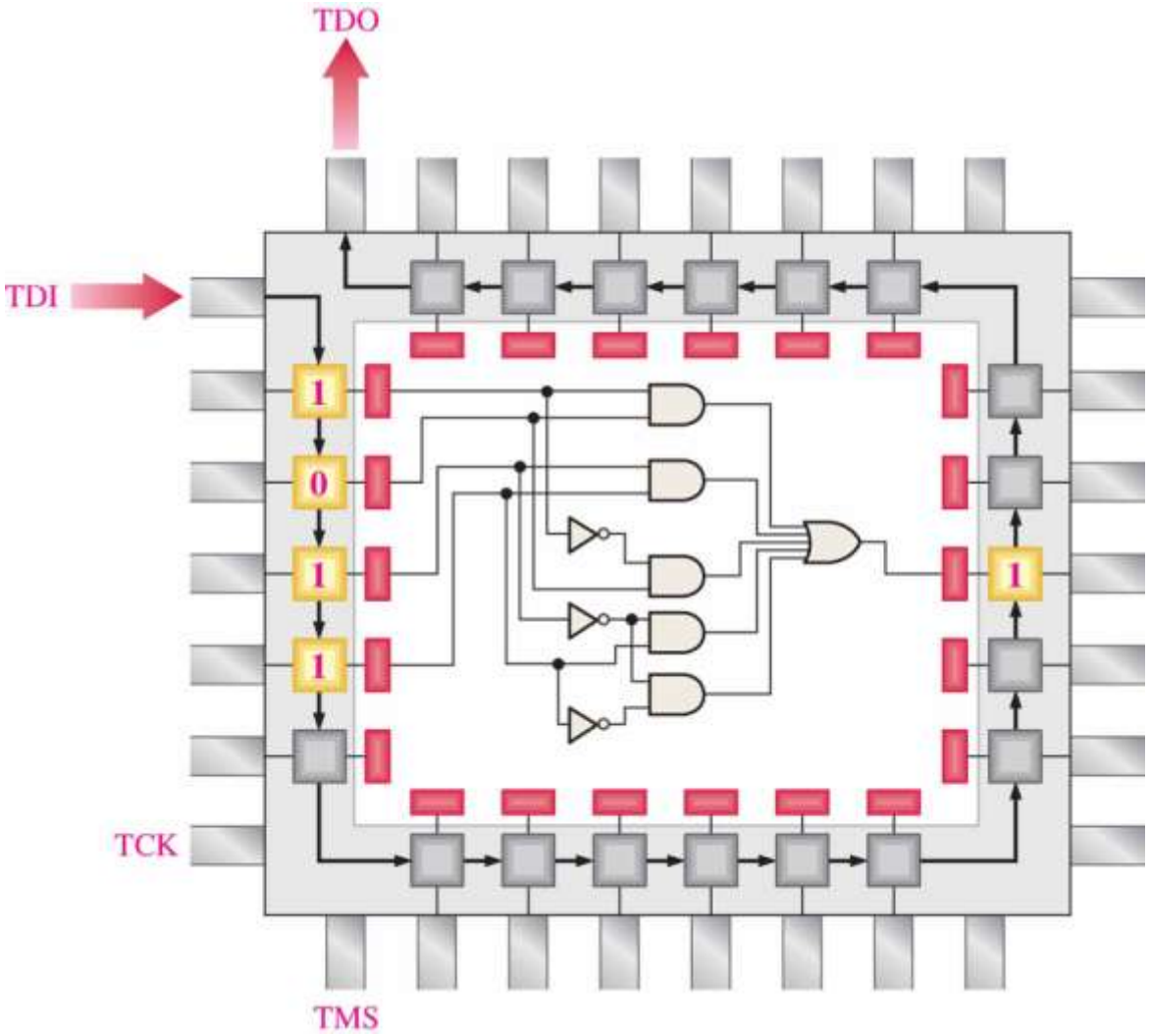
(a) 3-axis movement

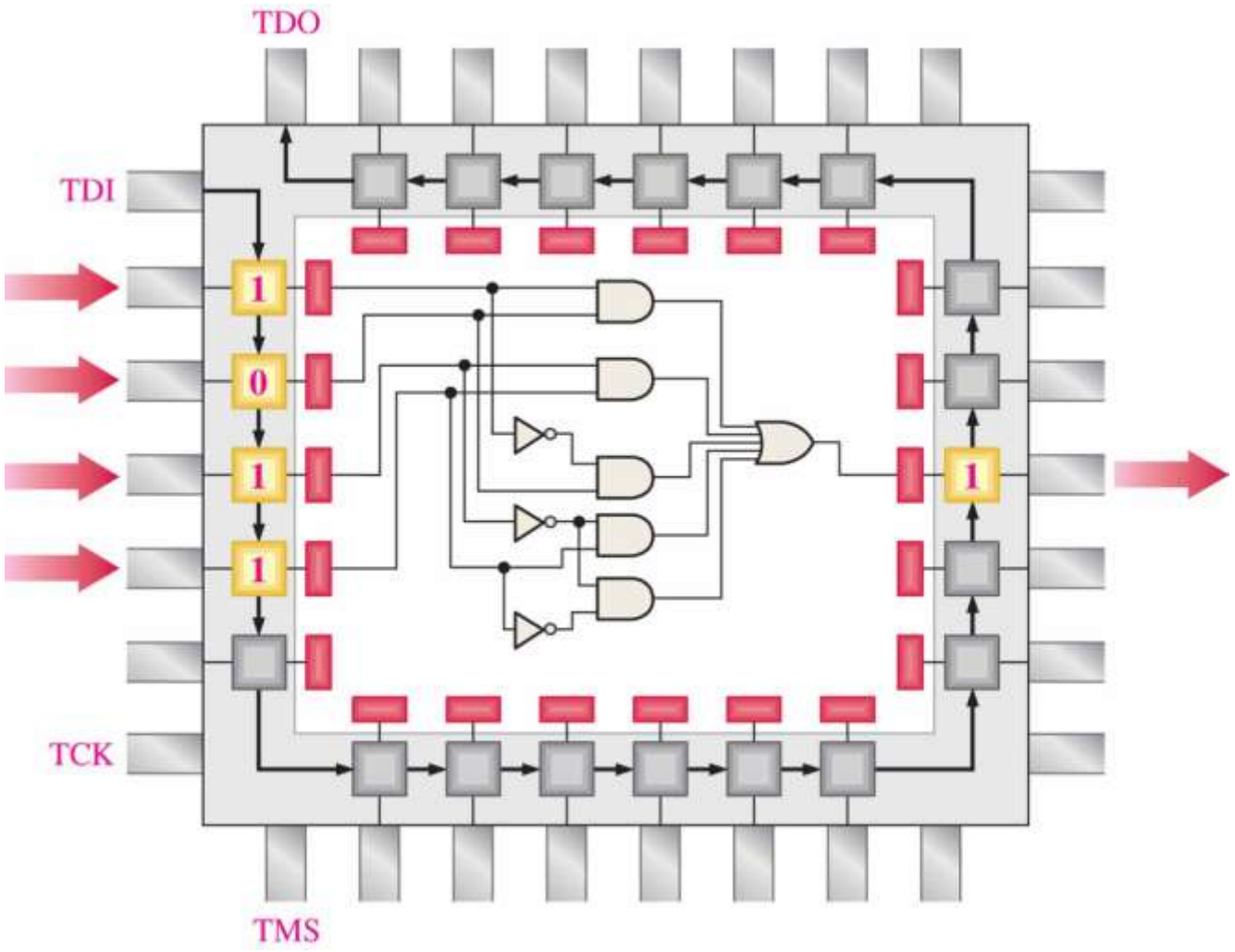


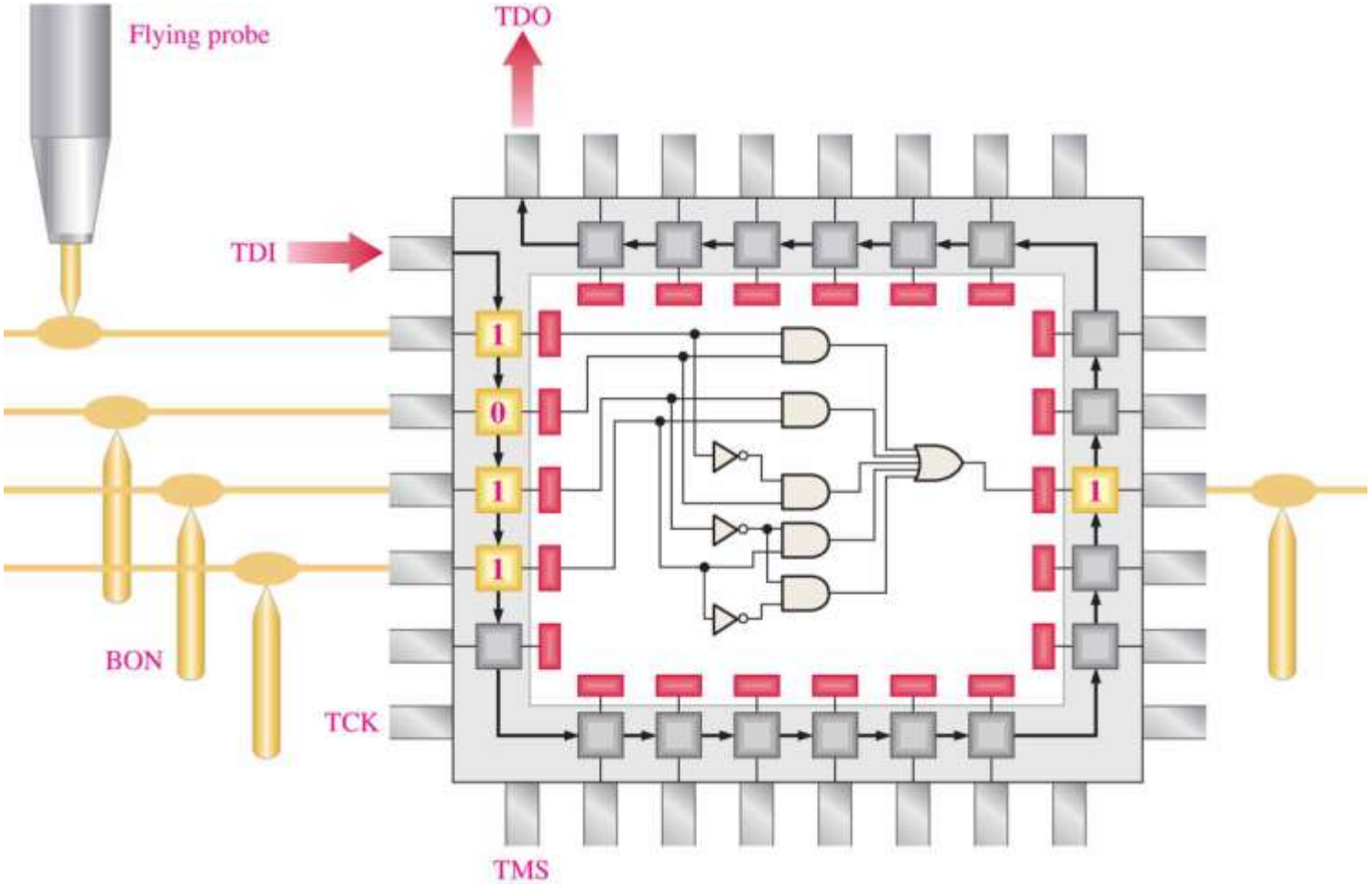
(b) Movement from point to point

Çerçeve Tarama mantığı

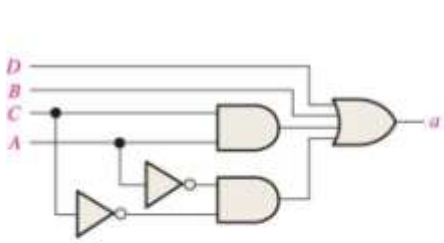



TDO
TDI
TCK
TMS
1
0
1
1
1

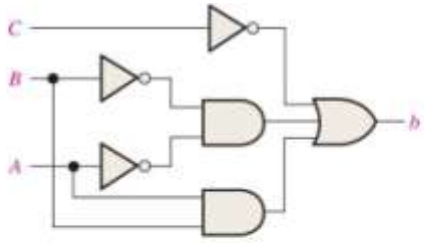

TDO
TDI
TCK
TMS
1
0
1
1
1


Flying probe
TDO
TDI
1
0
1
1
1
BON
TCK
TMS

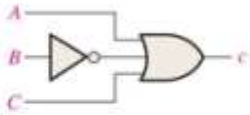
7-segment display driver



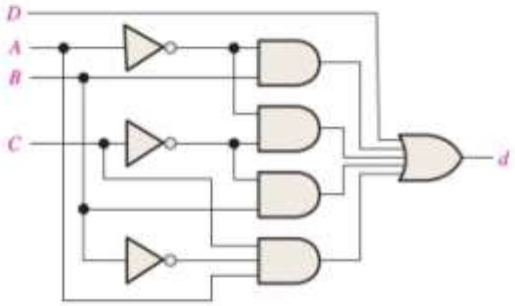
(a) Segment-*a* logic



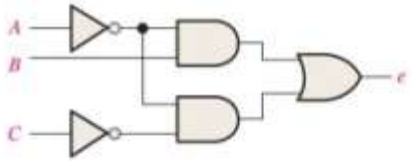
(b) Segment-*b* logic



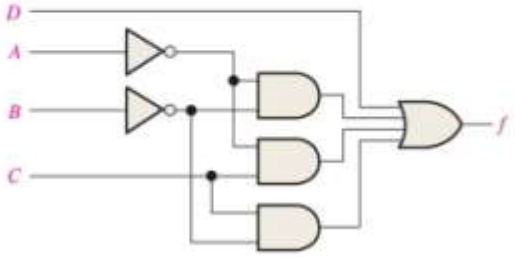
(c) Segment-*c* logic



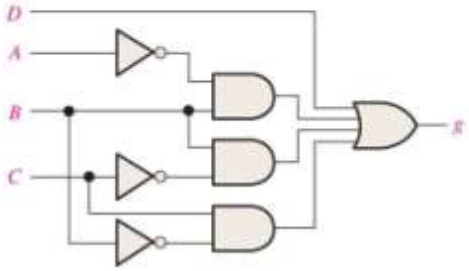
(d) Segment-*d* logic



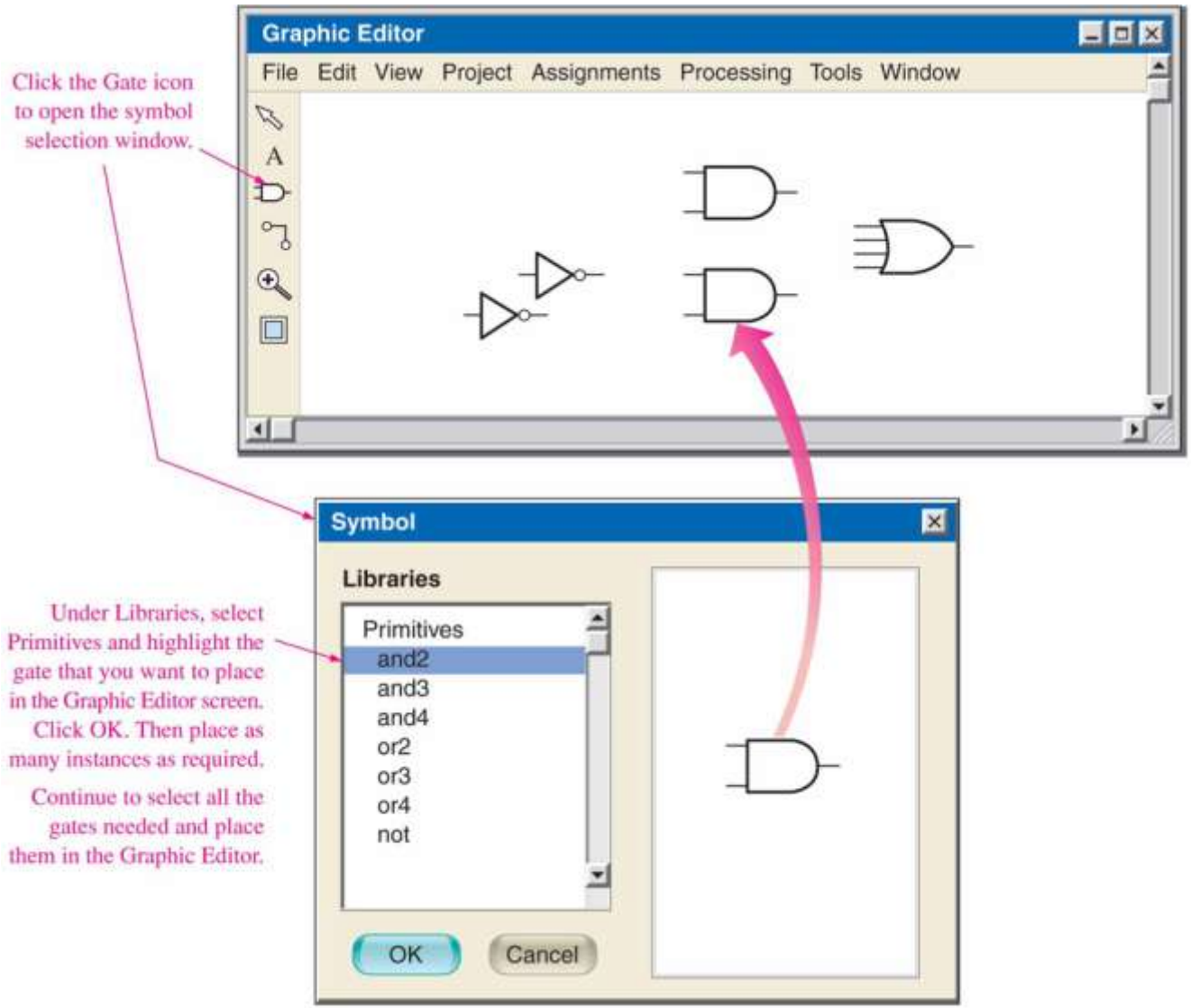
(e) Segment-*e* logic



(f) Segment-*f* logic

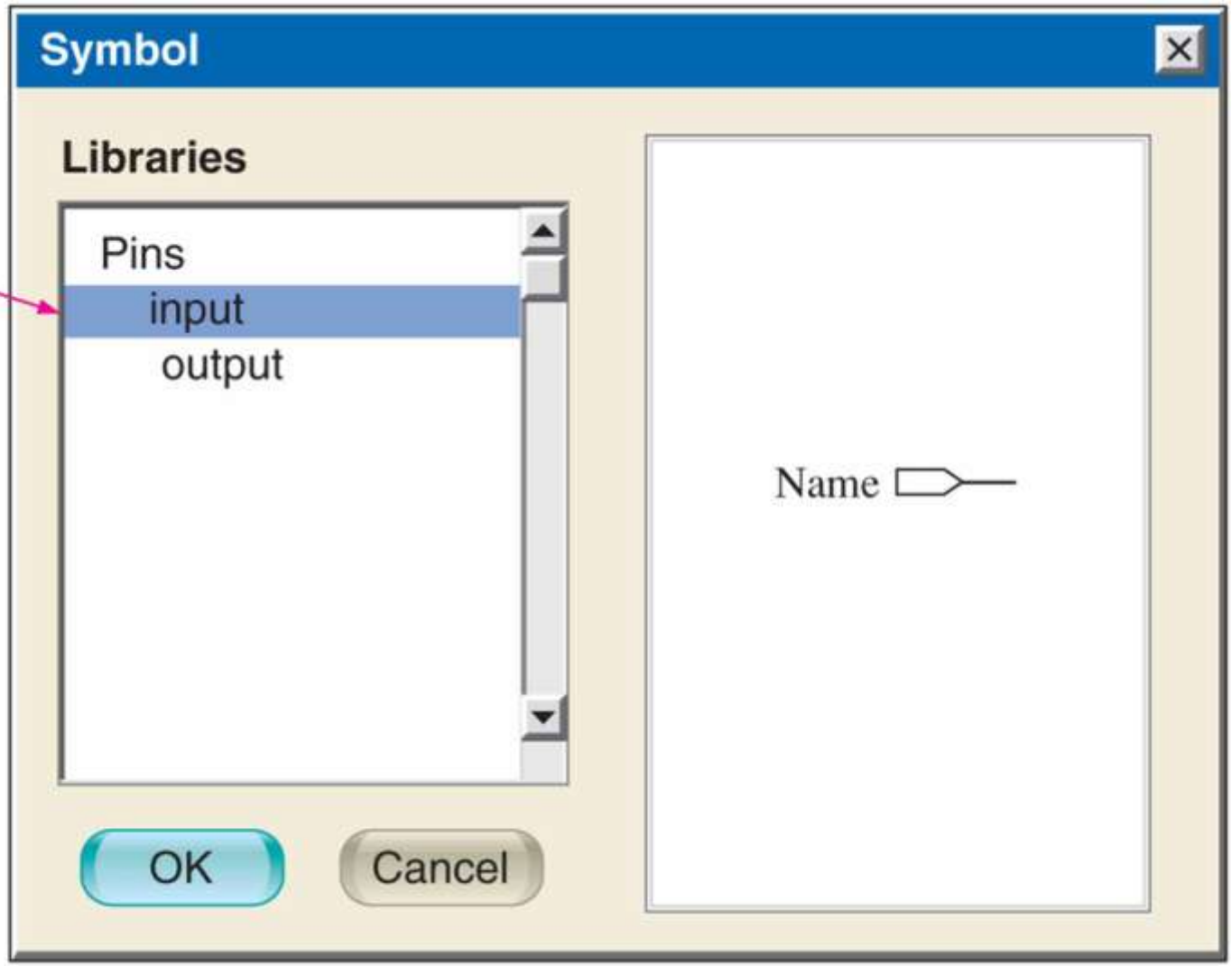


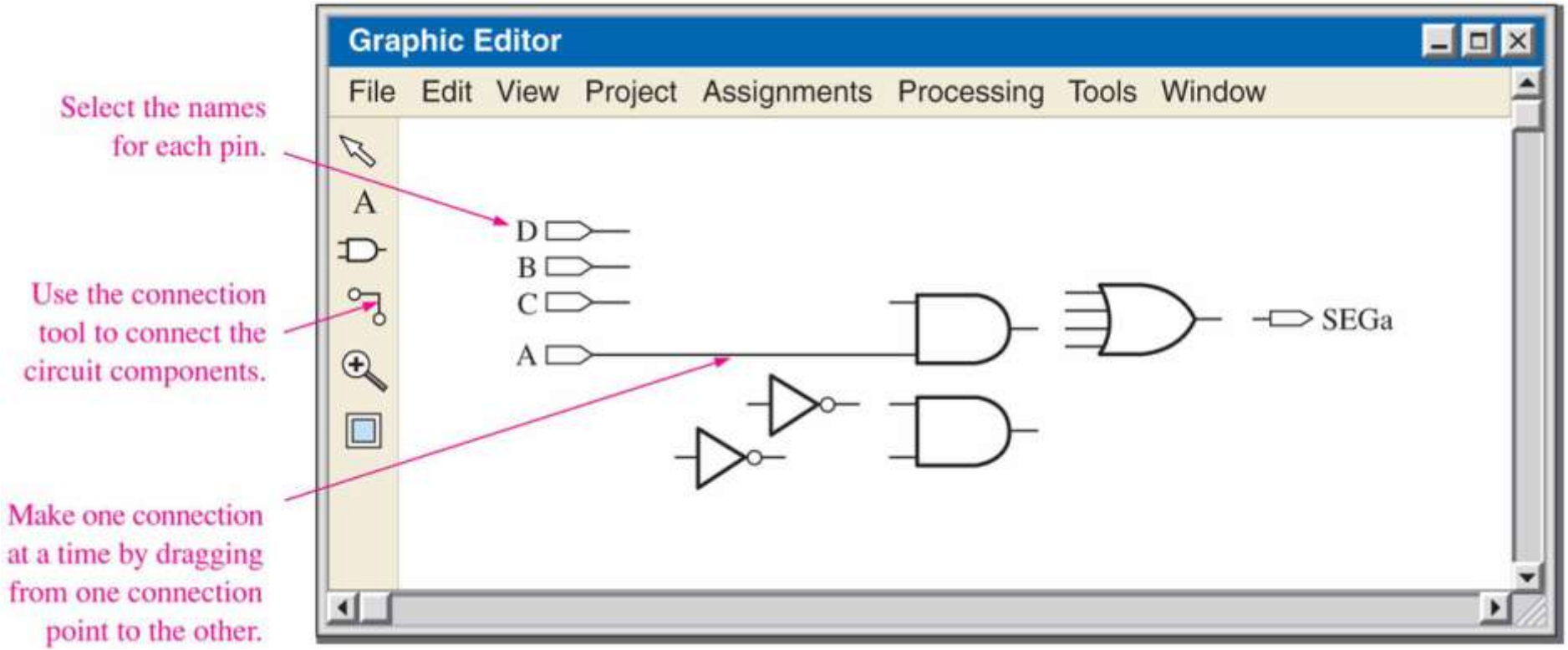
(g) Segment-*g* logic

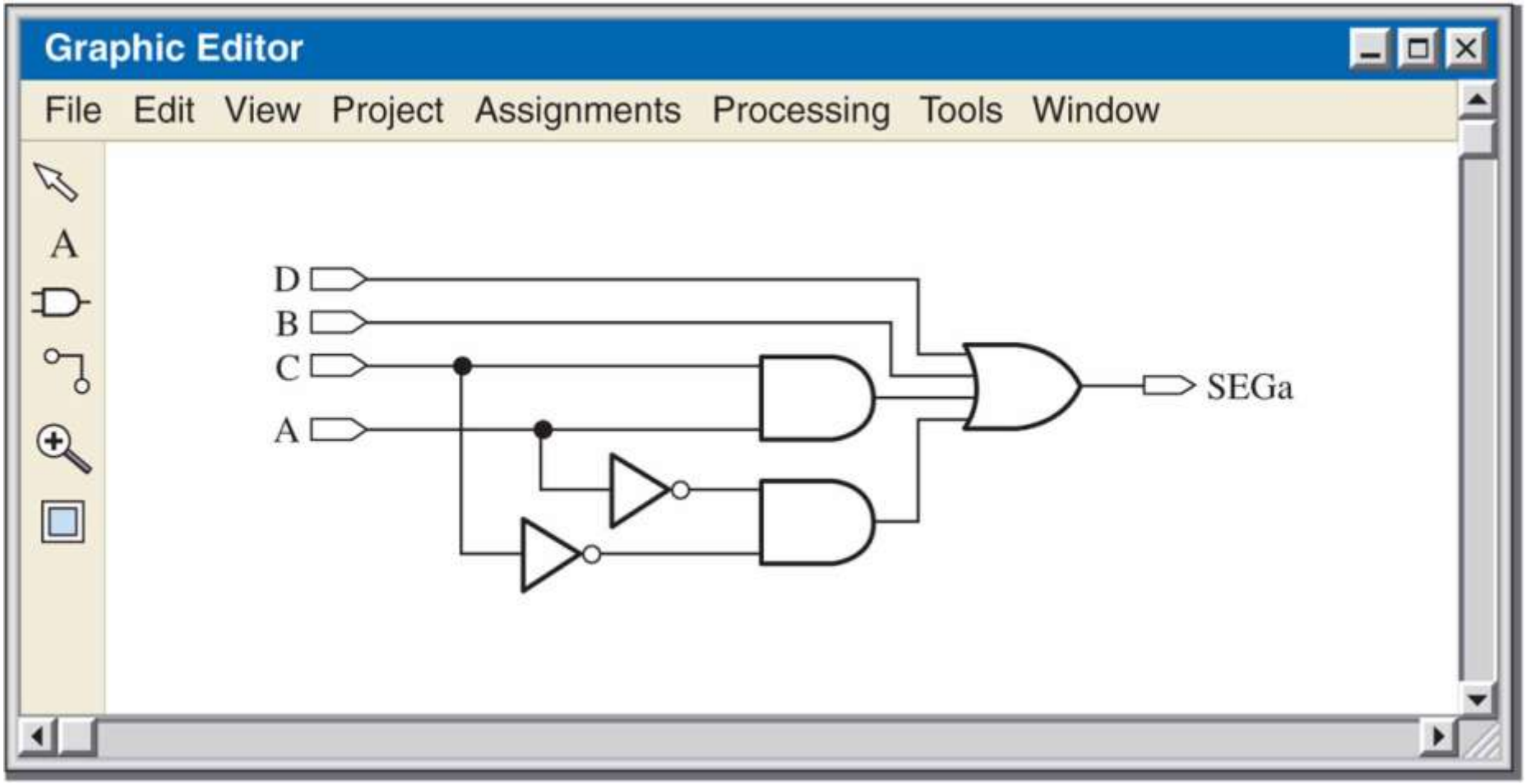

Graphic Editor
File Edit View Project Assignments Processing Tools Window
A
Click the Gate icon to open the symbol selection window.
Symbol
Libraries
Primitives
and2
and3
and4
or2
or3
or4
not
OK
Cancel
Under Libraries, select Primitives and highlight the gate that you want to place in the Graphic Editor screen. Click OK. Then place as many instances as required.
Continue to select all the gates needed and place them in the Graphic Editor.

Next select Pins and highlight input or output. Click OK to place in the Graphic Editor.

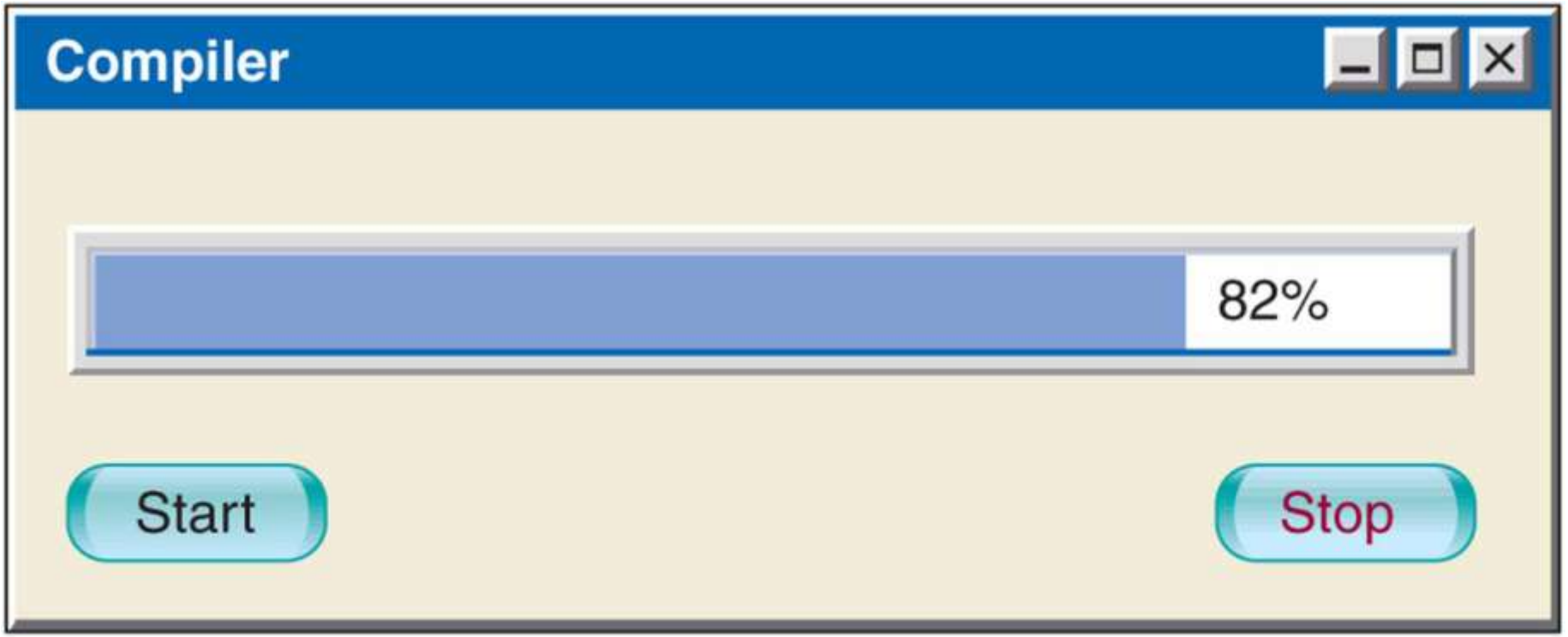
Continue to select all the pins needed and place them in the Graphic Editor.




Graphic Editor
File Edit View Project Assignments Processing Tools Window
A
D
B
C
A
SEGa
Select the names for each pin.
Use the connection tool to connect the circuit components.
Make one connection at a time by dragging from one connection point to the other.


Graphic Editor
File Edit View Project Assignments Processing Tools Window
A
D
B
C
A
SEGa

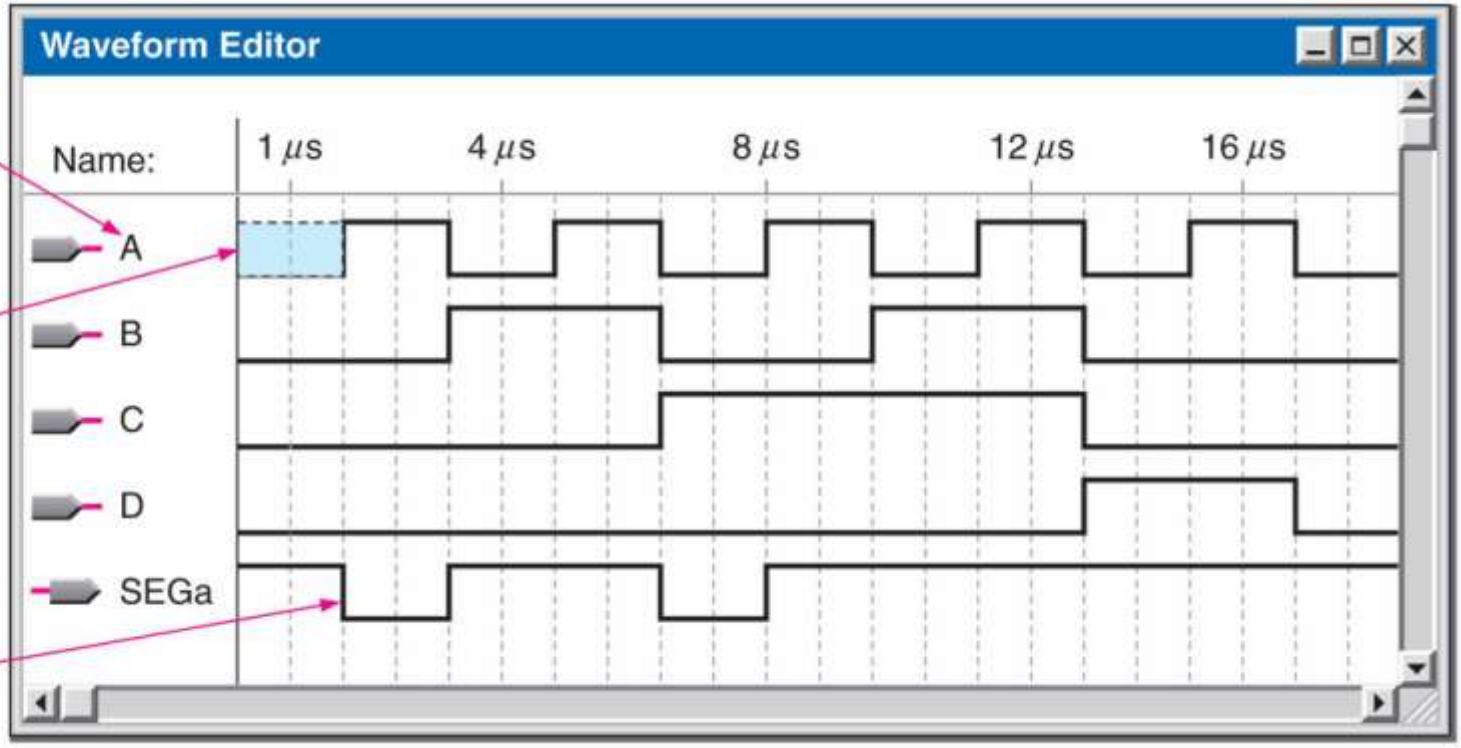
Compiler



Pins and waveform names are assigned to match the schematic.

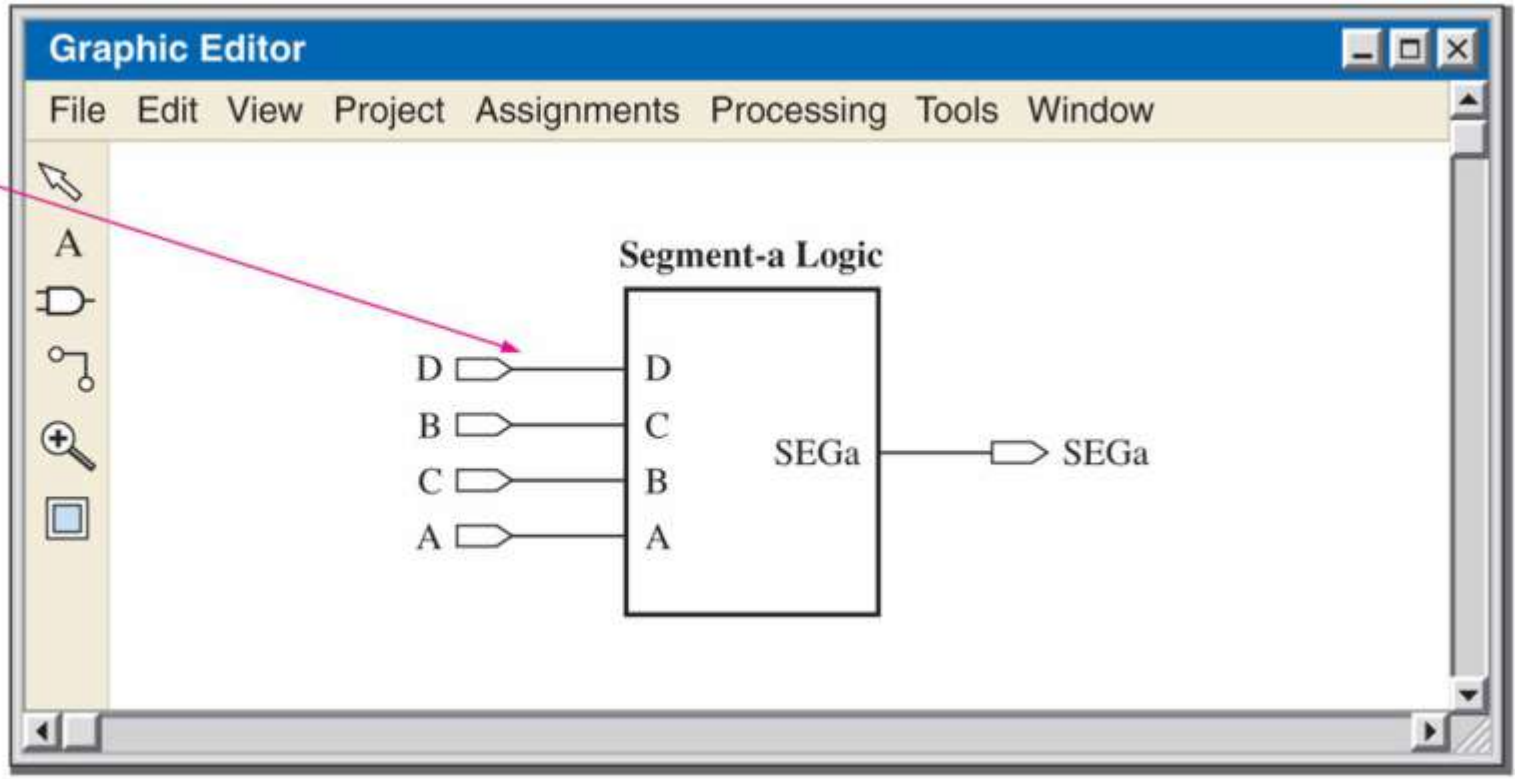
To create a waveform, select each time interval and specify a 0 or 1 for that time interval, one interval at a time.

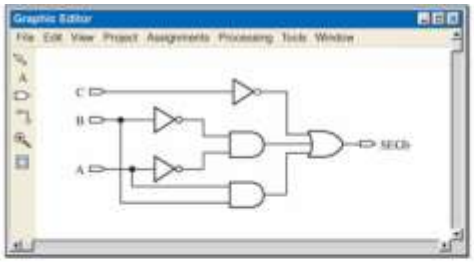
You specify the input waveforms. The simulation tool produces the output waveform.



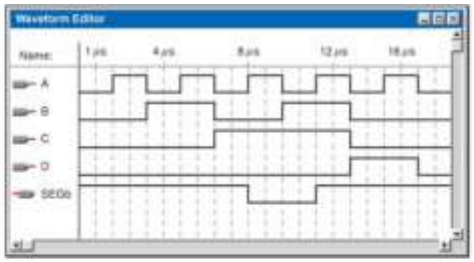
Devre bloğa dönüştürülür.

Place and connect input and output pins the same way as you did for the schematic.

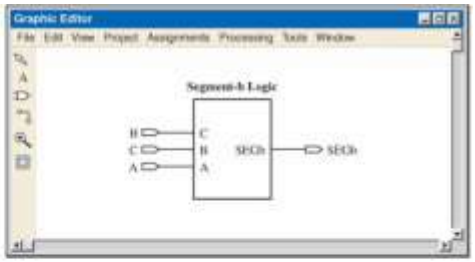




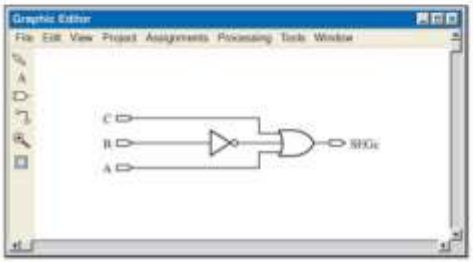
Enter segment-*b* schematic.



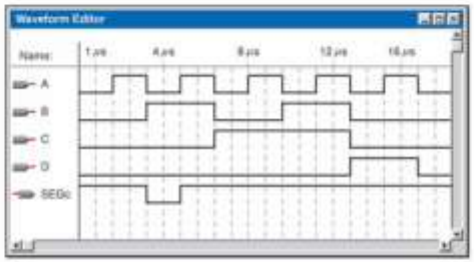
Run functional simulation.



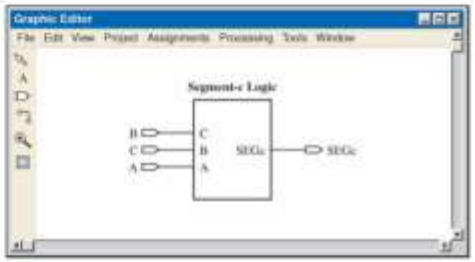
Convert schematic to block symbol and save.



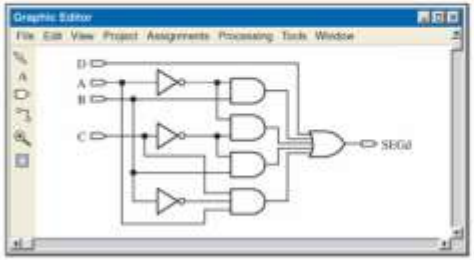
Enter segment-*c* schematic.



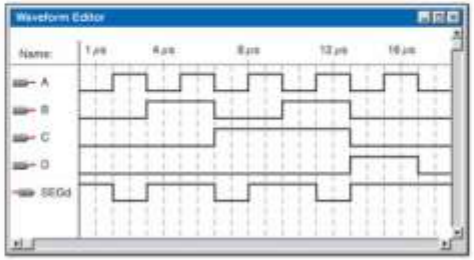
Run functional simulation.



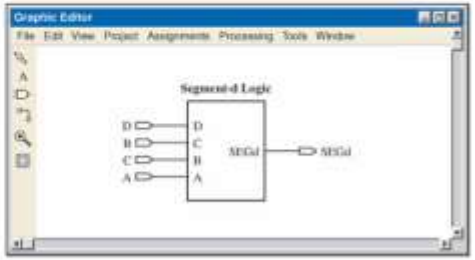
Convert schematic to block symbol and save.



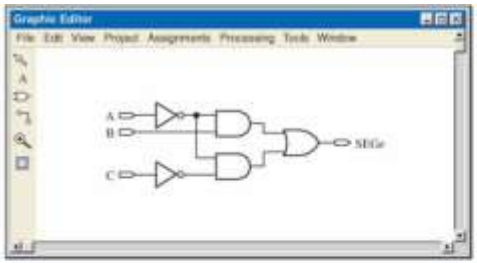
Enter segment-*d* schematic.



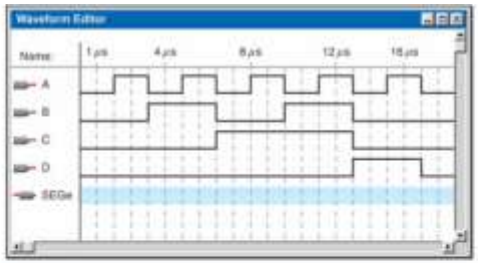
Run functional simulation.



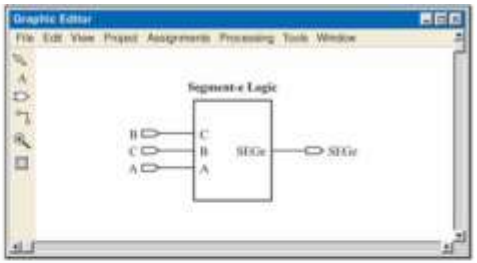
Convert schematic to block symbol and save.



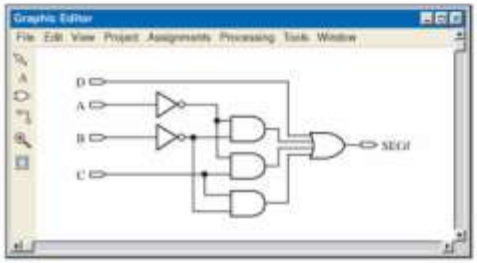
Enter segment-*e* schematic.



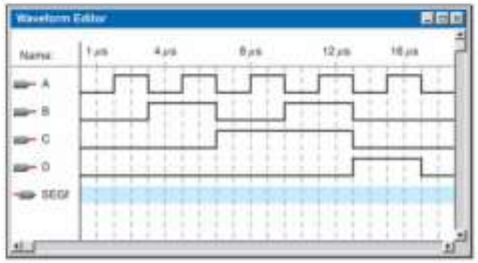
Run functional simulation.



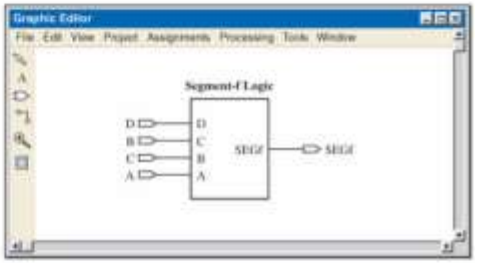
Convert schematic to block symbol and save.



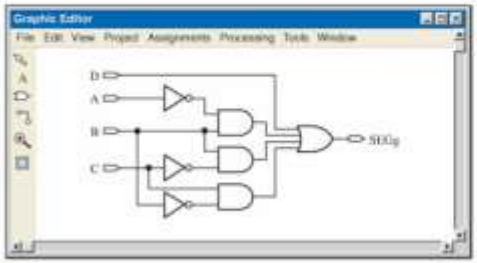
Enter segment-*f* schematic.



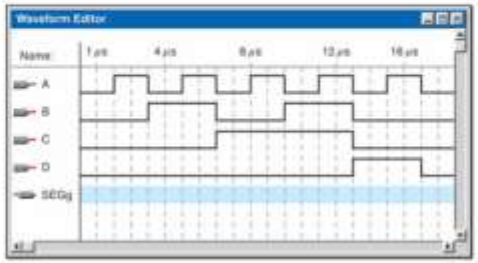
Run functional simulation.



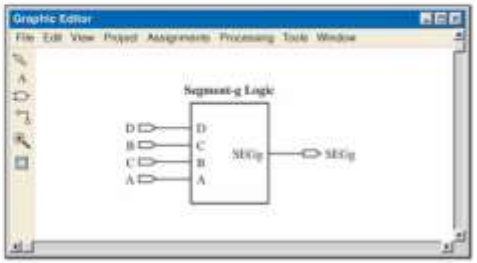
Convert schematic to block symbol and save.



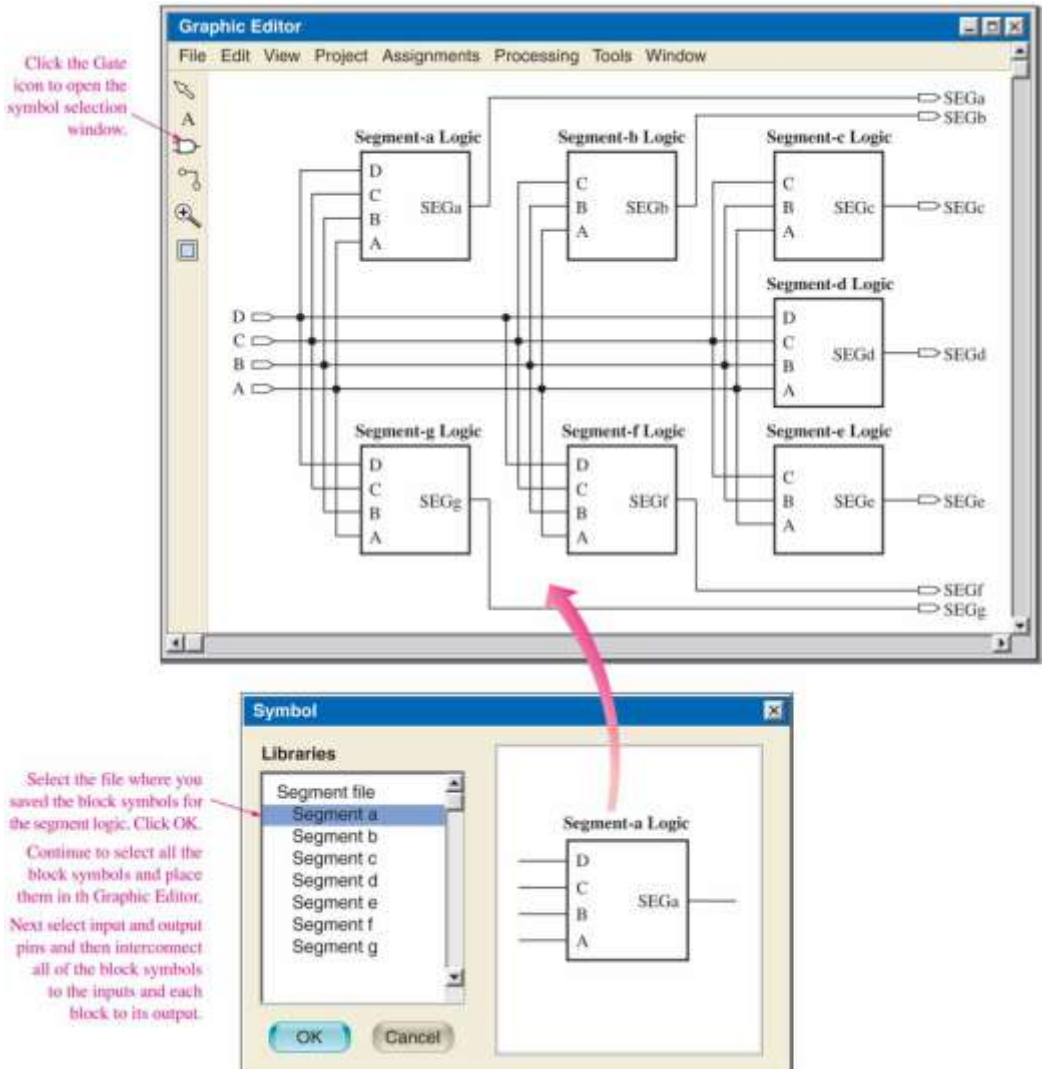
Enter segment-*g* schematic.

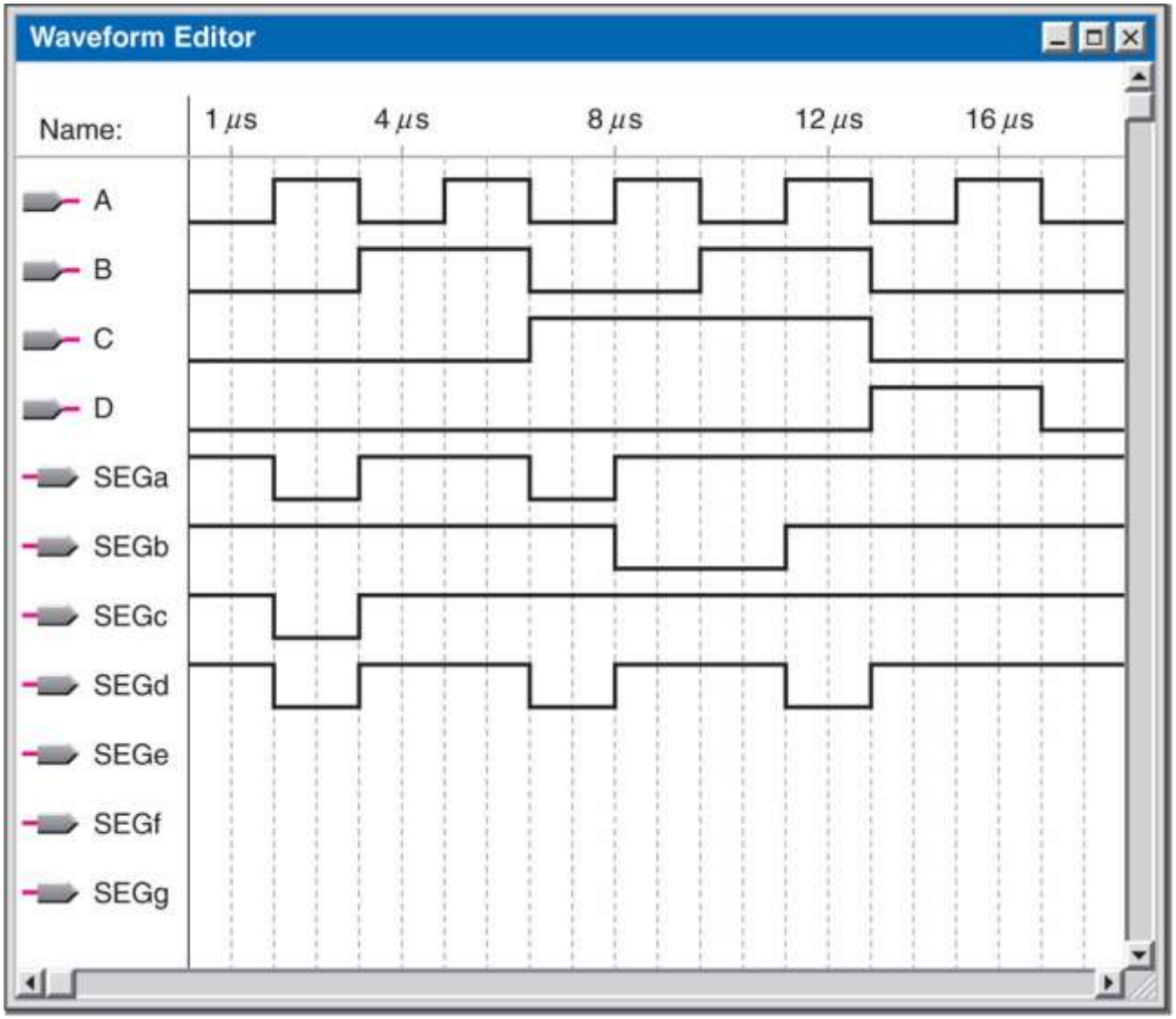


Run functional simulation.

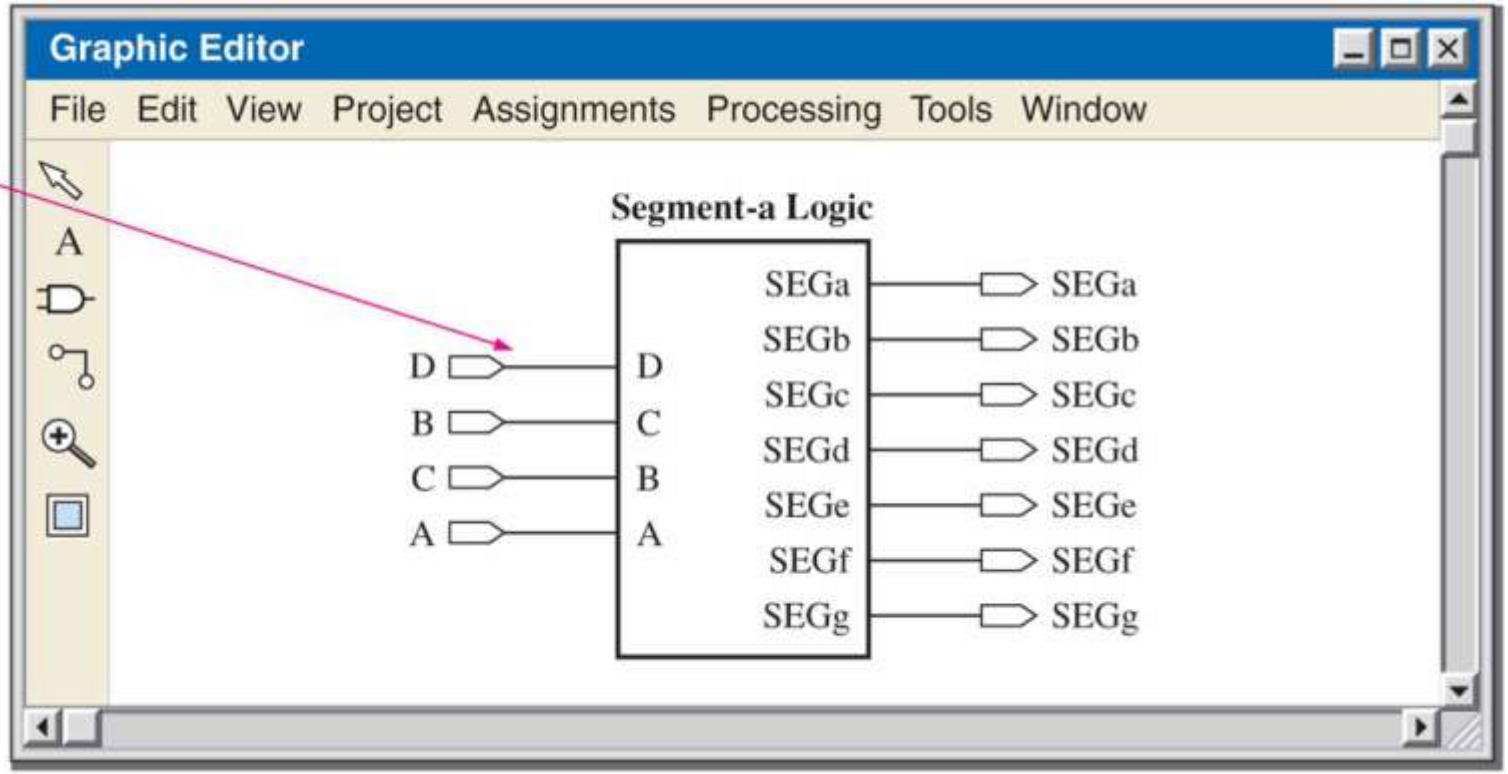


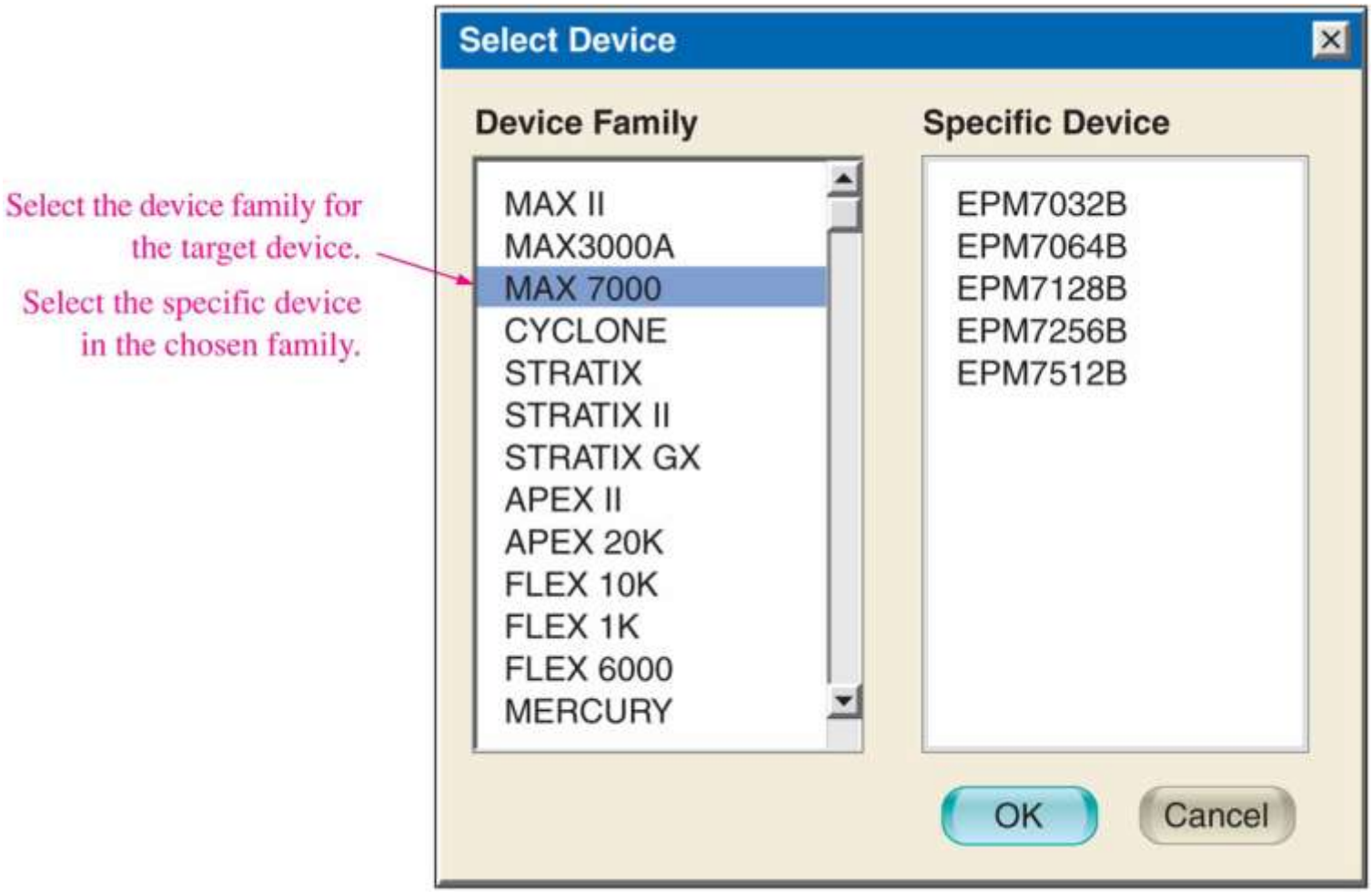
Convert schematic to block symbol and save.

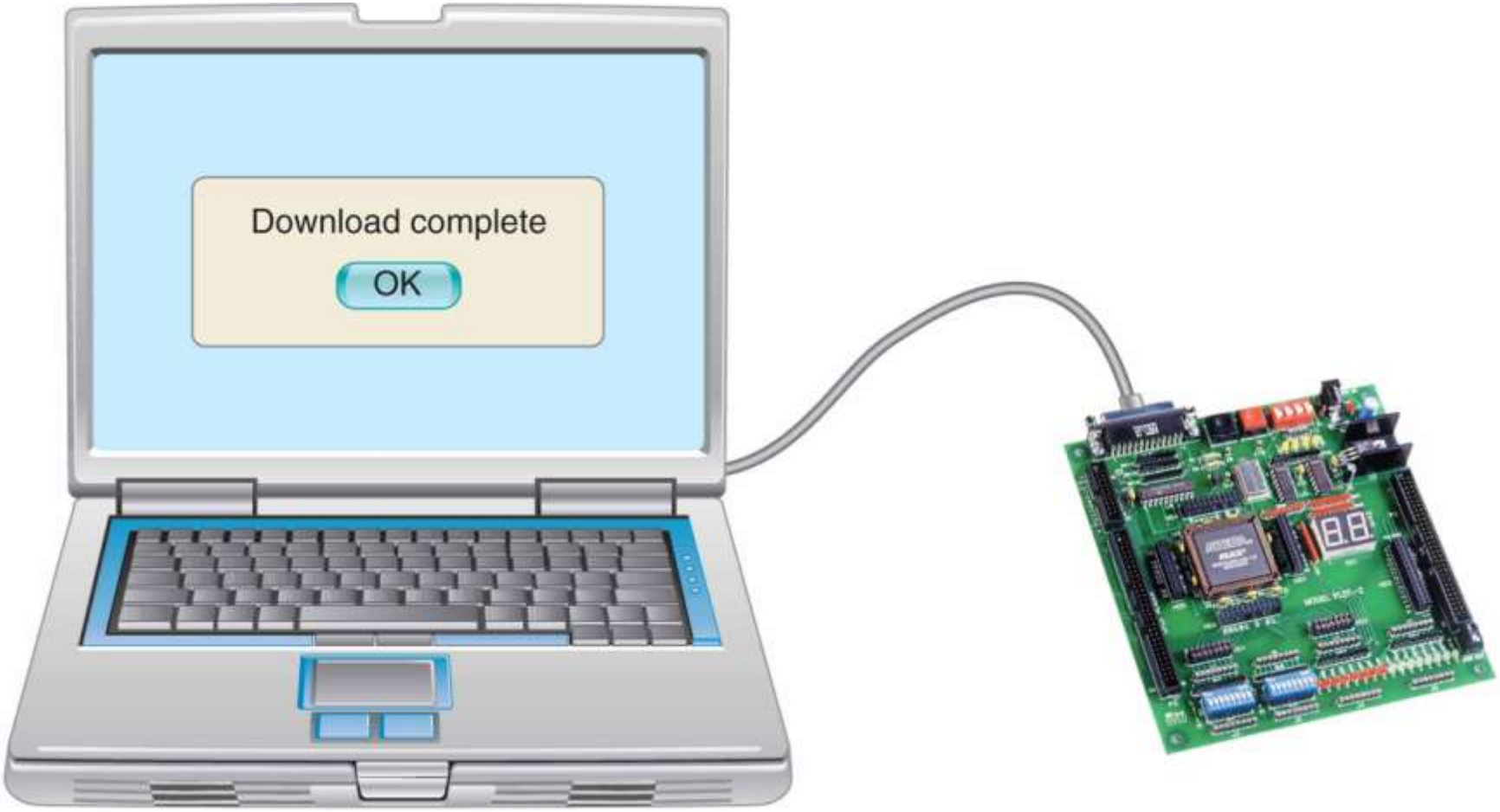

Graphic Editor
File Edit View Project Assignments Processing Tools Window
Click the Gate icon to open the symbol selection window.
Segment-a Logic
Segment-b Logic
Segment-c Logic
Segment-d Logic
Segment-e Logic
Segment-f Logic
Segment-g Logic
SEGa
SEGb
SEGc
SEGd
SEGe
SEGf
SEGg
D
C
B
A
Symbol
Libraries
Segment file
Segment a
Segment b
Segment c
Segment d
Segment e
Segment f
Segment g
OK
Cancel
Select the file where you saved the block symbols for the segment logic. Click OK.
Continue to select all the block symbols and place them in th Graphic Editor.
Next select input and output pins and then interconnect all of the block symbols to the inputs and each block to its output.


Waveform Editor
Name:
1 μs
4 μs
8 μs
12 μs
16 μs
A
B
C
D
SEGa
SEGb
SEGc
SEGd
SEGe
SEGf
SEGg

Place and connect input and output pins the same way as you did for the schematic.




Select the device family for the target device.
Select the specific device in the chosen family.
Select Device
Device Family
MAX II
MAX3000A
MAX 7000
CYCLONE
STRATIX
STRATIX II
STRATIX GX
APEX II
APEX 20K
FLEX 10K
FLEX 1K
FLEX 6000
MERCURY
Specific Device
EPM7032B
EPM7064B
EPM7128B
EPM7256B
EPM7512B
OK
Cancel


Download complete
OK

Çalıştırma ve Test etme