EHM2141 LOJİK DEVRELER

2024-2025 BAHAR DÖNEMİ

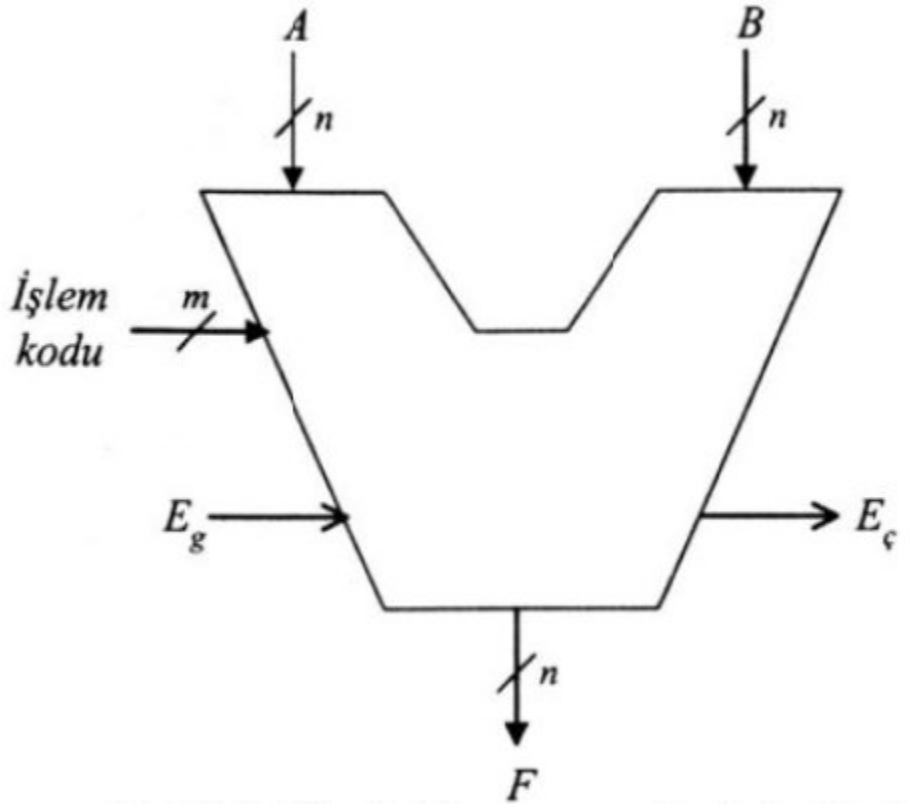
HAFTA 9 – DERS 1

15 Nisan 2025

Dr. Sibel ÇİMEN

KOMBİNEZONSAL ORTA ÖLÇEKLİ TÜMLEŞİK LOJİK DEVRELER

- **Aritmetik Lojik Birim (ALU)**

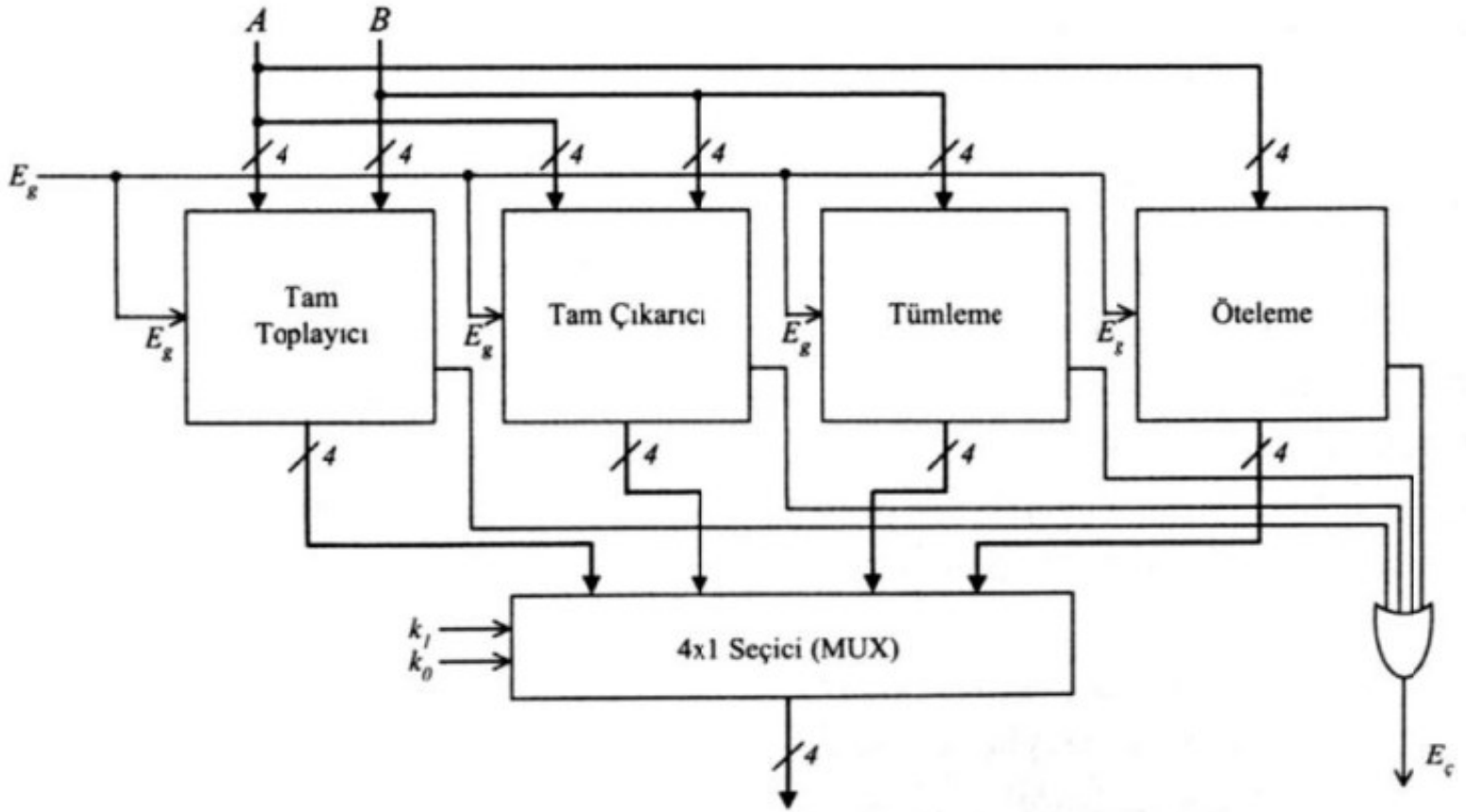


ALU'nun genel gösterilimi

Aritmetik Lojik Birim (ALU)

İşlem Kodu k_1 k_0	İşlem	Açıklama
0 0	$F = A+B$	A ve B'nin içerikleri toplanır.
0 1	$F = A-B$	A'dan B çıkarılır.
1 0	$F = \overline{B}$	B'nin tüm bitleri tümlenir; 1'ler 0, 0'lar 1 yapılır.
1 1	$F = A_{\text{1 bit sağa öteleme}}$	A'nın içeriği 1 bit sağa ötelenir; 0110 ise 0011 olur.

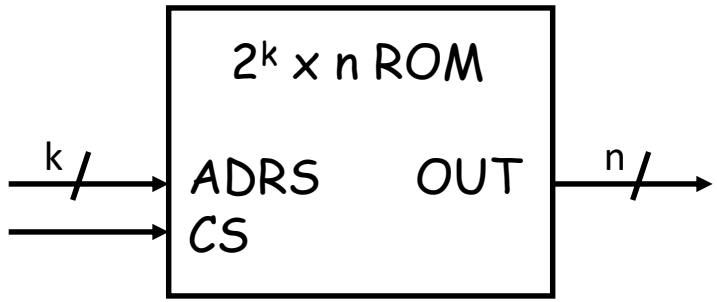
Örnek ALU işlem listesi ve kodları



PROGRAMLANABİLİR LOJİK DEVRELER

- **Sadece Okunabilir Bellek (Read Only Memory-ROM)**
- **Programlanabilir Lojik Diziler (Programmable Logic Array-PLA ve PAL)**

Sadece Okunabilir Bellek (Read Only Memory-ROM)

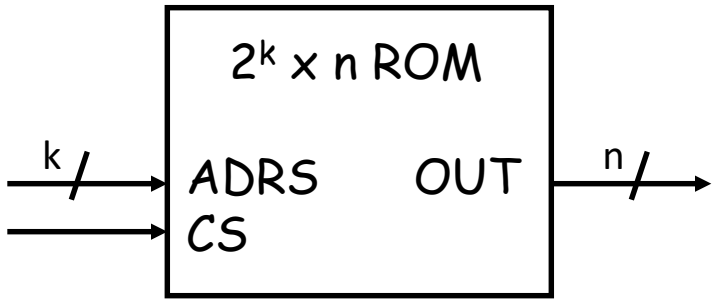


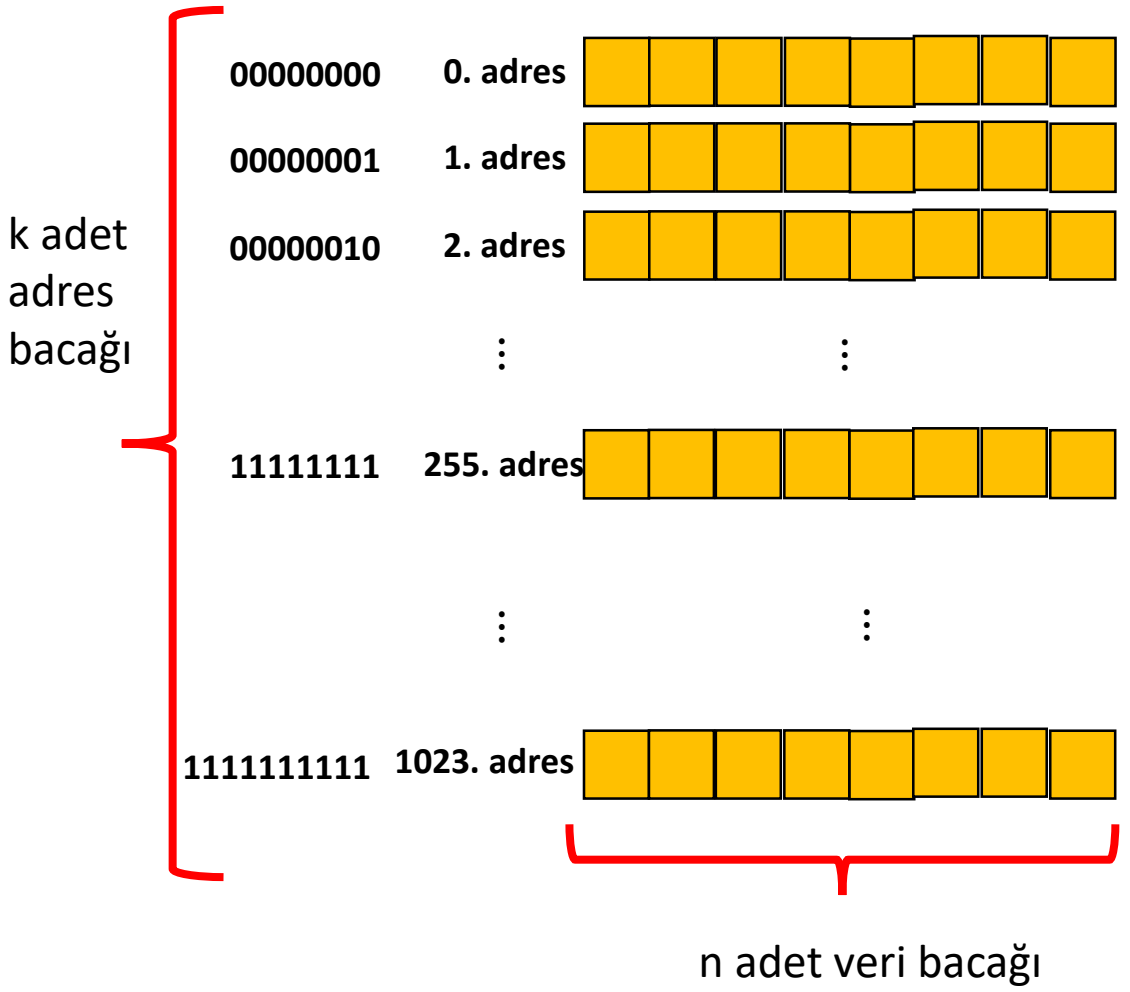
k adet adres girişi (bacağı, pini, ucu) olan, n adet çıkış ucu bulunan ROM elemanı. Chipselect ucu da CS ile gösterilmiştir.

Eğer n=8 ise her bir adreste 8 bit veri saklanmaktadır. Adres bacağı sayısı ile de ulaşılabilecek adres miktarı belirlenebilir. Örneğin k=10 ise 2^{10}= 1024= 1 K dır. N=8 ise 1K byte'lık veri saklayabilen bir ROM elemanıdır demektir.

PROGRAMLANABİLİR LOJİK DEVRELER

Sadece Okunabilir Bellek (Read Only Memory-ROM)





10 adet adres bacağı ile 2^{10}=1024 = 1K

20 adet adres bacağı ile 2^{20}=1024x1024 = 1M

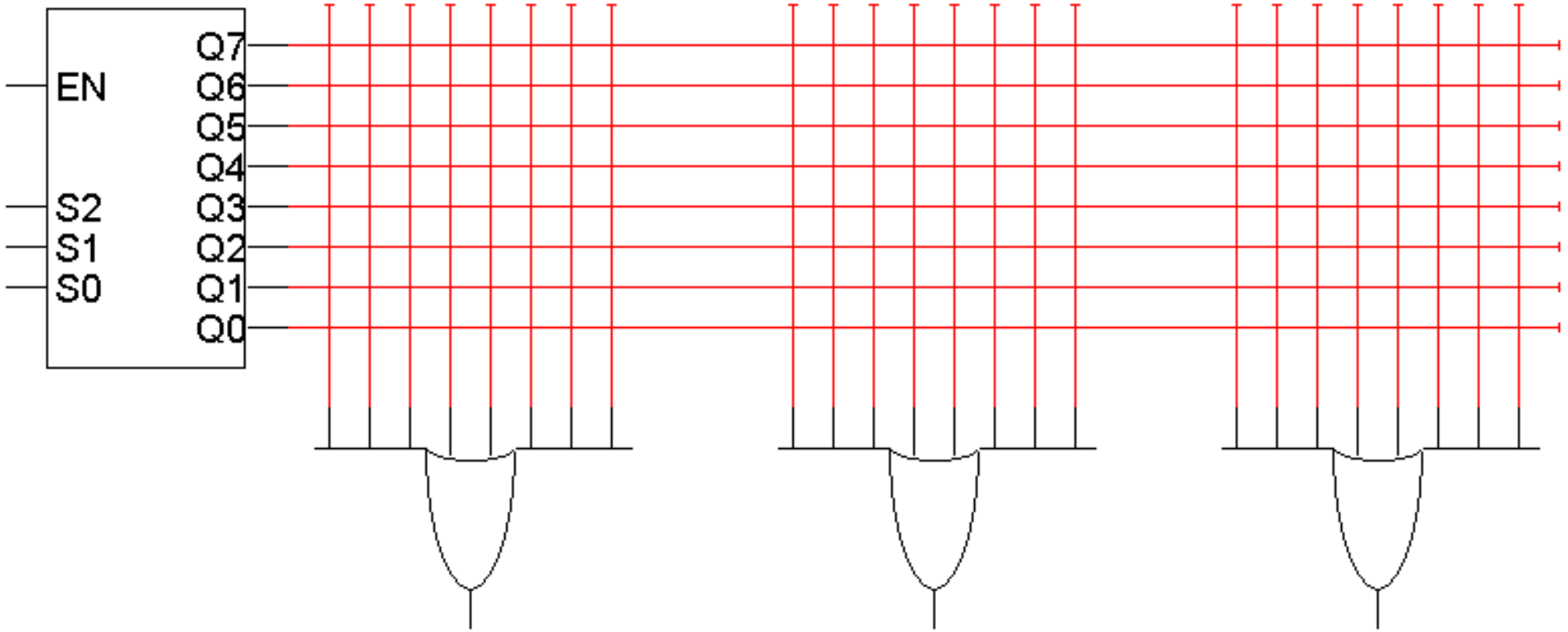
30 adet adres bacağı ile 2^{30}=1024x1M = 1G

32 adet adres bacağı ile 2^{32}=2^2x1G = 4G

2^{20} x 8 bitlik ROM. Yani 1 Mbyte'lık bir ROM için 20 adet adres bacağı gereklidir.

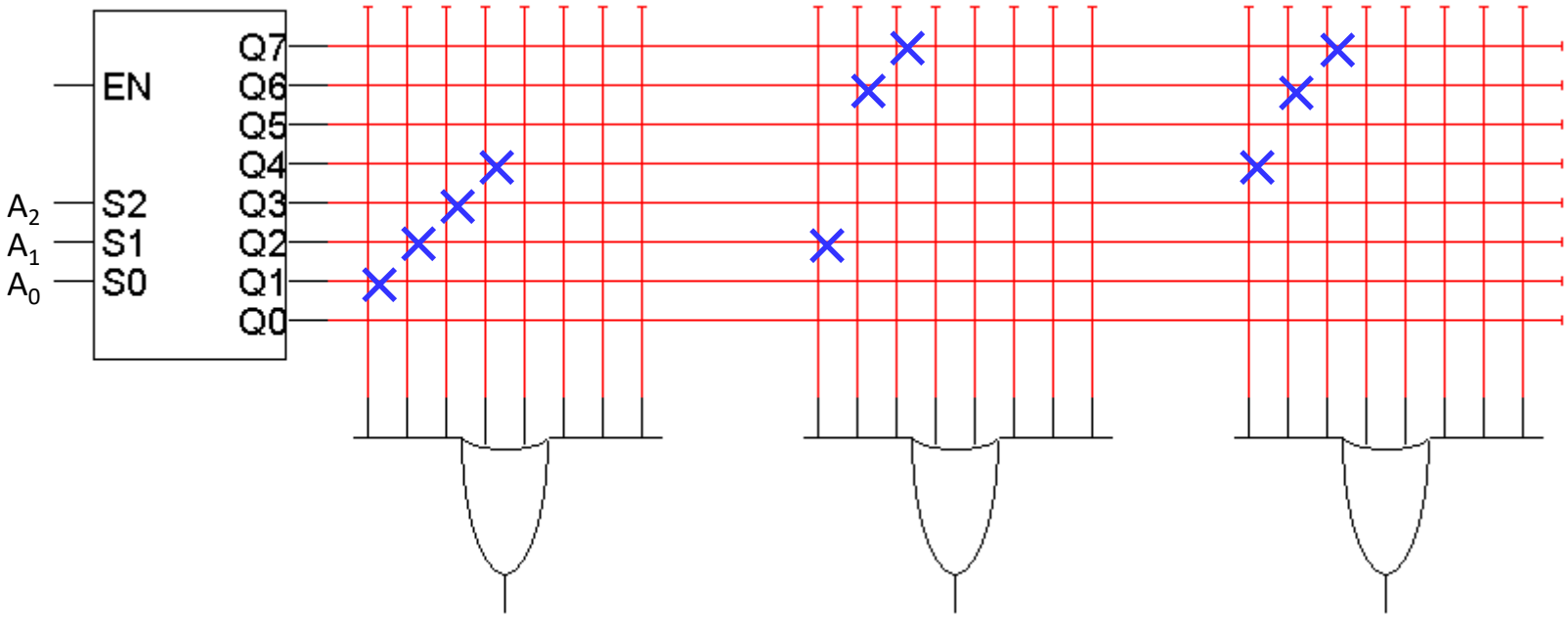
PROGRAMLANABİLİR LOJİK DEVRELER

Sadece Okunabilir Bellek (Read Only Memory-ROM)

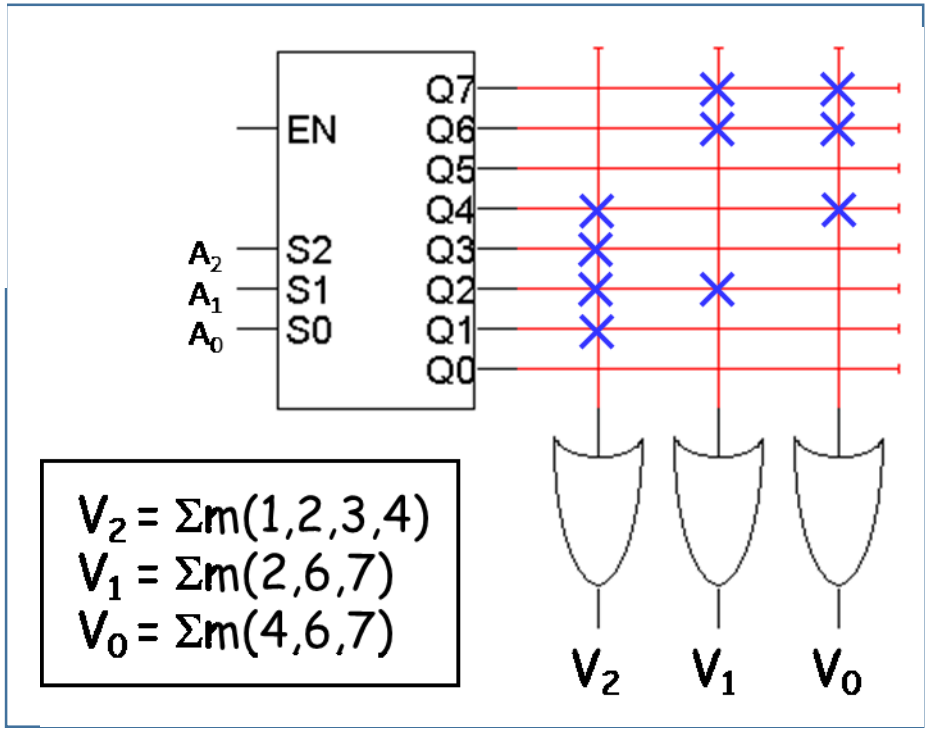


PROGRAMLANABİLİR LOJİK DEVRELER

Sadece Okunabilir Bellek (Read Only Memory-ROM)

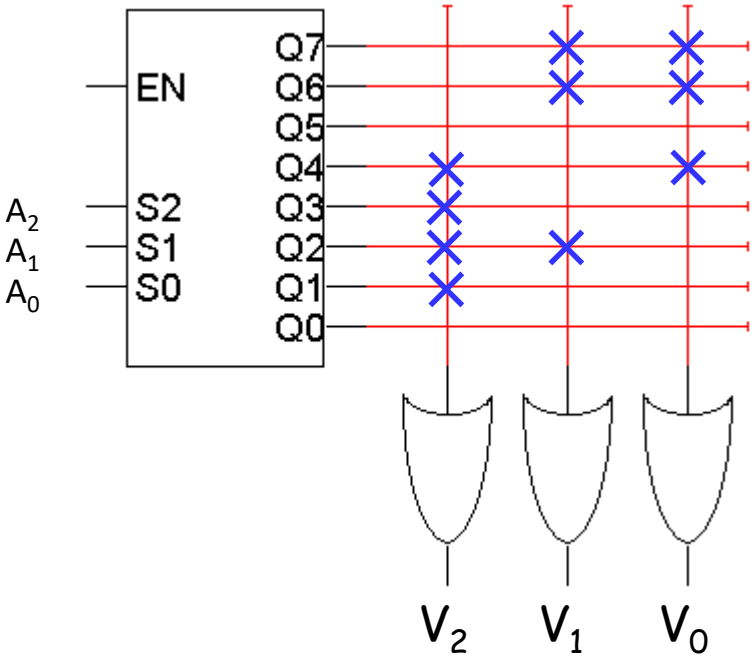


$V_2 = \Sigma m(1,2,3,4)$ $V_1 = \Sigma m(2,6,7)$ $V_0 = \Sigma m(4,6,7)$



PROGRAMLANABİLİR LOJİK DEVRELER

Sadece Okunabilir Bellek (Read Only Memory-ROM)



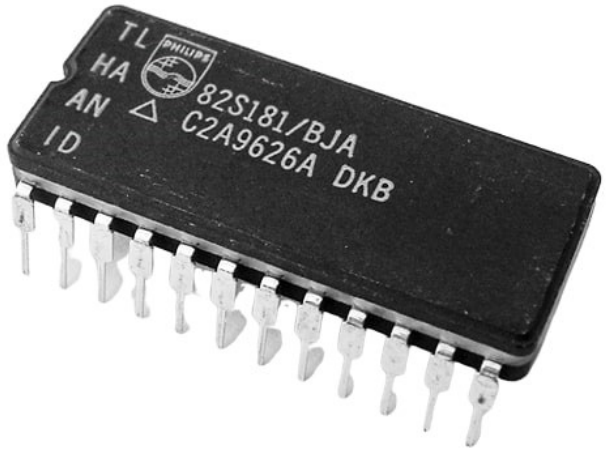
Address $A_2A_1A_0$	Data $V_2V_1V_0$
000	000
001	100
010	110
011	100
100	101
101	000
110	011
111	011

PROGRAMLANABİLİR LOJİK DEVRELER

ROM Çeşitleri

- Programmable ROM (PROM), OTP (One-Time Programmable)
- Erasable Programmable ROM (EPROM)
- Electrically Erasable Programmable ROM (EEPROM)
- Flash Memory EPROM
- Mask ROM

Programmable ROM (PROM), OTP (One-Time Programmable)



PROM'un her biti için bir sigorta vardır. PROM, sigortalar yakılarak programlanır. PROM'a yazılan bilgi yanlışsa, bu PROM'un dahili sigortaları kalıcı olarak yakıldığı için atılması gerekir.
Bu nedenle, PROM aynı zamanda OTP (Bir Kez Programlanabilir) olarak da adlandırılır.

PROGRAMLANABİLİR LOJİK DEVRELER

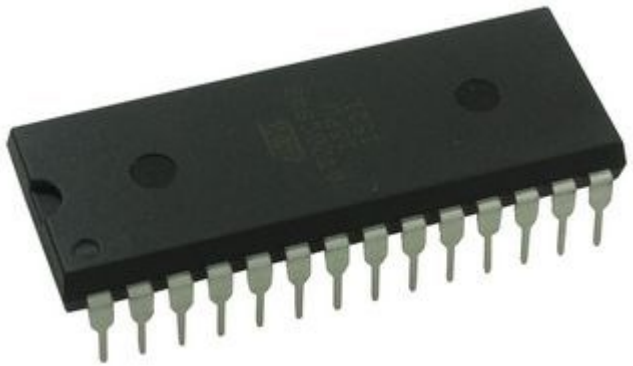
ROM Çeşitleri

Erasable Programmable ROM (EPROM)

EPROM'da bellek yongası programlanabilir ve binlerce kez silinebilir.

Tüm EPROM çiplerinde, programcının çipin içeriğini silmek için ultraviyole (UV) radyasyon uygulayabilecği bir pencere vardır. Bu nedenle EPROM, UV-EPROM olarak da adlandırılır. EPROM'un içeriğinin silinmesi için UV ışınları altında 20 dakika kadar bir süre gereklidir.

Electrically Erasable Programmable ROM (EEPROM)



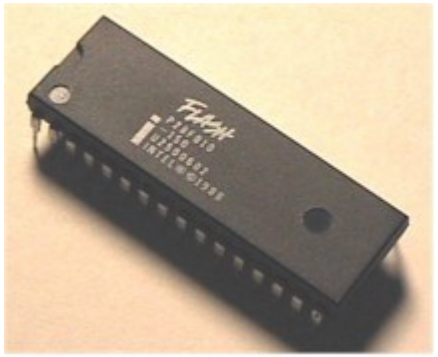
EEPROM'larda istenilen adres seçilerek elektriksel olarak silme işlemi yapılabilir. Bu işlem EEPROM devre kartı üzerinde takılı iken yapılabilir.

ATMEL EEPROM

PROGRAMLANABİLİR LOJİK DEVRELER

ROM Çeşitleri

Flash Memory EPROM



1990'ların başından beri, Flash EPROM'lar yaygın olarak kullanılmaya başlanmıştır. Flash EPROM'lar da tüm içeriğin silinmesi elektriksel olarak gerçekleştirilir ve bir saniyeden daha az sürmektedir. Bu nedenle Flash EEPROM veya Flash bellek olarak adlandırılmıştır.
Flash bellekte, içerikler bloklara bölünmüş ve silme işlemi, EEPROM'un aksine, blok blok yapılabilmektedir.

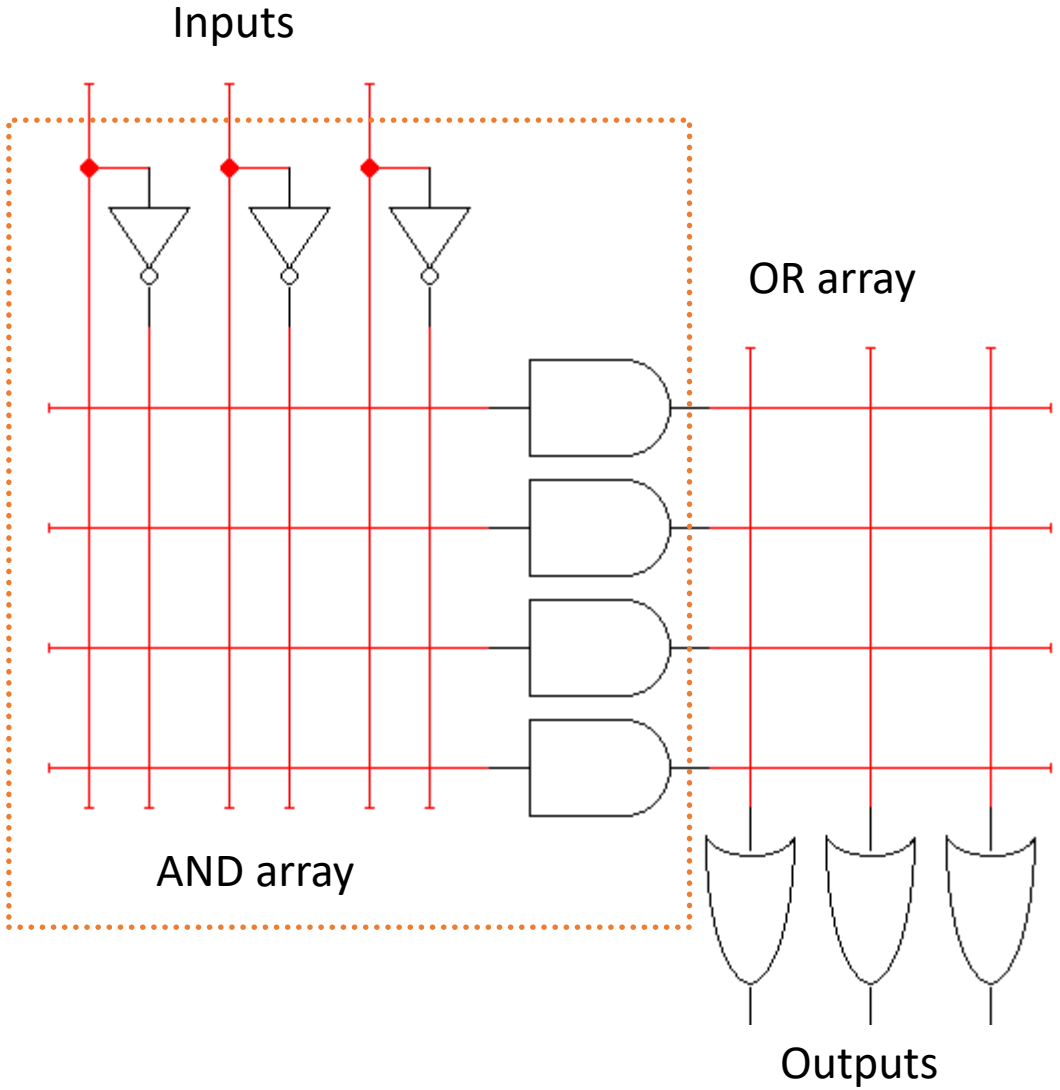
Mask ROM

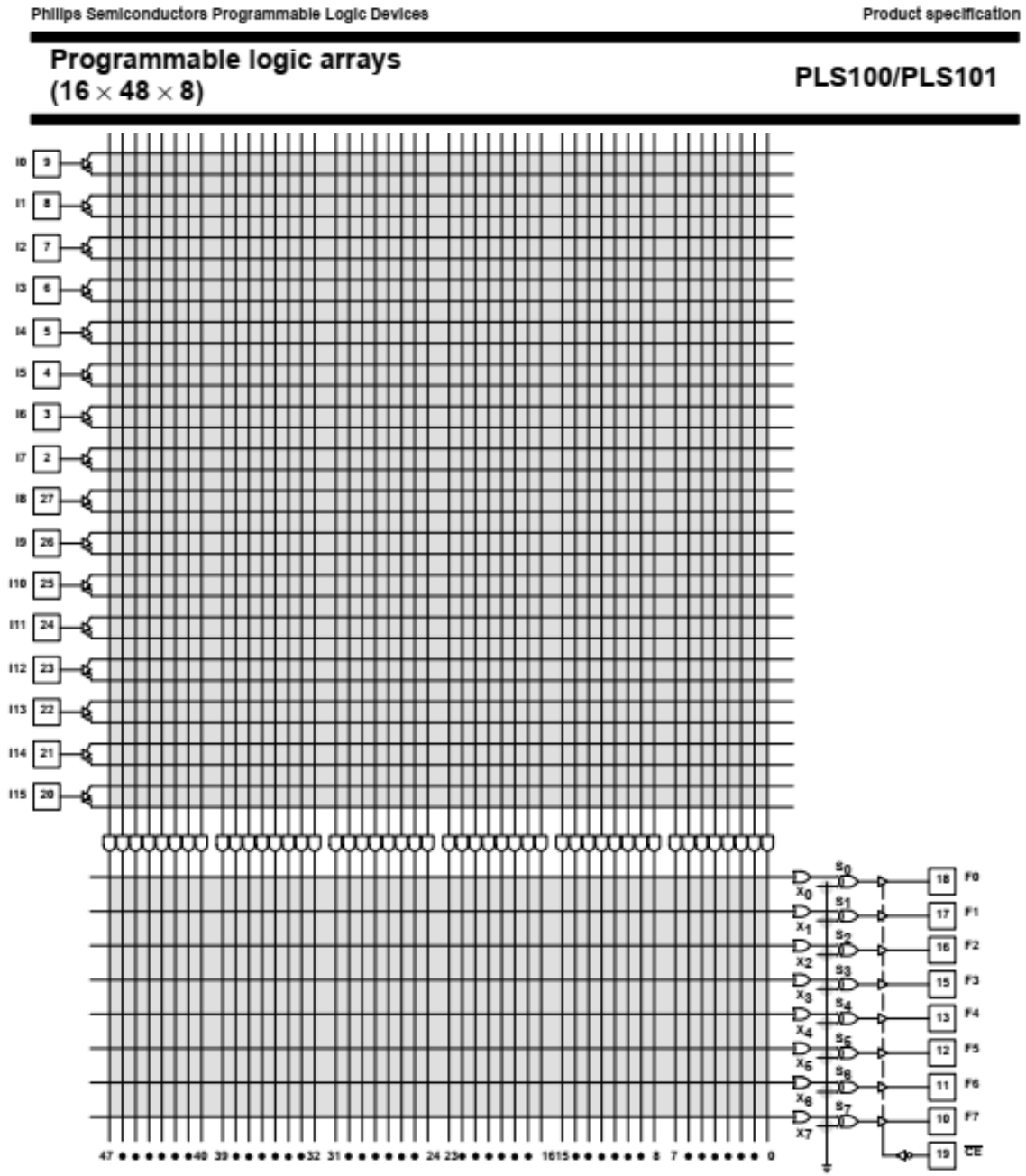
Mask ROM, içeriğin IC üreticisi tarafından programlandığı bir ROM tipidir. Kullanıcı tarafından programlanabilen bir ROM değildir.

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

3 x 4 x 3 PLA





PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

Örnek:

$V_2 = xy'z' + x'z + x'yz'$

X \ YZ	00	01	11	10
0		1	1	1
1	1	0	0	0

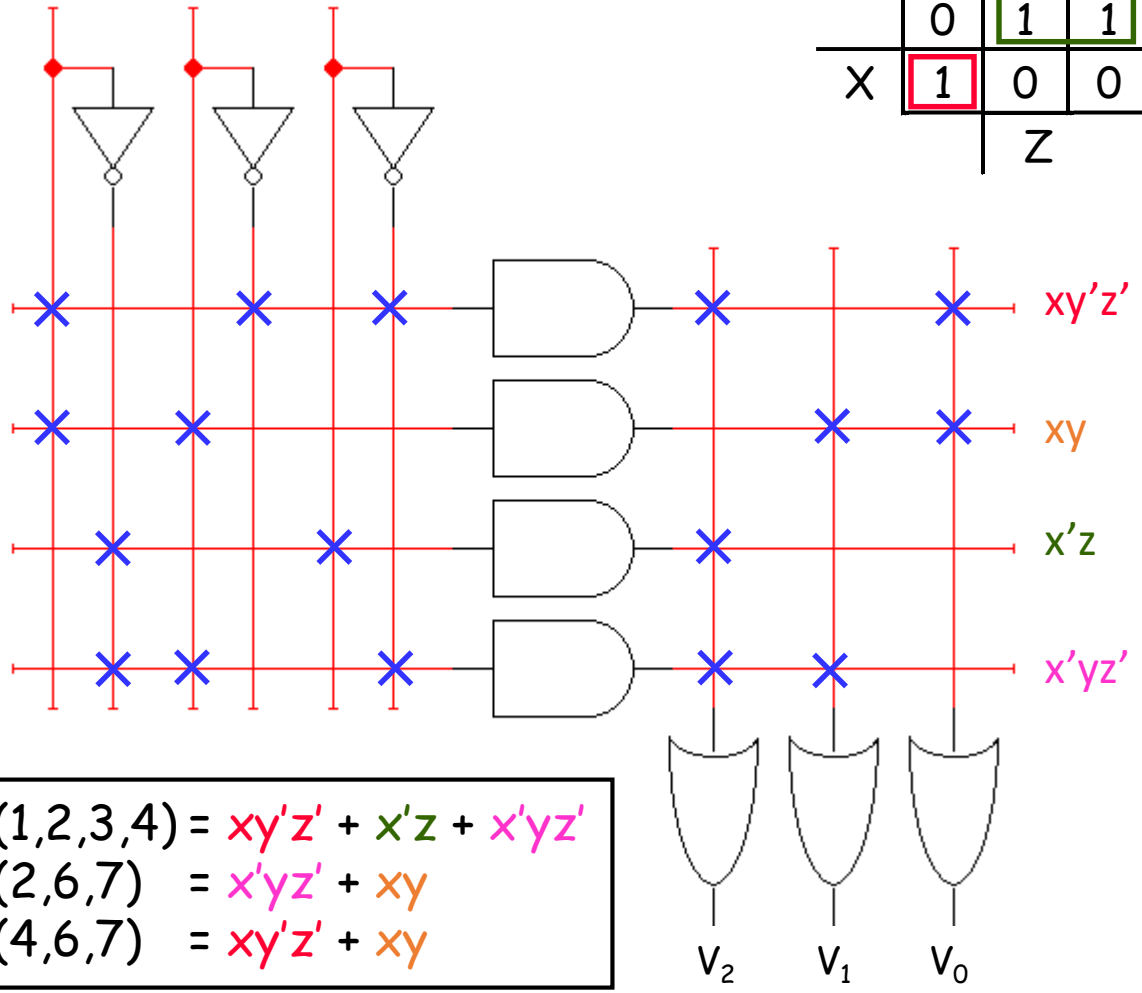
$V_1 = x'yz' + xy$

X \ YZ	00	01	11	10
0	0	0	0	1
1	0	0	1	1

$V_0 = xy'z' + xy$

X \ YZ	00	01	11	10
0	0	0	0	0
1	1	0	1	1

$V_2 = \Sigma m(1,2,3,4)$
$V_1 = \Sigma m(2,6,7)$
$V_0 = \Sigma m(4,6,7)$



$V_2 = \Sigma m(1,2,3,4) = xy'z' + x'z + x'yz'$
$V_1 = \Sigma m(2,6,7) = x'yz' + xy$
$V_0 = \Sigma m(4,6,7) = xy'z' + xy$

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

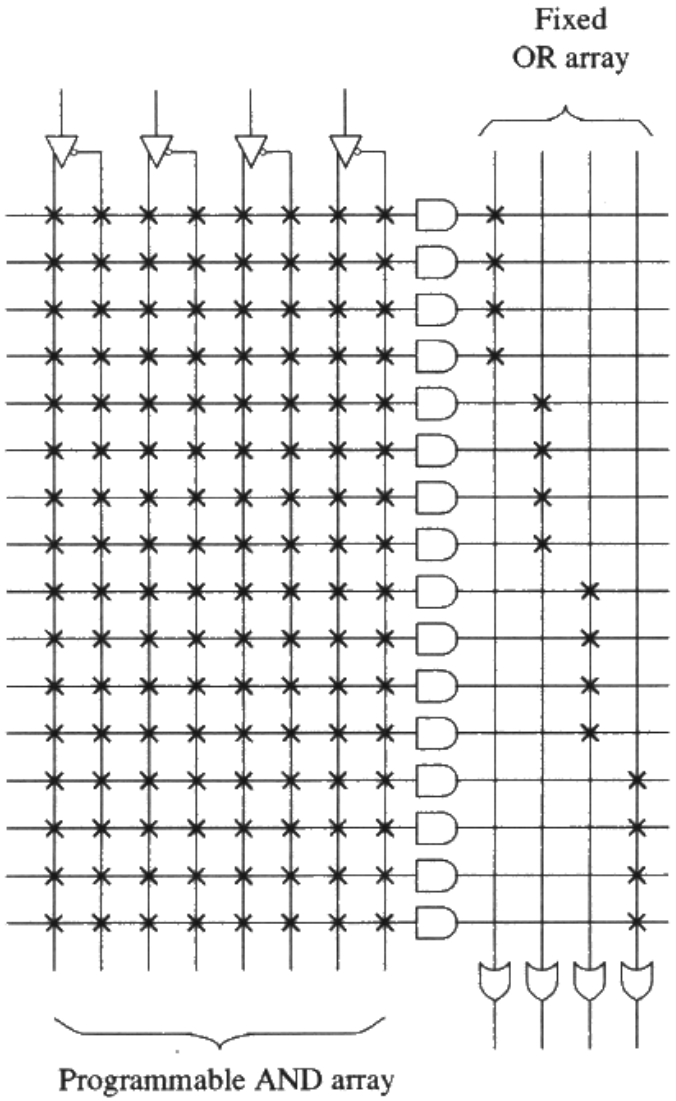


figure 1.8

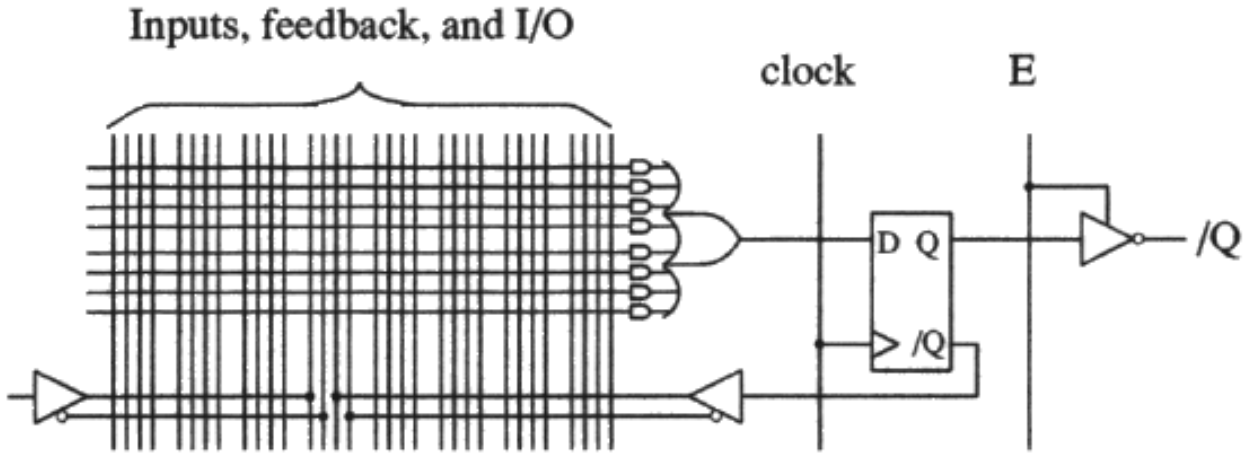


figure 1.9

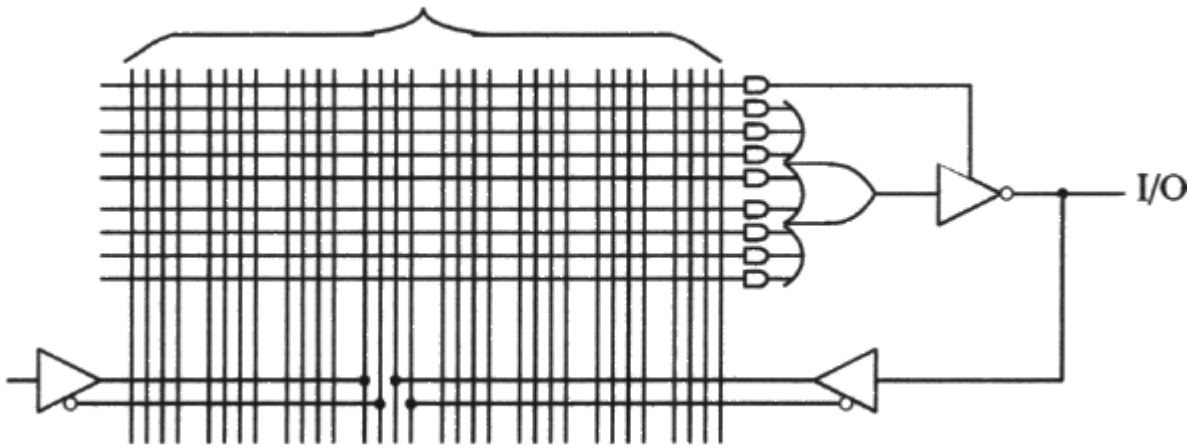
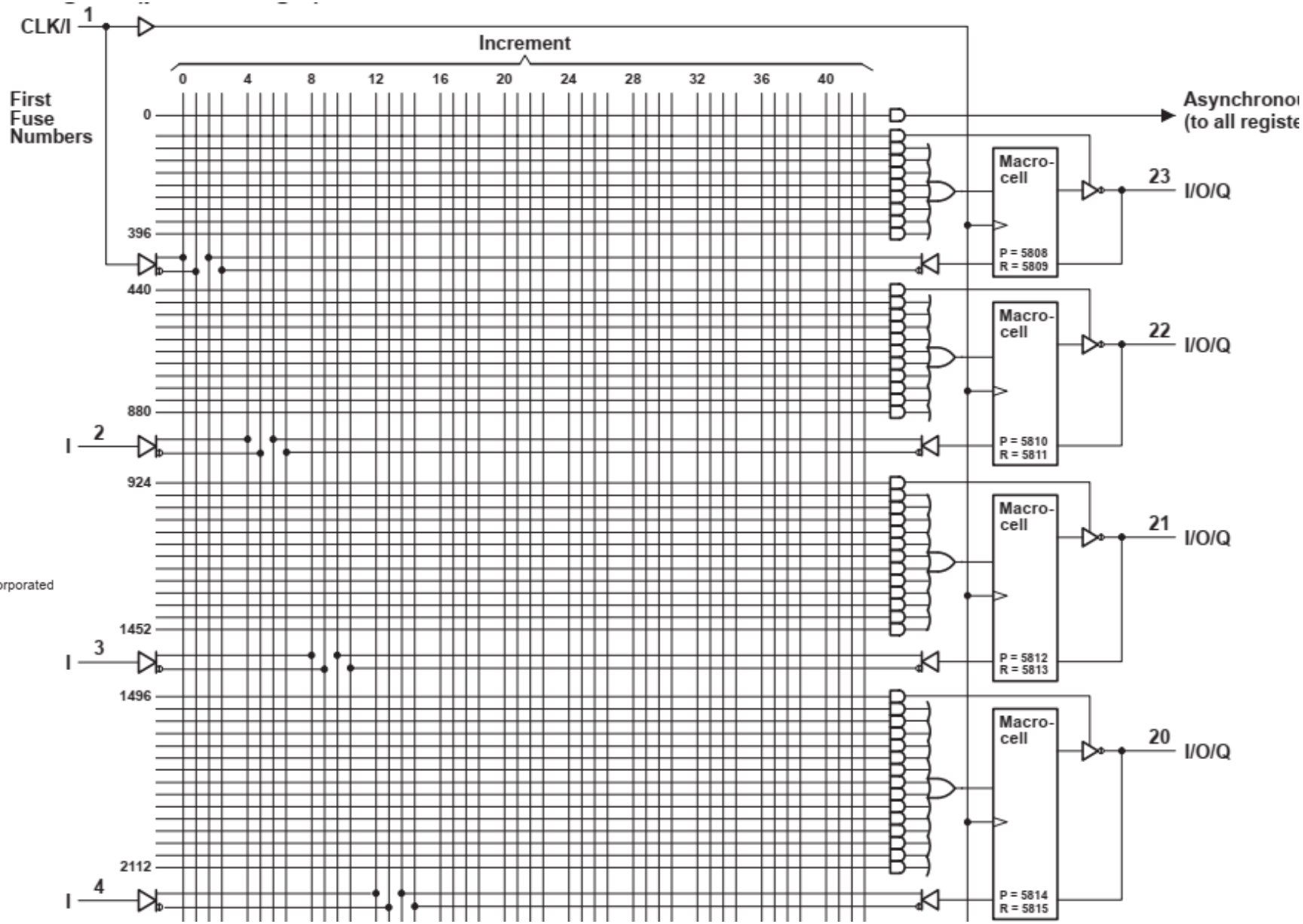


figure 1.10

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

PAL'ler



TEXAS INSTRUMENTS

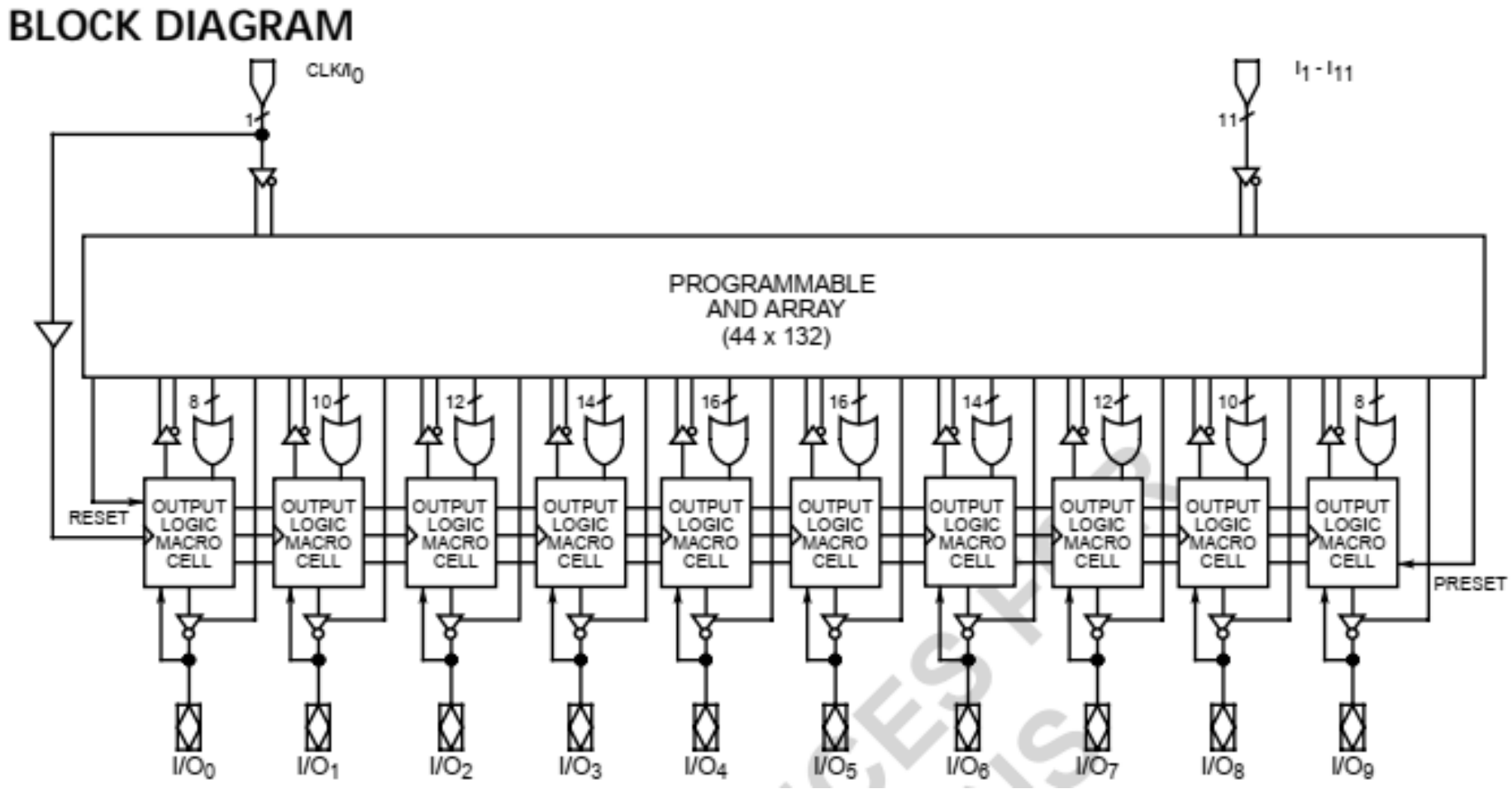
Lattice Semiconductor Corporation

Advanced Micro Devices

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

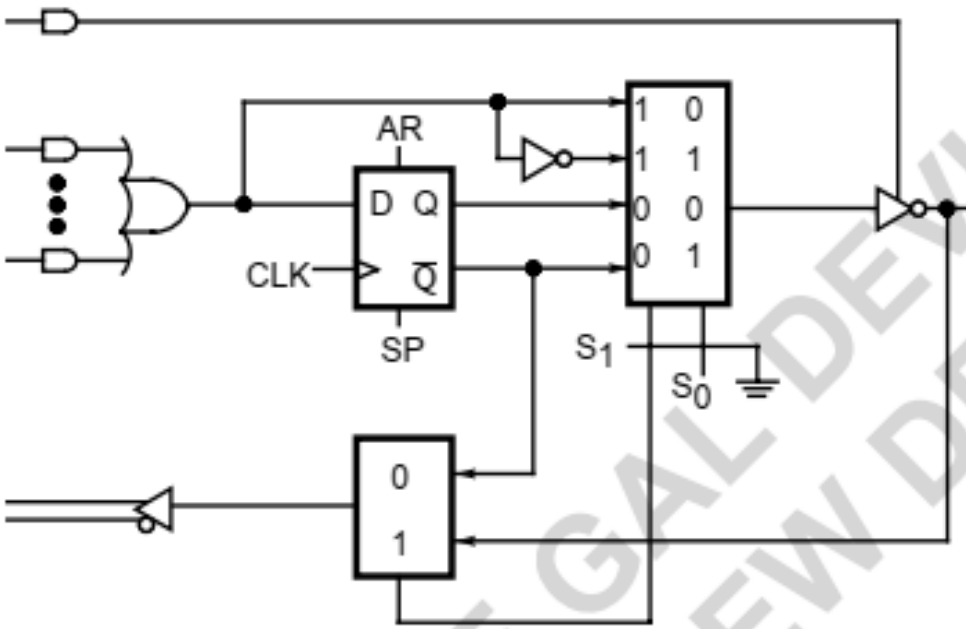
PAL'ler



PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

PAL'ler



S_1	S_0	Output Configuration
0	0	Registered/Active Low
0	1	Registered/Active High
1	0	Combinatorial/Active Low
1	1	Combinatorial/Active High

0 = Programmed EE bit
1 = Erased (charged) EE bit

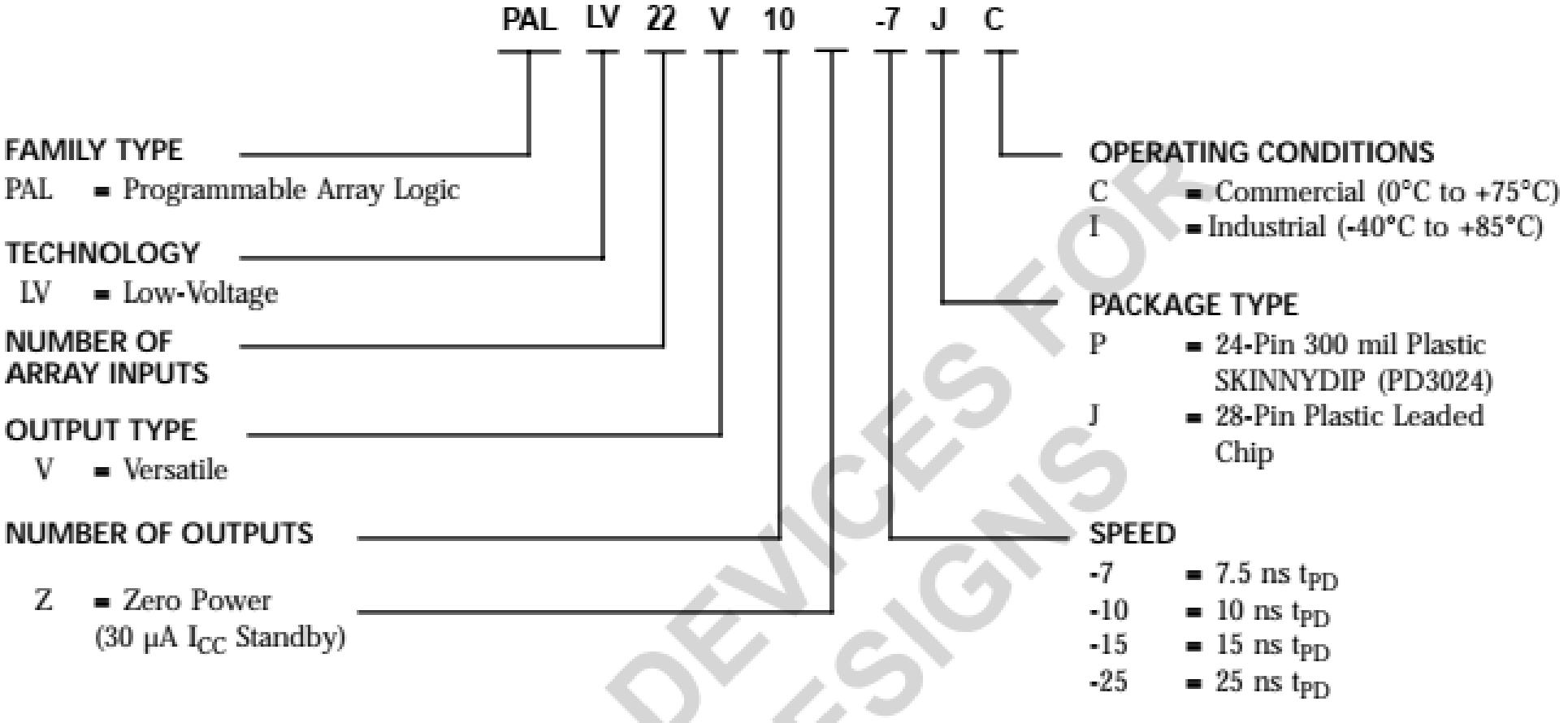
18956C-004

Figure 1. Output Logic Macrocell Diagram

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

PAL'ler



PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

PAL'ler

C:\WINCUPL\EXAMPLES\ATMEL\GATES.PLD

```
Name            Gates;
Partno          CA0001;
Revision        04;
Date            9/12/89;
Designer        G. Woolhiser;
Company         Logical Devices, Inc.;
Location        None;
Assembly        None;
Device          g16v8a;
/*************************************************************************/
 * Inputs:  define inputs to build simple gates from
 */
Pin 1 =  a;
Pin 2 =  b;
/*
 * Outputs:  define outputs as active HI levels
 *
```

```
* Logic:  examples of simple gates expressed in CUPL
inva = !a;                  /* inverters */
invb = !b;
and  = a & b;               /* and gate */
nand = !(a & b);            /* nand gate */
or   = a # b;               /* or gate */
nor  = !(a # b);            /* nor gate */
xor  = a $ b;               /* exclusive or gate */
xnor = !(a $ b);            /* exclusive nor gate */
```

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

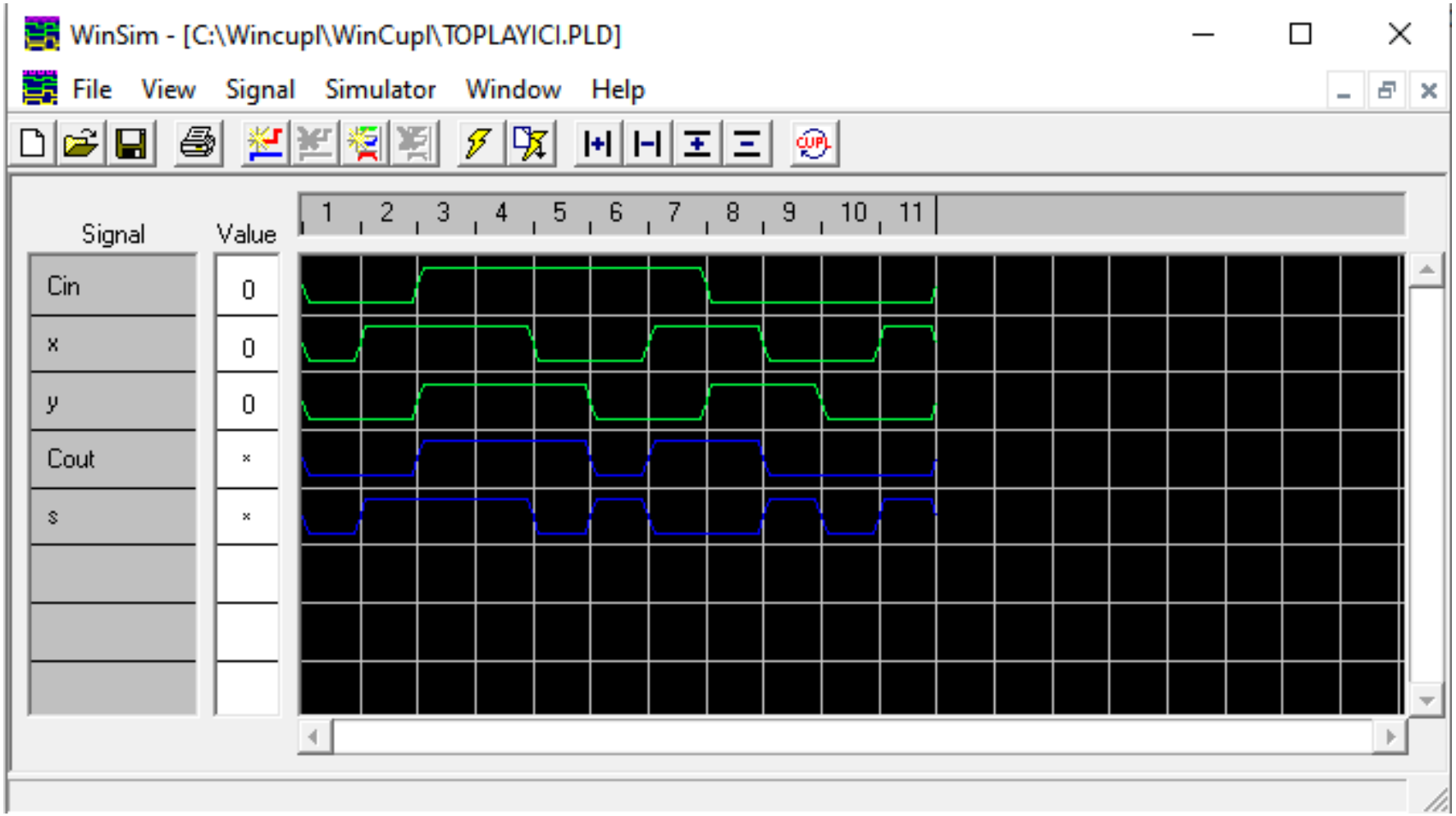
PAL'ler

```
Name     toplayici ;
PartNo   00 ;
Date     29.11.2020 ;
Revision 01 ;
Designer Engineer ;
Company  Y.T.U. ;
Assembly None ;
Location  ;
Device   g22v10 ;

/* *************** INPUT PINS *********************/
PIN   2   =   Cin                      ; /*                    */
PIN   3   =   x                      ; /*                    */
PIN   4   =   y                      ; /*                    */

/* *************** OUTPUT PINS *********************/
PIN   22   =   s                      ; /*                    */
PIN   21   =   Cout                      ; /*                    */

s=Cin$x$y;
Cout=(x&y)#((x$y)&Cin);
```

PROGRAMLANABİLİR LOJİK DEVRELER

Programlanabilir Lojik Diziler

PAL'ler



REFERANSLAR:

1. ‘Lojik Devreler’, Tuncay UZUN Ders Notları, http://tuncayuzun.com/Dersnot_LDT.htm, 2020.
2. ‘Lojik Devre Tasarımı’, Taner ASLAN ve Rifat ÇÖLKESEN, Papatya Yayıncılık, 2013.
3. M. Morris Mano, Sayısal Tasarım (Çeviri), Literatür Yayıncılık: İstanbul, 2003.