3. gaia:
Bloke konbinazionalak

Sistema konbinatzional logikoak

- Zirkuitu konbinatzionalek Z_j funtzioak sortzen dituzte, X_i aldagaien menpean
- X_i balioen konbinaziek Z_j balio bakar bat definitzen dituzte
- Z_i balioak bakarrik aldatzen dira X_j balioen aldaketa bat gertatzen denean

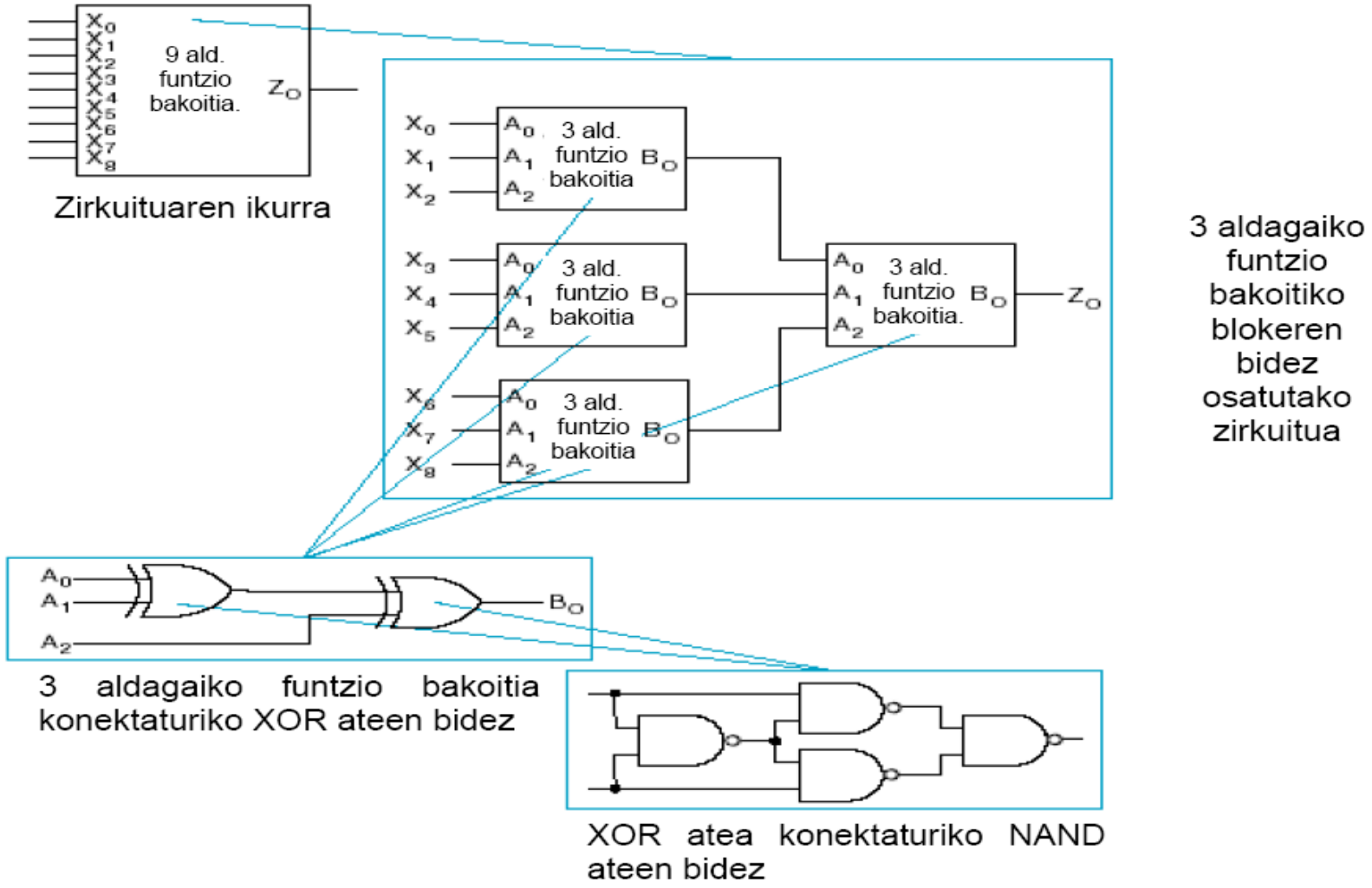
Integrazio eskala

- Zirkuitu integratuak, barruan daramaten ate logikoen kopuruaren arabera sailkatzen dira
- Lau sailkapen daude, integrazio eskalak:
  - SSI (10 ate baino gutxiago)
  - MSI (10 eta 100 ate arteko kopurua)
  - LSI (100 eta 1000 ate arteko kopurua)
  - VLSI (1000 ate logiko baino gehiago)
- Mikroprozesadoreak eta memoriak VLSI zirkuituak dira, eta barruan milloi ate baino gehiago daukate

Diseinu hierarkia

- Baina, nola diseinatu daiteke funtzio logiko bat, mila eta mila ate logiko erabiltzen duena?
- Funtzio logikoen diseinua atal txikiagoetan banatzen badugu, atal horiek aljebra boolearraren bidez ebazteko bezain txikiak, diseinu problema osoa ebaztuko dugu, atalez atal diseinatuta
- Metodo honi diseinu hierarkia deitzen diogu, diseinu mailak sortzen ditugulako; azpiko diseinu maila aljebra boolearraren bidez konponduko dugu, eta bere gainean daudenak, horrela definitutako bloke funtzional horren bidez

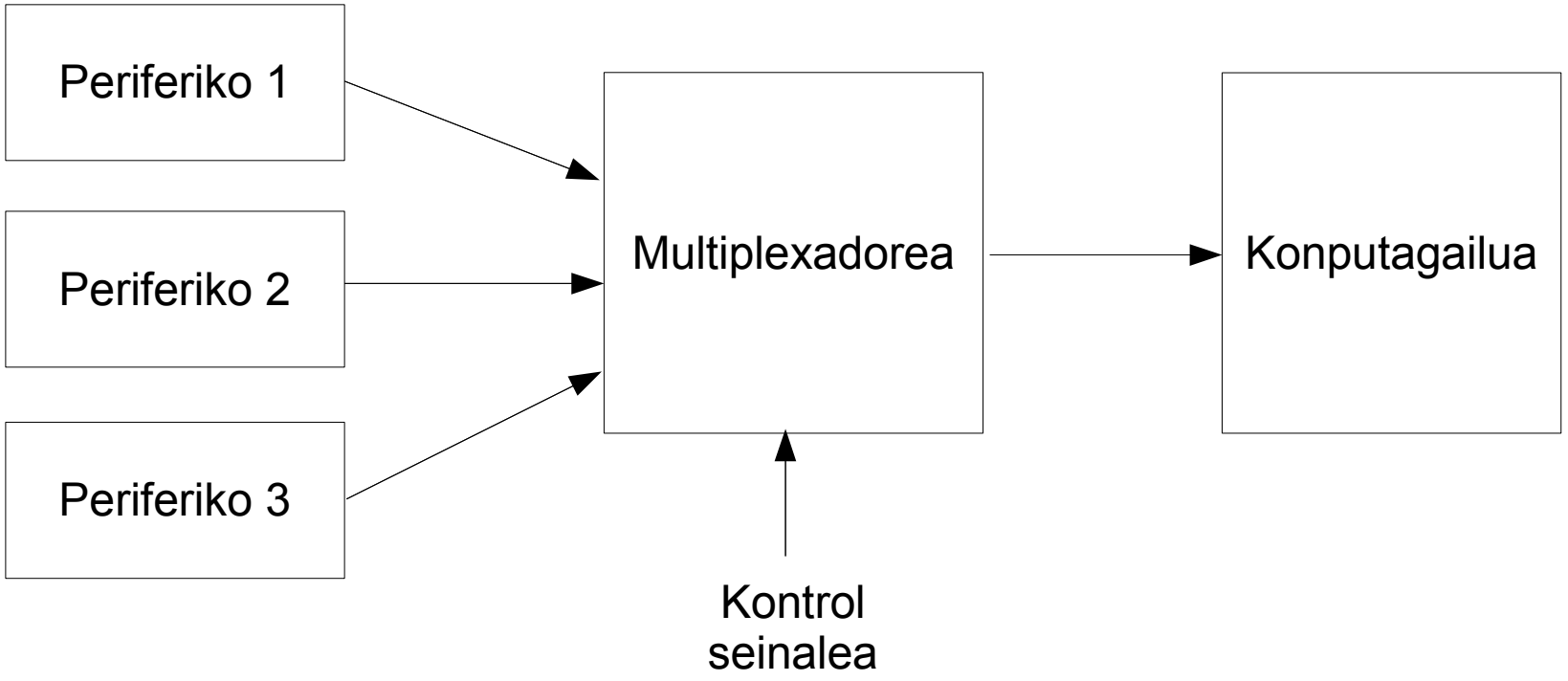
Diseinu hierarkia



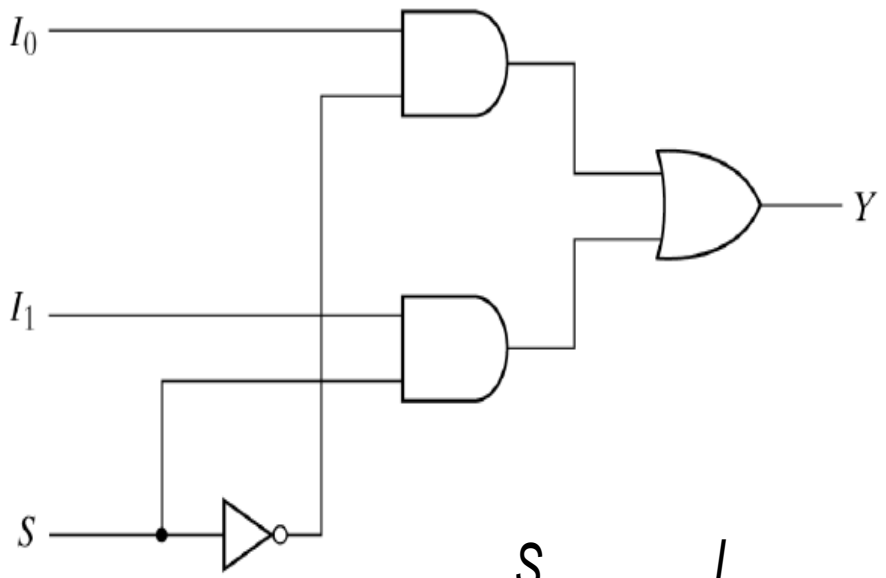
Funtzio osoa 9 aldagaikoa da, baina aljebra boolearraren bidez bakarrik 2 aldagaiko funtzioa definitu egin dugu

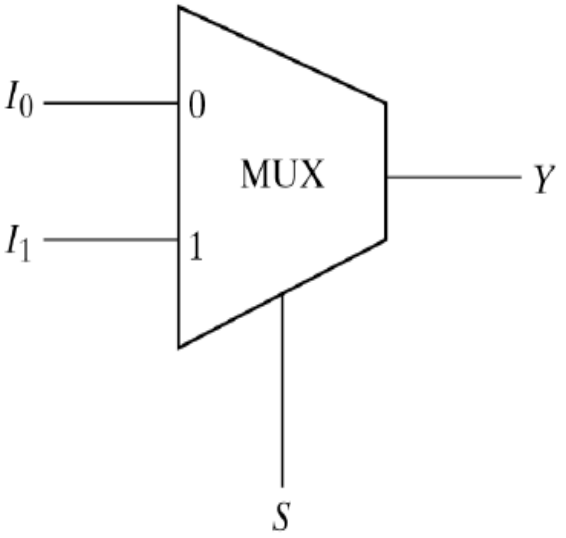
Multiplexadorea

- Aldagai sorta baten arteko bat aukeratzeko erabiltzen da multiplexadorea
- Kontrol aldagaien balioen arabera, sarrerako aldagaien artean bat aukeratzen da
- Sarrera aukeratutaren balioa, irteeran agertuko da



2:1 multiplexadorea

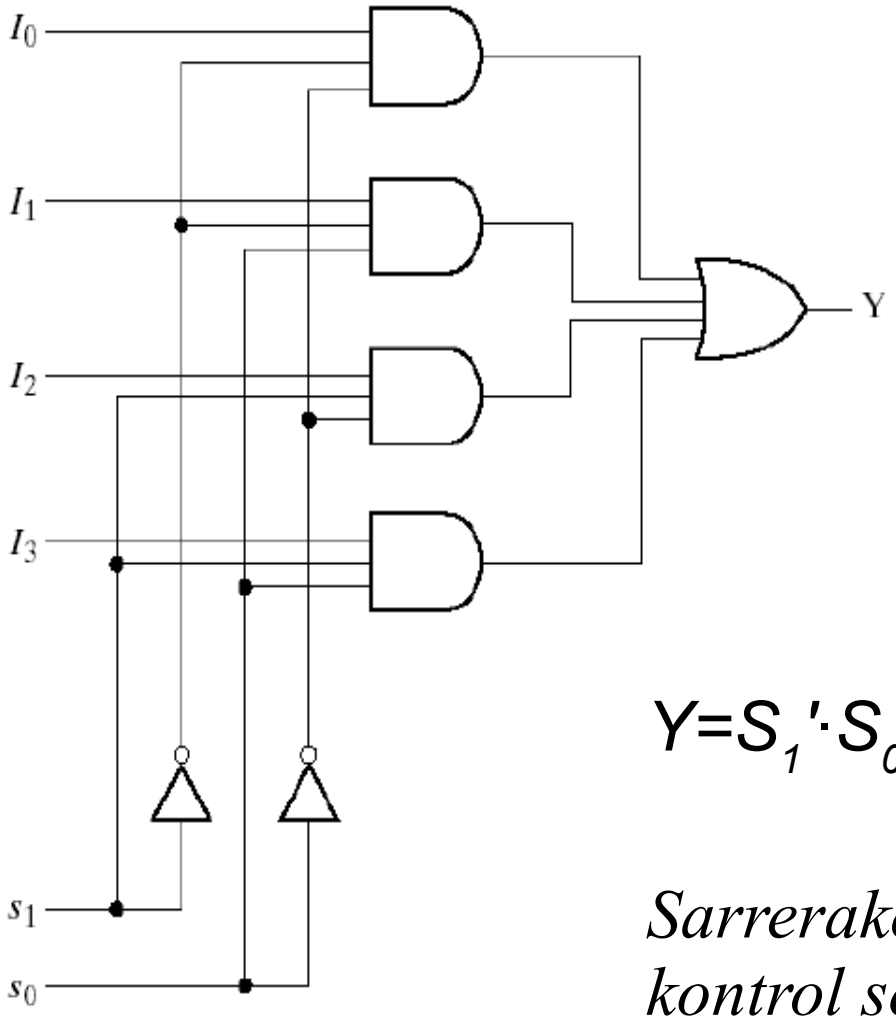




S	I_0	I_1	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$Y = S' \cdot I_0 + S \cdot I_1$$

4:1 multiplexadorea



s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

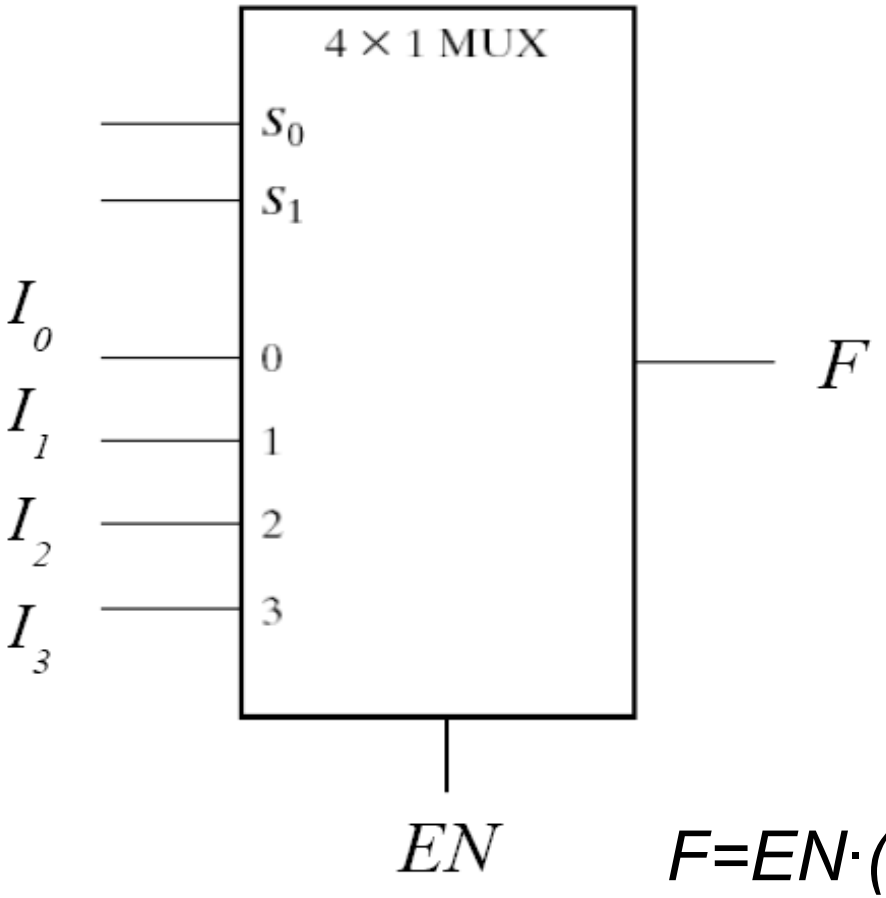
$$Y=S_1'\cdot S_0'\cdot I_0+S_1'\cdot S_0\cdot I_1+S_1\cdot S_0'\cdot I_2+S_1\cdot S_0\cdot I_3$$

Sarrerako aldagai kopurua 2^n izanik, kontrol seinalearen kopurua n da

Gaikuntza sarrera

- Sistema handietan erabiltzen dira bloke funtzionalak
- Sistema handi horietan, blokearen funtzionamendua konbinatzen da, bloke batzuk erabiliz zenbait unetan
- Gaikuntza seinalea erabiltzen da kontrolatzeko noiz erabiltzen den bloke bakoitza
- Desaktibatuta dagoenean, blokeak ez du funtziorik betetzen, irteera konstante (0an) mantenduz
- Orokorrean, gaikuntza seinalearen izena *E* edo *EN* da (*ENABLE*-ren laburdura, ingelesez gaitu)

Gaikuntza sarrerako 4:1 multiplexadorea



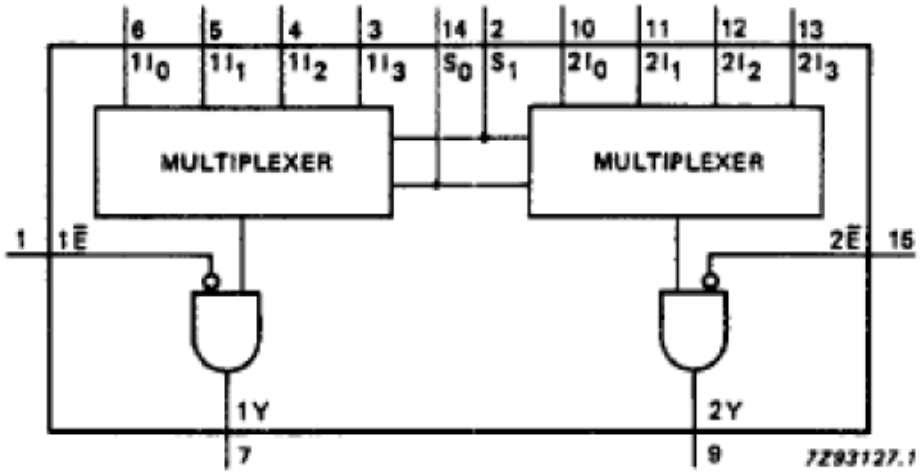
EN	S_1	S_0	F
0	X	X	0
1	0	0	I_0
1	0	1	I_1
1	1	0	I_2
1	1	1	I_3

$$F=EN\cdot(S_1'\cdot S_0'\cdot I_0+S_1'\cdot S_0\cdot I_1+S_1\cdot S_0'\cdot I_2+S_1\cdot S_0\cdot I_3)$$

Busedun multiplexadoreak

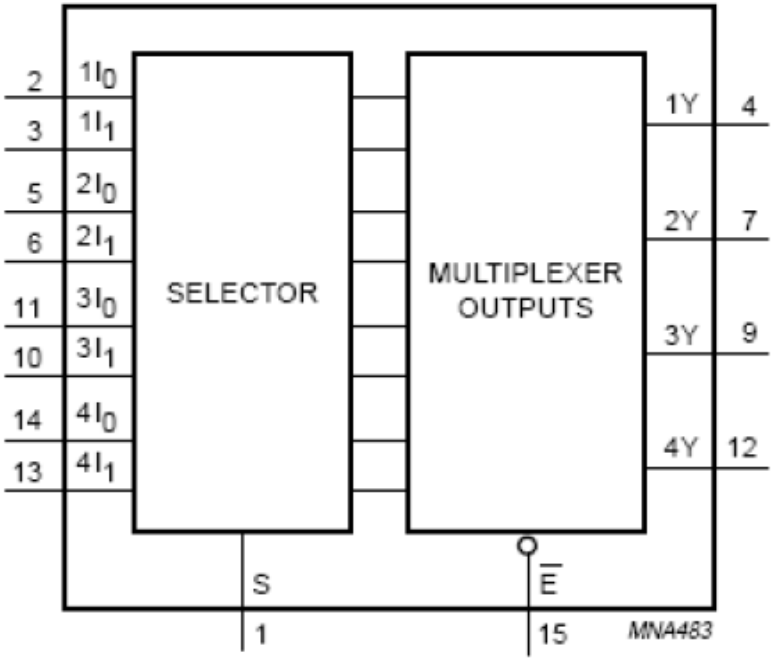
- Seinale elektrikoen multzoa busa deitzen da
- Aldagai digital bat baino gehiago behar dituzten datuak definitzeko erabiltzen dira busak
- Multiplexadorearen sarrerak eta irteerak busak izan daitezke
- Kontrol seinalearen kopurua n bada, sarrerako seinalearen kopurua eta irteerako seinalearen kopuruaren arteko zatidura 2^n da

Bi multiplexadoreko zirkuitu integratuak



SELECT INPUTS		DATA INPUTS				OUTPUT ENABLE	OUTPUT
S_0	S_1	nI_0	nI_1	nI_2	nI_3	$n\bar{E}$	nY
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
H	L	X	L	X	X	L	L
H	L	X	H	X	X	L	H
L	H	X	X	L	X	L	L
L	H	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

74HC/HCT153



INPUT				OUTPUT
$\bar{E}$	S	nI_0	nI_1	nY
H	X	X	X	L
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

74AHC157

Multiplexadorea VHDLren bidez

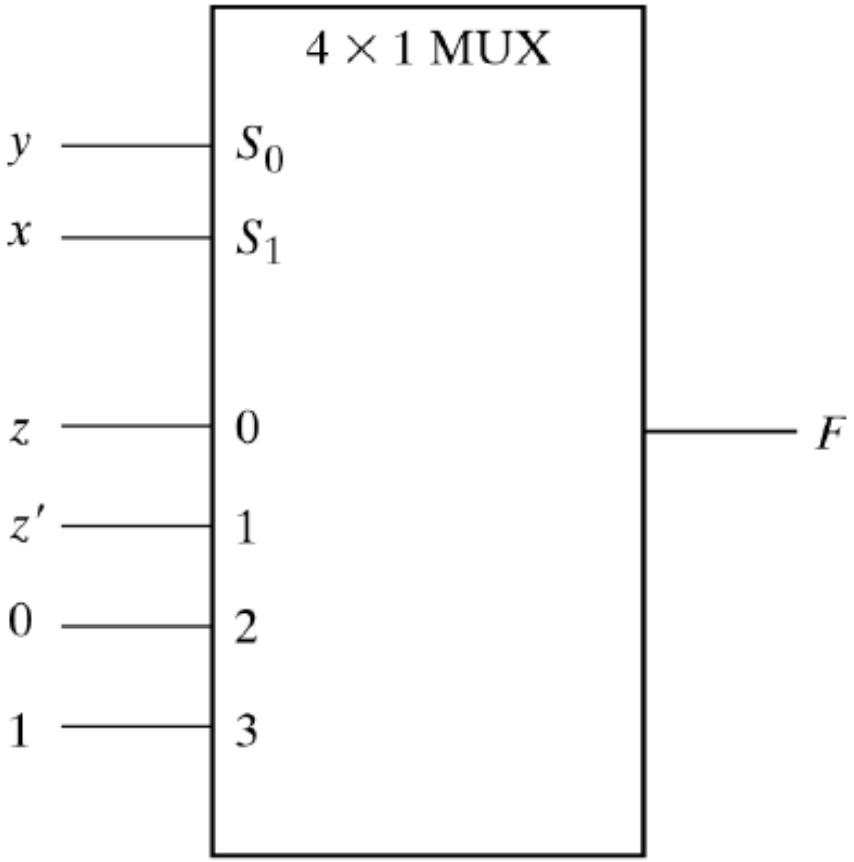
- Hauxe da 4:1 multiplexadore baten deskripzioa VHDLn:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all ;

ENTITY mux4to1 IS
PORT (  w0, w1, w2, w3   : IN       STD_LOGIC ;
s     : IN       STD_LOGIC_VECTOR(1 DOWNTO 0) ;
f     : OUT    STD_LOGIC ) ;
END mux4to1 ;

ARCHITECTURE a OF mux4to1 IS
BEGIN
WITH s SELECT
f <=       w0 WHEN "00",
           w1 WHEN "01",
           w2 WHEN "10",
           w3 WHEN OTHERS ;
END a ;
```

Funtzioen inplementazioa multiplexadoreen bidez

- Multiplexadore baten irteeraren adierazpena beti da sarrera batetik biderkatuta kontrol seinalearen minterm-eko batuketa
- Funtzio baten adierazpen kanonikoa beti da minterm-eko batuketa
- Multiplexadore baten kontrol seinaleak sarrera gisa erabiltzen baditugu, funtzio baten adierazpen kanokikoa irudikatu dezakegu
- Multiplexadorearen sarrerak erabili ditzakegu minterm-ak beste aldagai batekin osatzeko

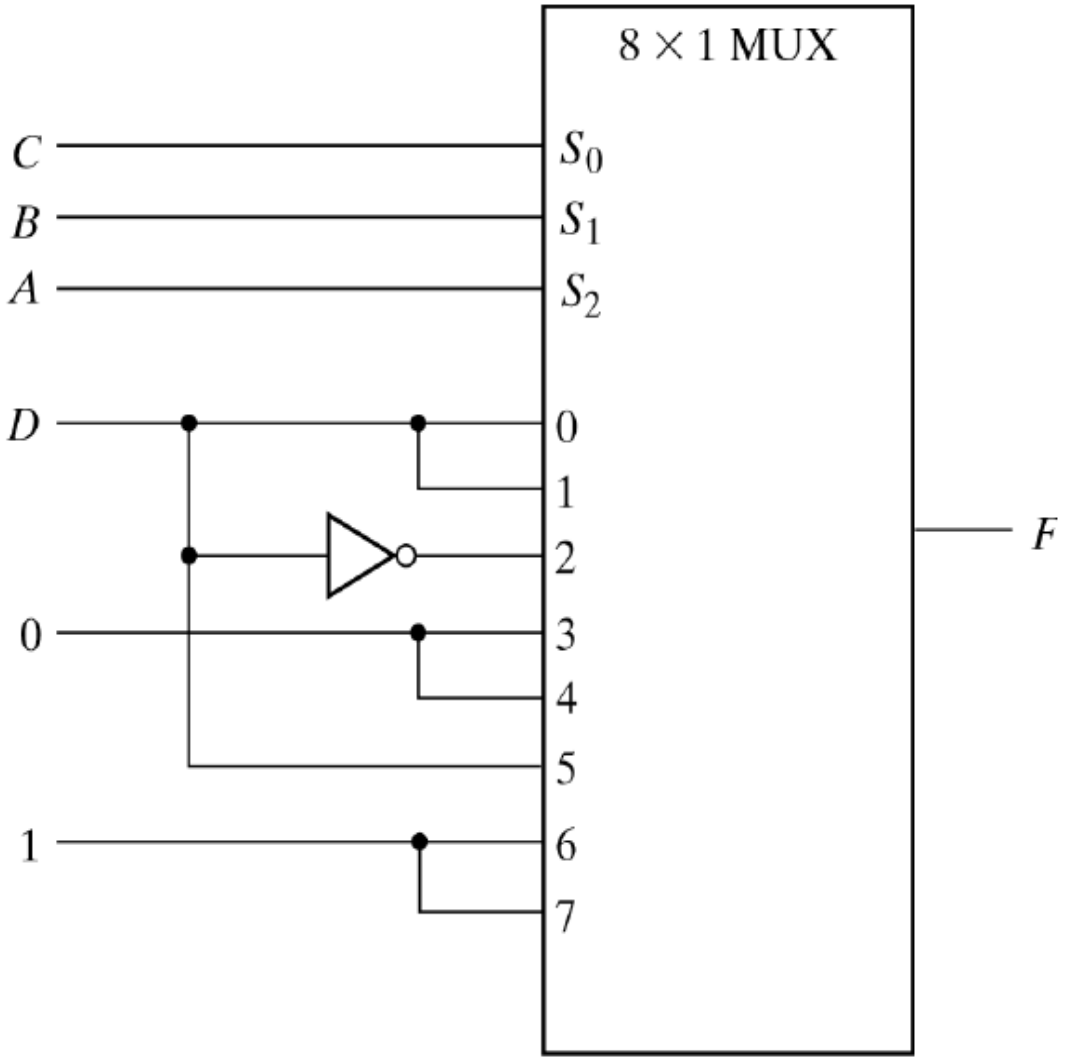
Funtzioen inplementazioa multiplexadoreen bidez



x	y	z	F	
0	0	0	0	$F = z$
0	0	1	1	
0	1	0	1	$F = z'$
0	1	1	0	
1	0	0	0	$F = 0$
1	0	1	0	
1	1	0	1	$F = 1$
1	1	1	1	

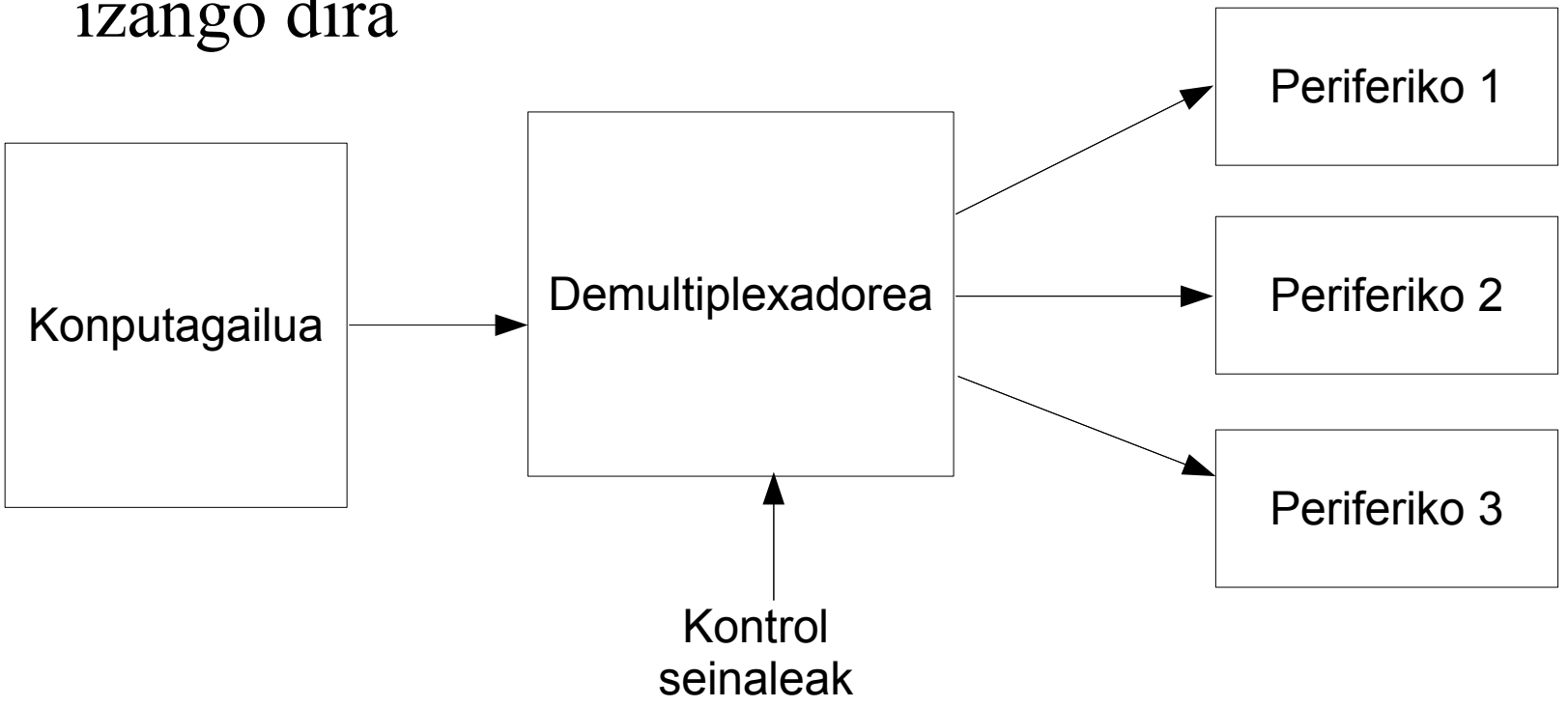
Funtzioen inplementazioa multiplexadoreen bidez

A	B	C	D	F	
0	0	0	0	0	$F = D$
0	0	0	1	1	
0	0	1	0	0	$F = D$
0	0	1	1	1	
0	1	0	0	1	$F = D'$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = D$
1	0	1	1	1	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	



Demultiplexadorea

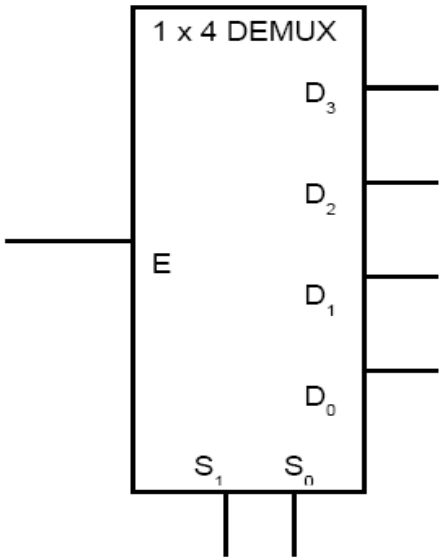
- Aldagai bat helbideratzeko funtzio sorta baten arteko baterantz, erabiltzen da demultiplexadorea
- Kontrol aldagaien balioen arabera, sarrerako aldagaien balioa funtzio bakar batera eramaten da
- Blokearen irteerak, beste hainbeste blokeetarako sarrerak izango dira

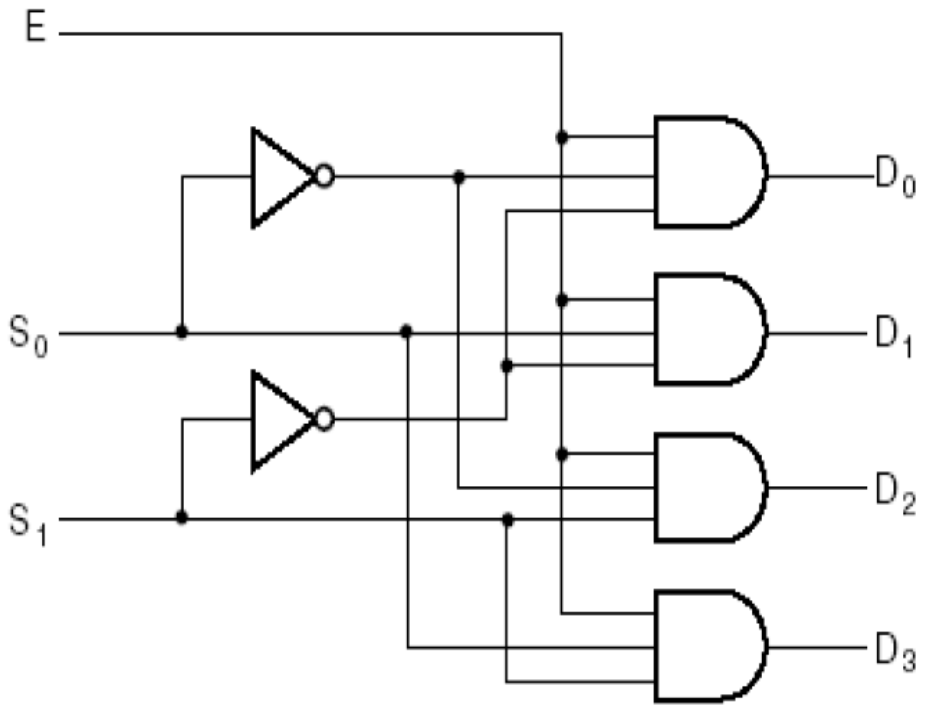


1:4 demultiplexadorea

E	S_1	S_0	D_0	D_1	D_2	D_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

$$D_0 = E \cdot S_1' \cdot S_0'$$
$$D_1 = E \cdot S_1' \cdot S_0$$
$$D_2 = E \cdot S_1 \cdot S_0'$$
$$D_3 = E \cdot S_1 \cdot S_0$$





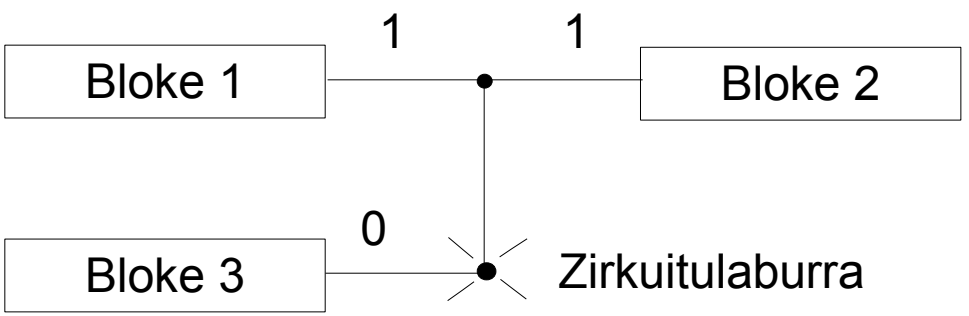
Demultiplexadorea VHDLren bidez

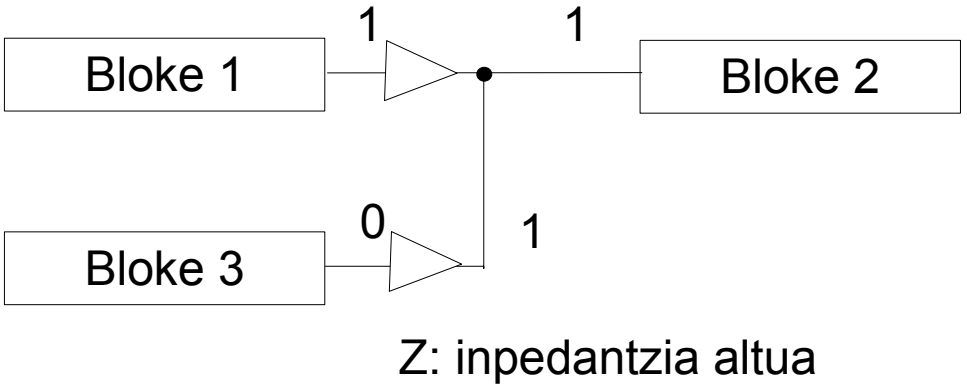
- Hauxe da 1:4 demultiplexadore baten adierazpena VHDLn:

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY demux1to4 is
PORT(    w0, w1, w2, w3      : out std_logic;
         f                   : in  std_logic;
         s                   : in  std_logic_vector(1 downto 0));
END demux1to4;

ARCHITECTURE a OF demux1to4 IS
BEGIN
         w0<= f when s="00" else '0';
         w1<= f when s="01" else '0';
         w2<= f when s="10" else '0';
         w3<= f when s="11" else '0';
END a;
```

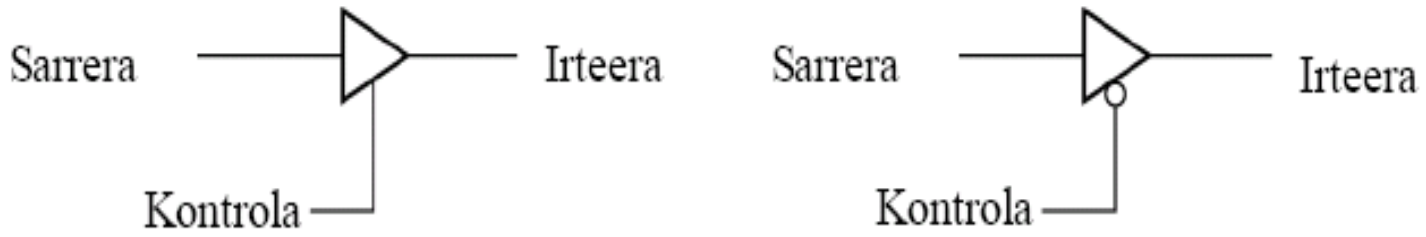
Hiru egoerako ateak

- Askotan busak zenbait bloke funtzionalen artean konpartitzen dira, baina lerro bat hiru blokegatik erabilia izatea zirkuitulaburraren bat sortu dezake
- Beharrezkoa da zihurtatzea bloke pare bakar bat erabiltzen duela busa (seinalearen igorlea eta jasotzailea)
- Elektrikoki moztu daiteke bloke baten konexioa busekin, hiru egoera ate baten bidez (inpedantzia altua)

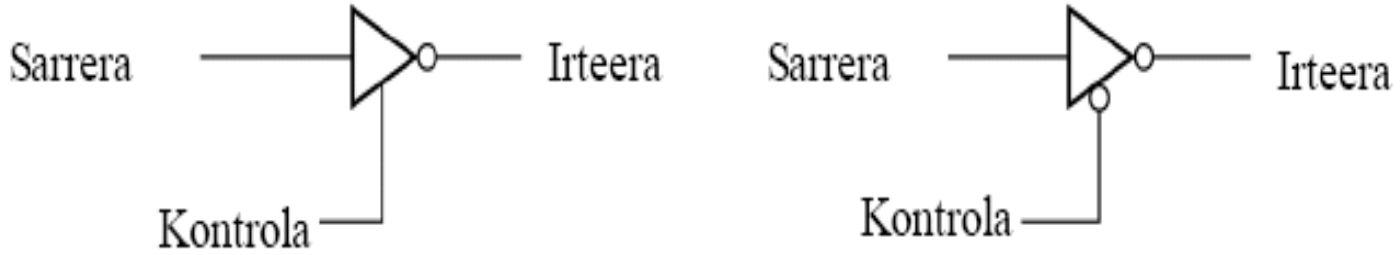




Hiru egoerako ateak



Ate ez inbertsoreak (Sarrera= Irteera), kontrol seinalea maila altuko eta maila baxukoa



Ate inbertsoreak (Sarrera= Irteera'), kontrol seinalea maila altuko eta maila baxukoa

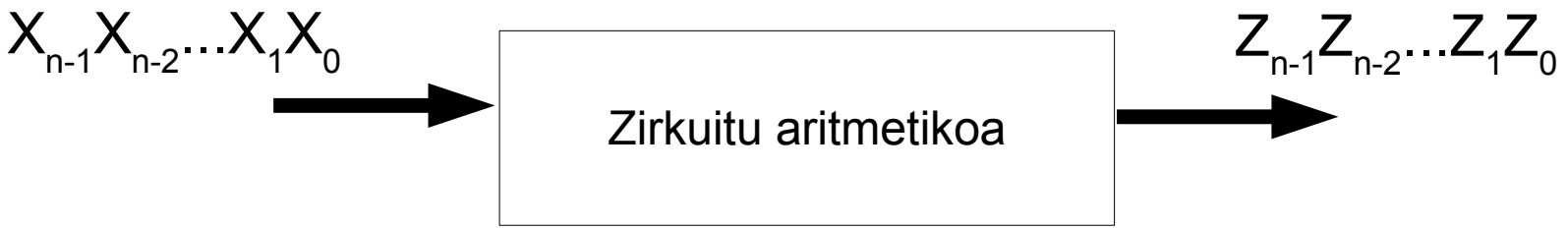
Hiru egoerako atea VHDLren bidez

- Hauxe da kontrol seinalea maila altuko ate ez inbertsorearen adierazpena VHDLn

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY tri_st is
PORT(   I      : out std_logic;
        S      : in  std_logic;
        K      : in  std_logic);
END tri_st;

ARCHITECTURE a OF tri_st IS
BEGIN
        I<= S when K='1' else 'Z';
END a;
```

Sistema konbinatzional aritmetikoak

- Sistema konbinatzional aritmetikoetan, sarrera edo irteerako datuak zenbakiak dira
- Zenbakiak aldagai digitalen bidez adierazten dira, 0 eta 1 zenbakiak aldagaien balio biren bidez adieraziz
- Zenbatzeko sistema bitarra erabiliko dugu, zeren eta 0 eta 1 baino ez du erabiltzen
- Zenbaki aldagaiak irudikatzeko azpiindize baten bidez adierazten dira zifra bitarrak



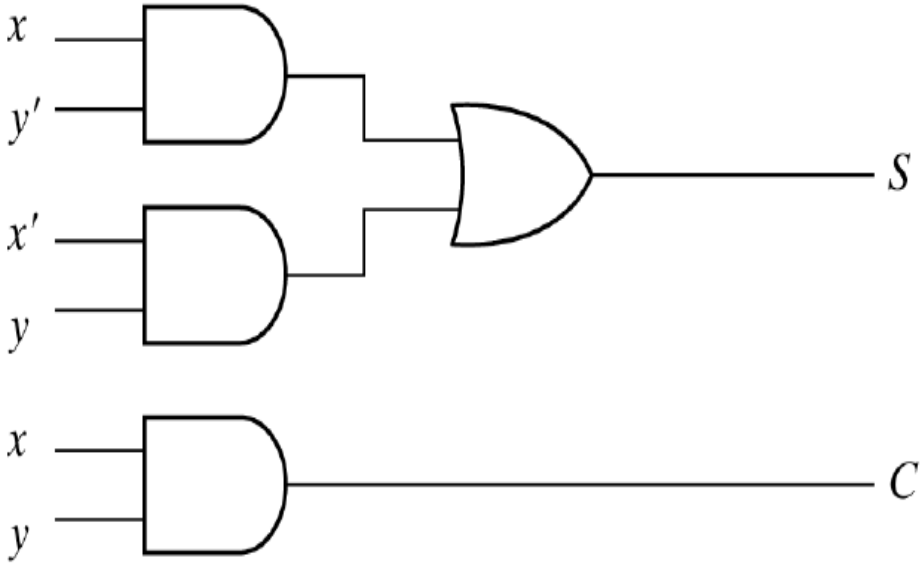
Erdi-batutzailea

- Bi zenbaki bitarreko batuketa funtzio logiko bat da, bi aldagaien menpeko funtzioa
- Batuketaren emaitzako aukera guztiak jasotzeko, bi funtzio behar dugu: S (batuketa) eta C (bururakoa)

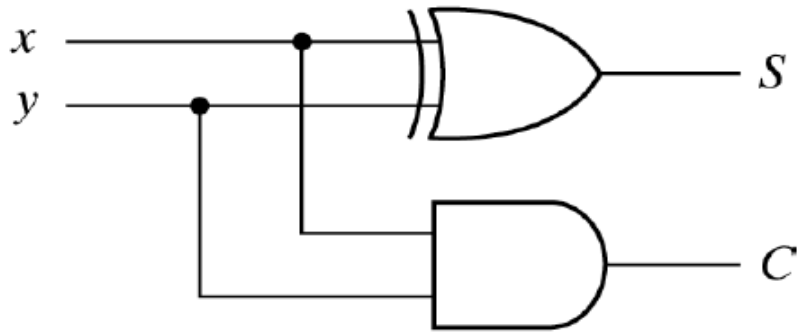
X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

```
  0
+ 0
 00

  0
+ 1
 01

  1
+ 1
 1 0
```

Erdi-batutzailearen implementazioa



(a) $S = xy' + x'y$
$C = xy$

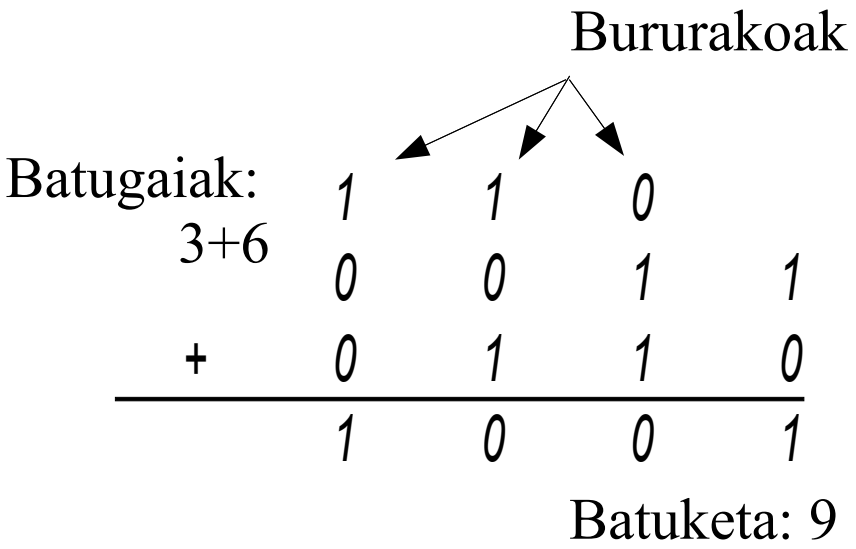


(b) $S = x \oplus y$
$C = xy$

- Batuketaren bi funtzio S eta C-ren adierazpen aljebraikoa da hau
- Erabilgarria bihurtzeko, n biteko batuketa garatu egin behar dugu

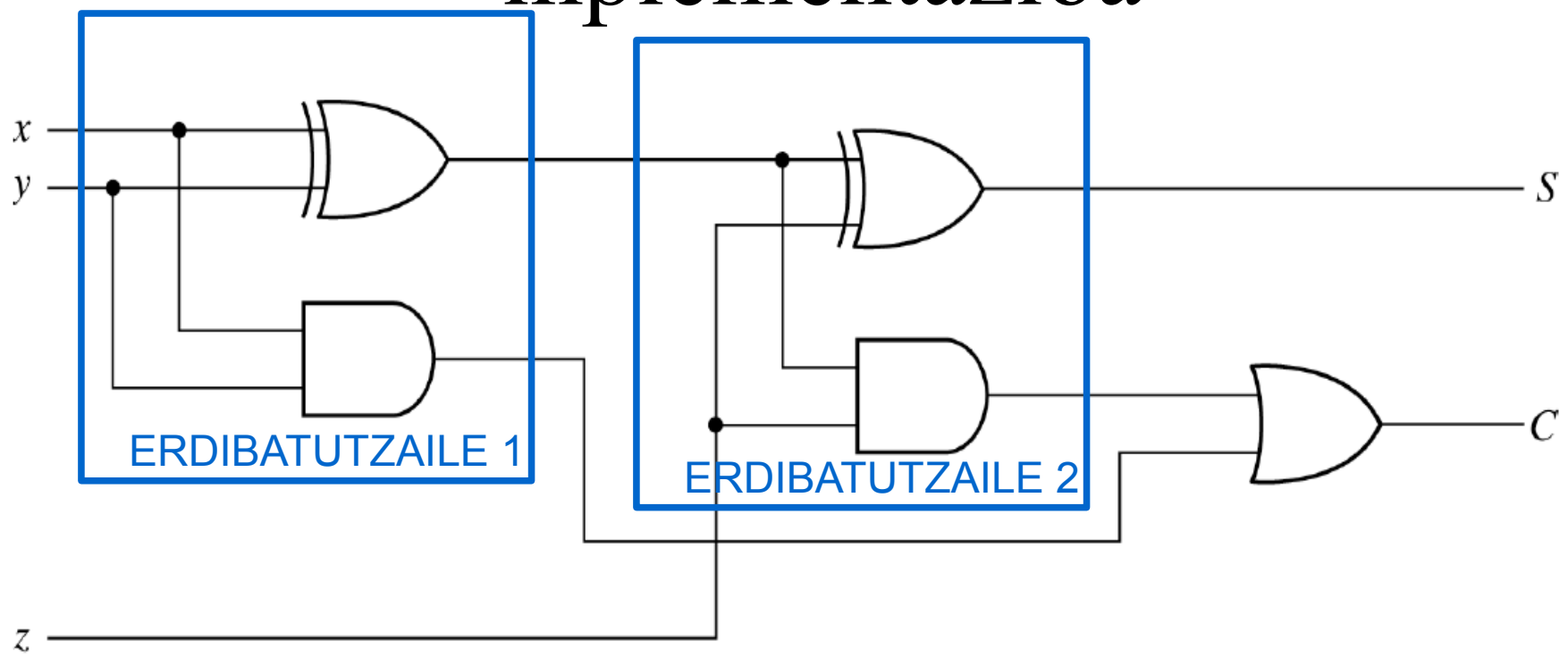
Erabateko batutzailea

- n biteko batuketak egiteko, batugai bakoitzeko zifra bakar bateko batuketak egingo ditugu
- Batugai bakoitzeko zifra bat eta aurreko zifraren bururakoa batuko dugu
- Beraz, hiru batugaiko batutzaile bat behar dugu: erabateko batutzailea



X	Y	Z	S	C
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

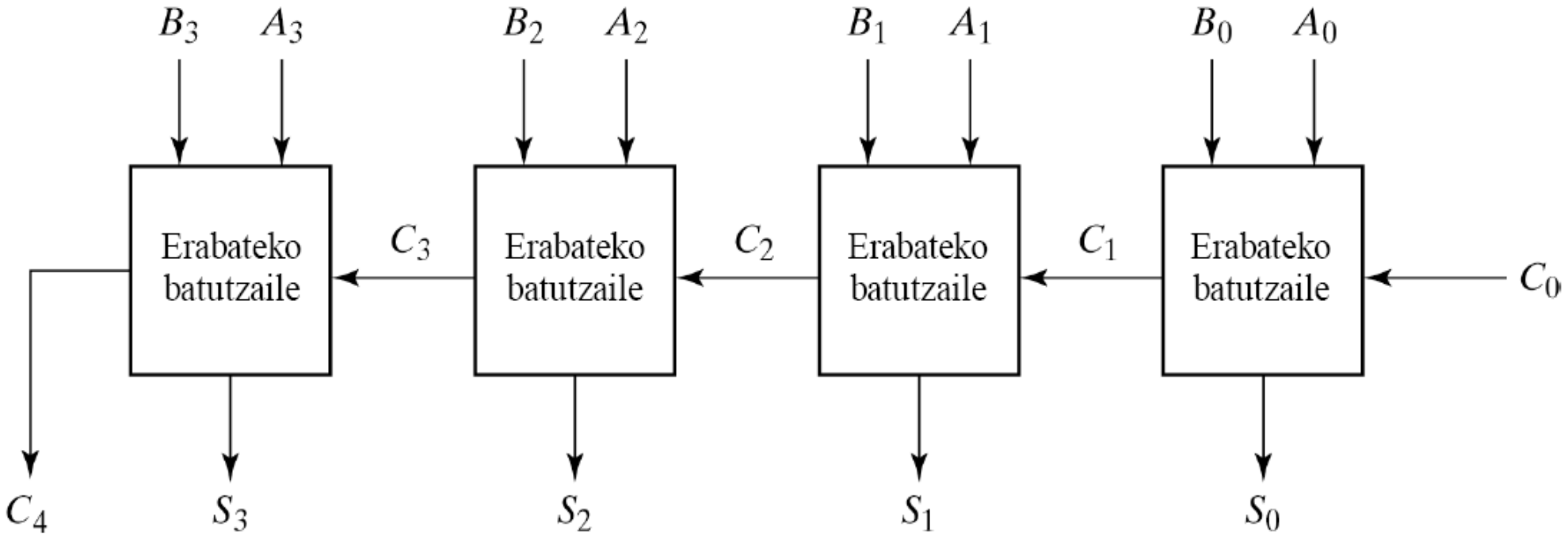
Erabateko batutzailearen inplementazioa



$$S = Z \oplus (X \oplus Y)$$
$$C = (X \cdot Y) + (X \oplus Y) \cdot Z$$

S eta C-ren egia taulatik lortutako adierazpena erakusten du erabateko batutzailea bi erdibatutzailearen konbinazioa dela

4 biteko batutzailea



- n erabateko batutzaileak konbinatuz, n biteko batutzailea sortzen da
- S_{n-1} zifra lortzeko, C_{n-1} ezagutzea behar dugu, eta hau aurreko bloketik etorritakoa da➔ atzerapenak batutzen dira
- Gero eta altuago n zenbakia, motelago batutzailea

Kentzailea

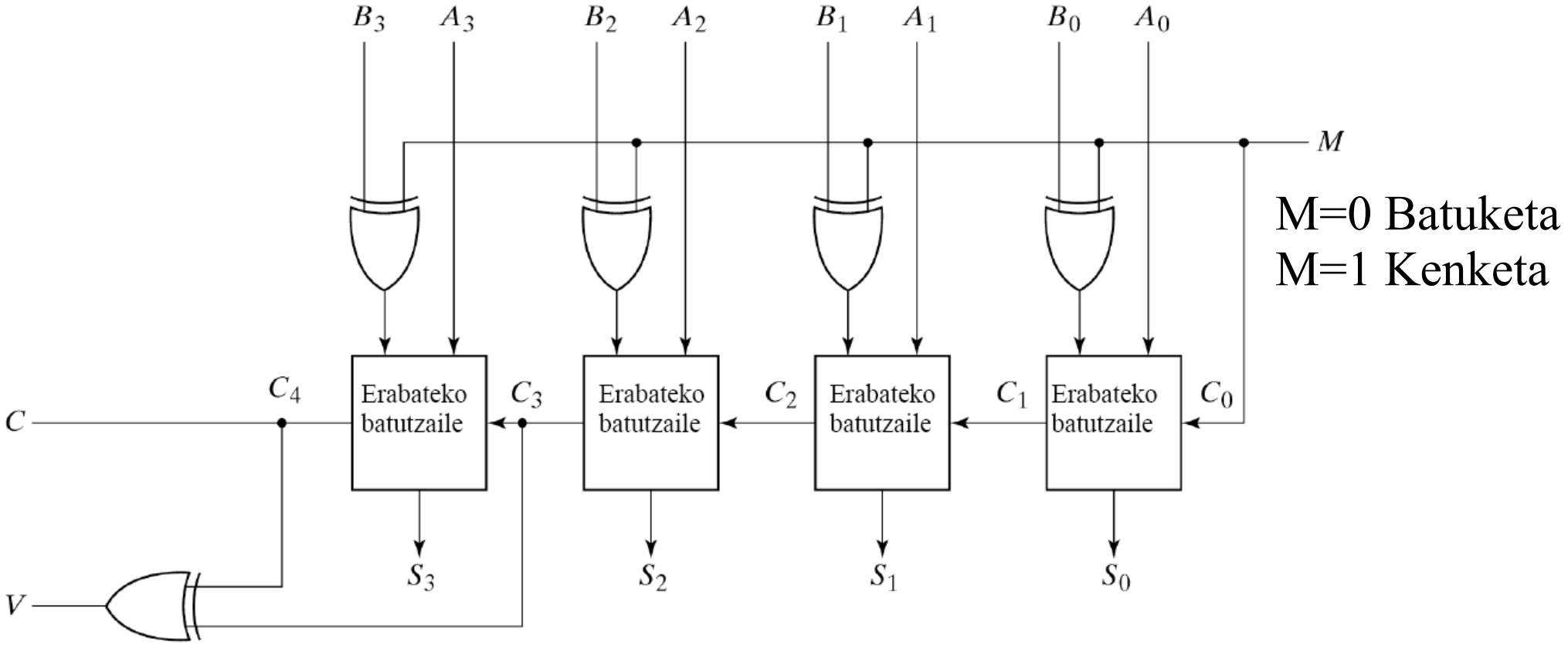
- Kenketa bitarra, batuketa bezala definitu daiteke
- Kenkizuna, kentzailea baino txikiagoa bada, emaitza negatiboa da
- Horrela bada, eragigaien ordena aldatuta egiten da kenketa, eta emaitzaren zeinua adierazten da erantsitako zifra bitar baten bidez (0 positiboa bada eta 1 negatiboa bada)
- Batuketa osatzeko baino elementu gehiago beharrezkoa da kenketa osatzeko metodo hau, eta horregatik ez dugu erabiliko

```
  0
- 0
---
  0 0

  1
- 0
---
  0 1

  1
- 1
---
  0

  0
- 1
---
  1 1
```

Biko osagarriko kenketa

- Biko osagarriko kenketa egiteko kentzailea zenbaki negatibo bat bihurtzen da, eta gero batuketa egin behar da: A-B=A+(-B)
- Eragiketa da: 2^n -B, eta hau burutzeko zenbakiaren 0-ren lekuan 1-ak jarriko ditugu, eta 1-en lekuan 0-ak (bateko osagarria), eta gehi bat egin
- Emaitza negatiboa bada, biko osagarrian agertzen da, eta horrela, batuketa eta kenketa eragigailu bakar batekin egin daiteke
- Magnitudea ezagutzeko, berriro biko osagarria egin behar zaio emaitzari
- Emaitza positiboa bada, azken zifran bururakoa 1 da, eta mesprezagarria da

4 biteko batutzaile-kentzaile (biko osagarrian)



- V seinalea 1 denean, kenketaren emaitza ezin da 4 biten bidez adierazi
- C seinalea 1 denean batuketaren emaitza ezin da 4 biten bidez adierazi

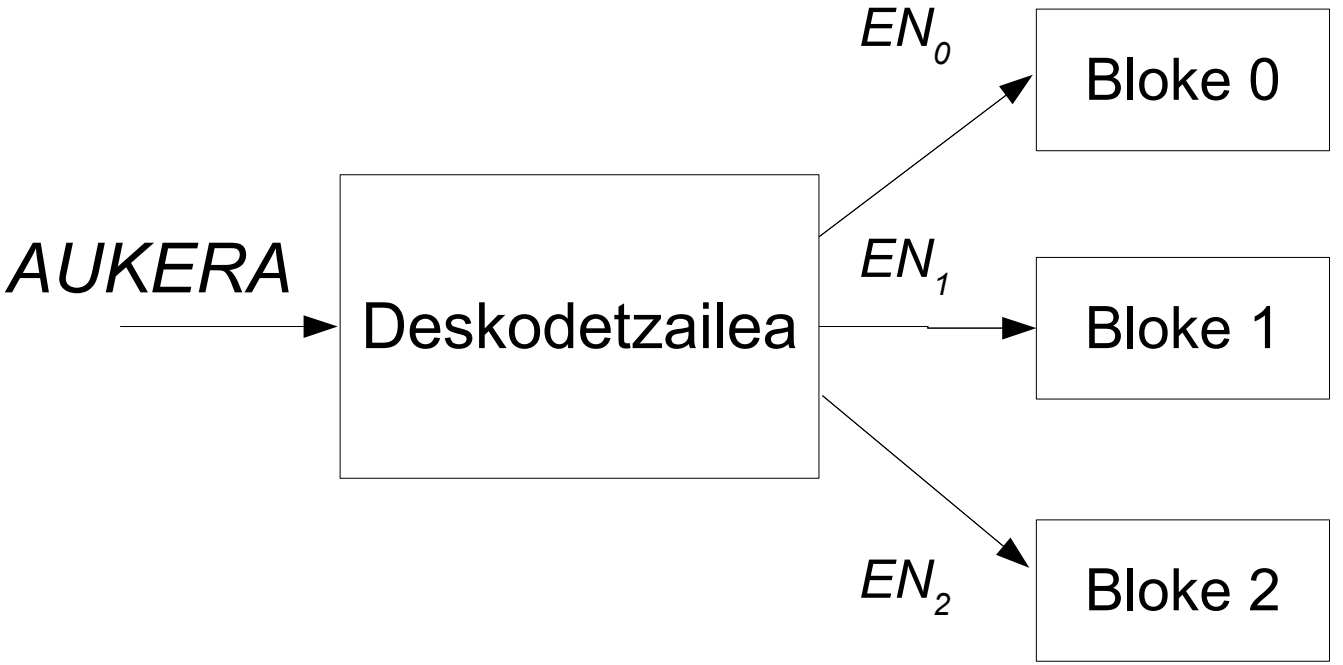
4 biteko batutzaile-kentzaile VHDLren bidez

- Hauxe da 4 biteko batutzaile-kentzaile baten adierazpena VHDLn

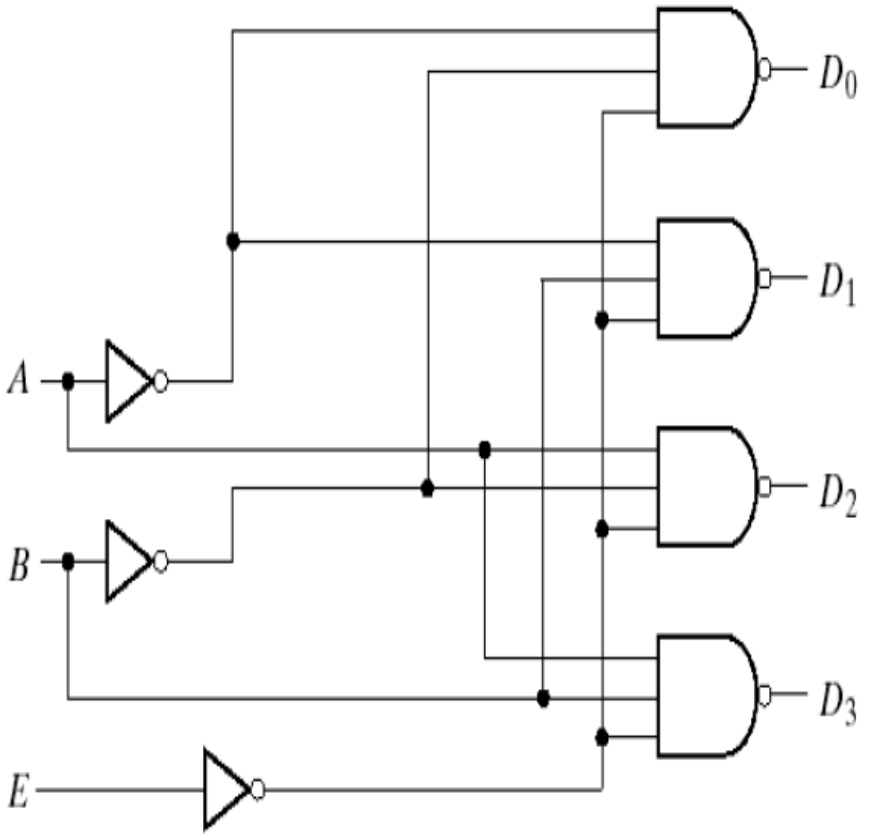
```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
USE ieee.std_logic_unsigned.all ;

ENTITY restsumIS
PORT (     M            : IN    STD_LOGIC ;
           A, B         :IN    STD_LOGIC_VECTOR(3 DOWNTO 0) ;
           F            : OUT       STD_LOGIC_VECTOR(3 DOWNTO 0) ) ;
END restsum ;

ARCHITECTURE a OF restsum IS
BEGIN
PROCESS ( M, A, B )
          BEGIN
                CASE M IS
                WHEN '0'                =>F <= A+B ;
                WHEN OTHERS         =>F <= A-B;
          END CASE ;
END PROCESS ;
END a ;
```

Deskodetzailea

- Deskodetzailearen funtzioetatik bat aktibatzen da, bere sarreran sartzen denean funtzio hori dagokien zenbakia (kodea)
- Bloke honetan 2^n irteerak dauzkagu, n sarrerako aldagaien kopurua izanik eta kode bitar natural erabiltzen dugunean



Gaikuntza sarrerako 2 biteko deskodetzailea

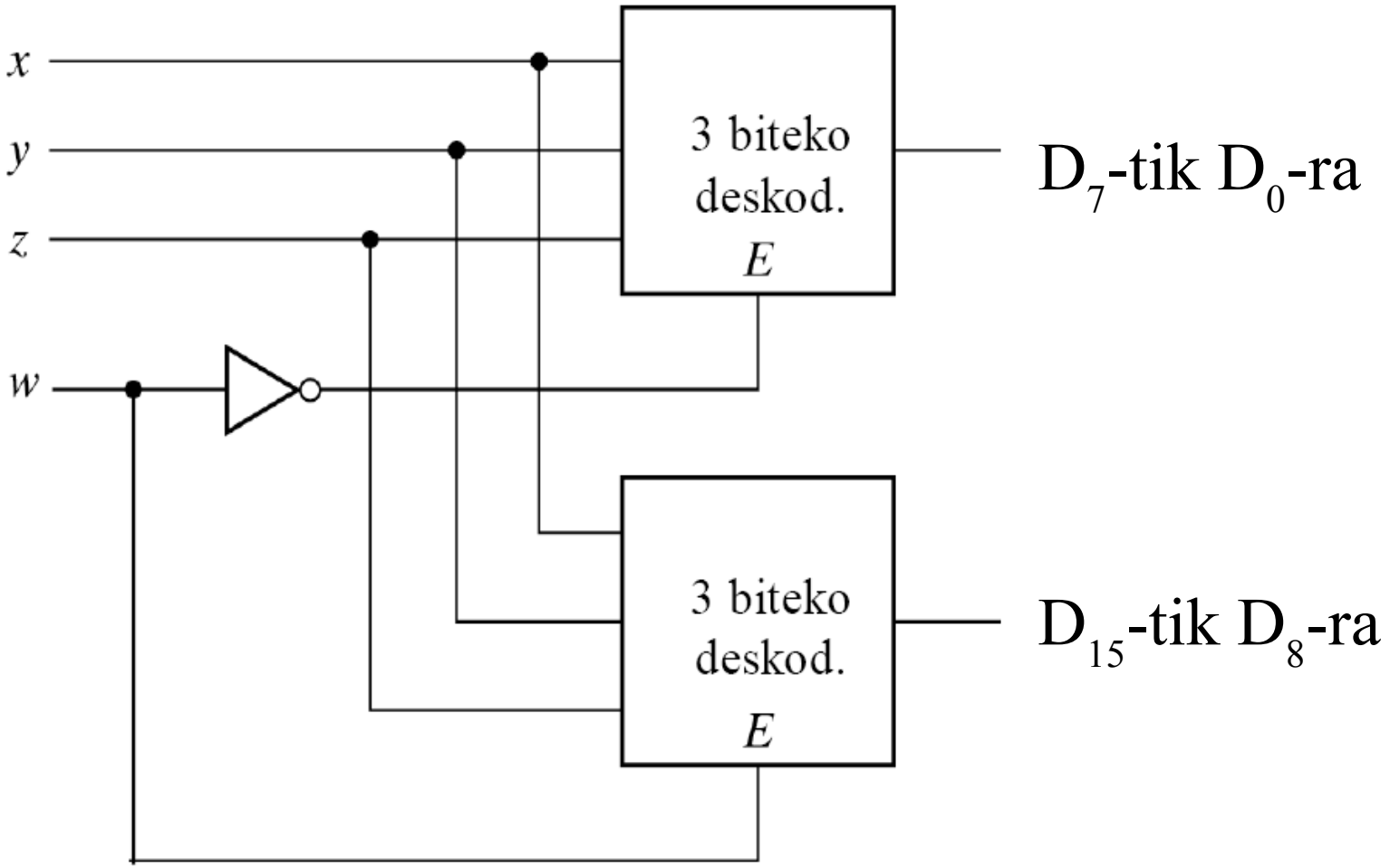


E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

$$D_0=(E\cdot A'\cdot B')'$$
$$D_1=(E\cdot A'\cdot B)'$$
$$D_2=(E\cdot A\cdot B')'$$
$$D_3=(E\cdot A\cdot B)'$$

Irteera hauek maila baxukoak dira, horregatik funtzioak ezeztatuak dira

4 biteko deskodetzailearen inplementazioa, 3 biteko bi deskodetzailearen bidez



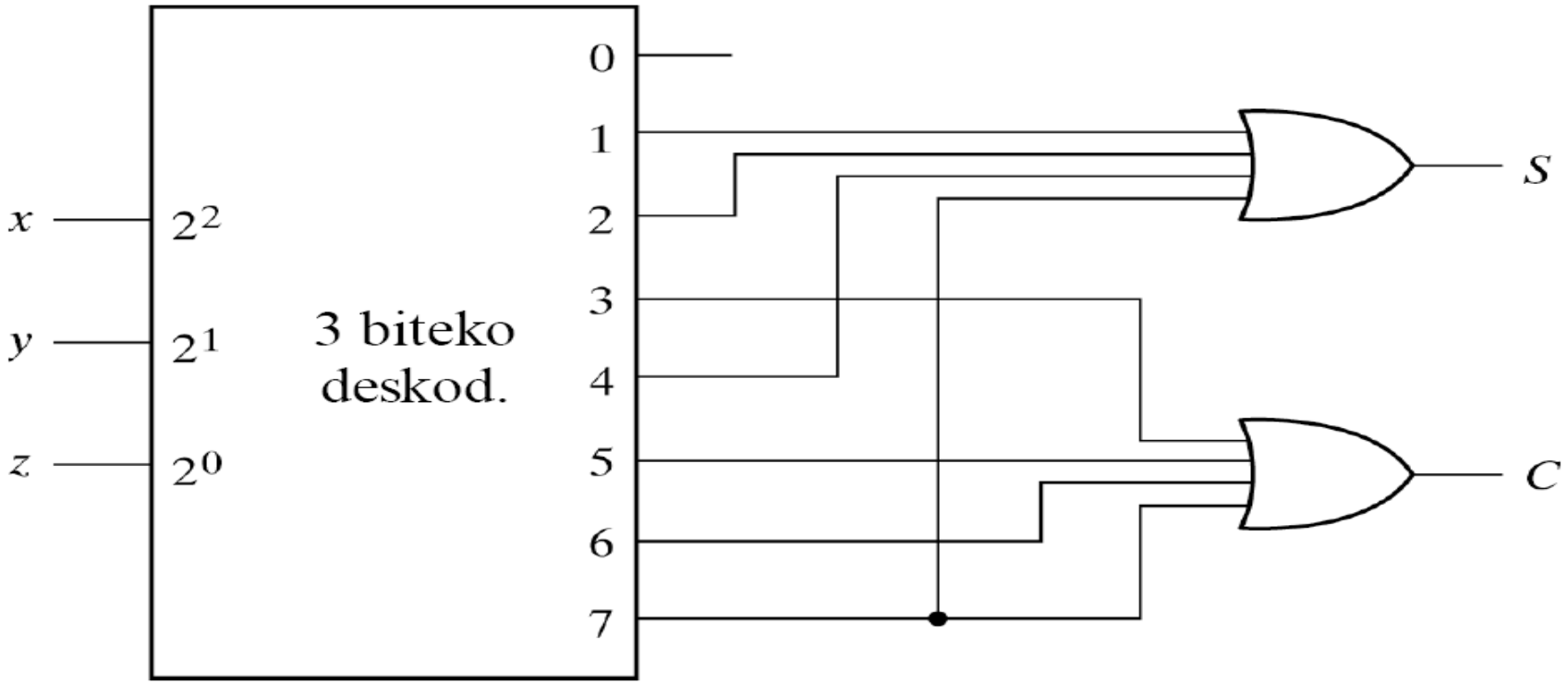
Metodo hau bloke guztietarako erabili daiteke (diseinu hierarkia)

4 biteko deskodetzailea VHDLren bidez

- Hauxe da 4 biteko deskodetzailea baten adierazpena VHDLn

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY dec2to4 IS
PORT (   w    : IN       STD_LOGIC_VECTOR(1 DOWNTO 0) ;
         y    : OUT      STD_LOGIC_VECTOR(0 TO 3) ) ;
END dec2to4 ;

ARCHITECTURE a OF dec2to4 IS
BEGIN
WITH w SELECT
y <=     "1000" WHEN "00",
         "0100" WHEN "01",
         "0010" WHEN "10",
         "0001" WHEN "11",
         "0000" WHEN OTHERS ;
END a ;
```

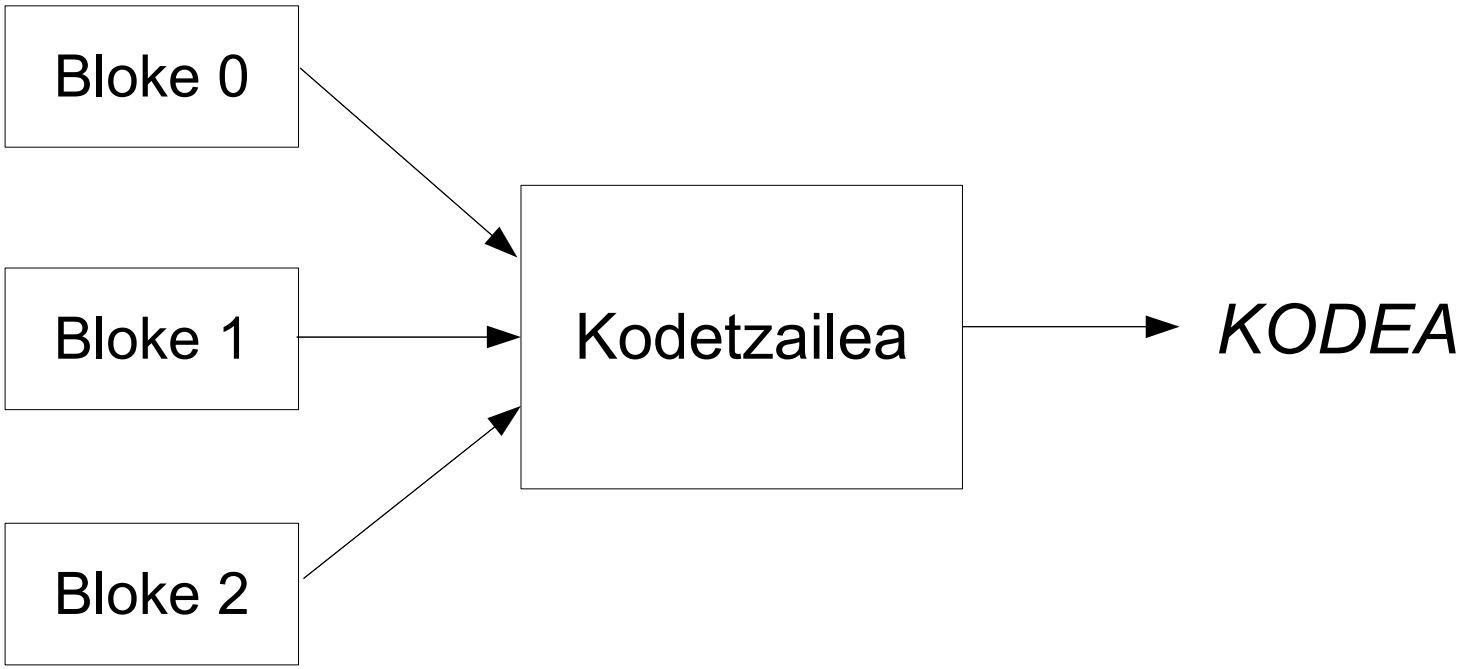
Funtzioak inplementatzeko deskodetzailea: erabateko batutzailea



Deskodetzailearen irteerak mintermak direnez, adierazpen kanoniko bat irudikatzeko erabili dezakegu, funtzioaren mintermei dagozkien irteerak batuz, OR ate baten bidez

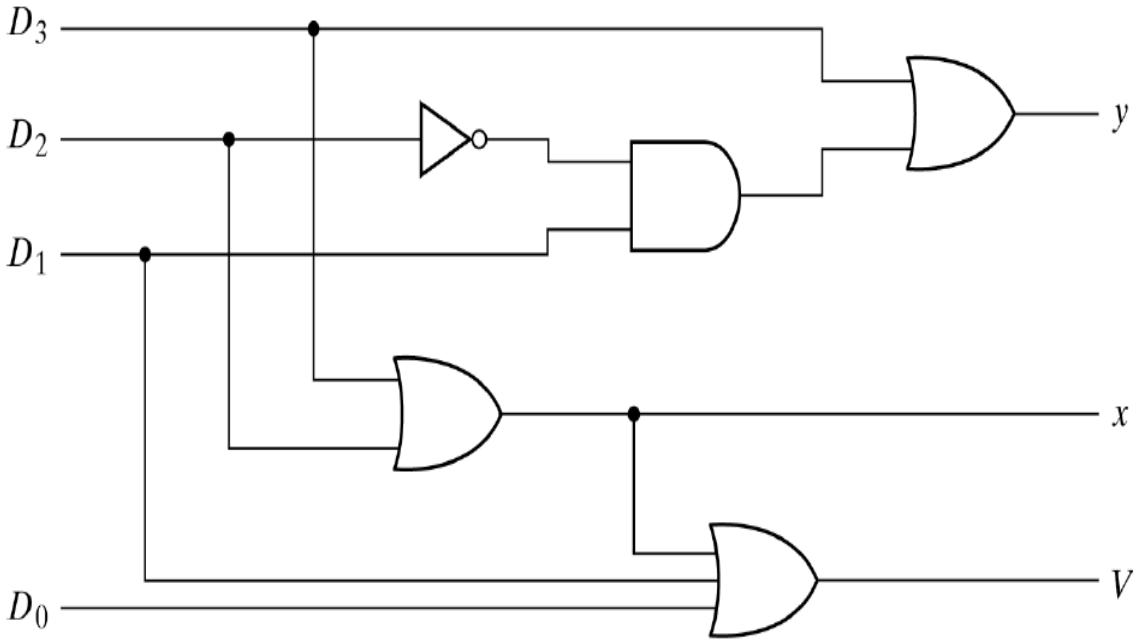
Kodetzailea

- Kodetzailearen irteeran, aktibatutako sarrerari dagokion zenbakia (kodea) agertzen da
- Kode bitar naturala bada, 2^n da sarreraren kopurua, n bada irteerako kopurua



Lehentasunezko kodetzailea

D_0	D_1	D_2	D_3	X	Y	V
0	0	0	0	0	0	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1



- Sarrera bat baino gehiago aktibatzen bada, irteera zein den erabakitzeko, sarrearen lehentasuna aplikatzen da
- Normalean, zenbaki altueneko sarrera lehentasun handienekoa da
- V irteera gehigarrien bidez, sarrera aktibatu bat badagoela adierazten da

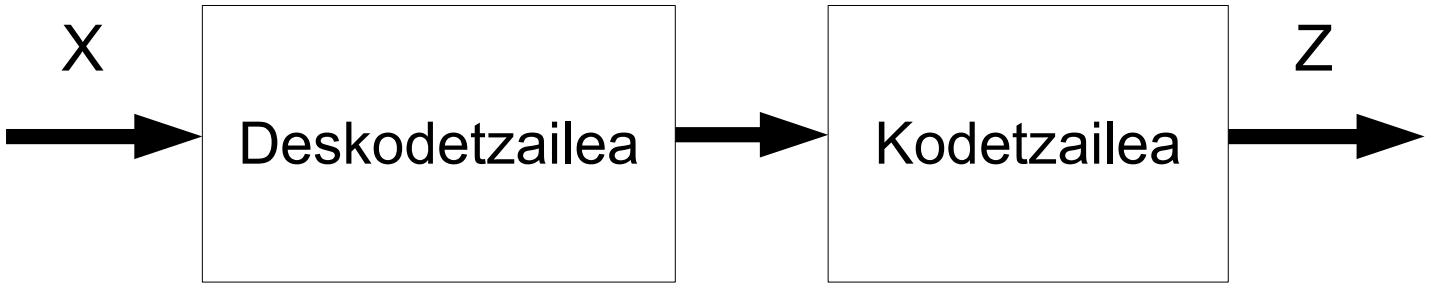
4 biteko kodetzailea VHDLren bidez

- Hauxe da 4 biteko lehentasunezko kodetzaile baten adierazpena VHDLn:

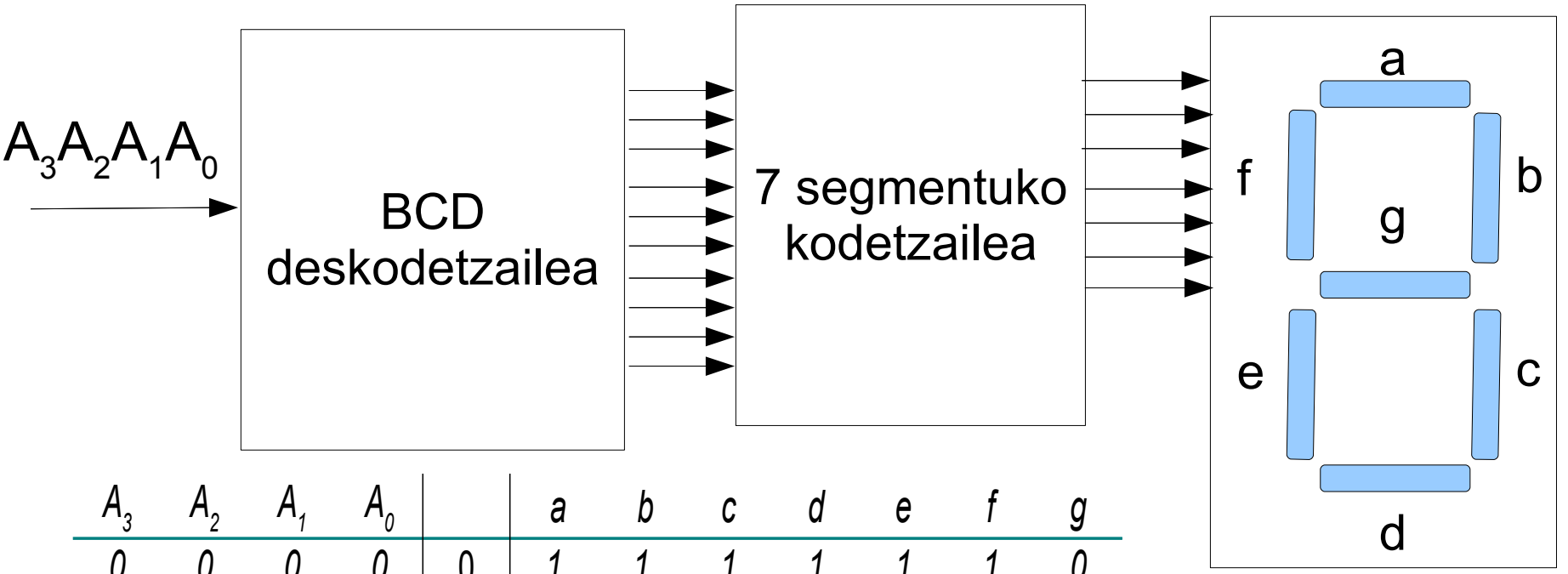
```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY cod4to2 IS
PORT (   w    : IN      STD_LOGIC_VECTOR(3 DOWNTO 0) ;
         y    : OUT     STD_LOGIC_VECTOR(1 DOWNTO 0) ;
         v    : OUT     STD_LOGIC ) ;
END cod4to2 ;

ARCHITECTURE a OF cod4to2 IS
BEGIN
y  <= "11"       WHEN w(3) = '1' ELSE
      "10"       WHEN w(2) = '1' ELSE
      "01"       WHEN w(1) = '1' ELSE
      "00" ;
v <=   '0'       WHEN w = "0000" ELSE '1' ;
END a;
```

Kode bihurgailua

- Deskodetzaile+kodetzailearen aplikazio bat da kode bihurgailua
- Zenbaki bat sartzen da deskodetzailean eta hori dagokion irteera aktibatzen da
- Seinale aktibatua beste kode bateko kodetzailearen sarrera da, bere irteeran kode berrian kodifikatutako datua sortuz



7 segmentuko display-a



7 segmentuko display-a

A_3	A_2	A_1	A_0		a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1