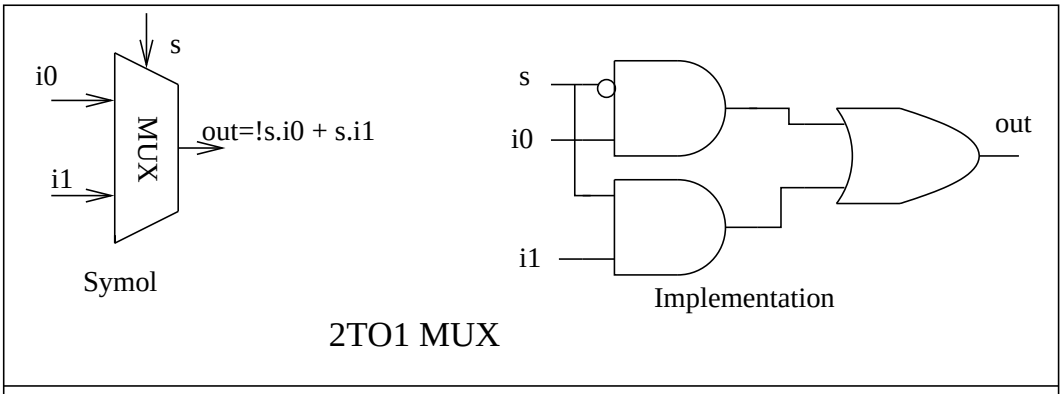
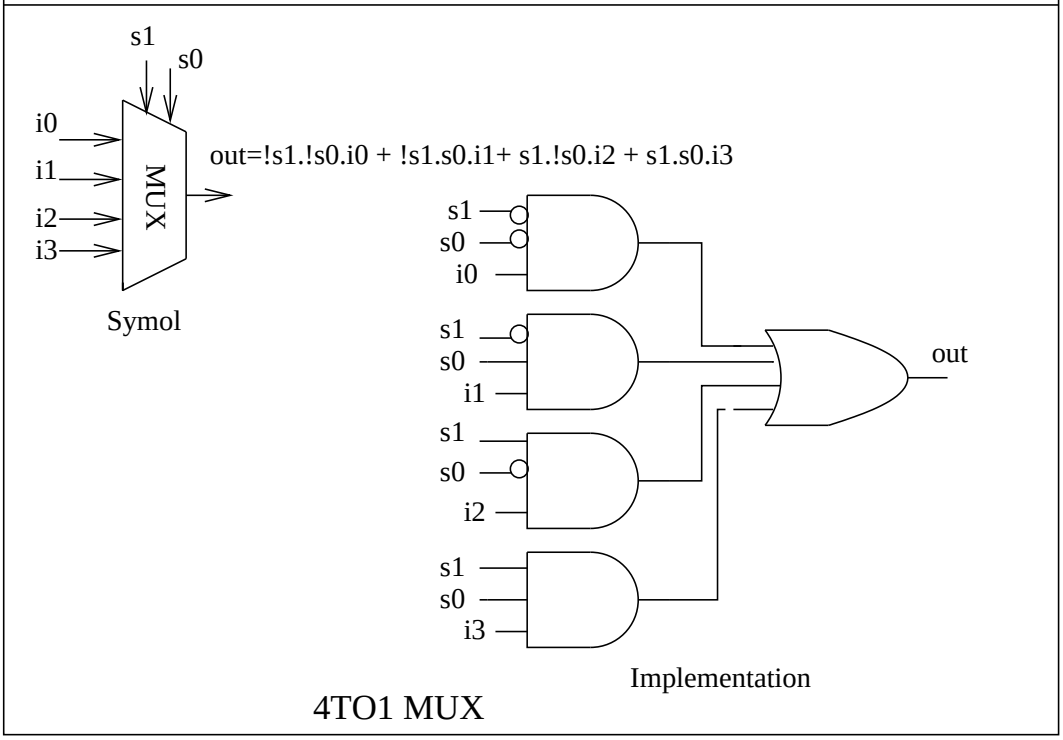
## Combinational Circuits 1-Multiplexers

Multiple inputs and only one selected by a select signal. The output = selected input

- **2 to 1 MUX and 4 to 1 MUX**



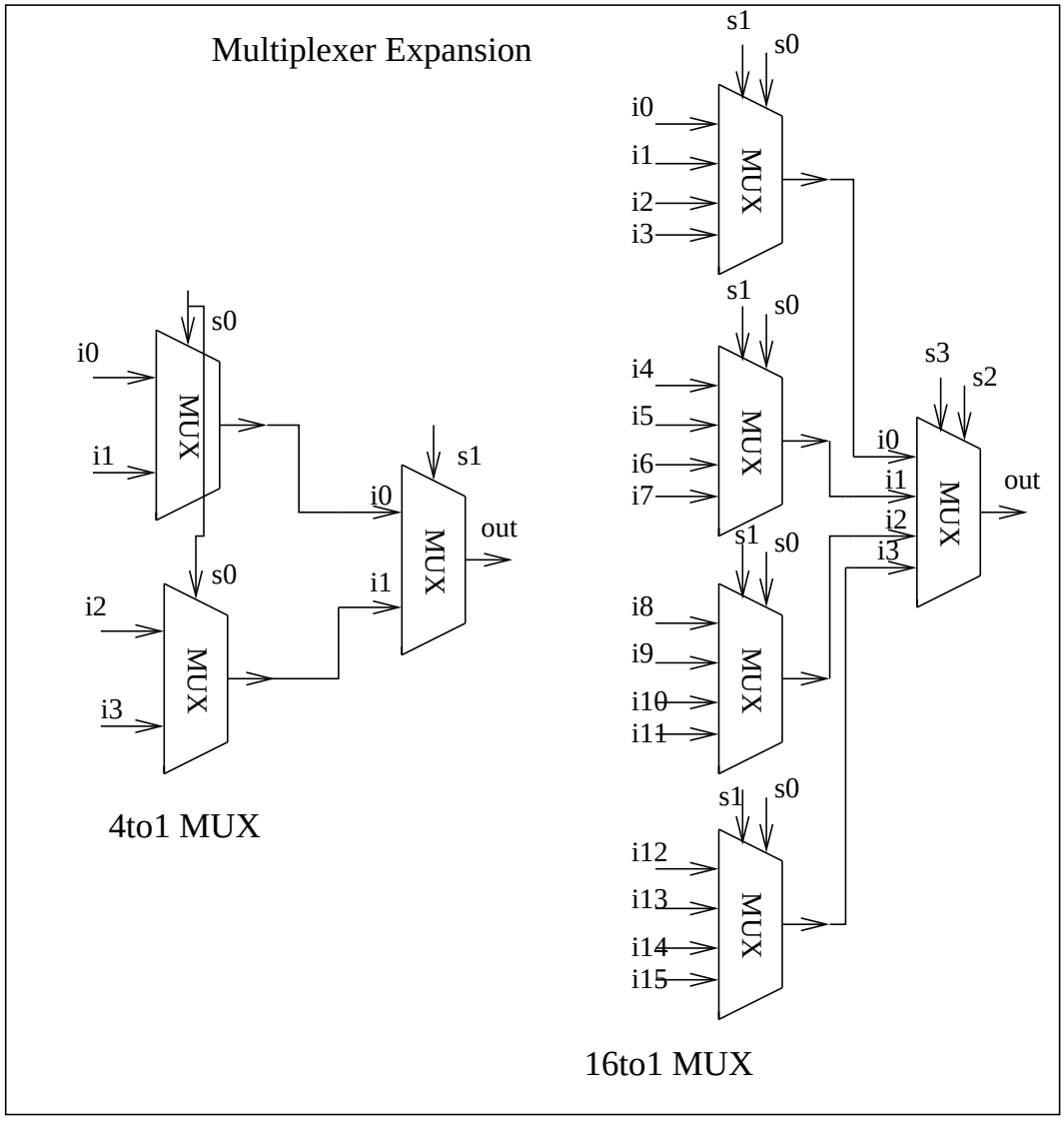

2TO1 MUX



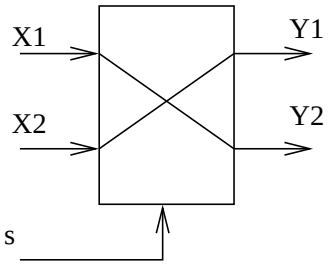

4TO1 MUX

# Multiplexer Expansion

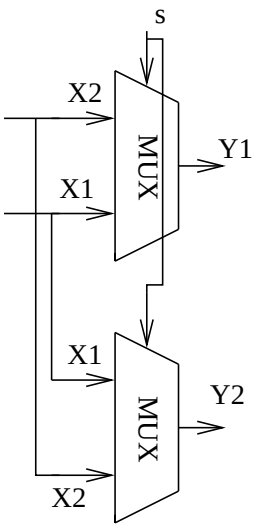
- use 2 to 1 to build 4 to 1
  use 4 to 1 to build 16 to 1 MUX



# Multiplexer Applications Crossbar switch




If s=0 then Y1=X2, Y2=X1
If s=1 then Y1=X1, Y2=X2



**Synthesis of logic functions**

1-Example: $f = \sum 2, 3$

**Shanon's Expansion**

$f(w1, w2, ..wn) = !w1.f(0, w2, ..wn) + w1.f(1, w2, ..wn)$

Example: $f = !w1.w2.w3 + w1.!w2.w3 + w1.w2.!w3 + w1.w2.w3$

$f = !w3(w1.w2) + w3.(!w1.w2 + w1.!w2 + w1.w2)$

Example: using 2 to 1 MUX to implement

$f = !w1.!w3 + w1.w2 + w1.w3$

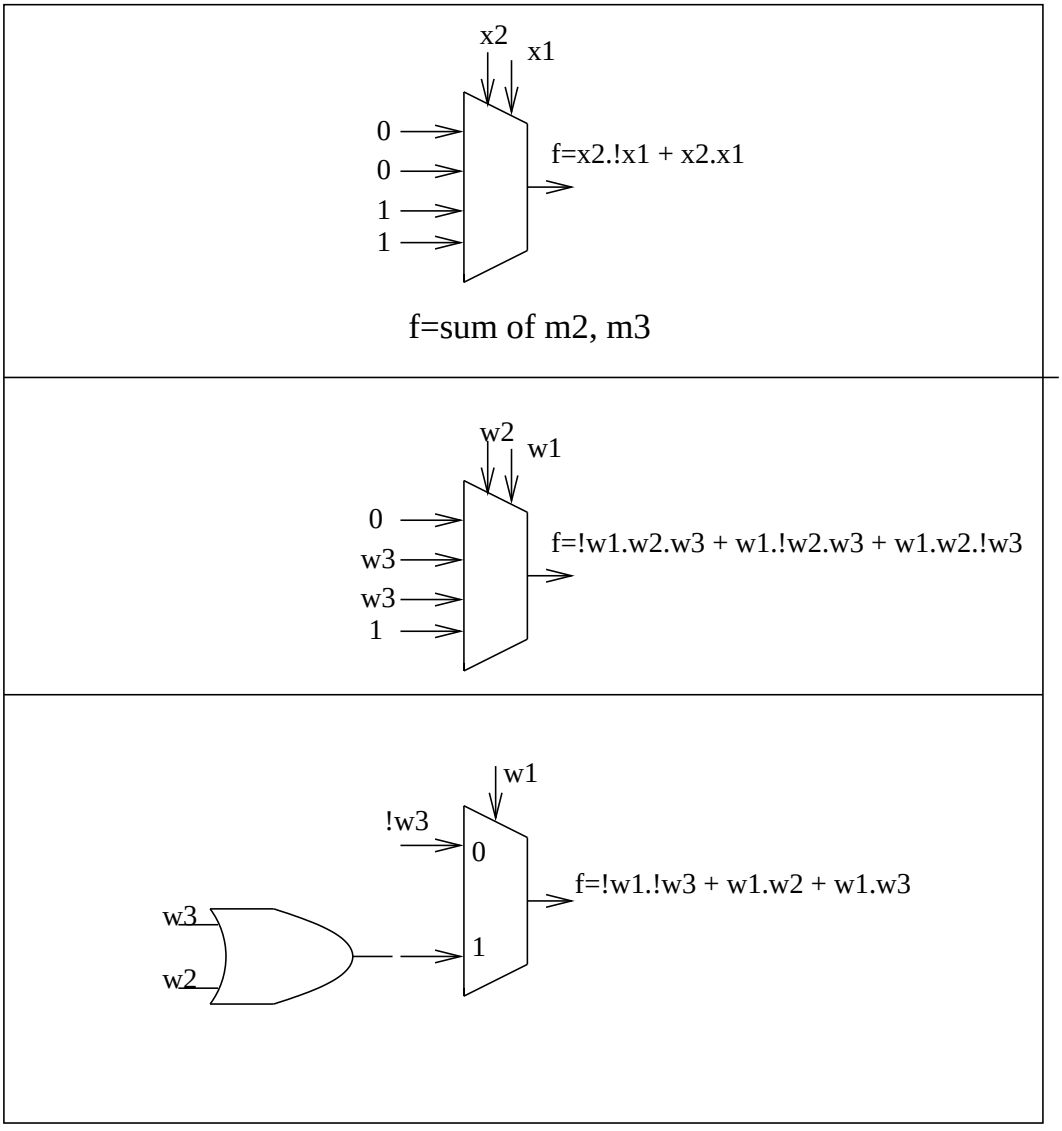
$f = !w3.!w2.f(0,0) + !w3.w2.f(0,1) + w3.!w2.f(1,0) + w3.w2.f(1,1)$

$f = !w3.!w2.(!w1) + !w3.w2.(!w1 + w1) + w3.!w2.(w1) + w3.w2.(w1 + w1)$

$f = !w1.(!w3.!w2 + !w3.w2) + w1.(!w3.w2 + w3.!w2 + w3.w2)$

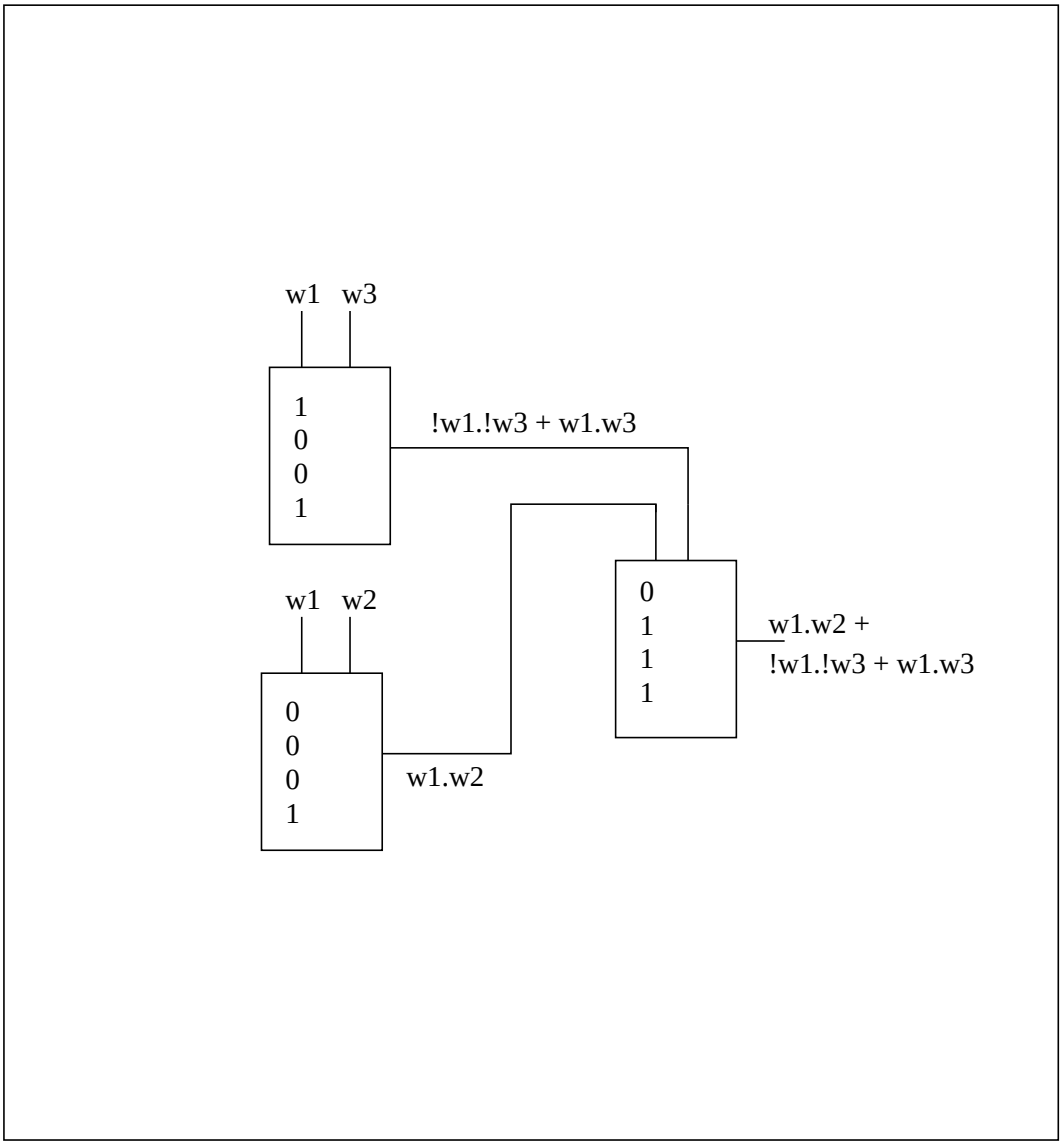
$f = !w1.!w3 + w1.(w3 + w2)$

# Shanon's expansion examples



# Function Implementation using LUT

Example: $f = !w1.!w3 + w1.w3 + w1.w2$

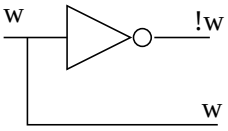


## Decoders

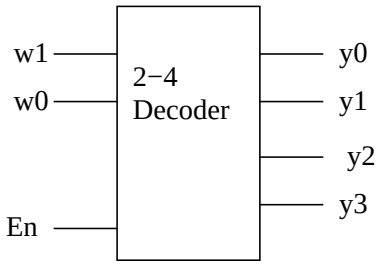
n inputs generate $2^n$ outputs, with only one = true based on value of n

For 2 inputs , we have 4 outputs

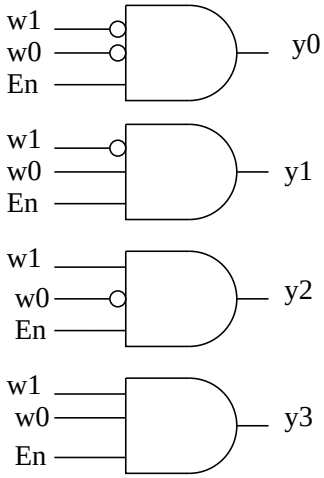
if input = 10, the output y2=1




1−2 decoder



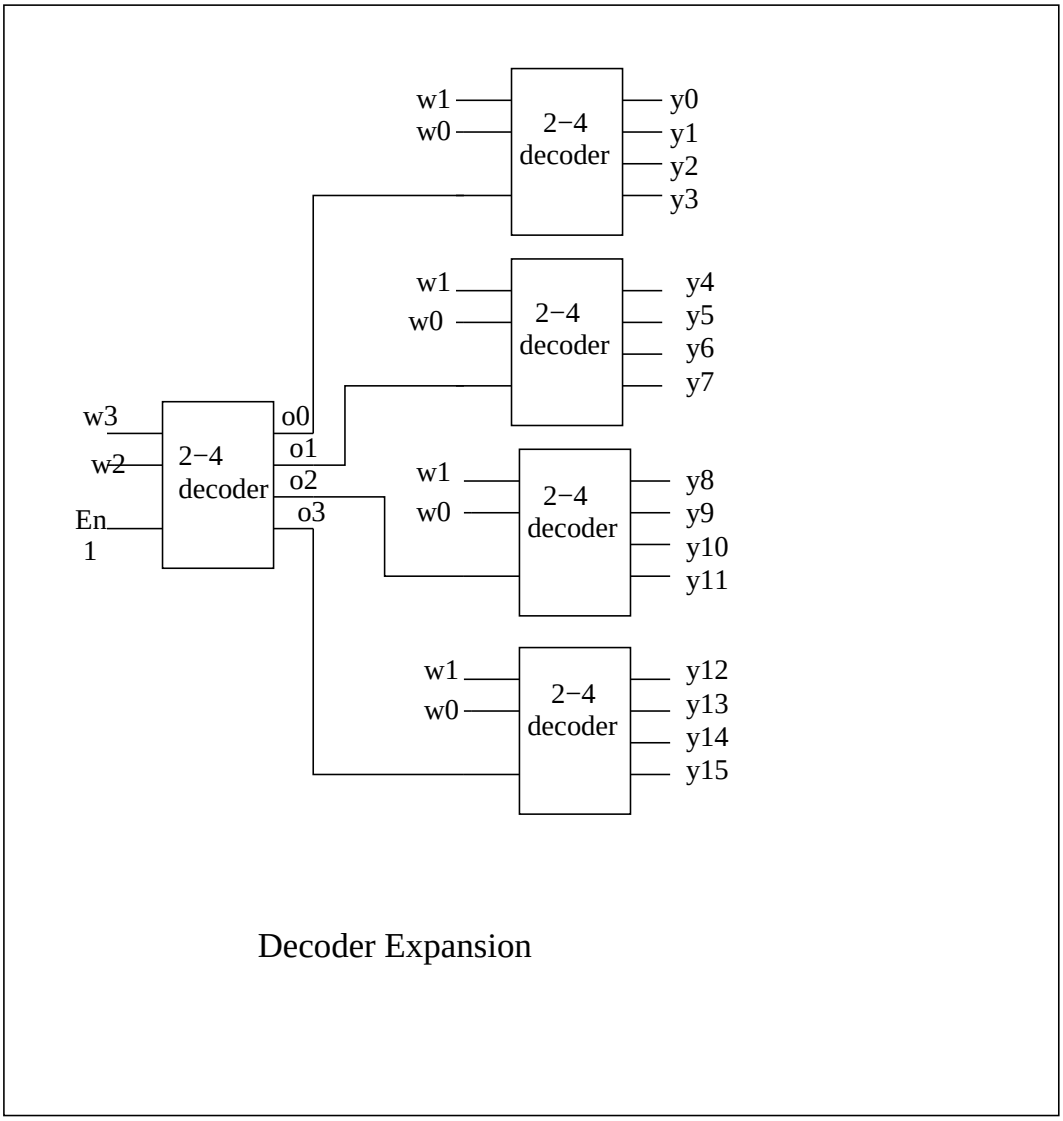

Symbol




Implementation

# Decoder Expansion

Design 16 output decoder using 4 output decoders



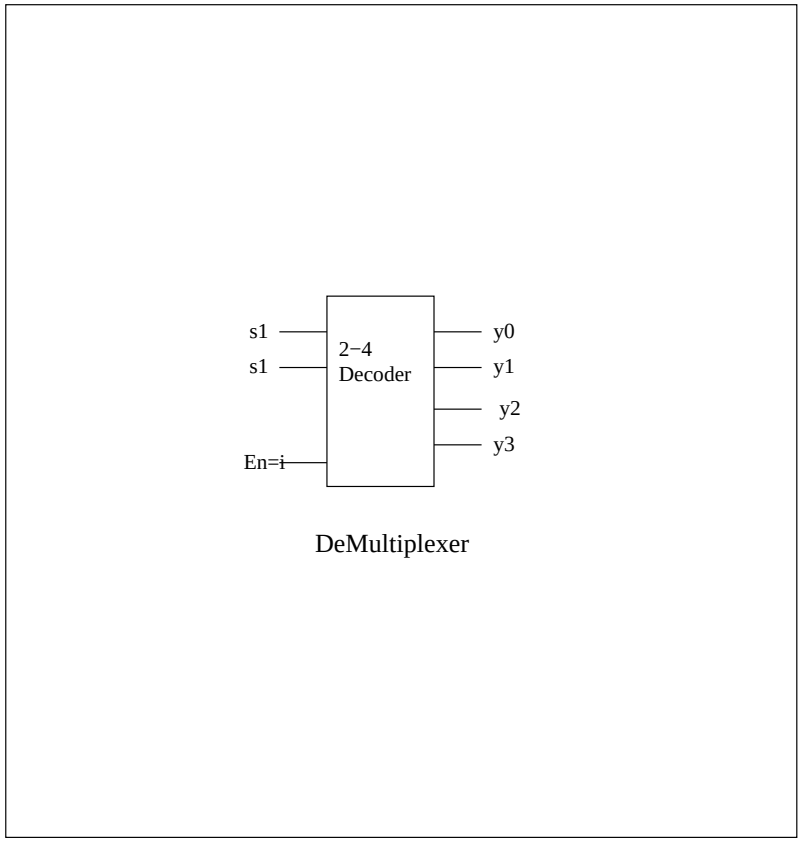

Decoder Expansion

## DeMultiplexer

Multiple outputs from single input selected by select line.

Could use a decoder, with En=input

Example: if i =1, s1 s0=10, then y2=1



## Encoders

from n inputs, one is active and the output is the code for the specific input

Example: Binary encoder

4 inputs w3, w2, w1, w0

output is y1, y0

If w3w2w1w0=1000, then y1y0=11

If w3w2w1w0=0100, then y1y0=10

If w3w2w1w0=0010, then y1y0=01

If w3w2w1w0=0001, then y1y0=00

## Priority Encoder

If w3w2w1w0=1xxx, then y1y0=11

If w3w2w1w0=01xx, then y1y0=10

If w3w2w1w0=001x, then y1y0=01

If w3w2w1w0=0001, then y1y0=00

$y0 = w3 + w1.!w2.!w3$

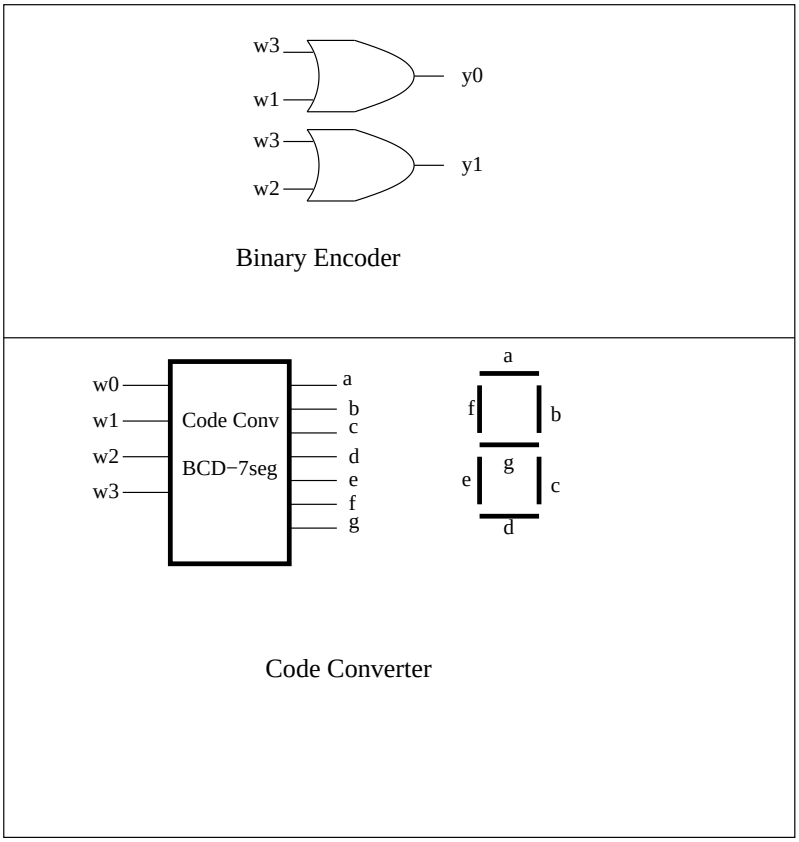
$y2 = w3 + !w3.w2$

## Code Conversion

From BCD to 7 Segment Display

segment a= 0+2+3+5+6+7+8+9

Use a 4 to 16 decoder , then OR gate for each segment



**Arithmetic Comparator**

If A=a3a2a1a0, and B= b3b2b1b0 then:

AeqB = !(a3 XOR b3). !(a2 XOR b2).
!(a1 XOR b1). !(a0 XOR b1)

AgtB=a3.!b3 + !(a3 XOR b3).a2.!b2
+ !(a3 XOR b3). !(a2 XOR b2).a1.!b1
+!(a3 XOR b3).
!(a2 XOR b2).!(a1 XOR b1).a0.!b0

AltB = !(AeqB + AgtB)

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

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

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



ARCHITECTURE Behavior OF dec2to4 IS
  SIGNAL Enw : STD_LOGIC_VECTOR (2 DOWNTO 0);
BEGIN
 Enw <= En & w;
  WITH Enw SELECT
     y<= "1000" WHEN "100",
         "0100" WHEN "101",
         "0010" WHEN "110",
         "0001" WHEN "111"
         "0000" WHEN OTHERS;
END Behavior;
```

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

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

ARCHITECTURE Behavior OF encod IS
BEGIN
  PROCESS(w)
  BEGIN
     IF w(3) = '1' THEN
              y <="11";
     ELSEIF  w(2) = '1' THEN
              y<= "10";
     ELSEIF  w(1) = '1' THEN
              y<= "01";
     ELSE     y<= "00";
     END IF;
     END PROCESS;
   z<= '0' WHEN w = "0000" ELSE "1";
END Behavior;
```

**Ch6**

6.2 Use 3 to 8 decoder and an OR gate to mplement:

$f = \sum m1, m2, m3, m5, m6$

Decoder inputs X1, X2, X3 and outputs y0..y7

$f = y1 + y2 + y3 + y5 + y6$

6.3 use 2 to 1 MUX to implement :

$f =$!$w1$.!$w3 + w2$.!$w3+$!$w1.w2$

w1 w2 w3 f

0 0 0 1

0 0 1 0

0 1 0 1

0 1 1 1

———————-

1 0 0 0

1 0 1 0

1 1 0 1

1 1 1 0

$f =$!$w1$.(!$w2$.!$w3+w2$.!$w3+w2.w3$)$+w1$.($w2$.!$w3$)$

$f =$!$w1$.(!$w3 + w2$) $+ w1$.($w2$.!$w3$)

6.11 use minimum of 2 inputs LUTs to implement:
$f$ =!$w1$.!$w2$+!$w2$.!$w3$ + $w1$.$w2$.$w3$
using Shanon expansion in w2
$f$ =!$w2$.(!$w1$+!$w3$) + $w2$.($w1$.$w3$)
assume g=w1.w3 then !g= !w1+!w3
$f$ = $g$.$w2$+!$g$.!$w2$

6-16 show how to implement :
$f$ = $w2$.!$w3$ + $w1$.$w3$+!$w2$.$w3$
use Shanon
$f$ =!$w3$.($w2$) + $w3$.($w1$+!$w2$) =!$w3$.($w2$)
+ $w3$.($w1$.$w2$+!$w2$)

6-18- given VHDL Code find the circuit
y0=1 if w =00, En=1
y2=1 if w=10 and En=1
This is a 2 to Decoder