A BCD ADDER

24.06.2011

(C)

ex:

X = 0111 7_{10}

y = 0101 5_{10}

+

1100

12 binary

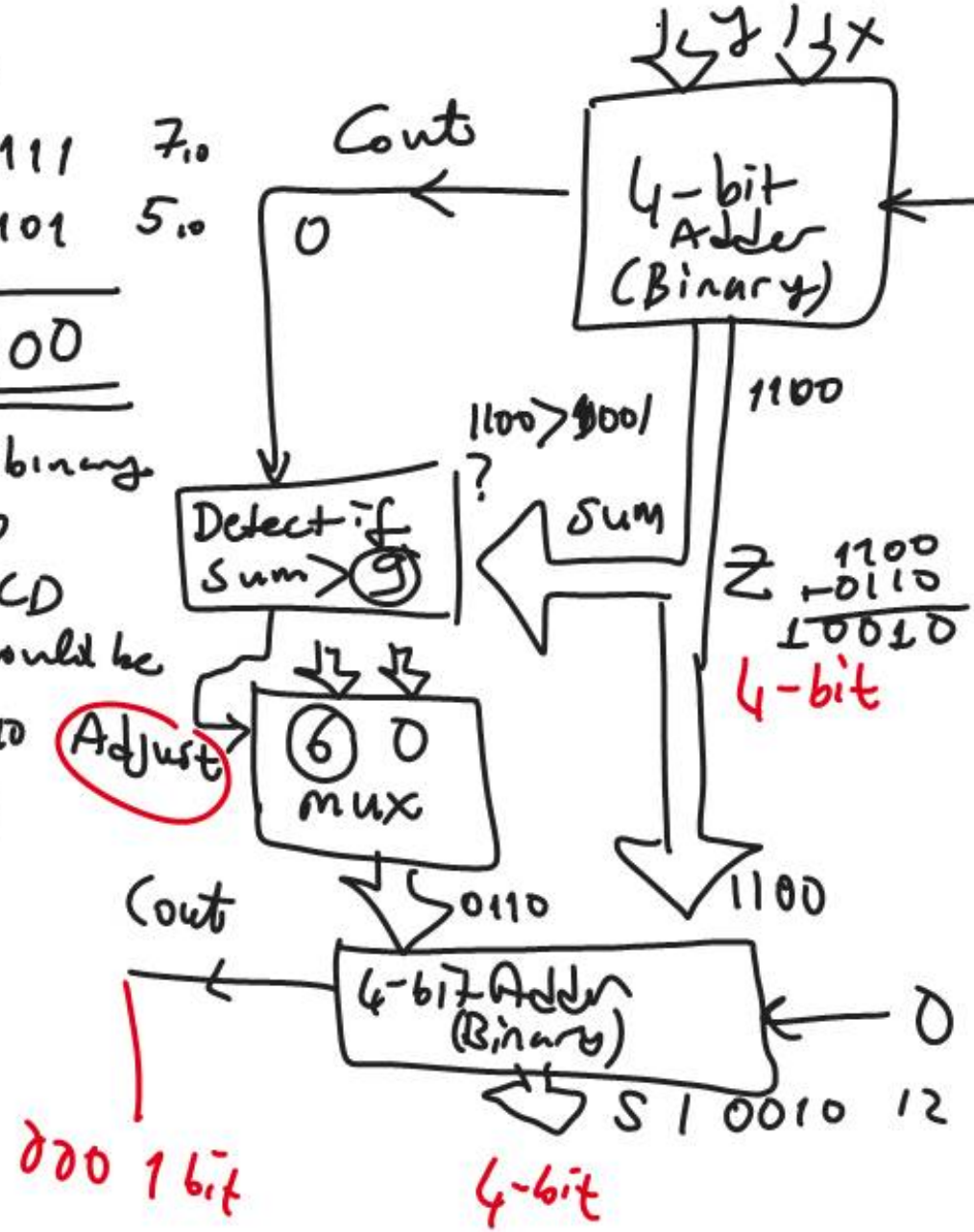
1100

in BCD

It should be

0001 0010

(1 2)



0	0000
1	0001
	0010
	0011
	0100
	0101
	0110
	0111
	1000
9	1001
not defined	1010
	⋮
	1111

0001 0000

1 0

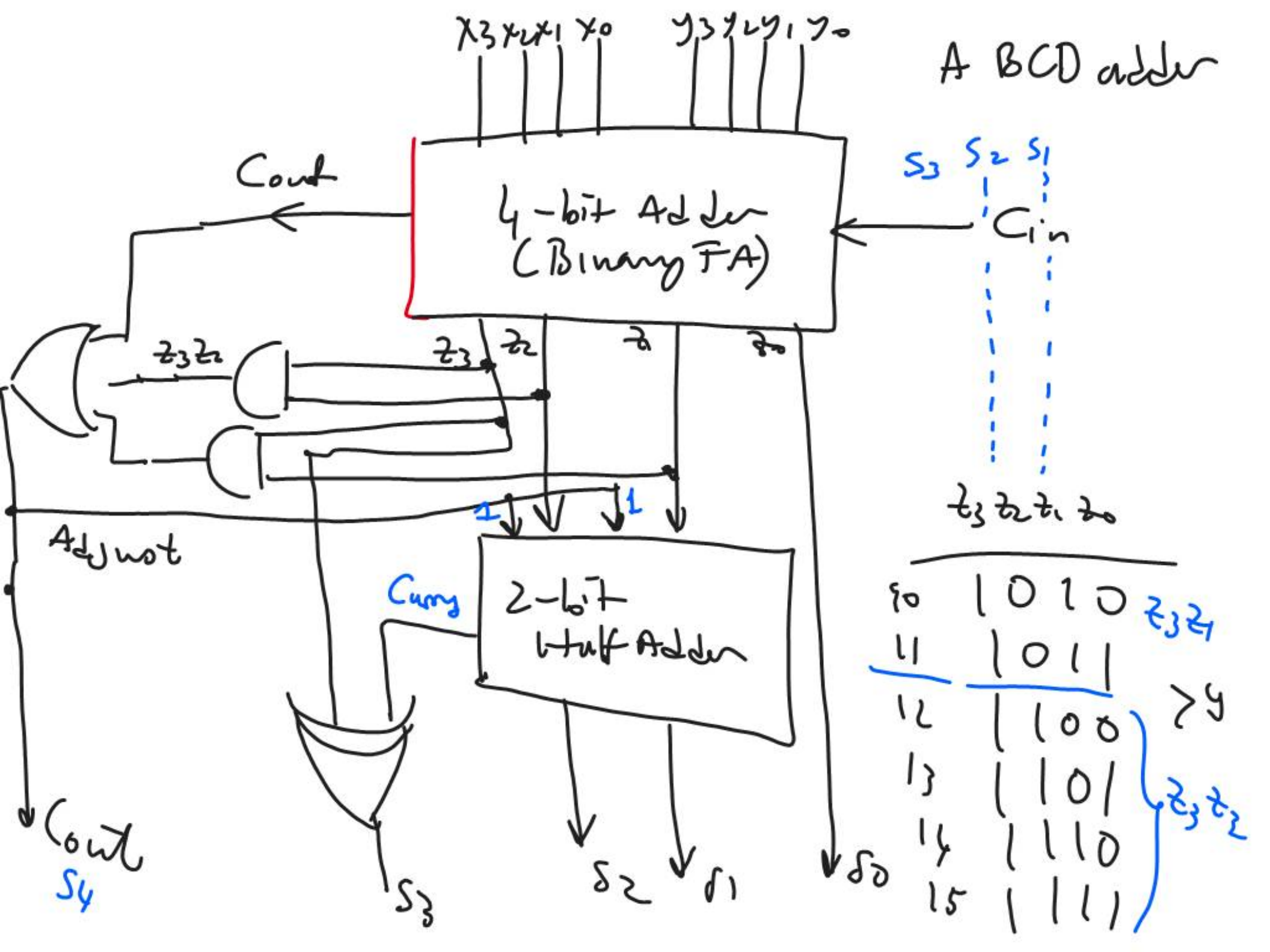
```
LIBRARY ieee;
USE ieee.STD_LOGIC_1164.all;
USE ieee.STD_LOGIC_UNSIGNED.all;
```

(This is arithmetic)

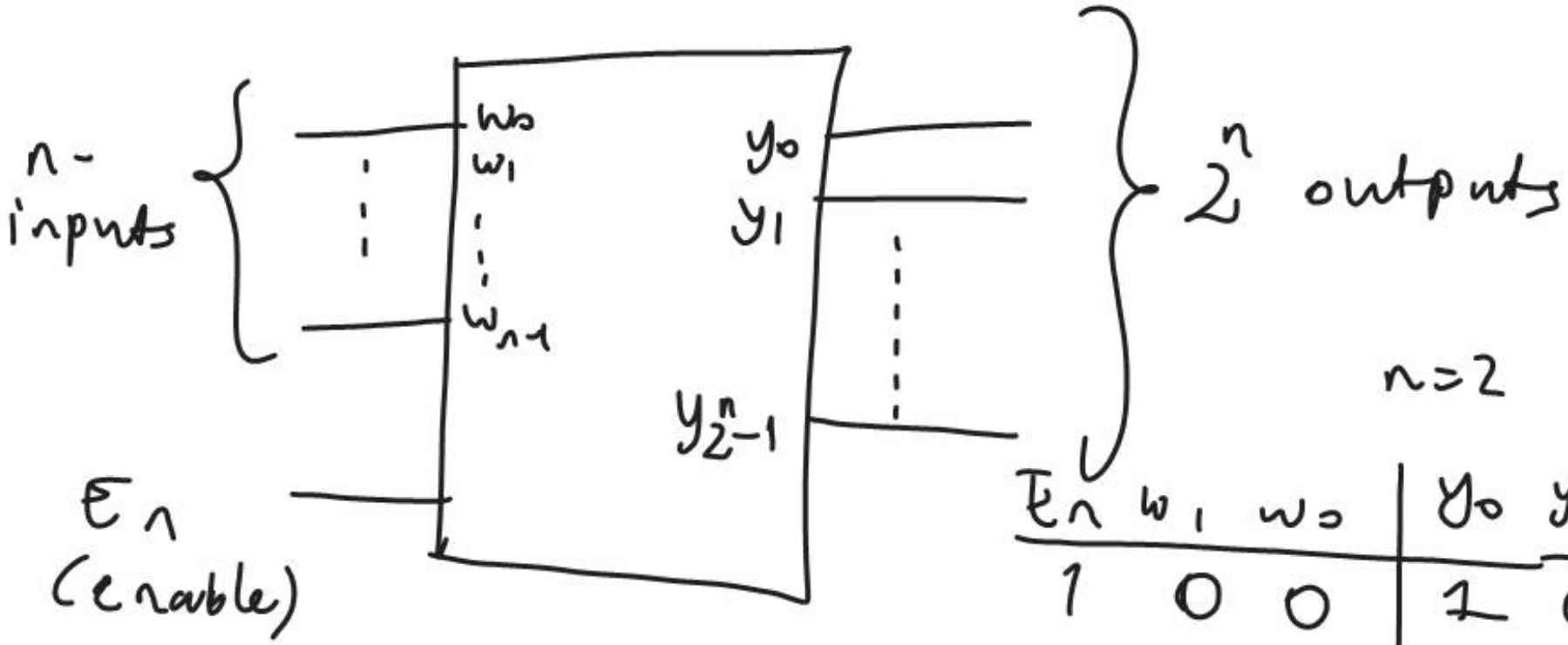
BCD Adder

```
ENTITY BCD IS
PORT(X, Y : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
     S : OUT STD_LOGIC_VECTOR(4 DOWNTO 0));
END BCD;
ARCHITECTURE Behavior OF BCD IS
SIGNAL Z : STD_LOGIC_VECTOR (4 DOWNTO 0);
SIGNAL ADJUST : STD_LOGIC;
BEGIN
Z <= ('0'&X+Y);
ADJUST <= '1' WHEN Z>9 ELSE '0';
S <= Z WHEN (ADJUST='0') ELSE Z+6;
END BEHAVIOR;
```

* Be careful
Z is 5-bit number
→ S is a 5-bit number


A BCD adder
$x_3 x_2 x_1 x_0$
$y_3 y_2 y_1 y_0$
Cout
4-bit Adder
(Binary FA)
S_3 S_2 S_1
Cin
$z_3 z_2$
z_3
z_2
z_1
z_0
1
1
Adjust
Carry
2-bit
Half Adder
Cout
S_4
S_3
S_2
S_1
S_0
$z_3 z_2 z_1 z_0$
10 | 1 0 1 0 $z_3 z_1$
11 | 1 0 1 1
12 | 1 1 0 0 >9
13 | 1 1 0 1
14 | 1 1 1 0
15 | 1 1 1 1
$z_3 z_2$

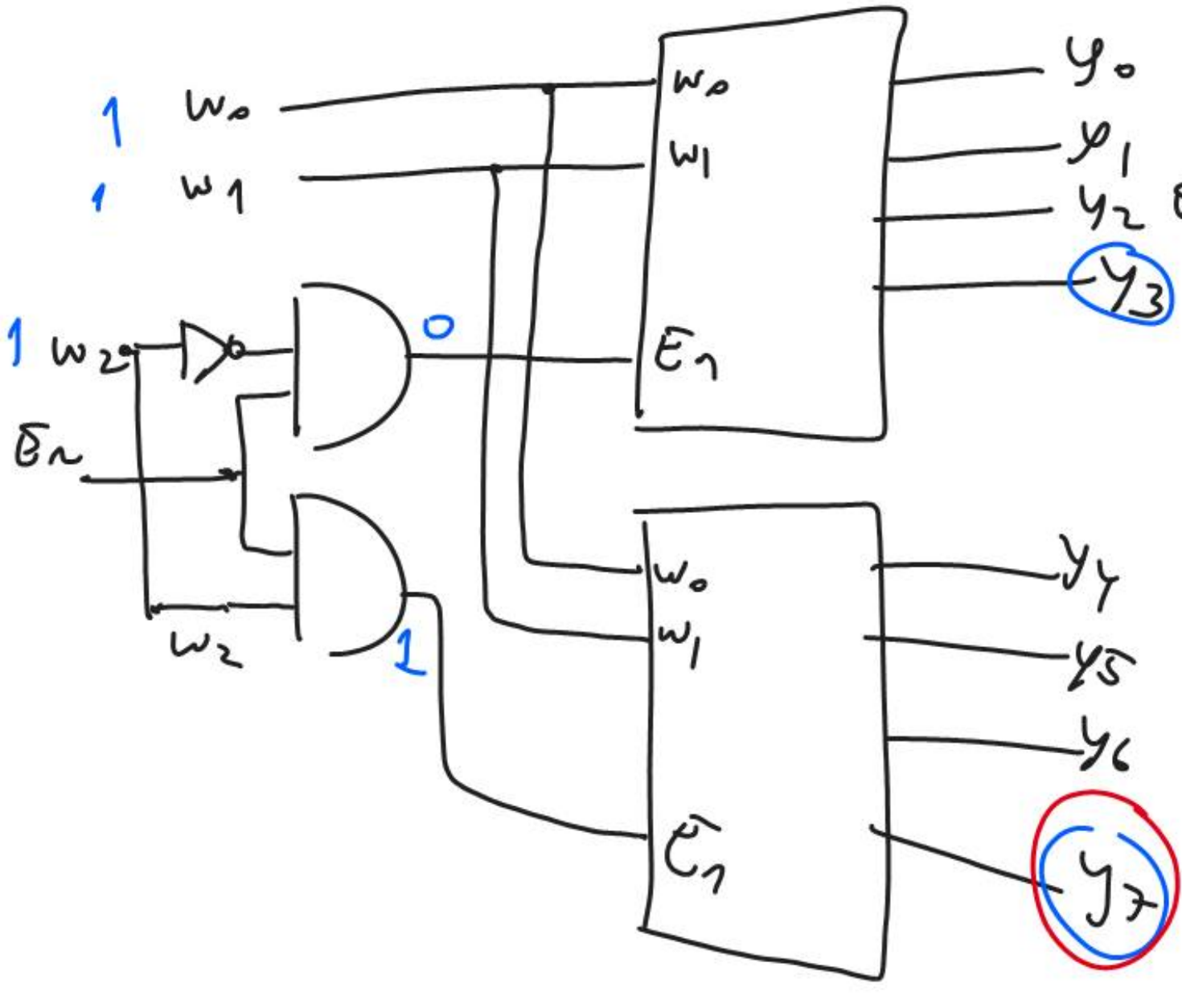
DECODERS



$n=2$

E_n	w_1	w_0	y_0	y_1	y_2	y_3
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
0	X	X	0	0	0	0

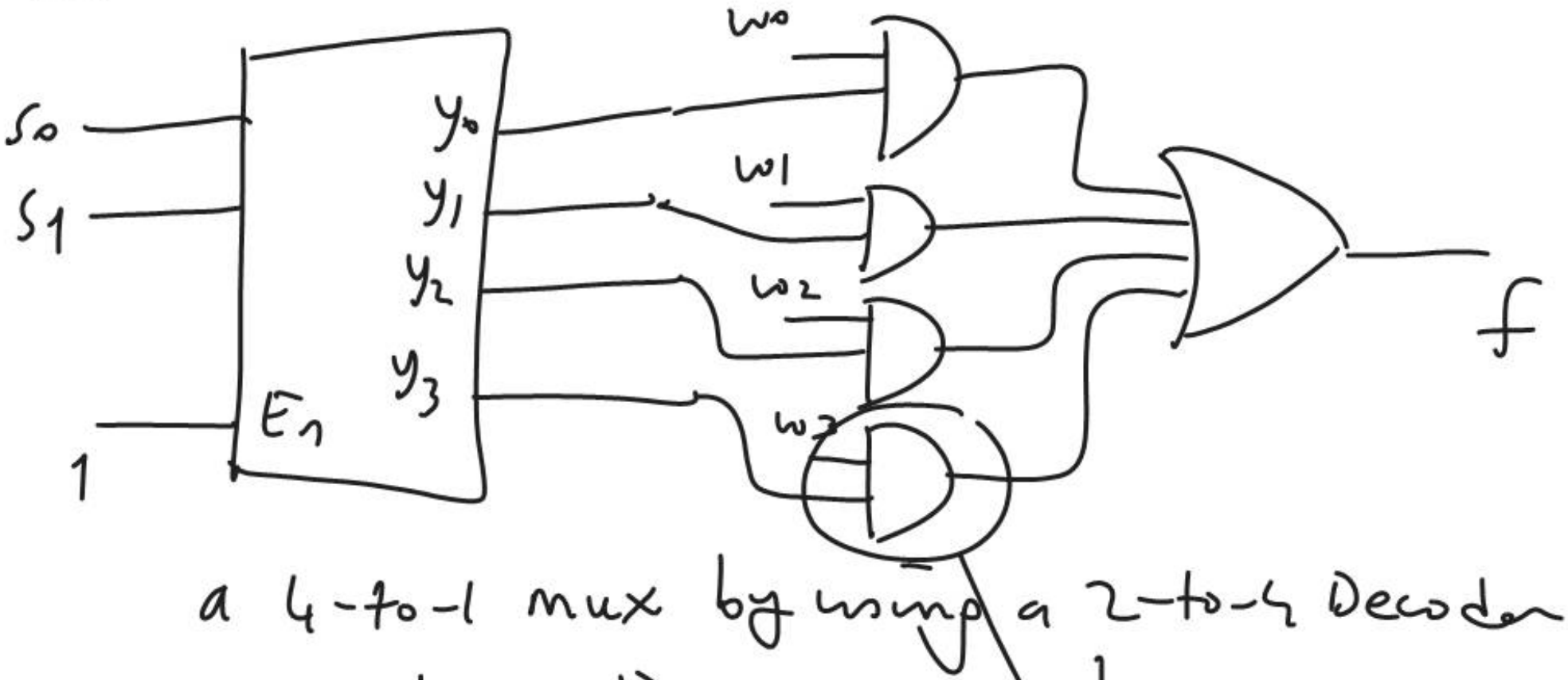
3-to-$(2^3=8)$ DECODER with 2-4 Decoders



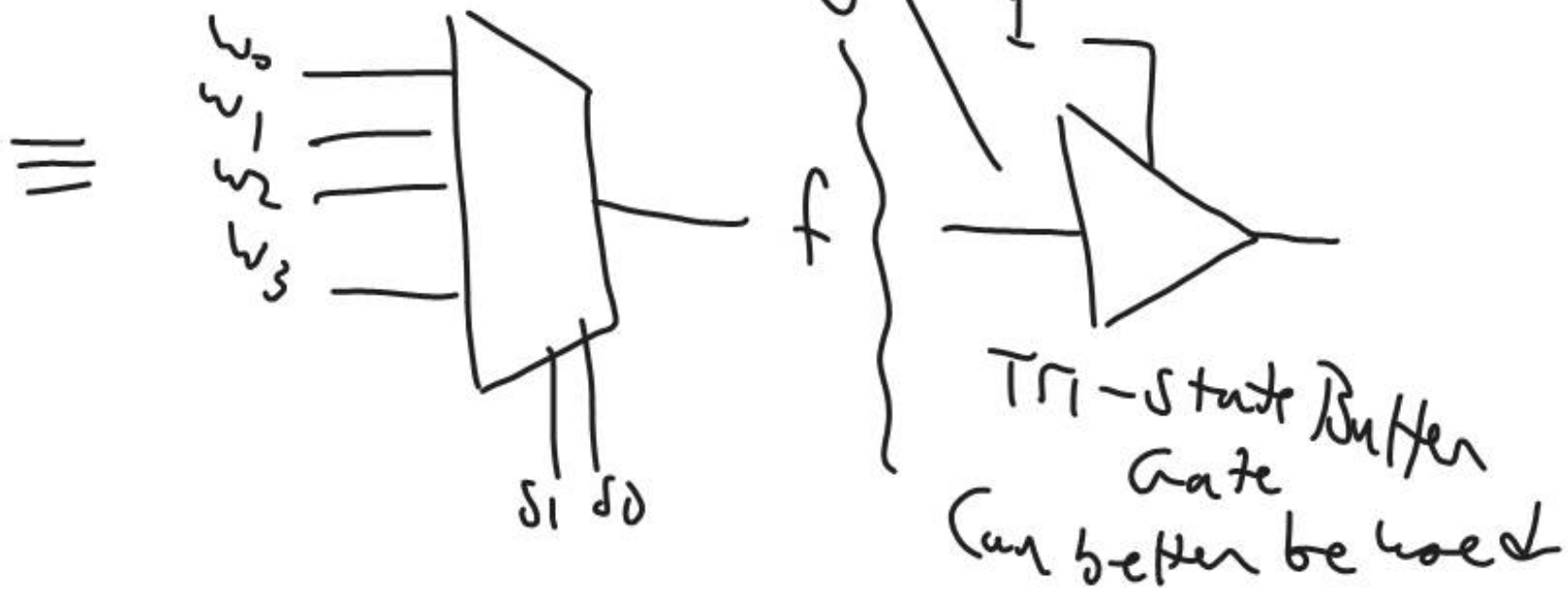
$\bar{E}_n$	w_2	w_1	w_0	y_0 -- -- y_7
1	0	0	0	1 0 0 0 0 -- 0
1	0	0	1	0 1 0 0 -- --
1	0	1	0	0 0 1 0 --
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	-- -- -- 1
0	x	x	x	0 0 0 0 -- 0

irrelevant

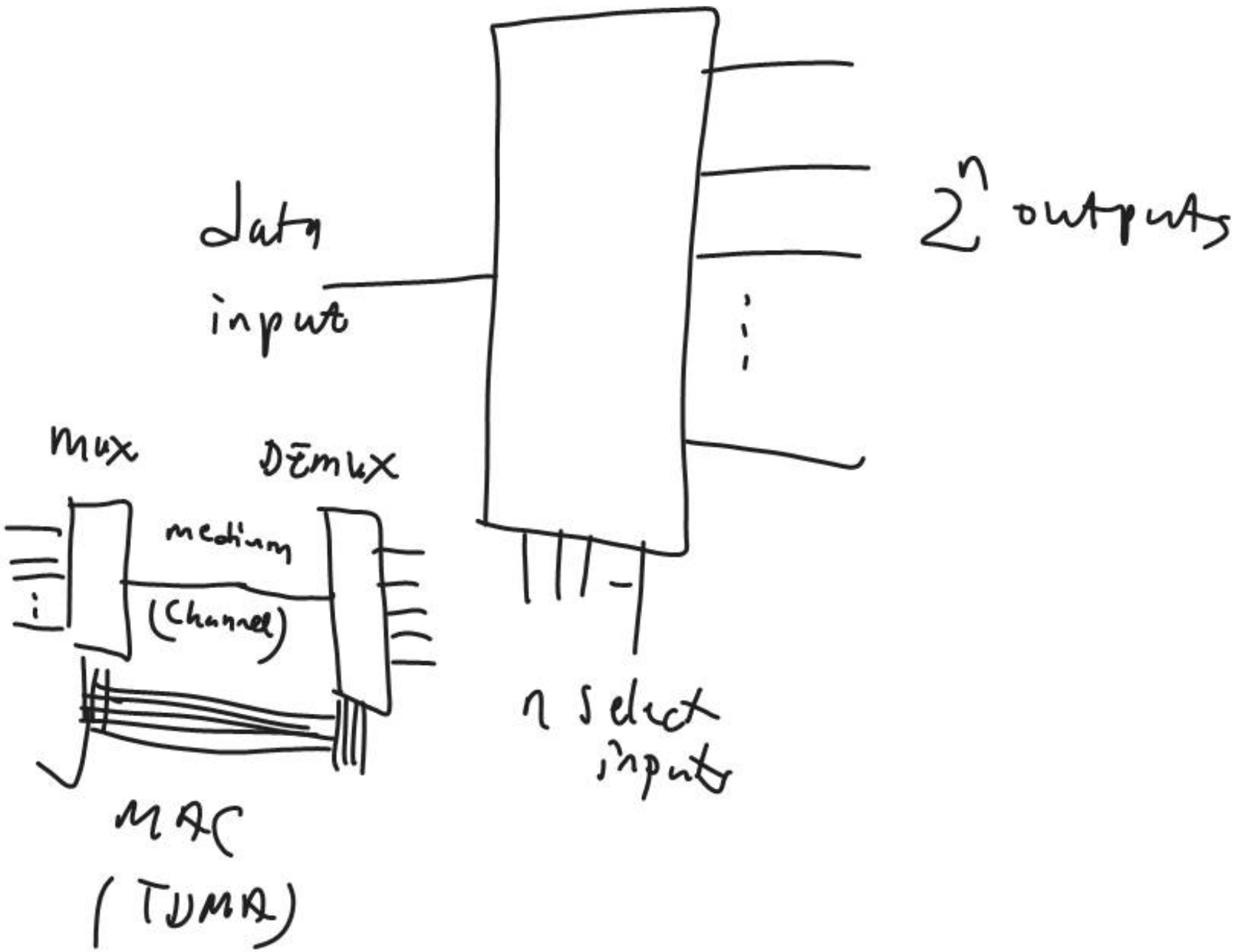
ex



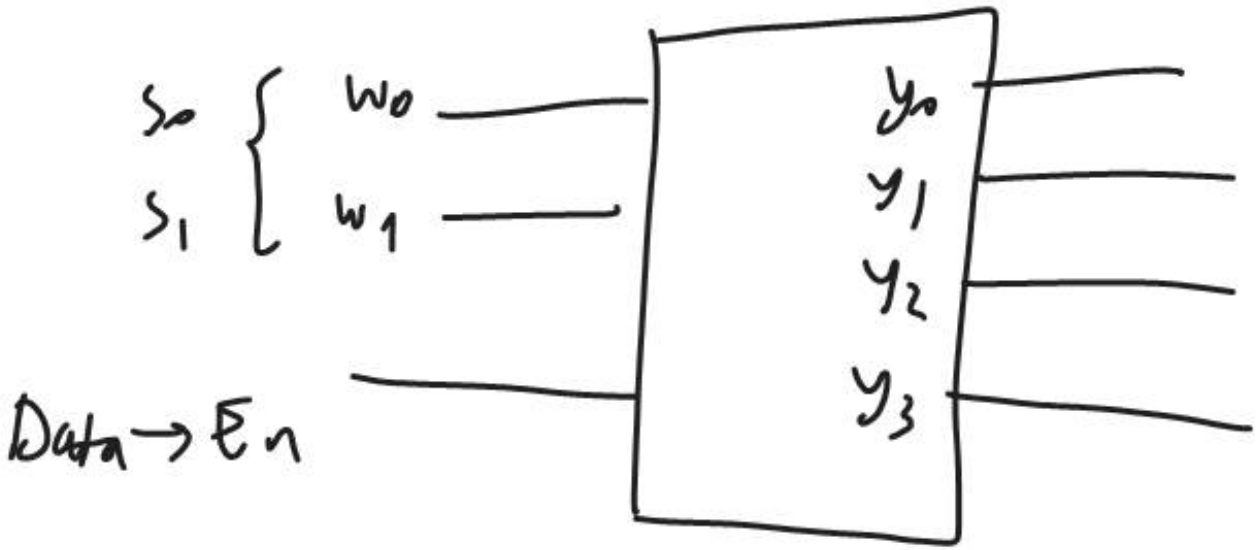
a 4-to-1 mux by using a 2-to-4 Decoder



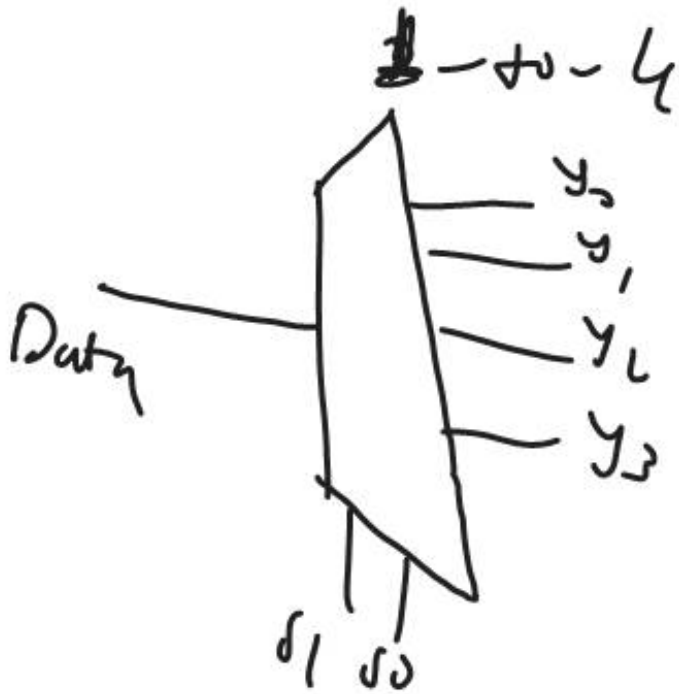
DEMULTIPLEXERS

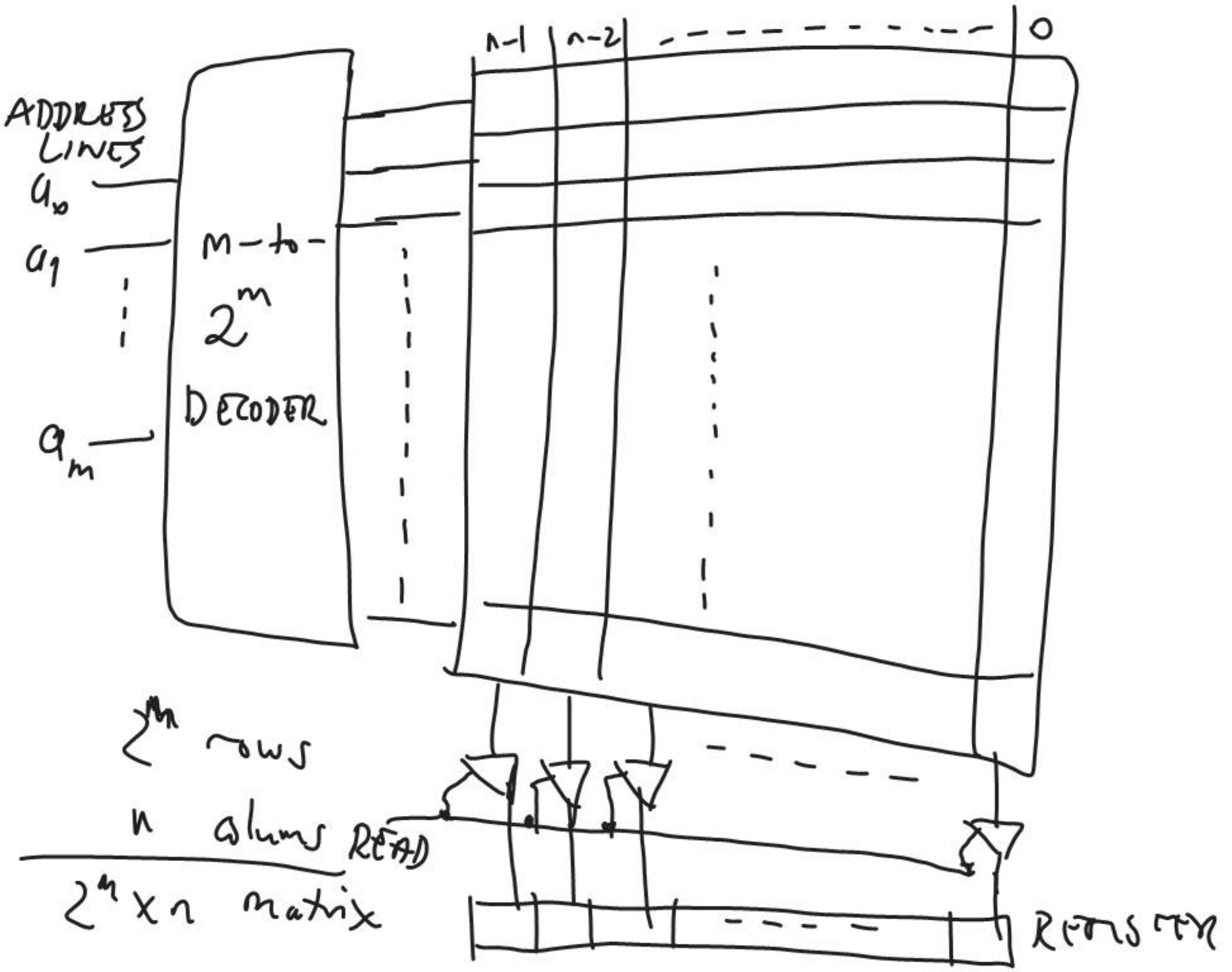


A Decoder can be used as a DEMUX

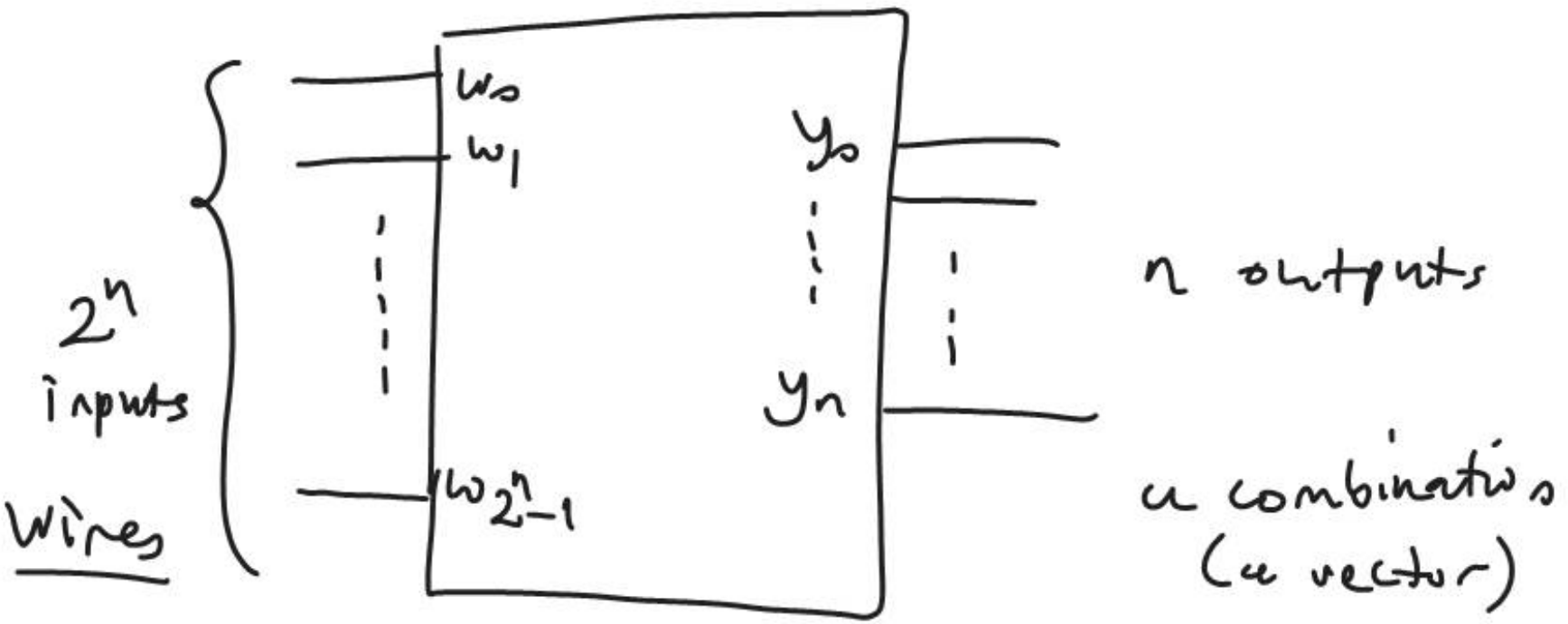


1-to-4 DEMUX



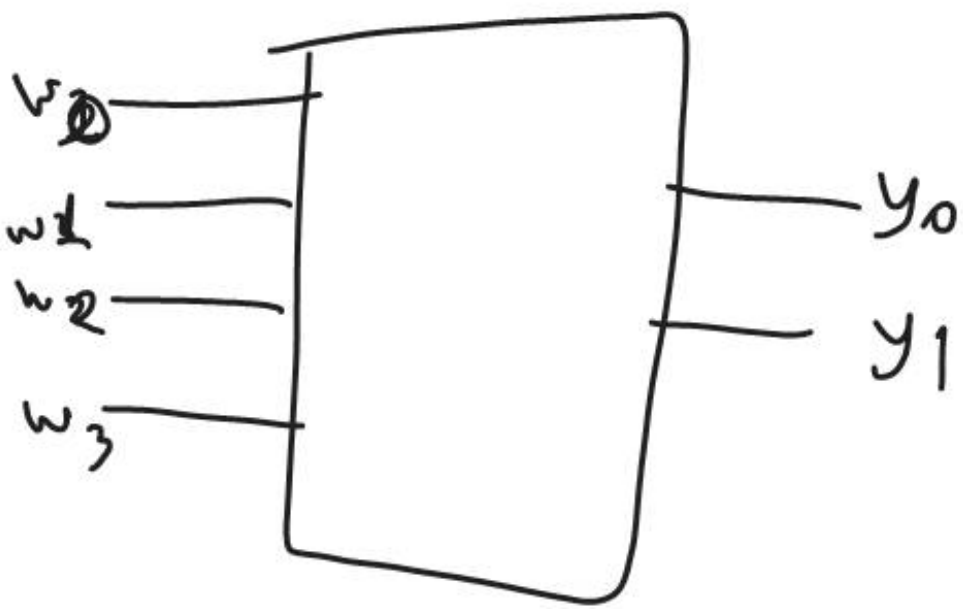

n-1
n-2
0
ADDRESS LINES
a_0
a_1
a_m
m-to-
2^m
DECODER
2^m rows
n cols
READ
$2^m \times n$ matrix
REGISTER

ENCODERS



It provides more compact form.

w_3	w_2	w_1	w_0	y_1	y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1



A 4-to-2 Encoder

more economical or compact representation is provided by an encoder.

A Priority Encoder

3-output functions

w_3: highest priority

w_0: lowest "

w_3	w_2	w_1	w_0	y_1	y_0	z
0	0	0	0	d	d	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

irrelevant

d: don't care either 0 or 1

intermediate signals

$i_0 = \bar{w}_3 \bar{w}_2 \bar{w}_1 w_0$

$i_1 = \bar{w}_3 \bar{w}_2 w_1$

$i_2 = \bar{w}_3 w_2$

$i_3 = w_3$

$y_0 = i_1 + i_3$

$y_1 = i_2 + i_3$

$z = i_0 + i_1 + i_2 + i_3$