Binary Decision Diagrams

A Binary Decision Diagram (BDD) is defined as a rooted, directed acyclic graph (DAG) with one or two terminal nodes corresponding to the logical values 0 and 1, and a set of non-terminal nodes, each of which has two outgoing edges.

Binary Decision Diagrams

These two edges correspond to the two functions F(x) and F(x'), as defined by Shannon's expansion theorem.

$F(x_1,x_2,\ldots,x_i,\ldots,x_n) =$

$x_i\ F(x_1,x_2,\ldots,1,\ldots,x_n) + x_i'\ F(x_1,x_2,\ldots,0,\ldots,x_n)$

Each node is associated with exactly one variable.

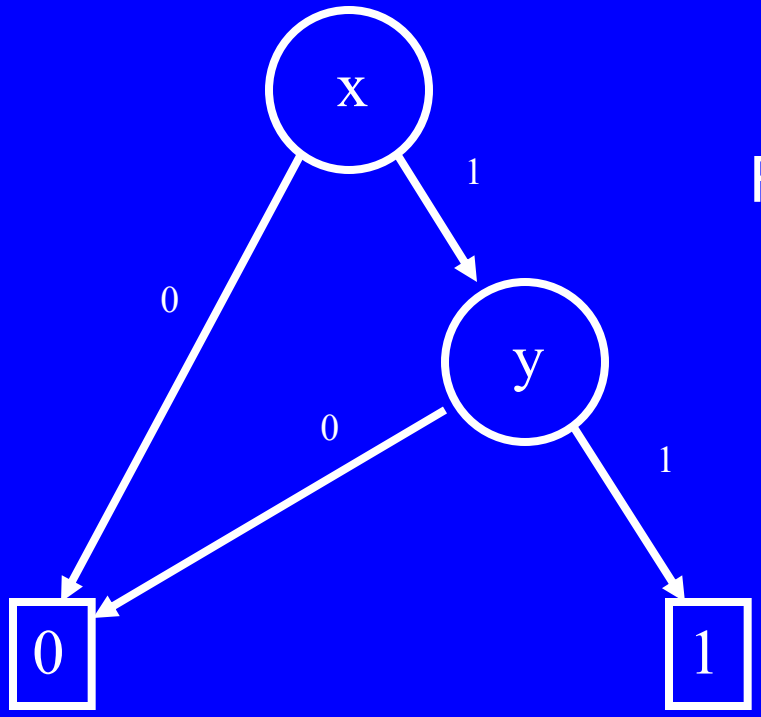
Binary Decision Diagrams

A BDD is said to be ordered if the variables appear in the same order along every path in the graph ($x_1 < x_2 < ... < x_i < ... < x_n$)

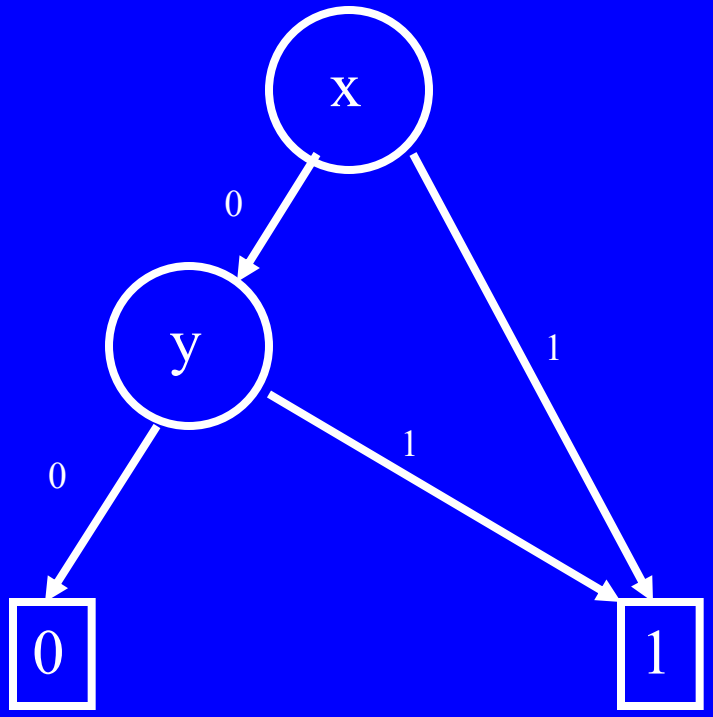
A BDD is said to be reduced if:

- there are no two nodes that correspond to the same variable and have identical subgraphs
- no node has two identical subgraphs

Binary Decision Diagrams



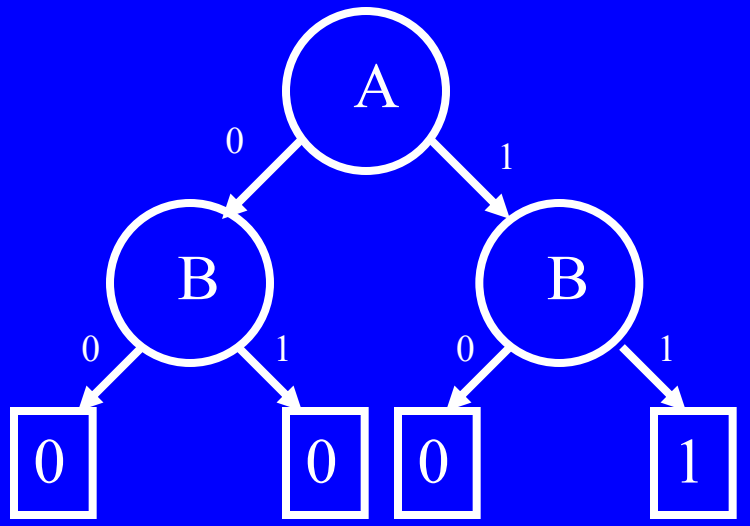
Binary Decision Diagrams



F = x+y

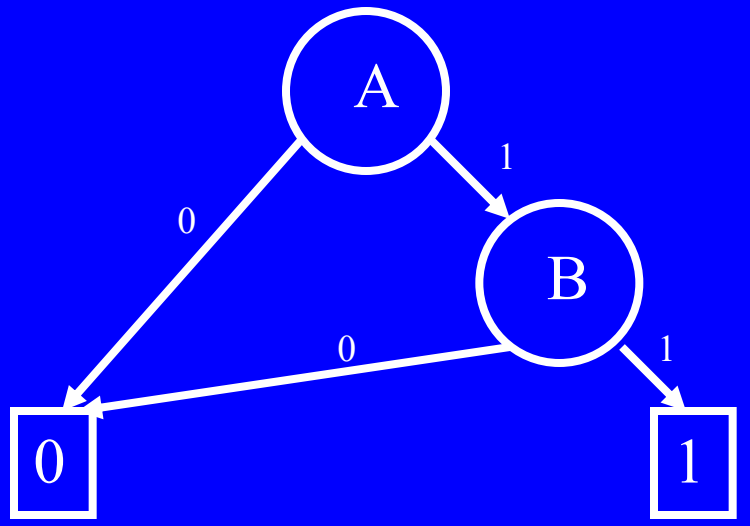
Binary Decision Diagrams

A	0	0	1	1
B	0	1	0	1
F	0	0	0	1



Binary Decision Diagrams

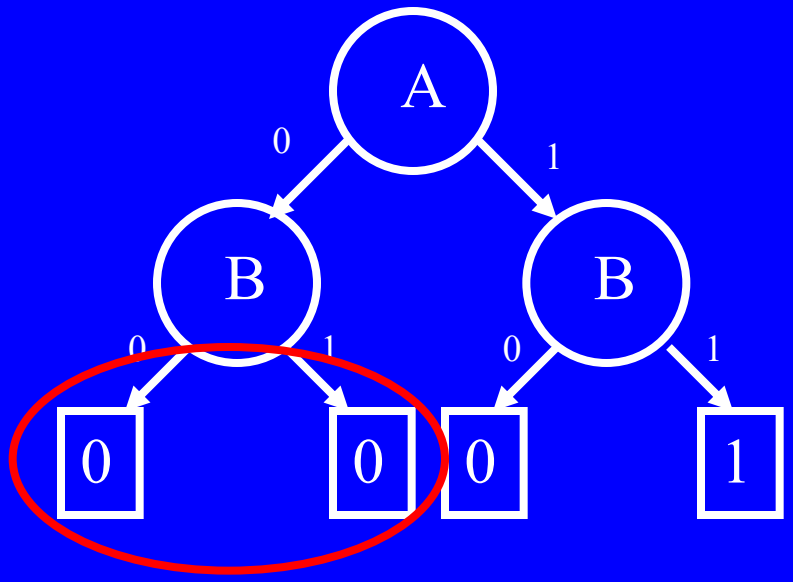
A	0	0	1	1
B	0	1	0	1
F	0	0	0	1



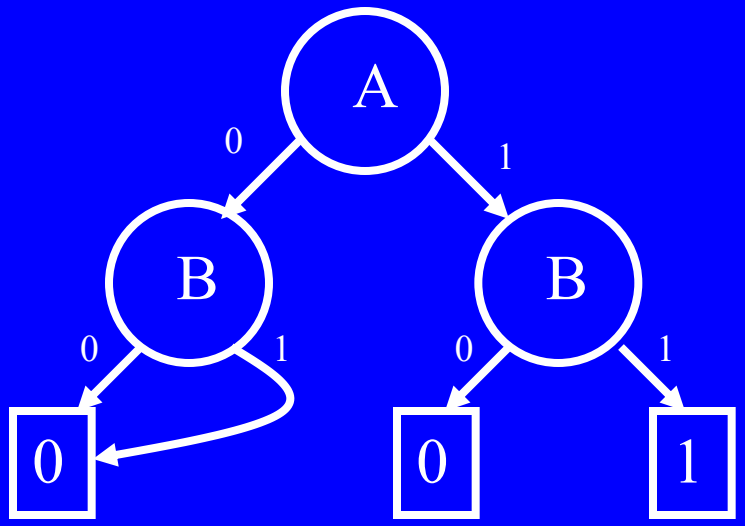
Reduction process

- Merge identical nodes.
- If both outgoing edges of a node X point to the same node Y, remove node X and connect its predecessor directly to node Y.
- Repeat the process until no further nodes can be merged or eliminated.

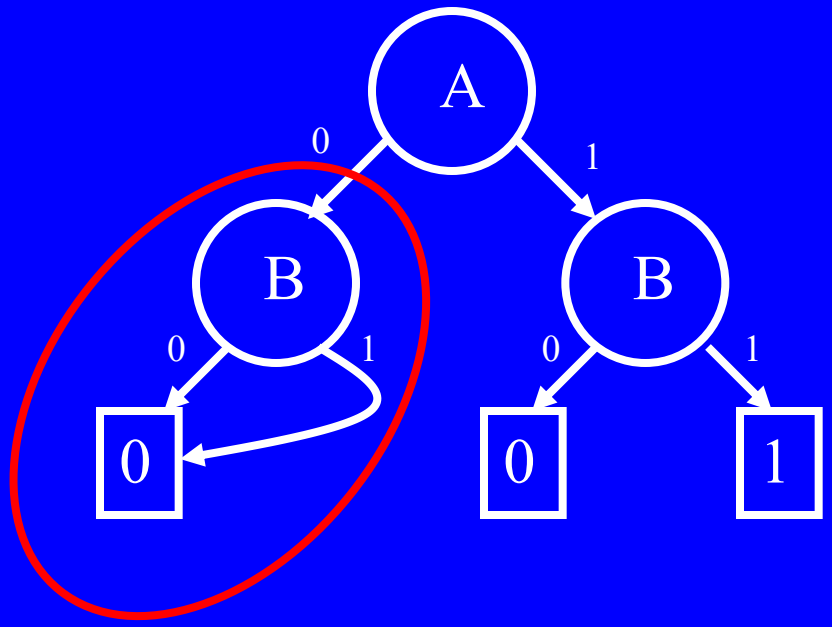
Reduction process



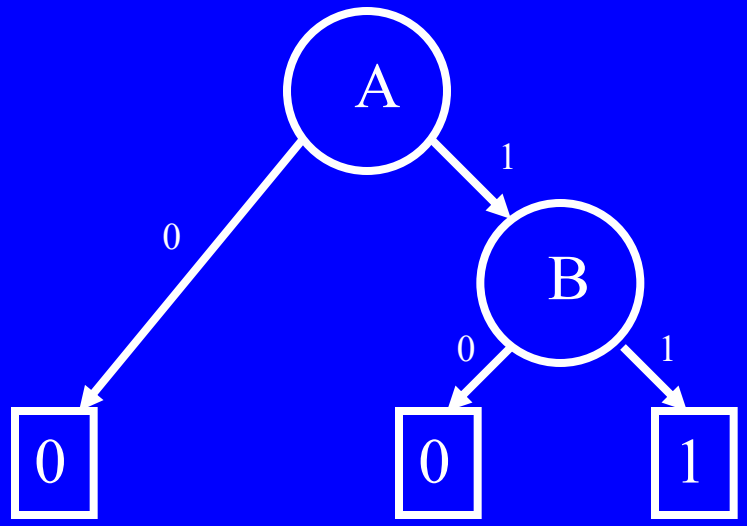
Reduction process



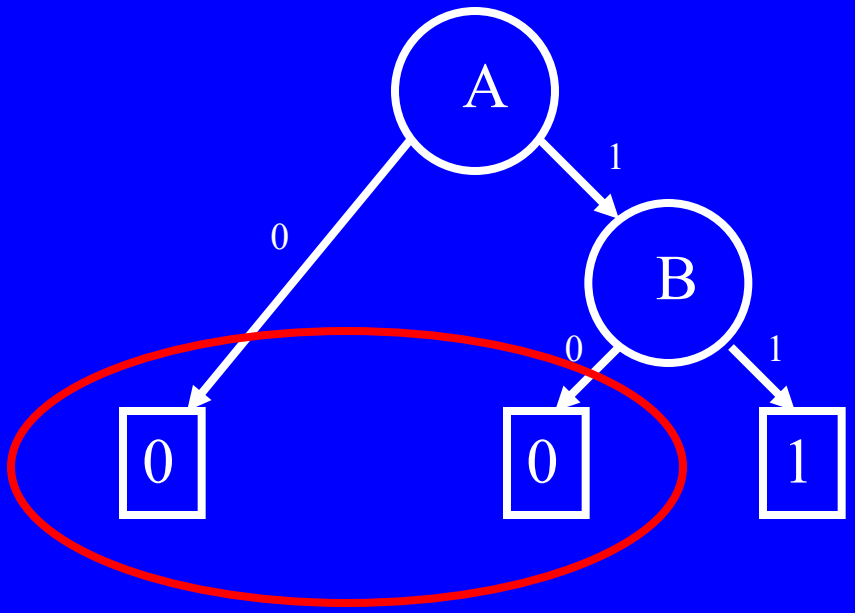
Reduction process



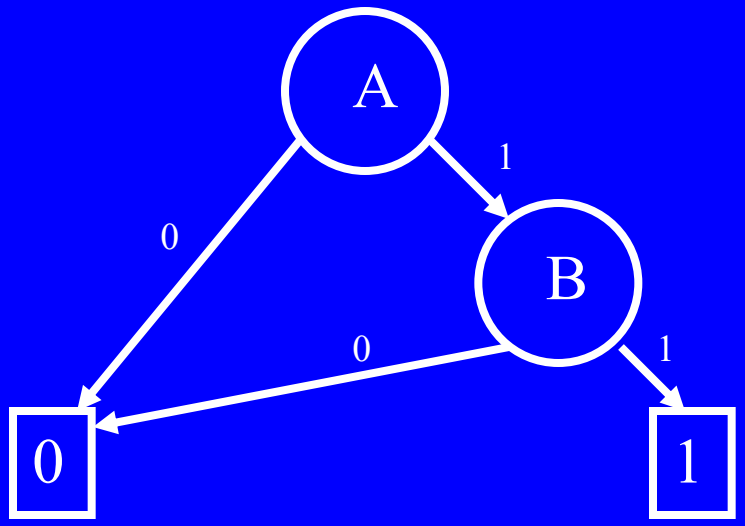
Reduction process



Reduction process



Reduction process

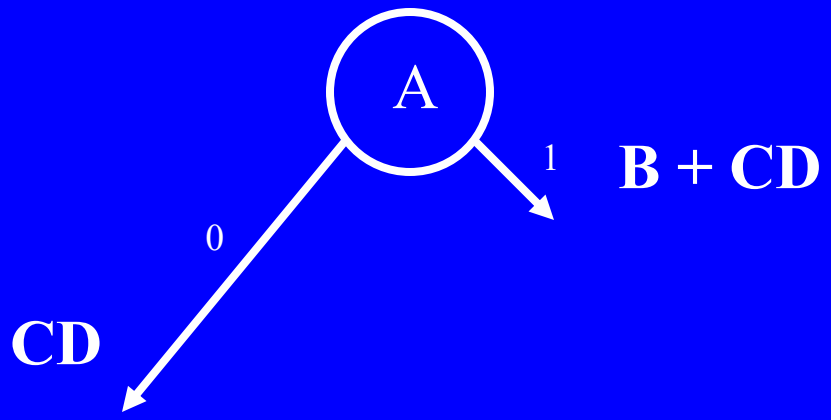


Example



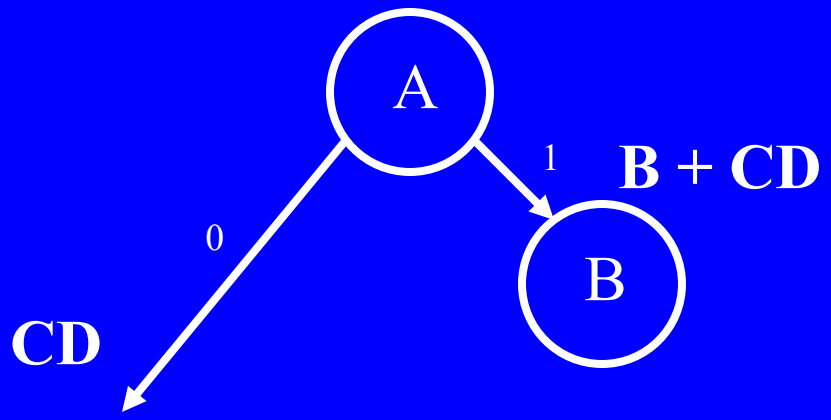
F = AB + CD
A<B<C<D

Example



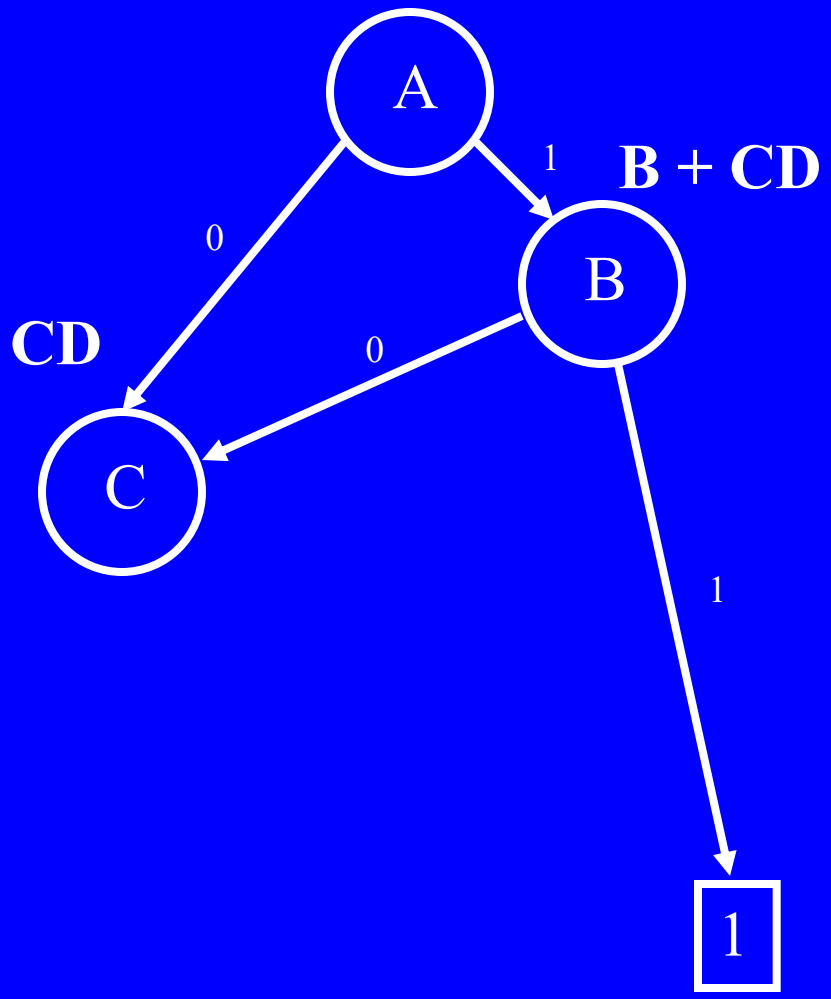
F = AB + CD
A<B<C<D

Example



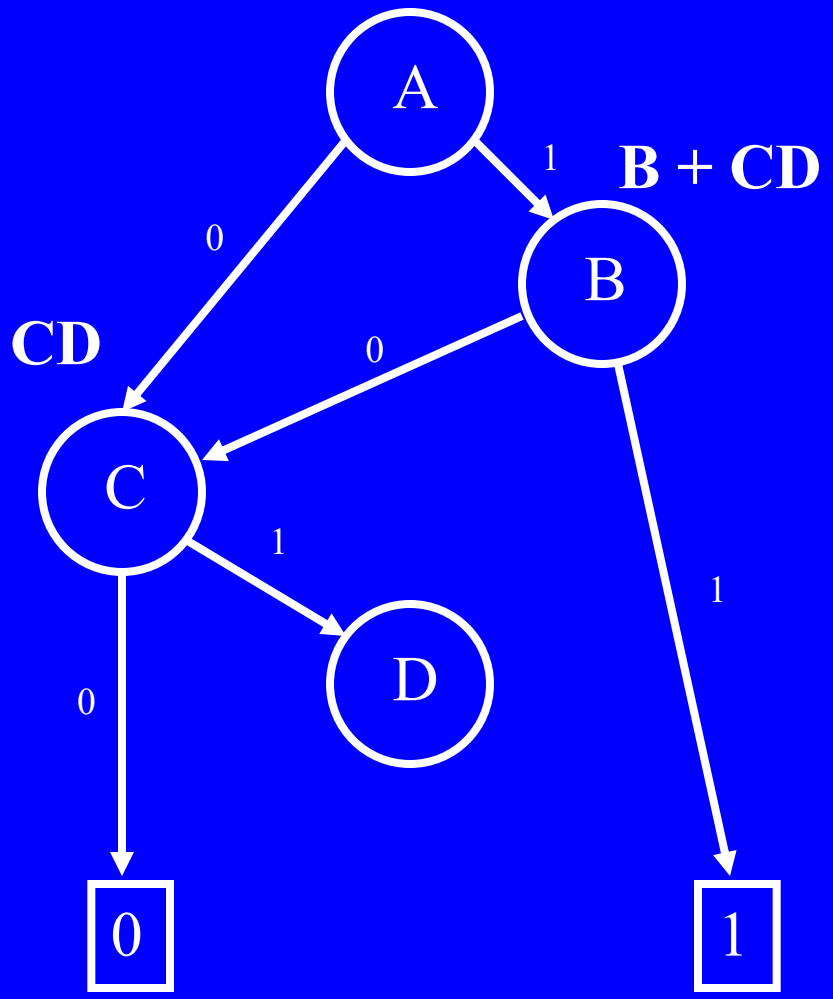
F = AB + CD
A<B<C<D

Example



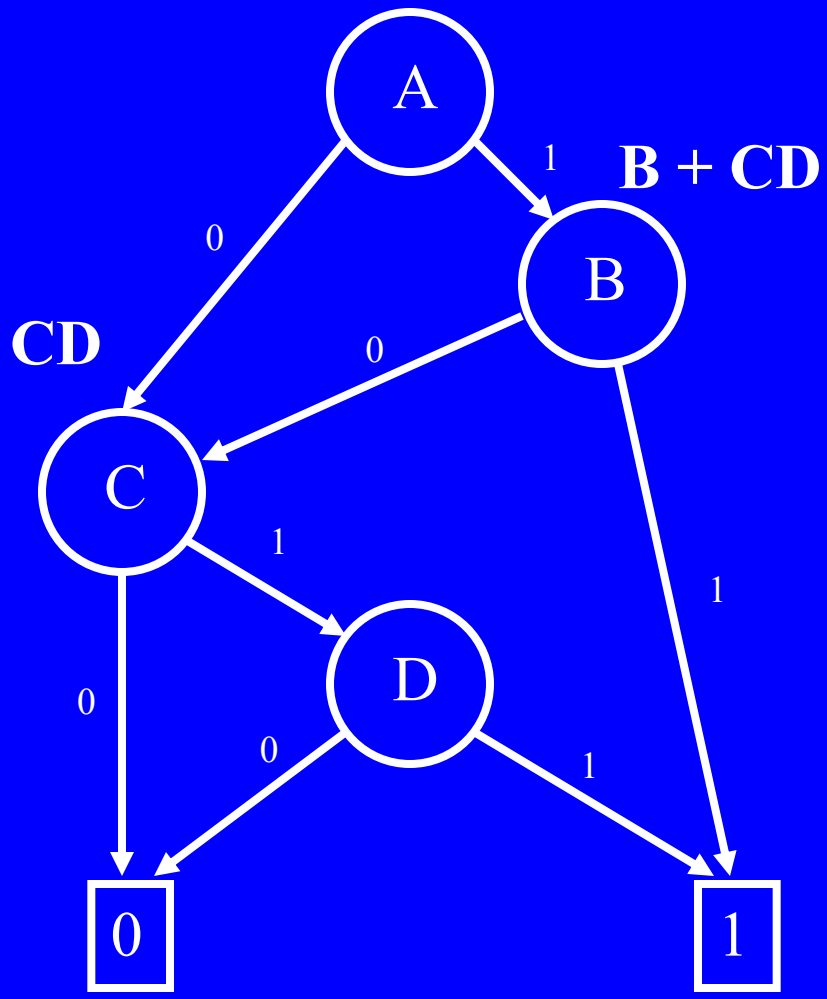
F = AB + CD
A<B<C<D

Example



F = AB + CD
A<B<C<D

Example



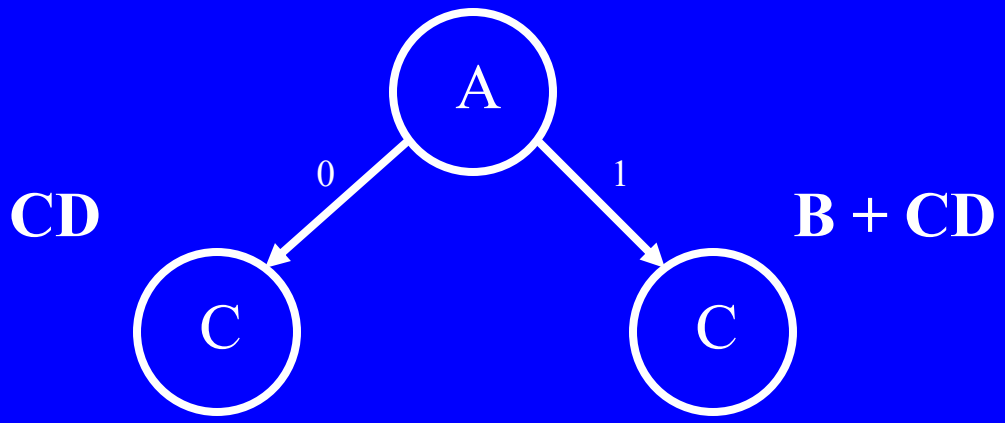
F = AB + CD
A<B<C<D

Another example



F = AB + CD
A<C<B<D

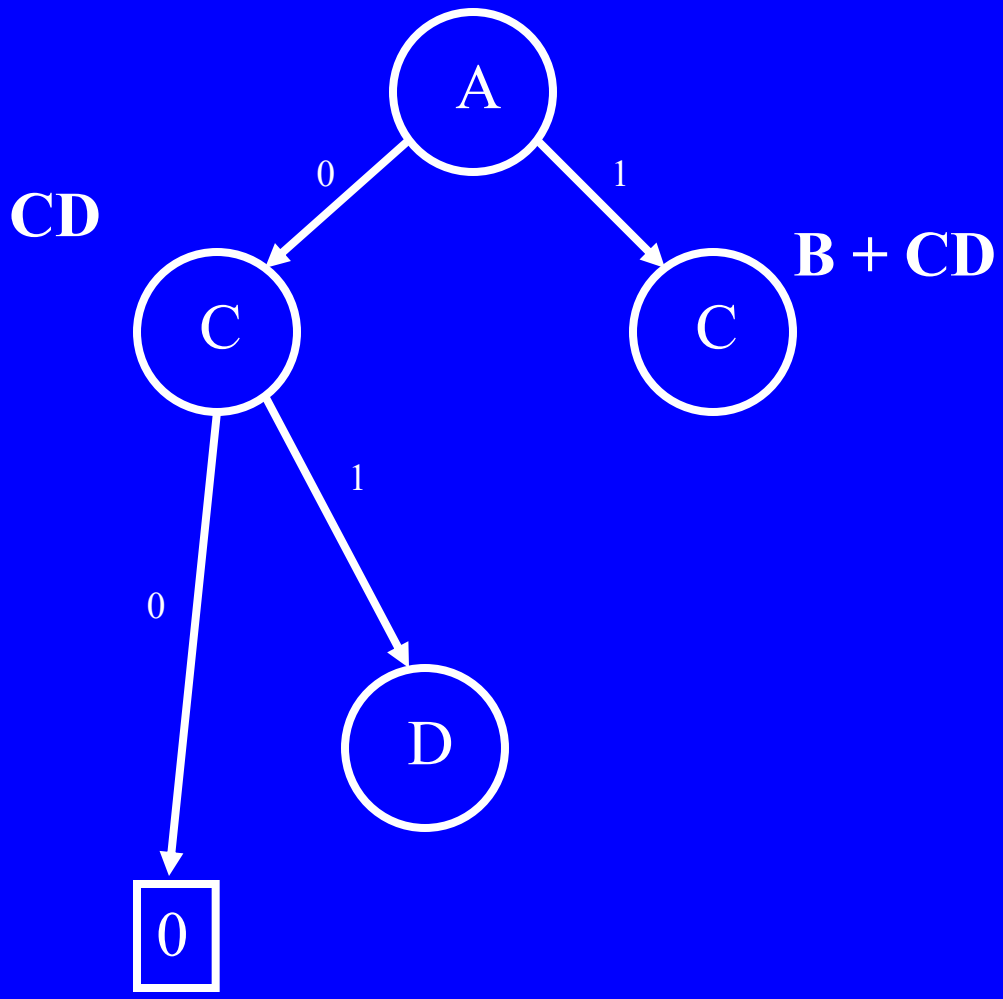
Another example



F = AB + CD

A<C<B<D

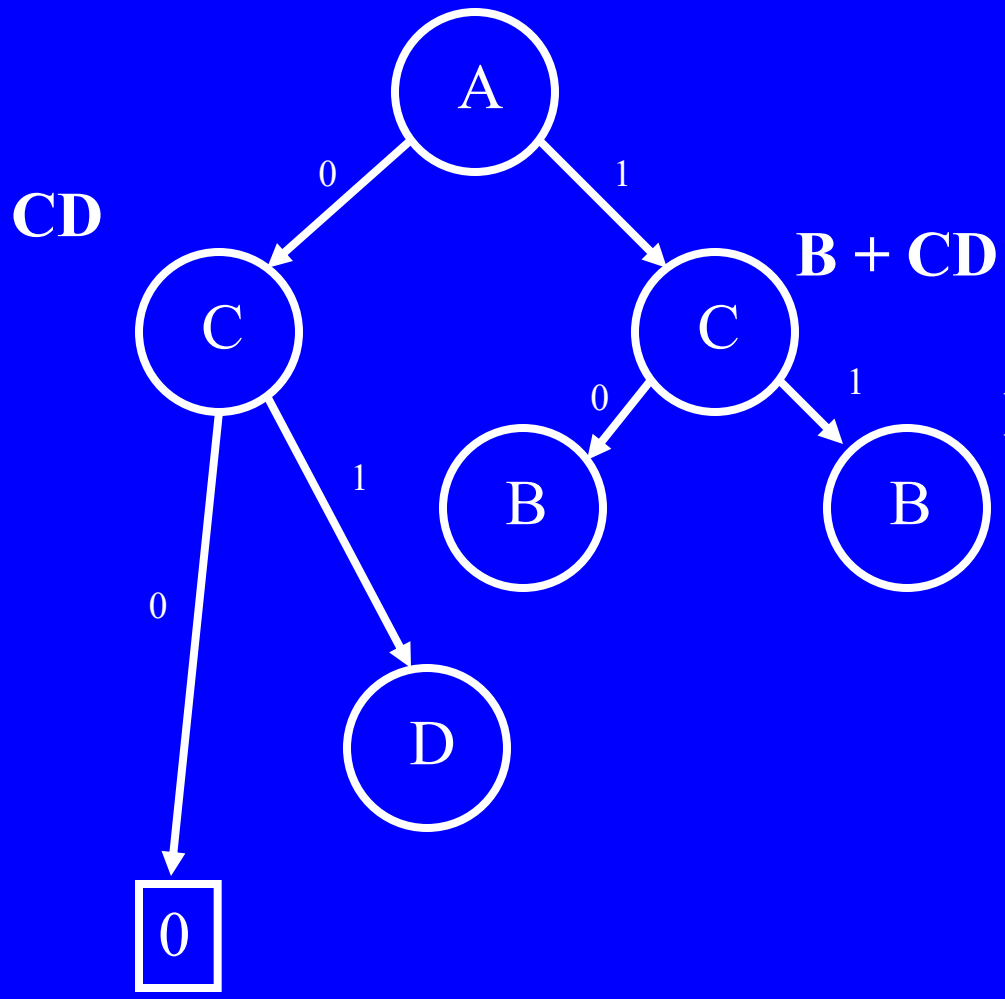
Another example



F = AB + CD

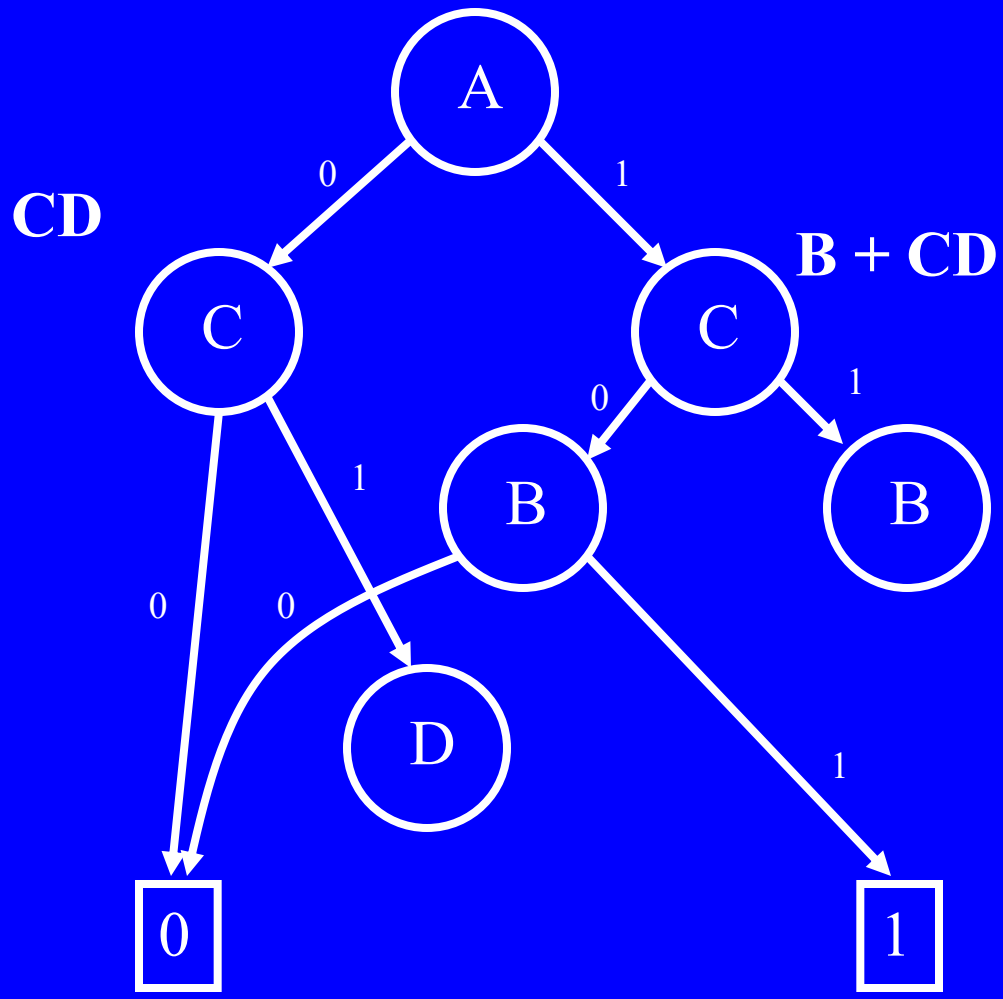
A<C<B<D

Another example



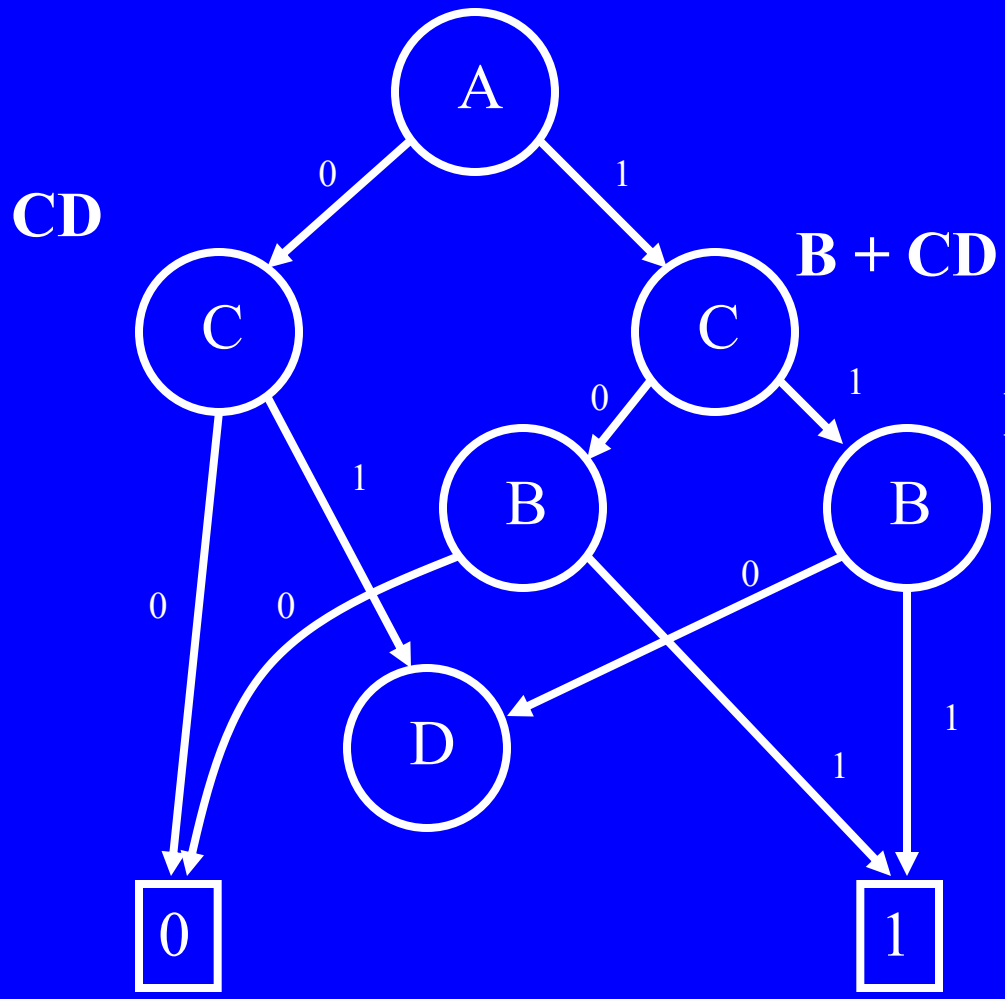
F = AB + CD
A<C<B<D

Another example



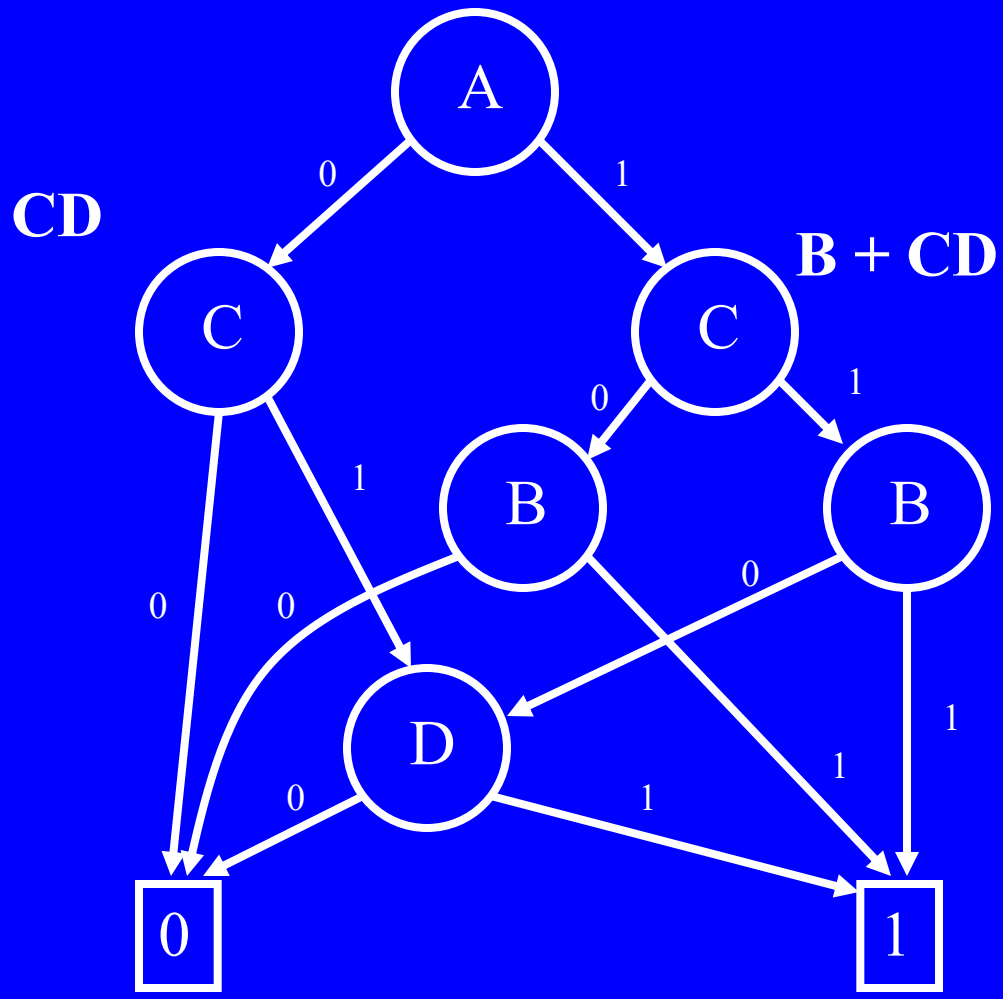
F = AB + CD
A<C<B<D

Another example



F = AB + CD
A<C<B<D

Another example

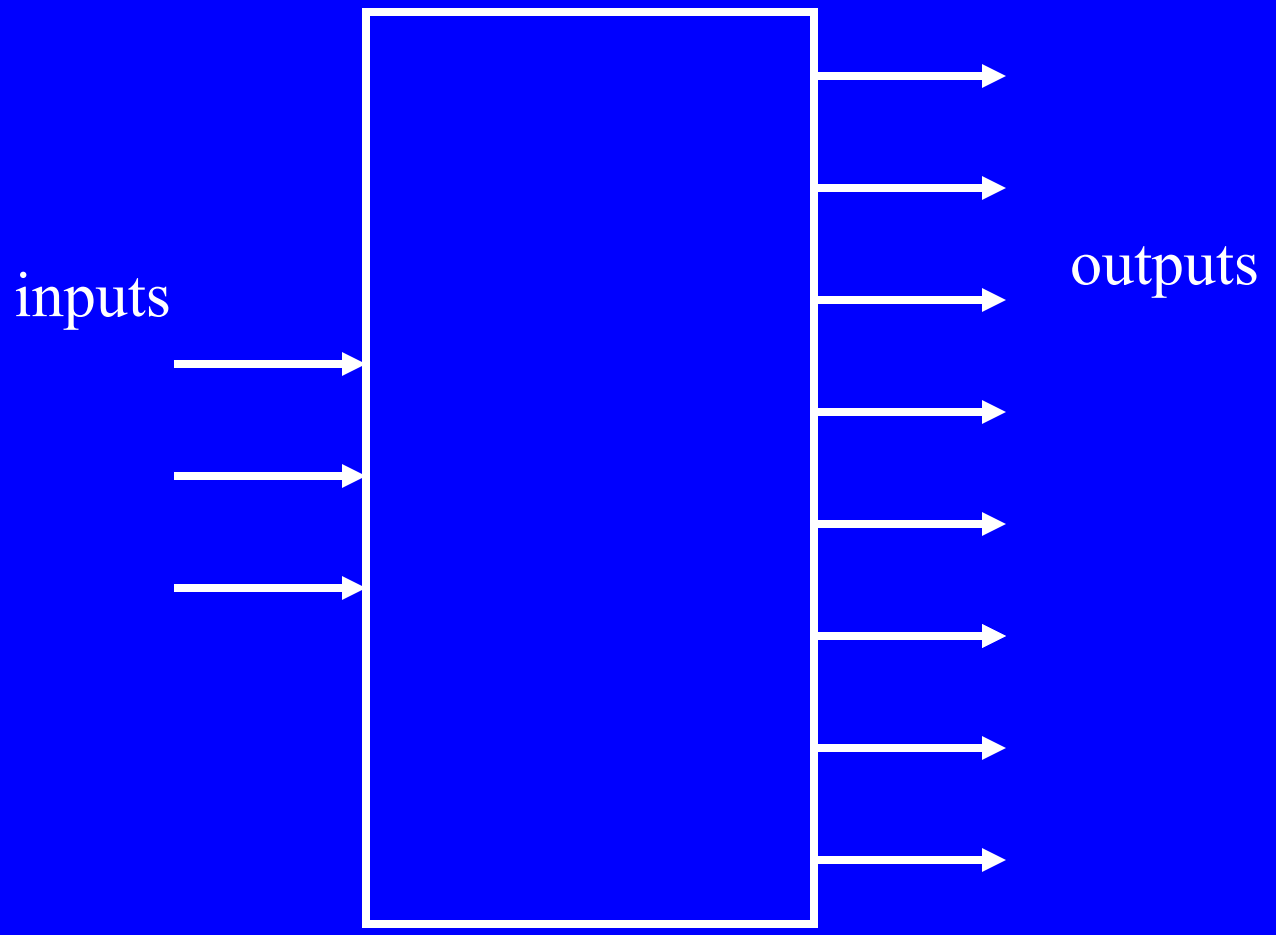


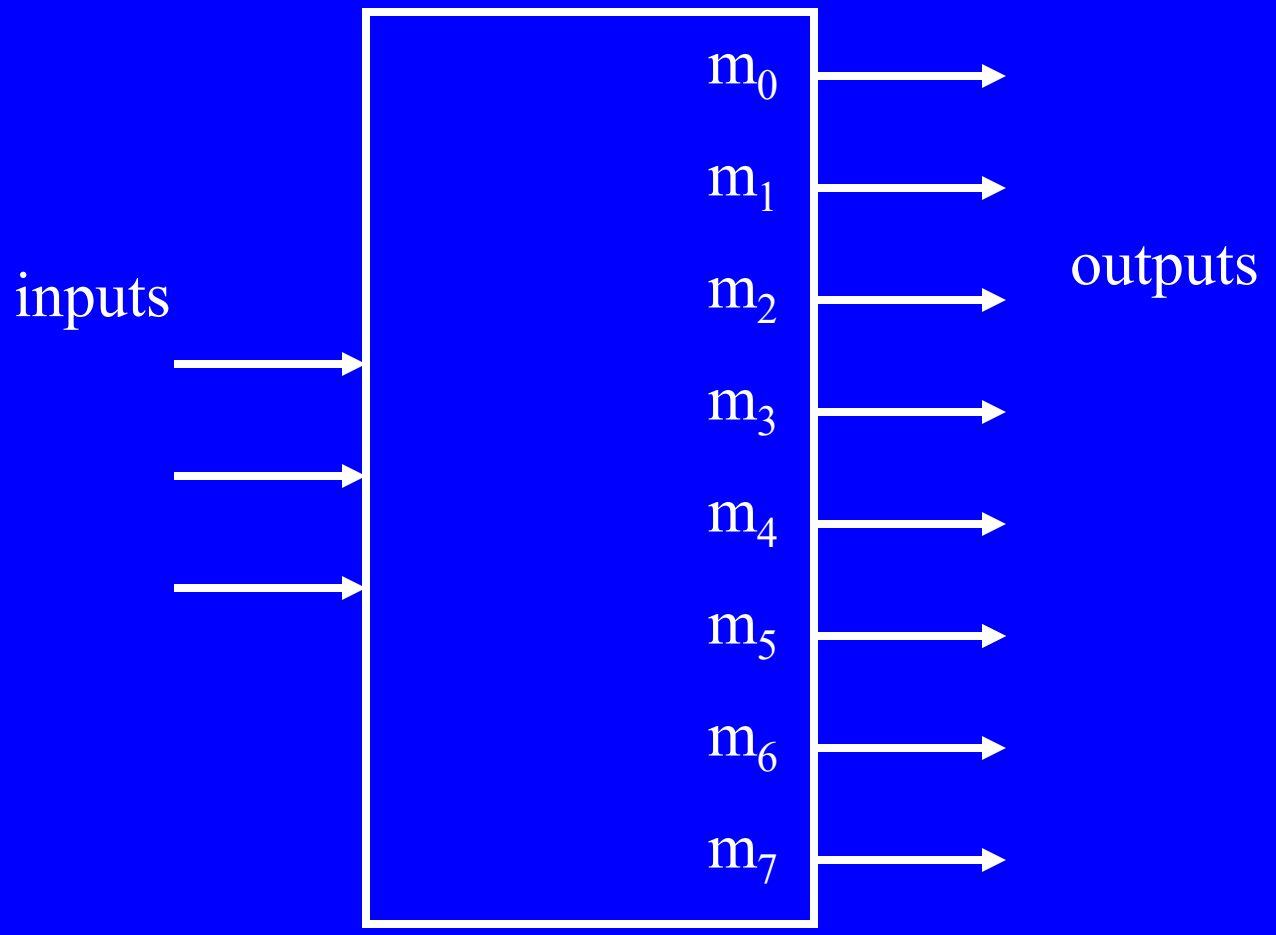
Minterms and Maxterms

x	y	z	term	name	term	name
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

Decoders

inputs	outputs							
x y z	e0	e1	e2	e3	e4	e5	e6	e7
0 0 0	1	0	0	0	0	0	0	0
0 0 1	0	1	0	0	0	0	0	0
0 1 0	0	0	1	0	0	0	0	0
0 1 1	0	0	0	1	0	0	0	0
1 0 0	0	0	0	0	1	0	0	0
1 0 1	0	0	0	0	0	1	0	0
1 1 0	0	0	0	0	0	0	1	0
1 1 1	0	0	0	0	0	0	0	1


inputs
outputs

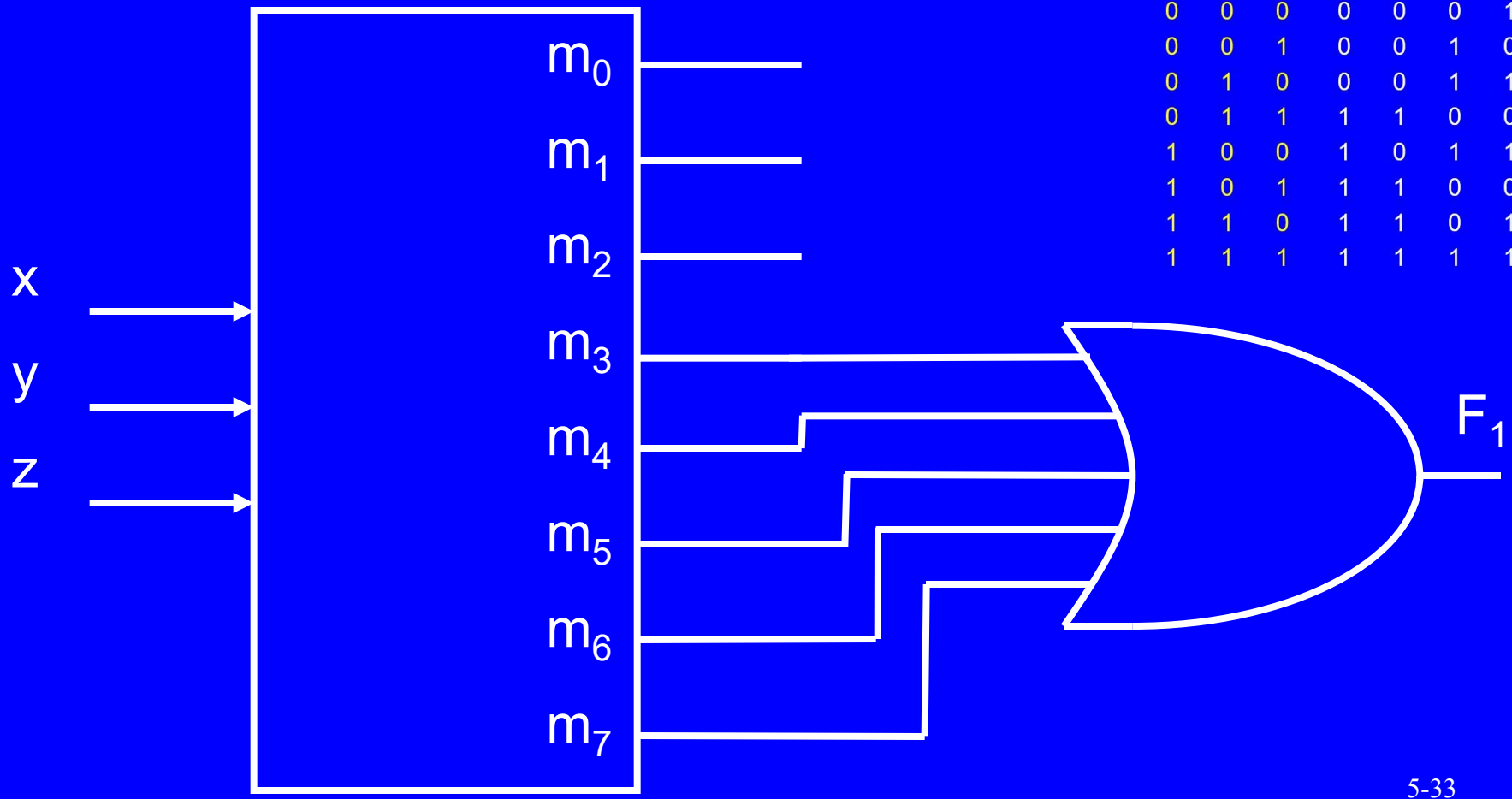

inputs
m_0
m_1
m_2
m_3
m_4
m_5
m_6
m_7
outputs

Truth table

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

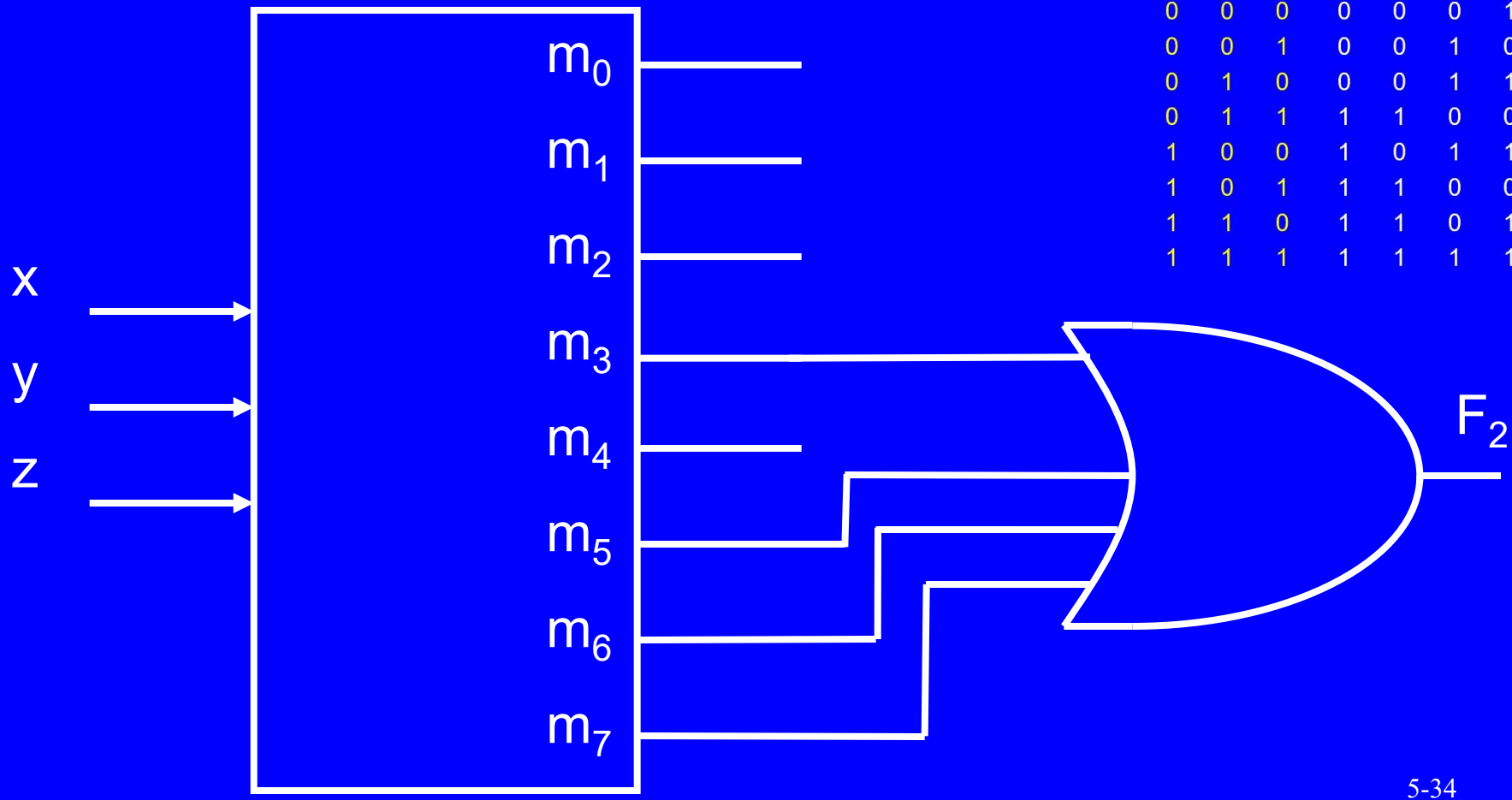
Function implementation with decoders

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1



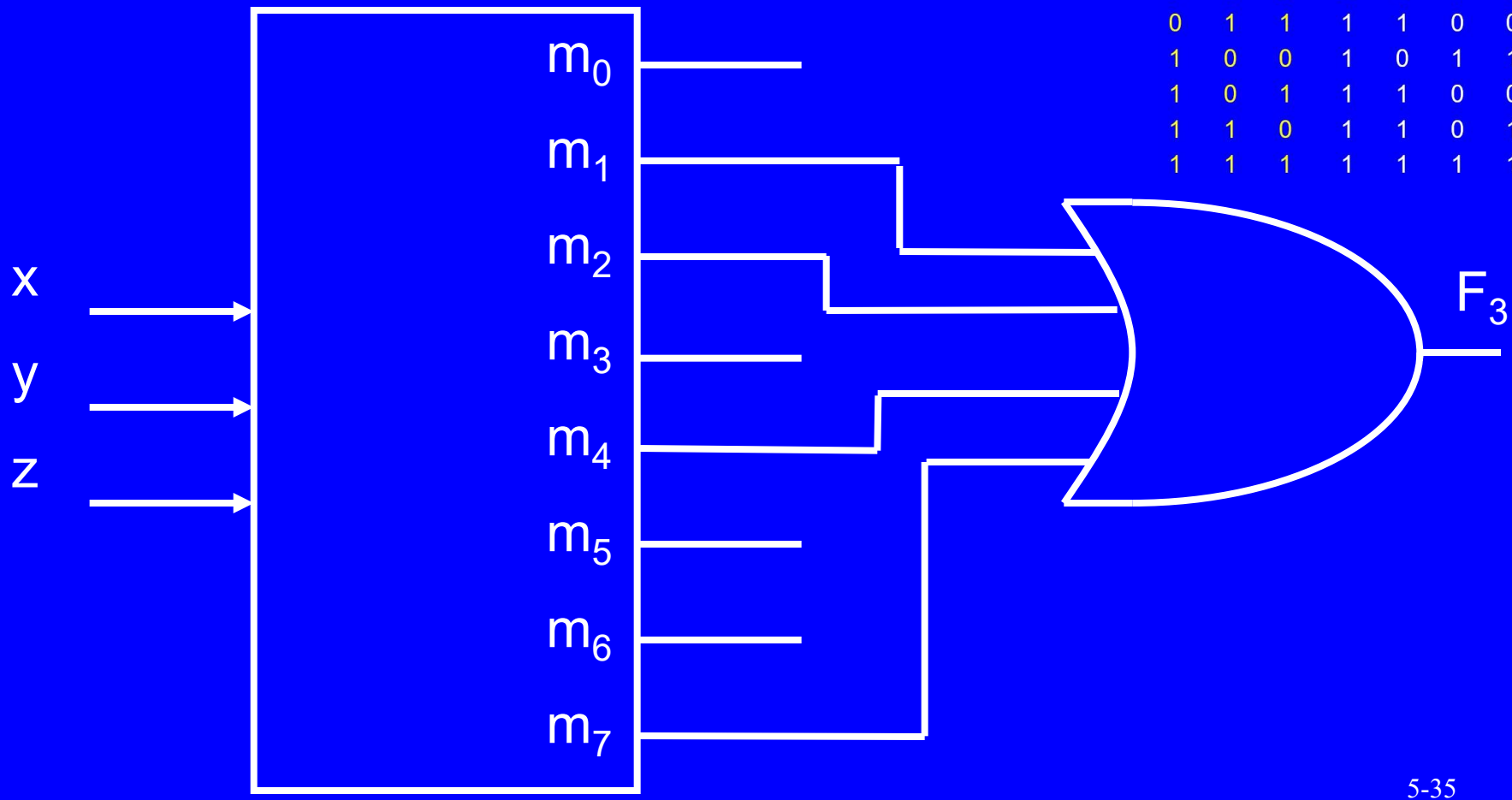
Function implementation with decoders

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1



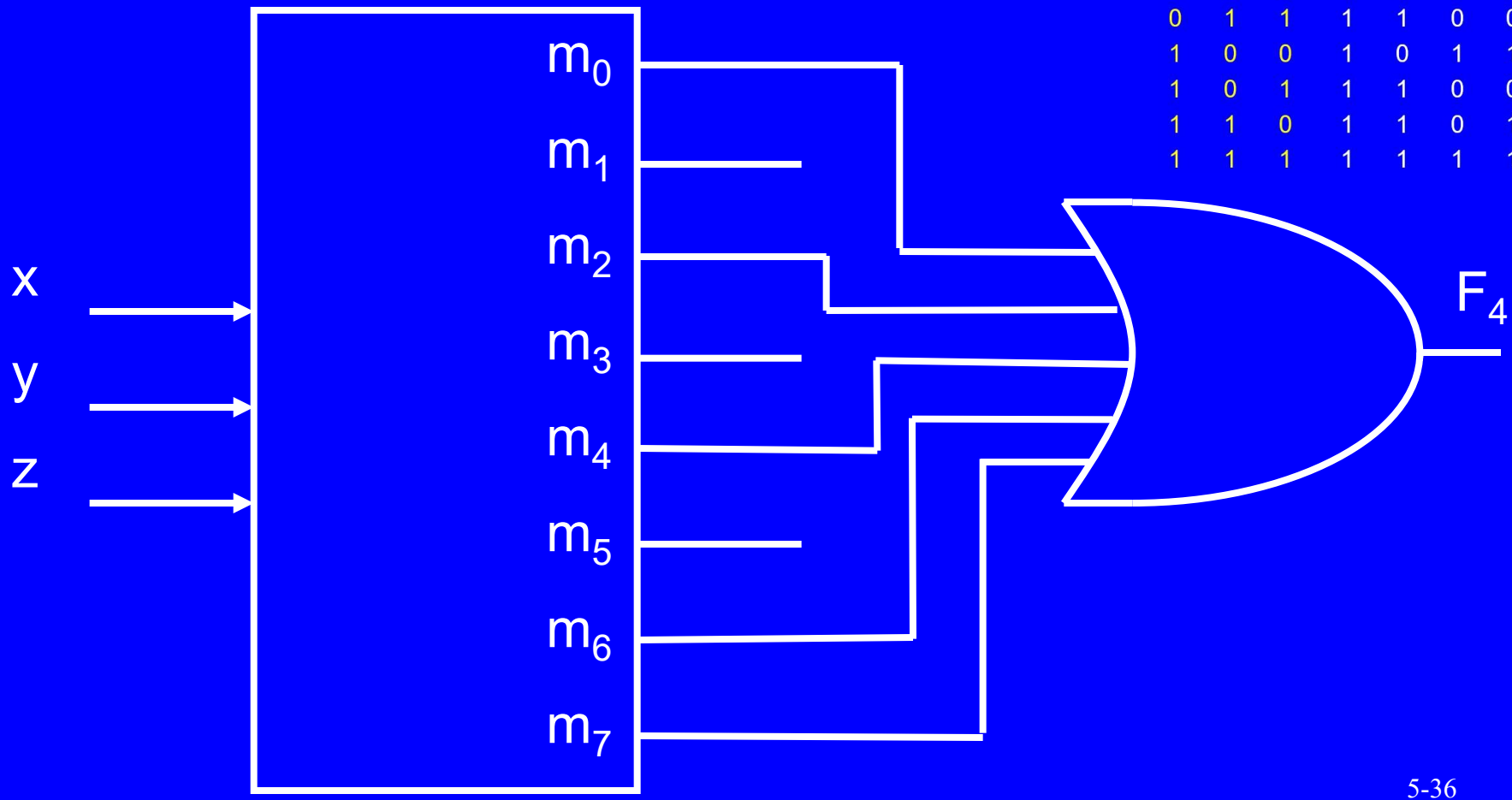
Function implementation with decoders

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1



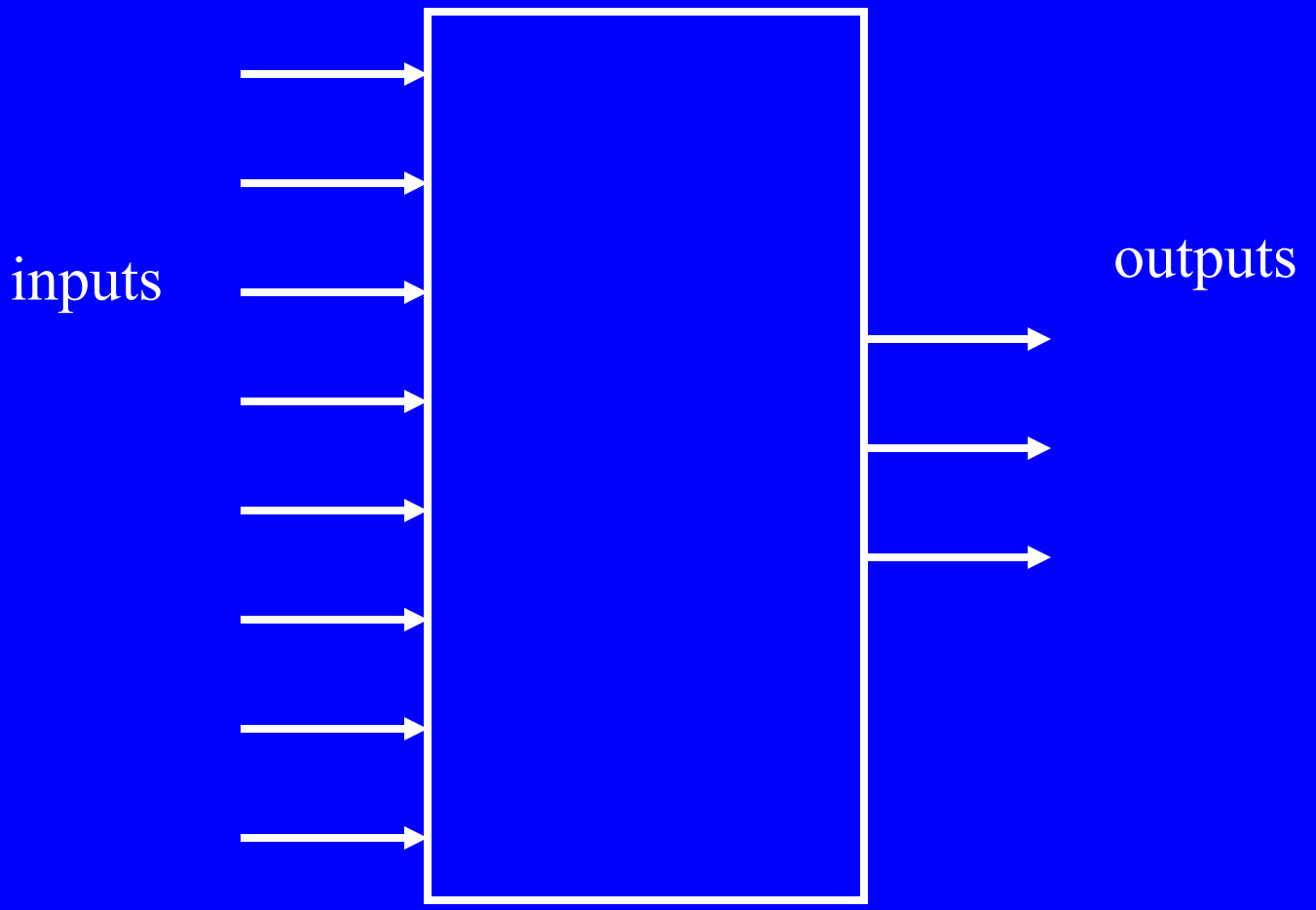
Function implementation with decoders

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

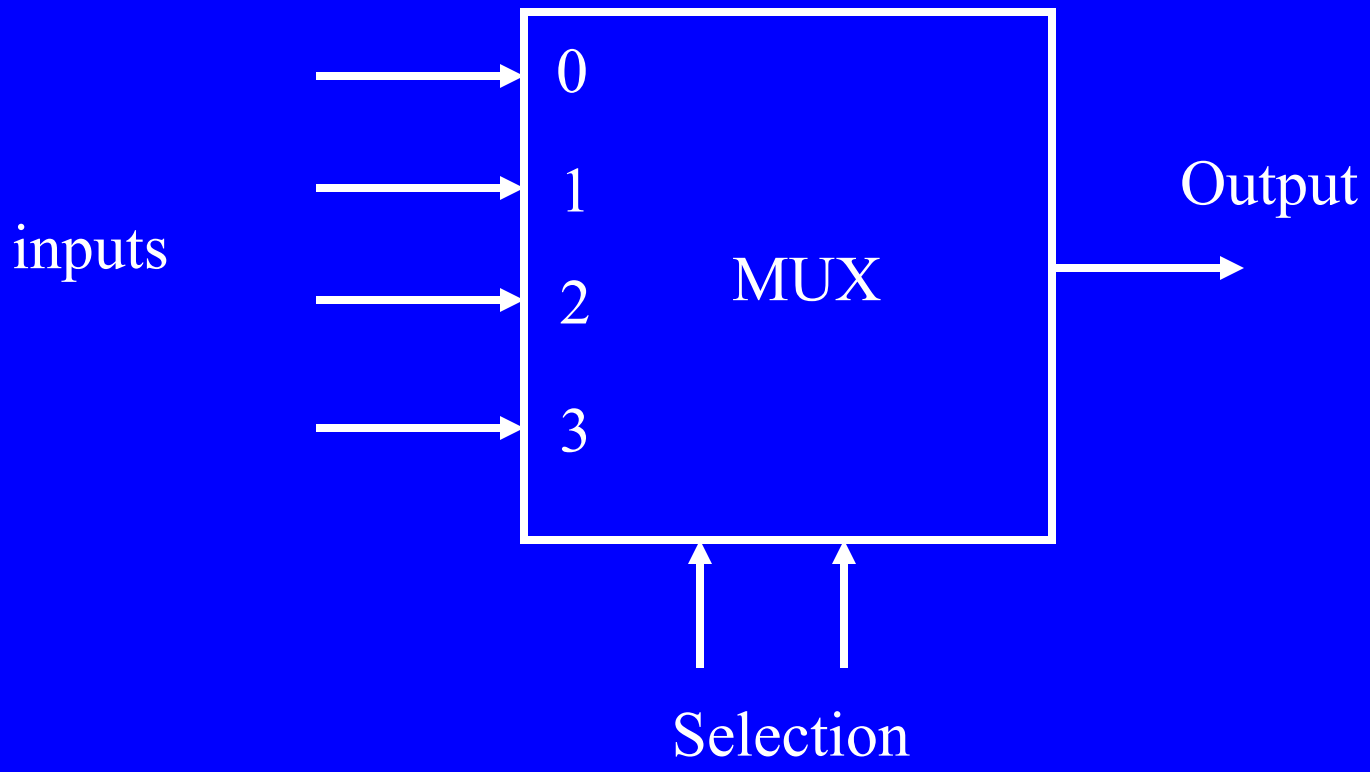


Encoders

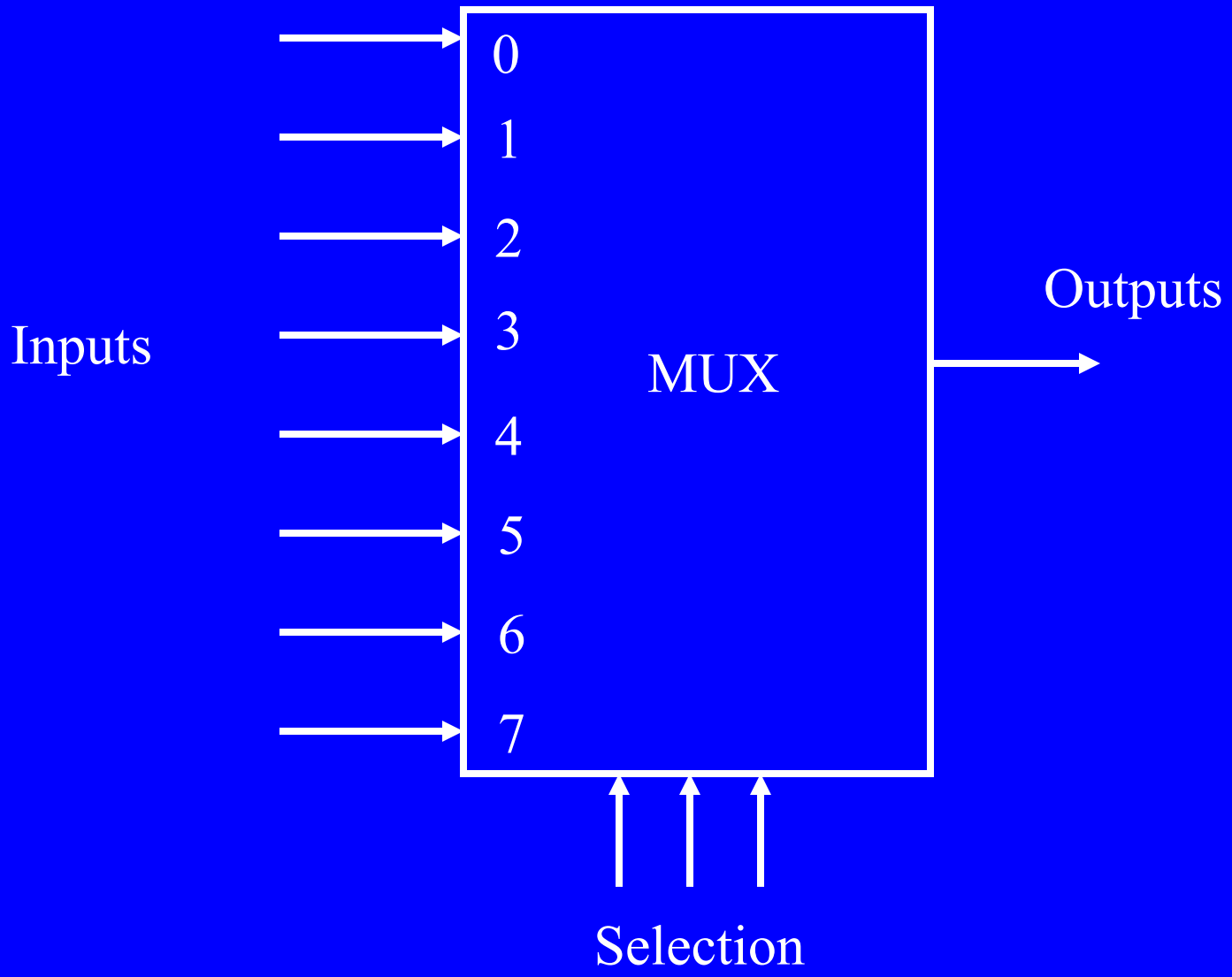
inputs	outputs							
x y z	e0	e1	e2	e3	e4	e5	e6	e7
0 0 0	1	0	0	0	0	0	0	0
0 0 1	0	1	0	0	0	0	0	0
0 1 0	0	0	1	0	0	0	0	0
0 1 1	0	0	0	1	0	0	0	0
1 0 0	0	0	0	0	1	0	0	0
1 0 1	0	0	0	0	0	1	0	0
1 1 0	0	0	0	0	0	0	1	0
1 1 1	0	0	0	0	0	0	0	1


inputs
outputs

Multiplexers



Multiplexers



Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

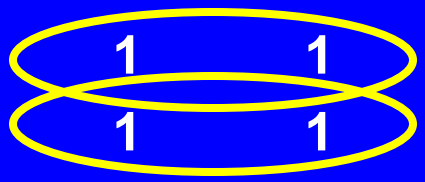
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

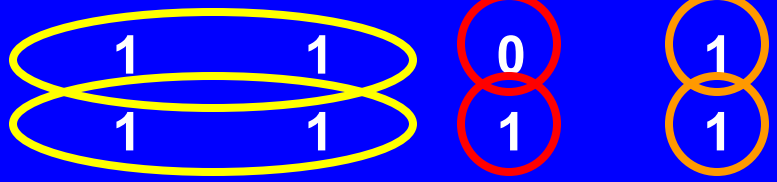


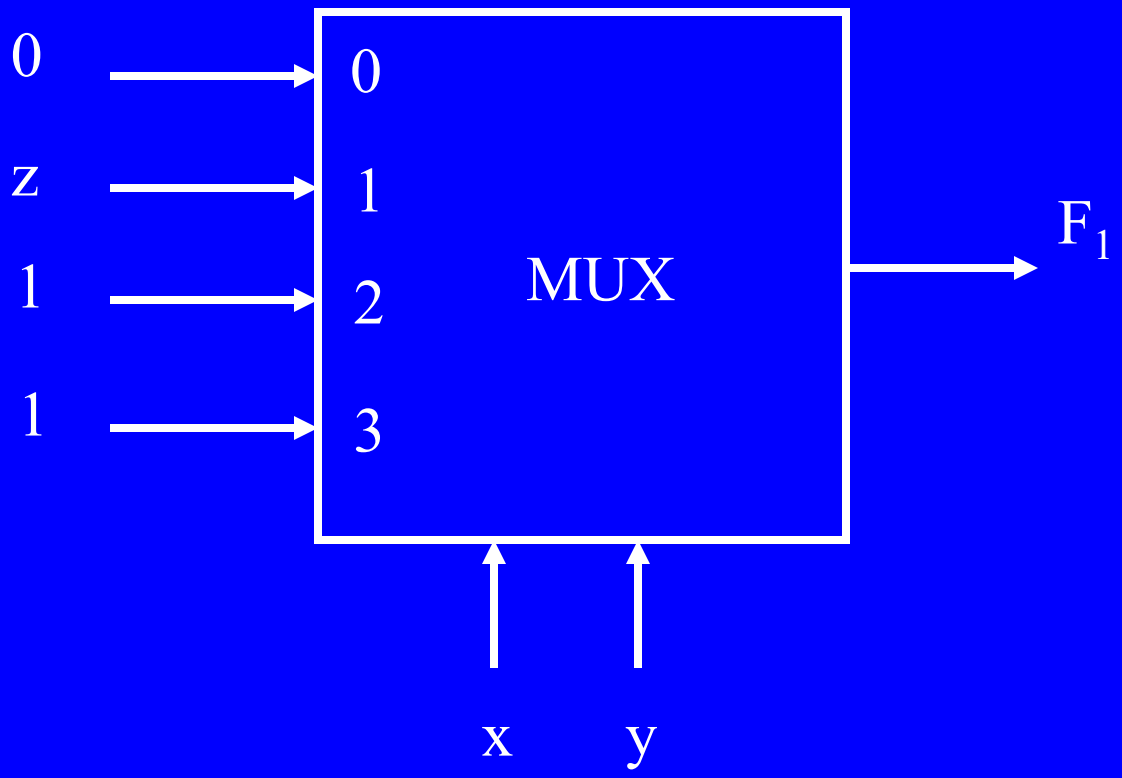
Function implementation with multiplexers

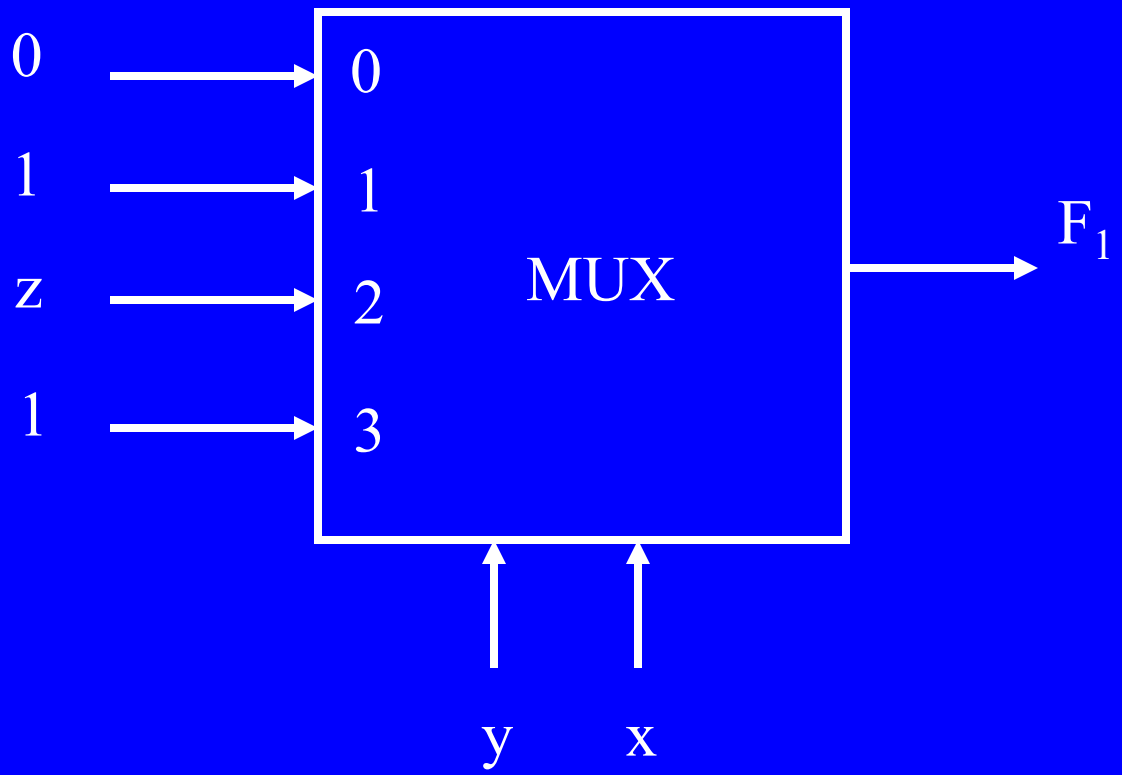
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

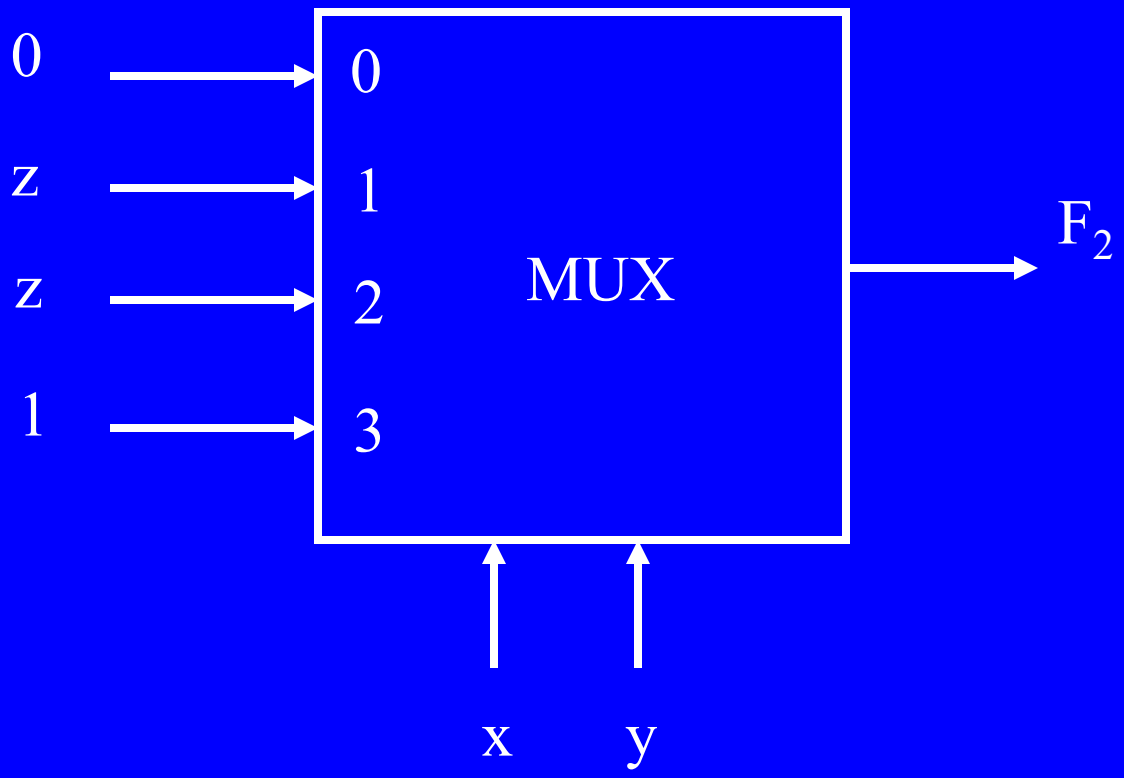
Function implementation with multiplexers

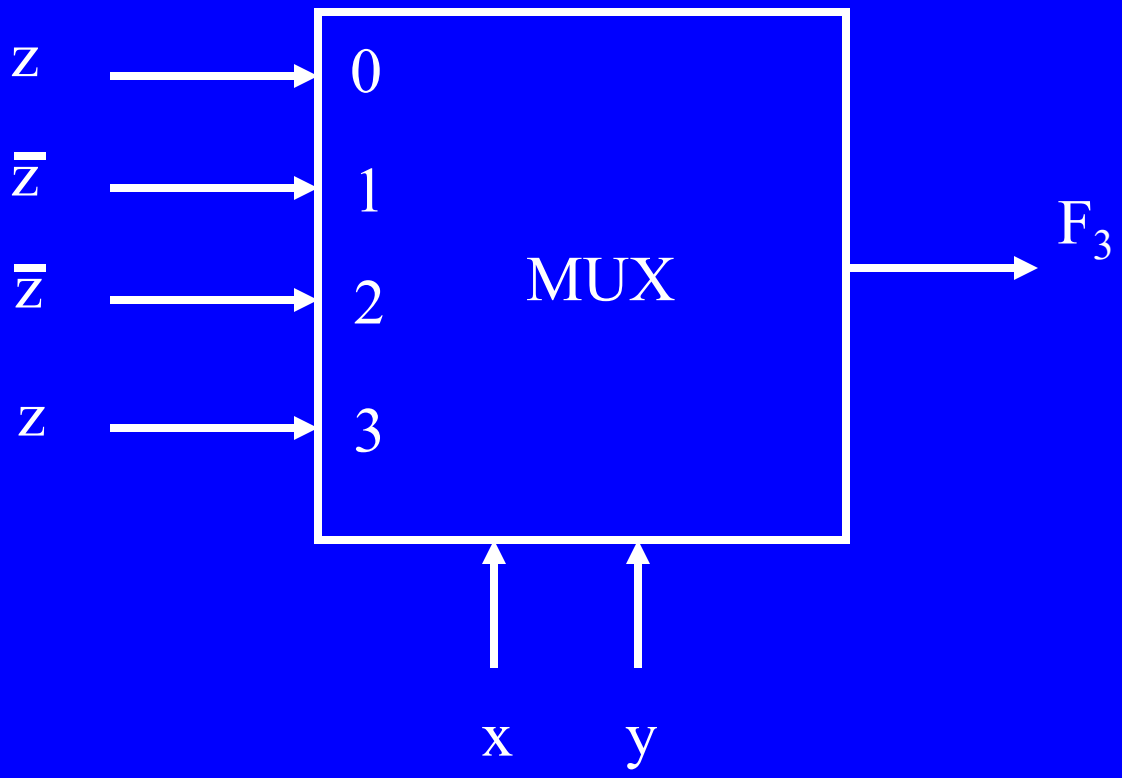
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

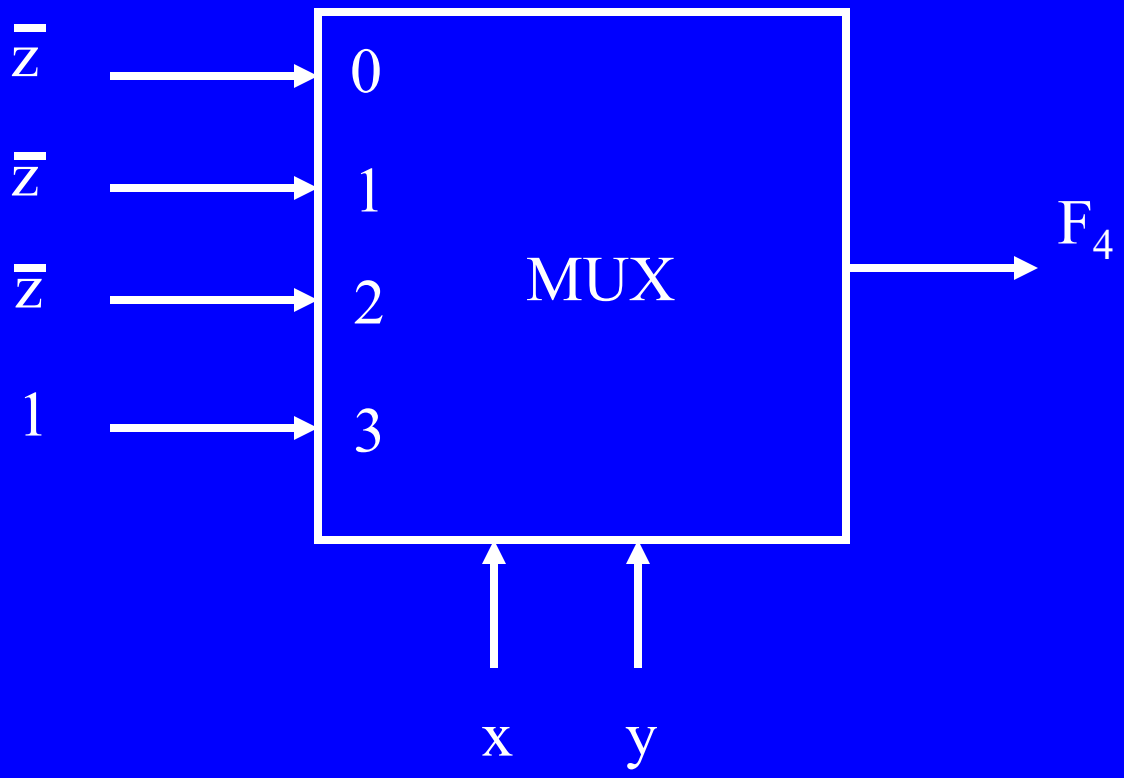



0
z
1
1
0
1
2
3
MUX
F_1
x
y


0
1
z
1
0
1
2
3
MUX
F_1
y
x


0
z
z
1
0
1
2
3
MUX
F_2
x
y


z
$\bar{z}$
$\bar{z}$
z
0
1
2
3
MUX
F_3
x
y


$\bar{z}$
$\bar{z}$
$\bar{z}$
1
0
1
2
3
MUX
F_4
x
y

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

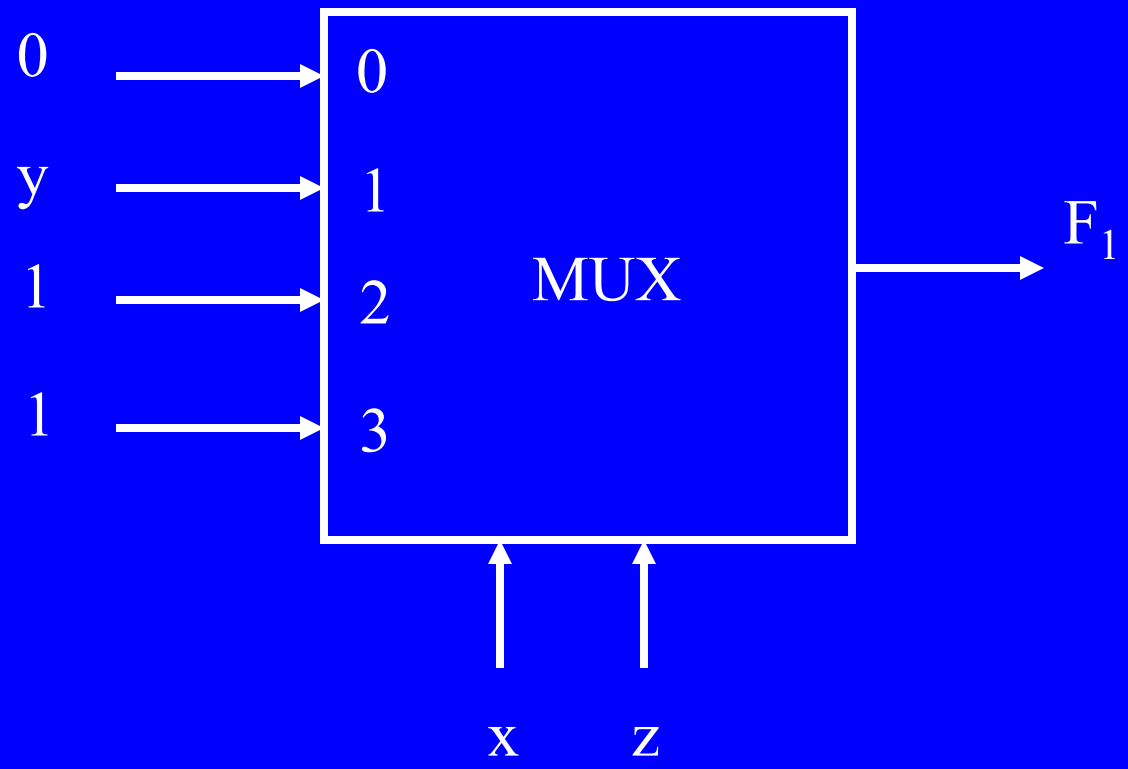
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1


0
y
1
1
0
1
2
3
MUX
F_1
x
z

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

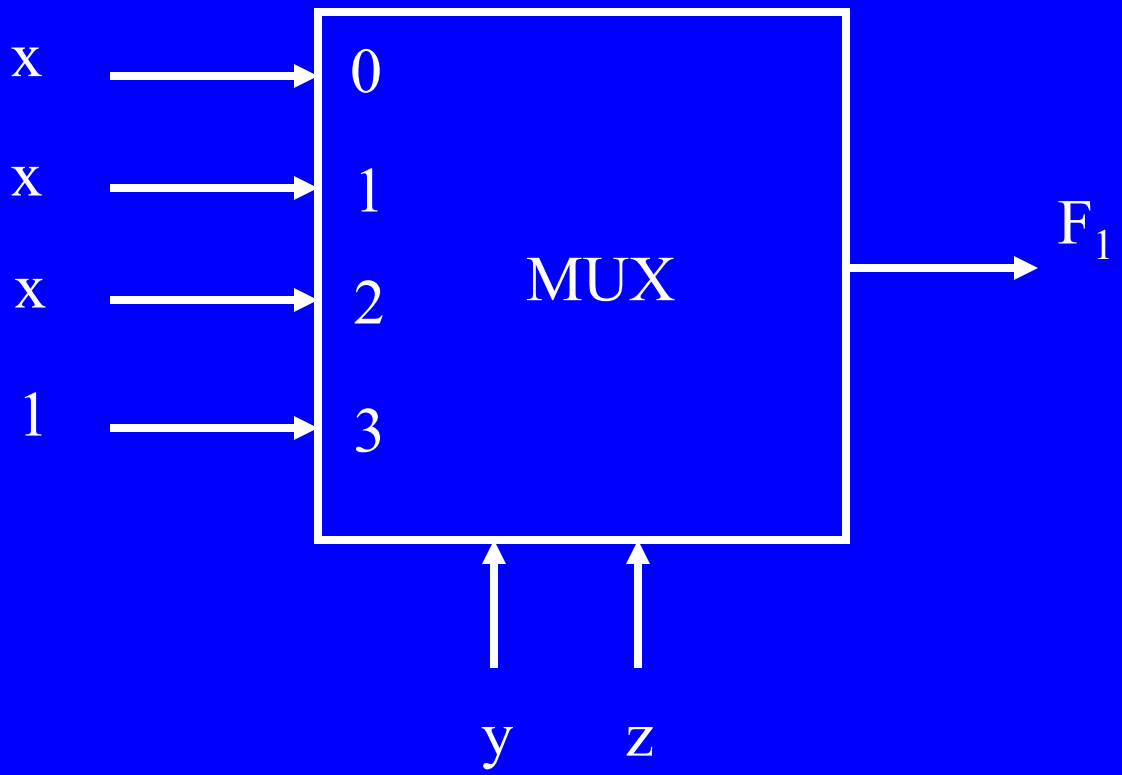
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

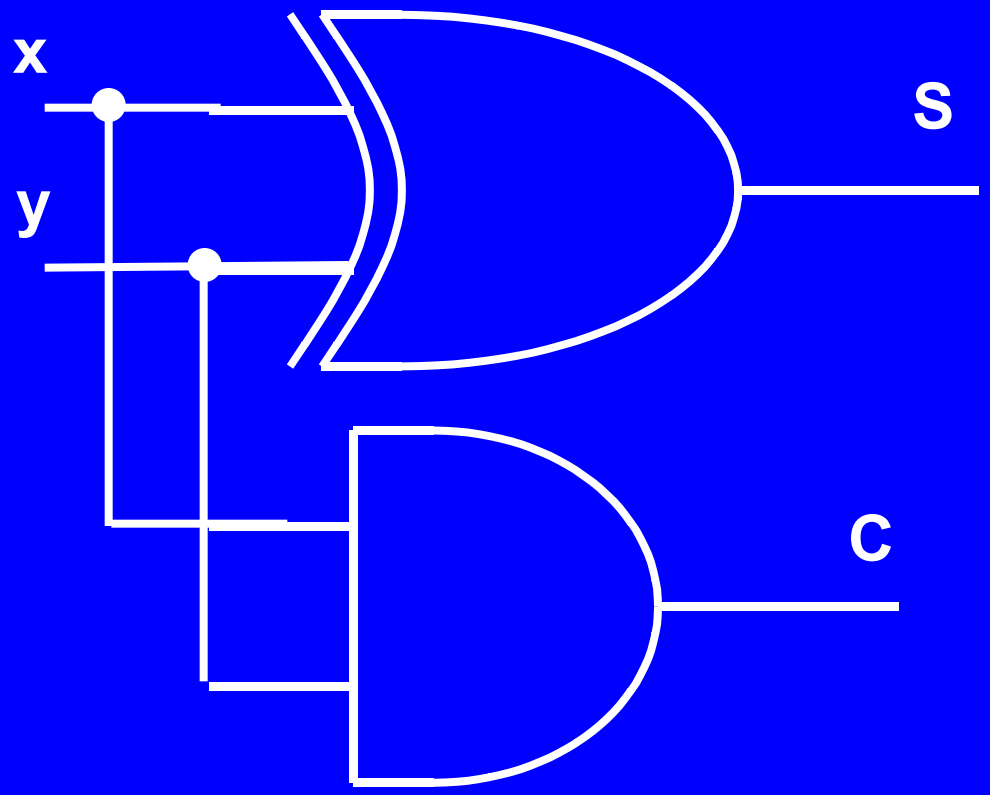
x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Function implementation with multiplexers

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	1
0	0	1	0	0	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	1
1	0	1	1	1	0	0
1	1	0	1	1	0	1
1	1	1	1	1	1	1

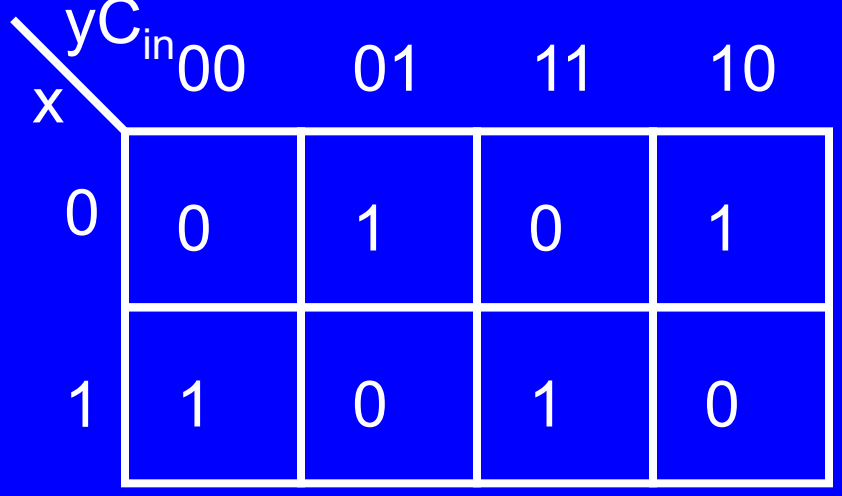

x
x
x
1
0
1
2
3
MUX
F_1
y
z

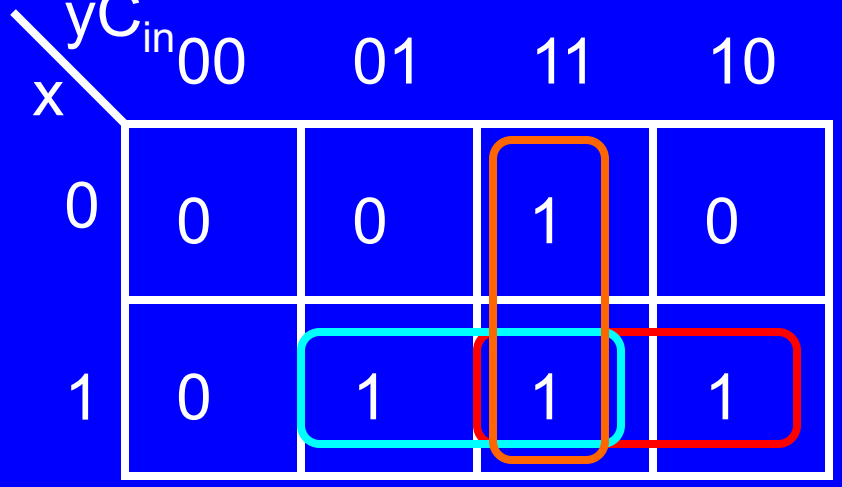
Half adder



x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

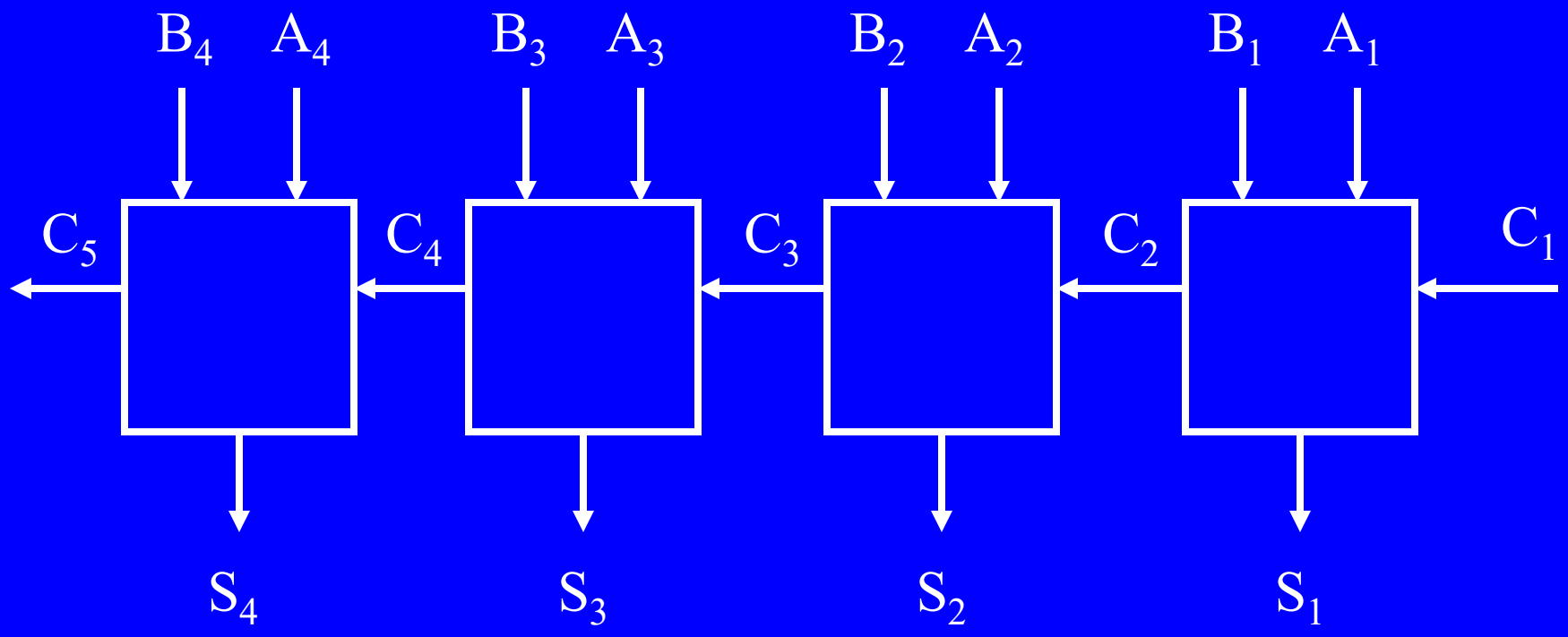
Full adder



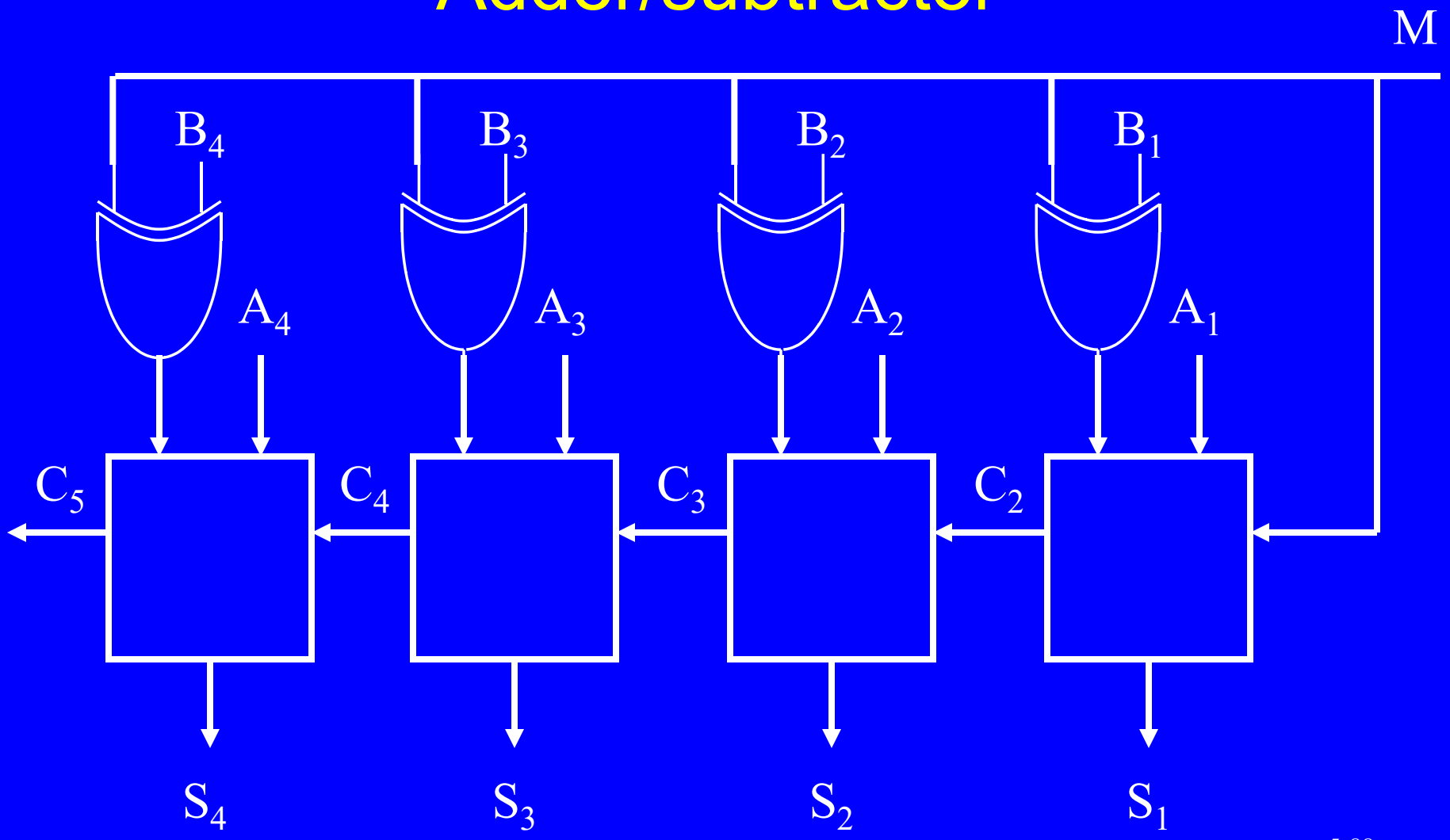


x	y	C_{in}	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Parallel adder



Adder/subtractor



Carry propagation

$P_i = A_i \oplus B_i$

$G_i = A_i B_i$

$S_i = P_i \oplus C_i$

$C_{i+1} = G_i + P_i C_i$

Carry propagation

$P_1 = A_1 \oplus B_1$

$G_1 = A_1 B_1$

$S_1 = P_1 \oplus C_1$

$C_2 = G_1 + P_1 C_1$

Carry propagation

$P_2 = A_2 \oplus B_2$

$G_2 = A_2 B_2$

$S_2 = P_2 \oplus C_2$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 C_2$

Carry propagation

$P_2 = A_2 \oplus B_2$

$G_2 = A_2 B_2$

$S_2 = P_2 \oplus C_2$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 C_1$

Carry propagation

$P_3 = A_3 \oplus B_3$

$G_3 = A_3 B_3$

$S_3 = P_3 \oplus C_3$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 C_1$

$C_4 = G_3 + P_3 C_3$

Carry propagation

$P_3 = A_3 \oplus B_3$

$G_3 = A_3 B_3$

$S_3 = P_3 \oplus C_3$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 C_1$

$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$

Carry propagation

$P_4 = A_4 \oplus B_4$

$G_4 = A_4 B_4$

$S_4 = P_4 \oplus C_4$

$C_2 = G_1 + P_1 C_1$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 C_1$

$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$

$C_5 = G_4 + P_4 C_4$

Carry propagation

$P_4 = A_4 \oplus B_4$

$G_4 = A_4 B_4$

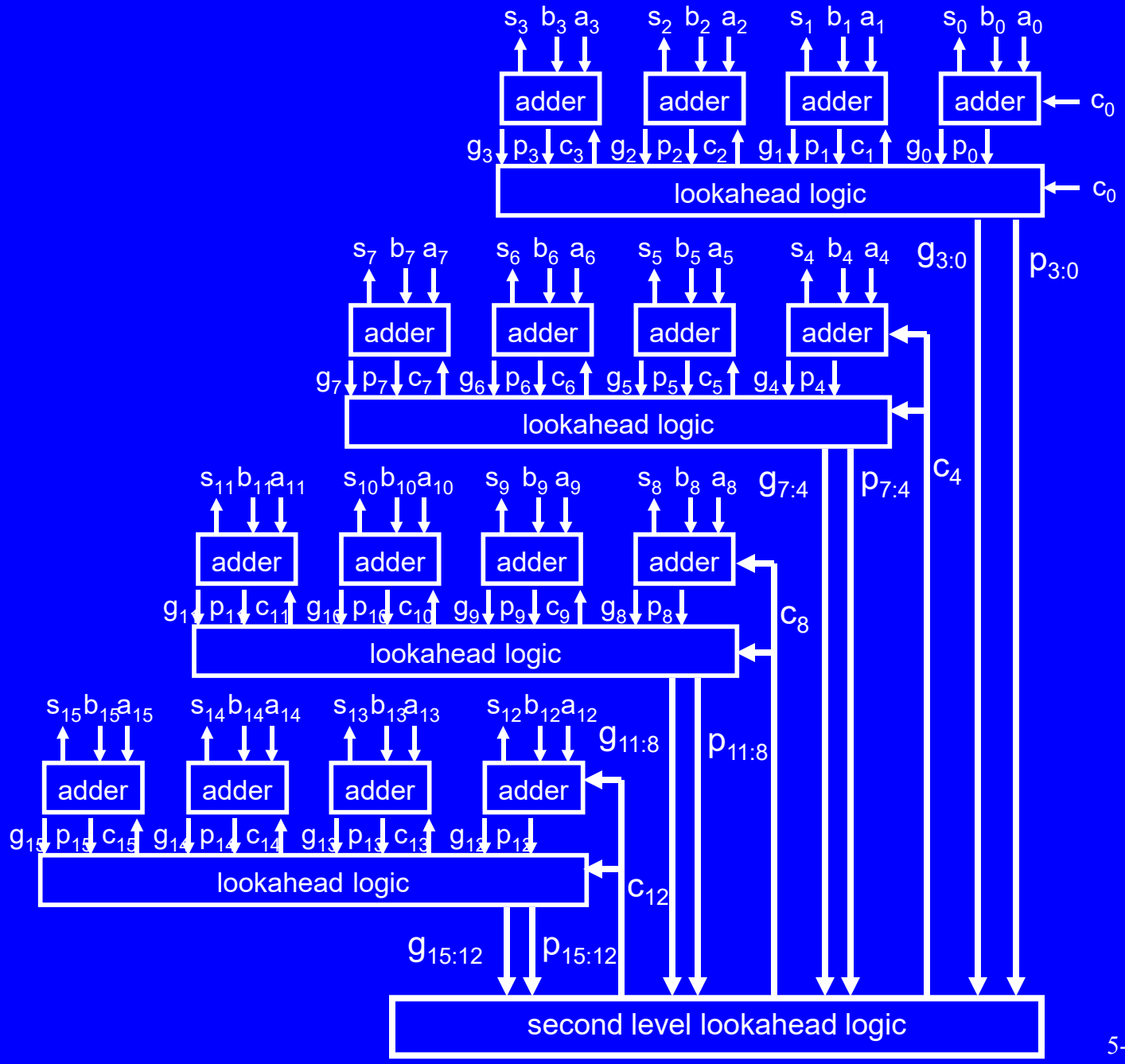
$S_4 = P_4 \oplus C_4$

$C_2 = G_1 + P_1 C_1$

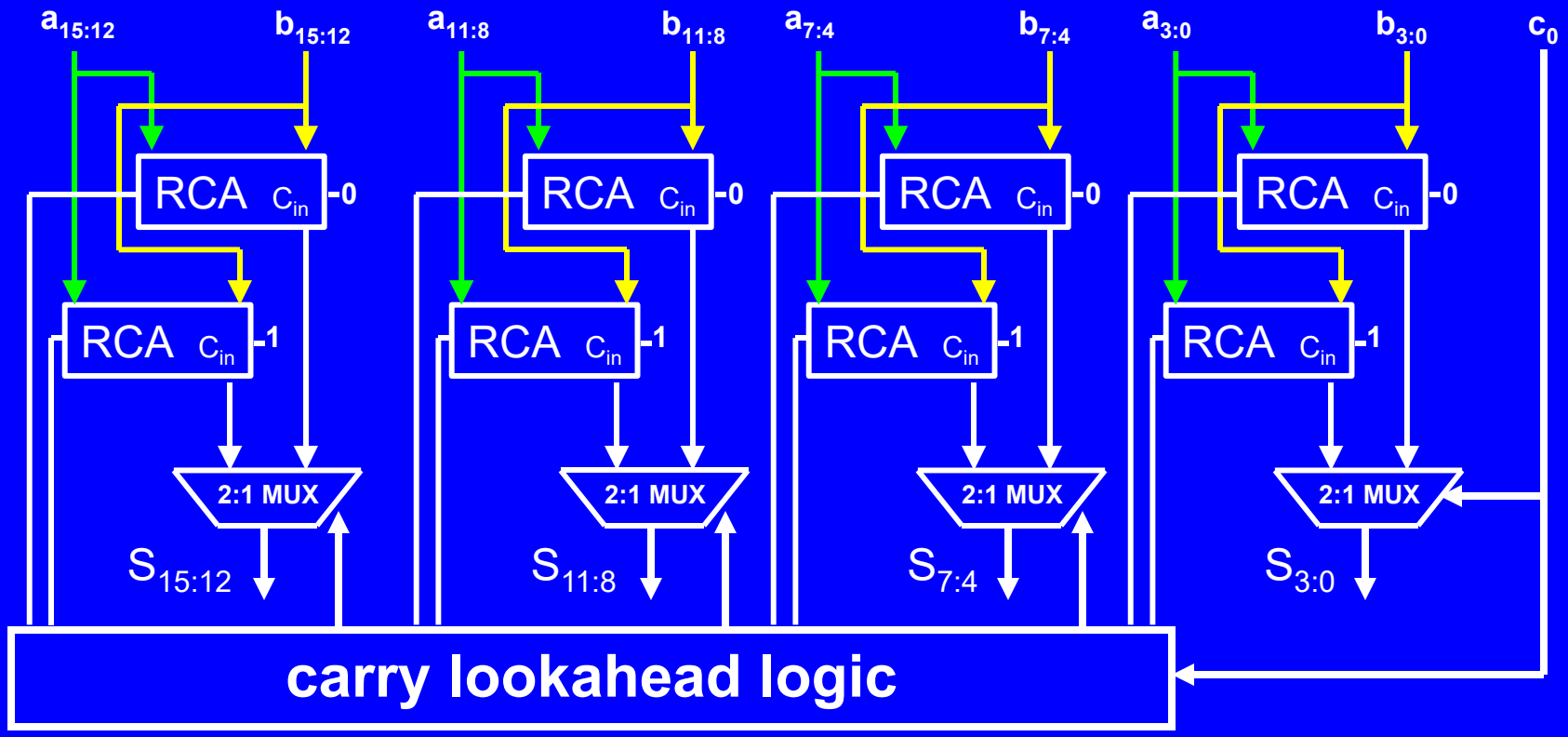
$C_3 = G_2 + P_2 G_1 + P_2 P_1 C_1$

$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$

$C_5 = G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_1$


s3 b3 a3
s2 b2 a2
s1 b1 a1
s0 b0 a0
adder
c0
g3 p3 c3
g2 p2 c2
g1 p1 c1
g0 p0
lookahead logic
c0
s7 b7 a7
s6 b6 a6
s5 b5 a5
s4 b4 a4
g3:0
p3:0
g7 p7 c7
g6 p6 c6
g5 p5 c5
g4 p4
c4
s11 b11 a11
s10 b10 a10
s9 b9 a9
s8 b8 a8
g7:4
p7:4
g11 p11 c11
g10 p10 c10
g9 p9 c9
g8 p8
c8
s15 b15 a15
s14 b14 a14
s13 b13 a13
s12 b12 a12
g11:8
p11:8
g15 p15 c15
g14 p14 c14
g13 p13 c13
g12 p12
c12
g15:12
p15:12
second level lookahead logic

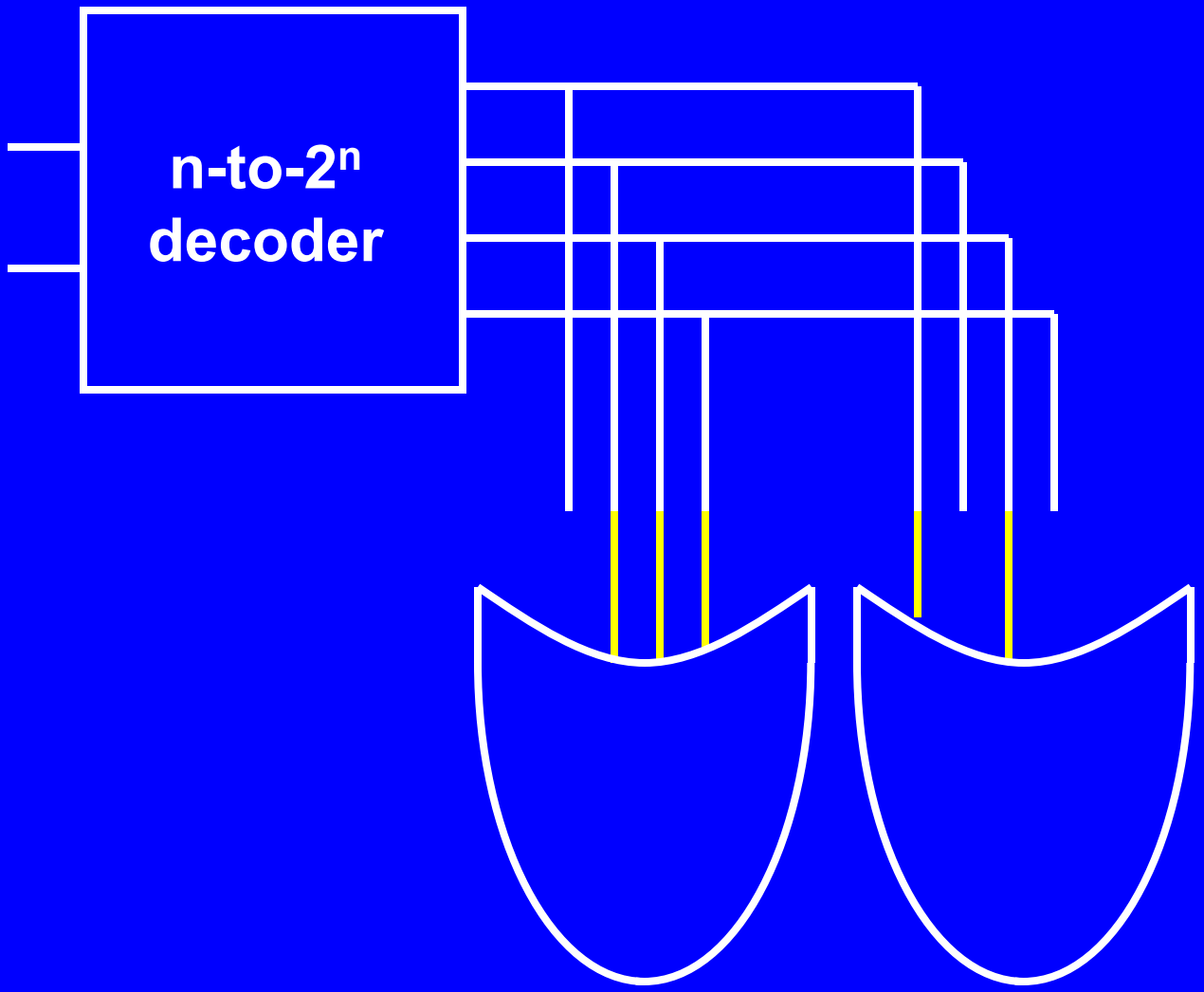
Carry select adder



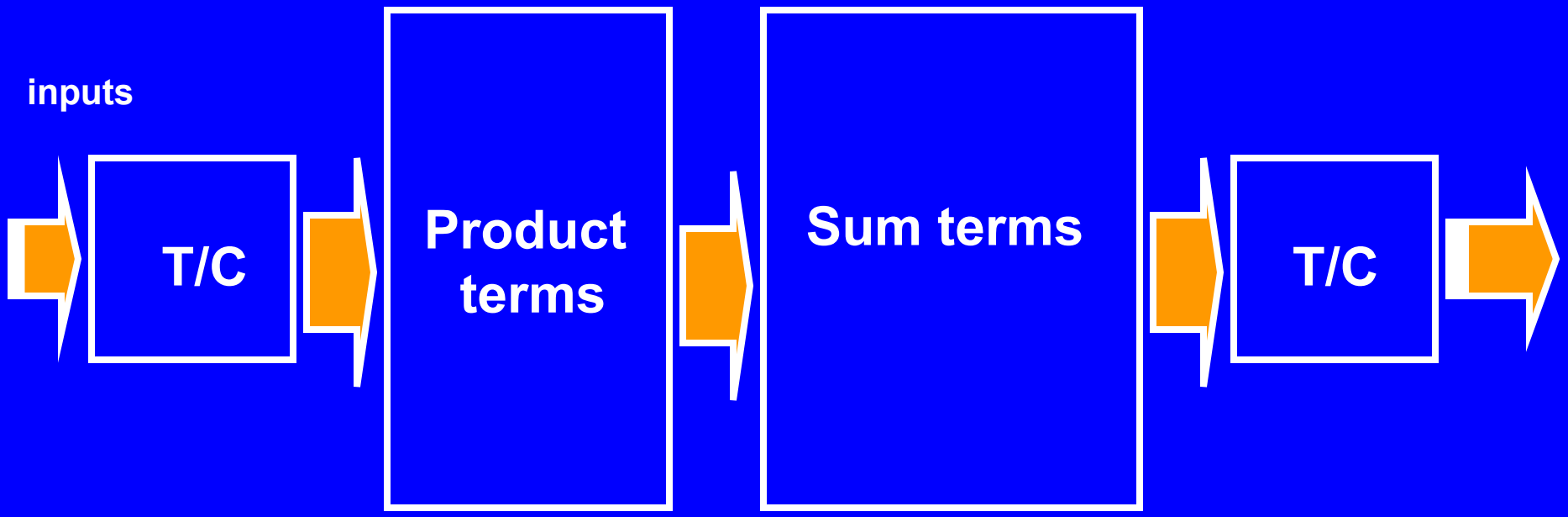
Priority encoder

Inputs				Outputs		
D0	D1	D2	D3	A	B	V
0	0	0	0	X	X	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

Read-Only Memory (ROM)



Programmable Logic Arrays



PLA programming table

Product term	Inputs	Outputs
	A B C	F1 F2
AB' 1	1 0 -	1 -
AC 2	1 - 1	1 1
BC 3	- 1 1	- 1

F1 = AB' + AC
F2 = AC + BC