Presentation Slides

Chapter 4

2NCL Combinational Expression

Logically Determined Design:
Clockless System Design With NULL Convention Logic

by Karl Fant

John Wiley & Sons, Inc.

Introducing NCL circuit synthesis

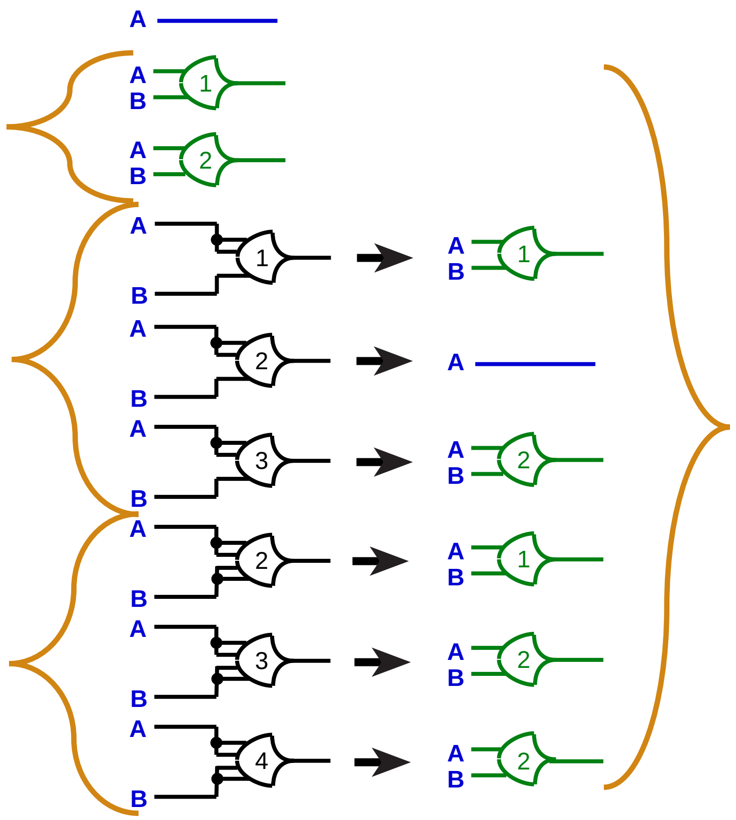
Classifying Threshold Functions

threshold 3 is nonfunctional

threshold 4 is nonfunctional

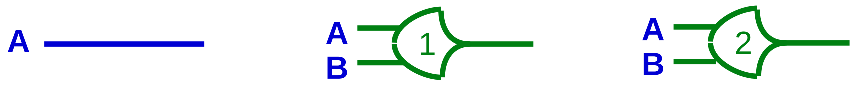
threshold 1 is nonfunctional and
threshold 5 is nonfunctional.

classifying all 2
input functions



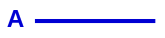
A weight of 3 with two inputs can always be reduced to an expression with a weight of two so there is no need to consider further configurations.

Threshold function classes of one and two inputs.

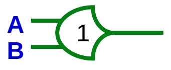


Following a similar procedeure one discovers 5 three input classes and 17 four input classes for 25 classes of 0ne, two, three and four input threshold functions. In the threshold function table functions 1 through 25 represent the 25 classes.

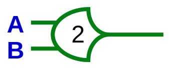
28 Library Operators



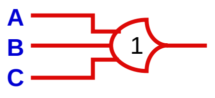
1. A



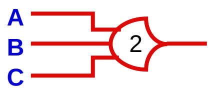
2. A + B



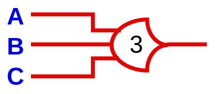
3. AB



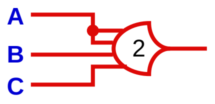
4. A + B + C



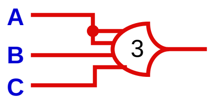
5. AB + BC + AC



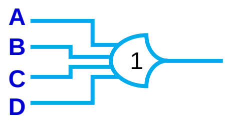
6. ABC



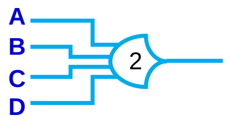
7. A + BC



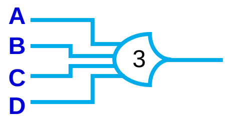
8. AB + AC



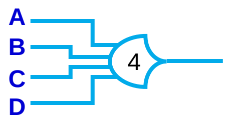
9. A + B + C + D



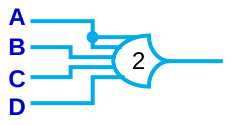
10. AB + AC + AD + BC + BD + CD



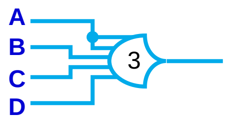
11. ABC + ABD + ACD + BCD



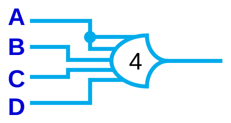
12. ABCD



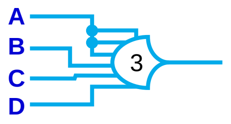
13. A + BC + BD + CD



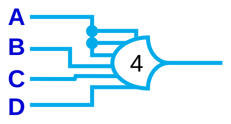
14. AB + AC + AD + BCD



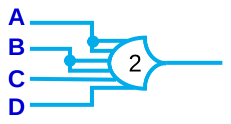
15. ABC + ABD + ACD



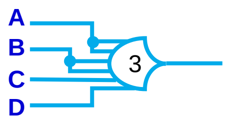
16. A + BCD



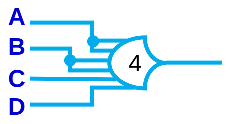
17. AB + AC + AD



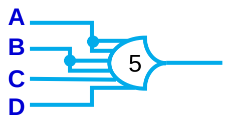
18. A + B + CD



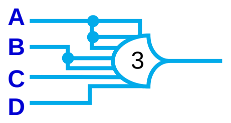
19. AB + AC + AD + BC + BD



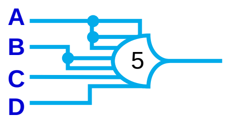
20. AB + ACD + BCD



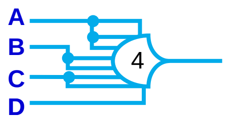
21. ABC + ABD



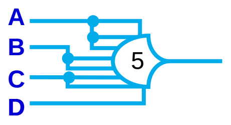
22. A + BC + BD



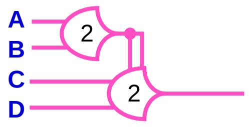
23. AB + ACD



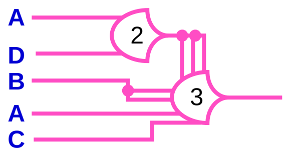
24. AB + AC + AD + BC



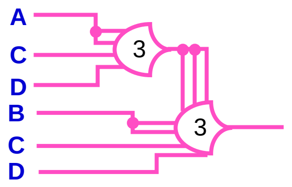
25. AB + AC + BCD



26. AB + CD



27. AB + BC + AD

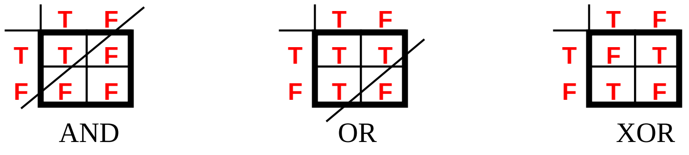


28. AC + BC + AD + BD

The Library Rationale

There is a one to one mapping between the NCL library operators and the classes of unate Boolean functions of 4 or fewer inputs

All threshold functions are linearly separable and map directly to the linearly separable Boolean functions.



All linearly separable Boolean functions are unate but not all unate Boolean functions are linearly separable.

There are 28 PN classes of unate Boolean functions of 4 or fewer variables. 25 of these classes are linearly separable and map directly onto the 25 classes of threshold functions with 4 or fewer inputs. There are three unate four input Boolean functions that are not linearly separable. These are included in the library as operators of two threshold functions (26, 27 and 28).

PN classes
permutation and negation of inputs
unate function classes are red

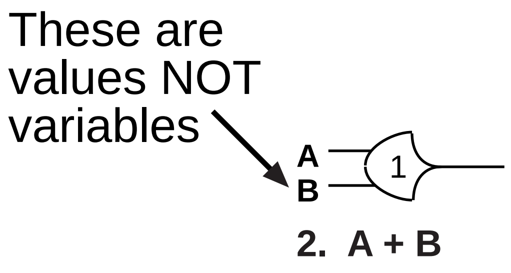
0 variable	M0	0	
	M1	1	
1 variable	M2	x_1	1. A
2 variable	M3	x_1x_2	3. AB
	M4	x_1 V x_2	2. A + B
	M5	$x_1 \oplus x_2 = x_1\bar{x}_2$ V $\bar{x}_1x_2$	
3 variable	M6	x_1x_2 V x_2x_3 V x_1x_3	5. AB + BC + AC
	M7	$x_1 \oplus x_2 \oplus x_3 = x_1(x_2\bar{x}_3$ V $\bar{x}_2x_3)$ V $\bar{x}_1(\bar{x}_2\bar{x}_3$ V $x_2x_3)$	
	M8	$x_1x_2x_3$	6. ABC
	M9	x_1 V x_2 V x_3	4. A + B + C
	M10	$x_1(x_2$ V $x_3)$	8. AB + AC
	M11	x_1 V x_2x_3	7. A + BC
	M12	$x_1x_2x_3$ V $\bar{x}_1\bar{x}_2\bar{x}_3$	
	M13	$(x_1$ V x_2 V $x_3)(\bar{x}_1$ V $\bar{x}_2$ V $\bar{x}_3)$	
	M14	$\bar{x}_1x_2x_3$ V $x_1\bar{x}_2$ V $x_1\bar{x}_3$	
	M15	$x_1(x_2x_3$ V $\bar{x}_2\bar{x}_3)$	
	M16	x_1 V $x_2\bar{x}_3$ V $\bar{x}_2x_3$	
	M17	x_1x_2 V x_2x_3 V $\bar{x}_1x_3$	
	M18	$\bar{x}_1x_2x_3$ V $x_1\bar{x}_2x_3$ V $x_1x_2\bar{x}_3$	
	M19	x_1x_2 V x_2x_3 V x_1x_3 V $\bar{x}_1\bar{x}_2\bar{x}_3$	
	M20	$x_1\bar{x}_2\bar{x}_3$ V x_2x_3	
	M21	$(x_1$ V $\bar{x}_2$ V $\bar{x}_3)(x_2$ V $x_3)$	

Threshold Operations and Boolean Equations

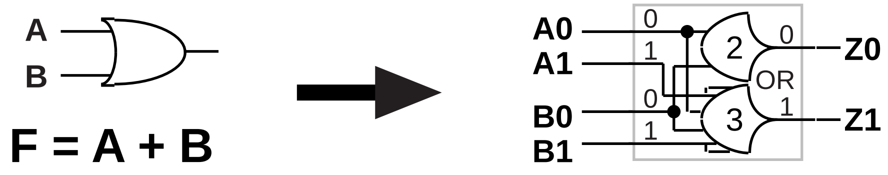
The question for each signal in an NCL expression is when should it transition to DATA. This corresponds to the Boolean Truth function which asks when it should transition to TRUE. Through this correspondence of form it is convenient to specify the transition to DATA function for each NCL signal as a Boolean sum of products.

With the monotonic behavior of the DATA path, there are no transitions from DATA to NULL during a DATA wavefront.There are no signal inversions. Consequently, a Boolean equation characterizing a transition to DATA function is unate.

The Boolean equation associated with each NCL operator is its transition to DATA equation, characterizing how input DATA values combine to produce an output DATA value. This Boolean equation does not fully characterize the behavior of the operator which must include its NULL function and its data holding behavior. Also the NCL operator does not implement its associated Boolean equation in the tradition sense of a function of binary data variables.



2NCL threshold operator #2 and its transition to DATA Boolean equation



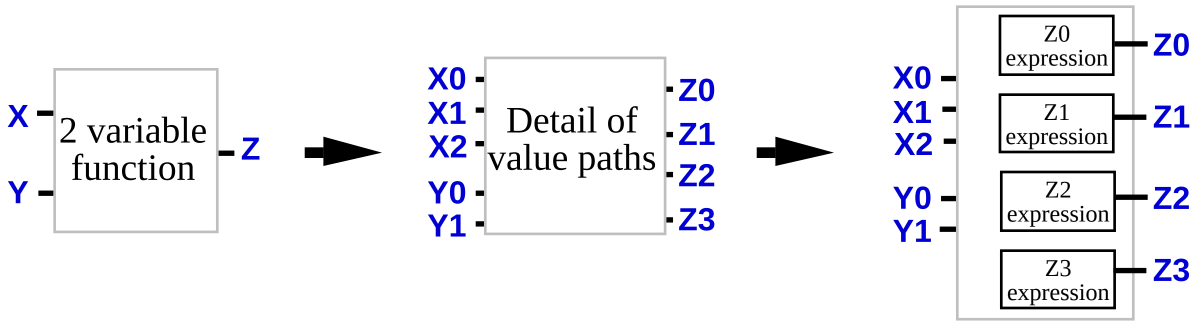
The Boolean function and its 2NCL combinational expression

Boolean Logic as Intermediate Synthesis Language

While an NCL expression as a whole, with multi-value variables, cannot be characterized with a Boolean equation, the transition to DATA of each individual output value can be specified as a unate Boolean expression of the input values.

An expression of multi-value variables is completely characterized by a set unate Boolean equations spanning all combinations of input values.

A threshold circuit can then be constructed by mapping these output equations to library operator equations.



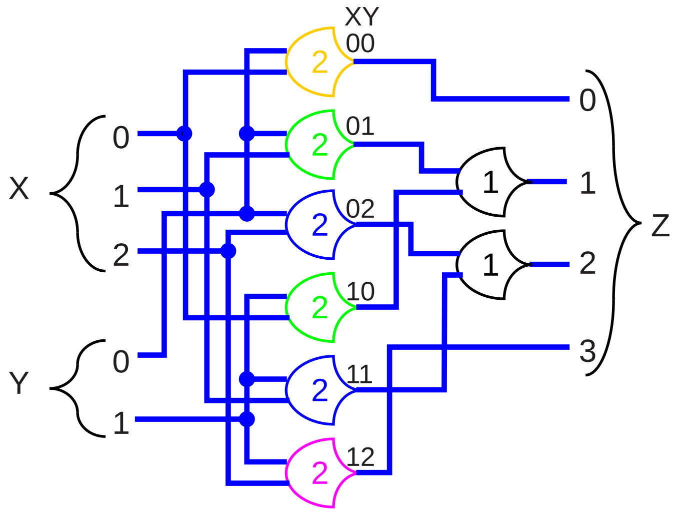
a. Variable level function   b. value level of function   c. internal structure of function

The Boolean equations should not be optimized when mapping to 2NCL operators. As will be shown later, optimizing the Boolean equations before mapping to NCL can greatly confuse the issues of completeness and orphan paths. Optimization must be performed in terms of 2NCL operators after the mapping.

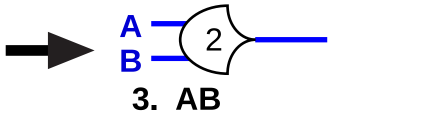
A General Multi-value Logic

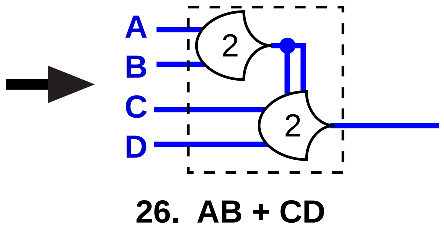
A binary-trinary-quaternary adder function table

	X0	X1	X2
Y0	Z0	Z1	Z2
Y1	Z1	Z2	Z3

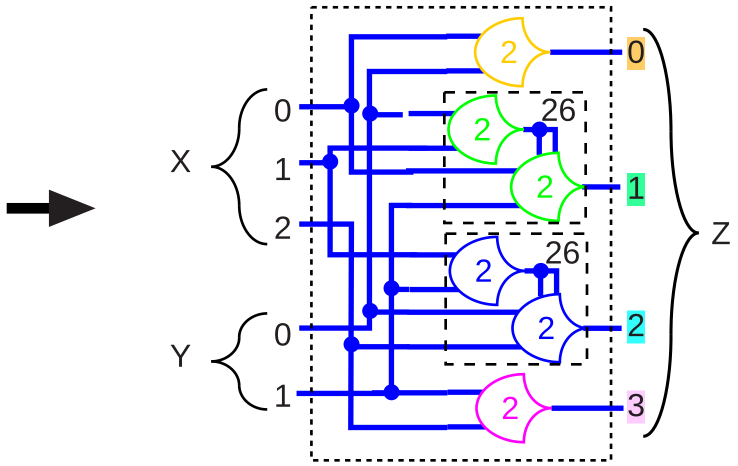


	Specific equation	Generic form
Z0	X0Y0	AB
Z1	X1Y0 + X0Y1	AB + CD
Z2	X2Y0 + X1Y1	AB + CD
Z3	X2Y1	AB





A binary-trinary-quaternary adder circuit



Combines one 2 value variable and one 3 value variable into one 4 value variable

A Logic Function Unit

A function unit that will perform AND, OR, XOR on two binary variables conditional on a command variable.

	C		
XY	A	O	X
00	Z0	Z0	Z0
01	Z0	Z1	Z1
10	Z0	Z1	Z1
11	Z1	Z1	Z0

Output transition to DATA equations

Z0 00 A +00 O + 00 X + 01 A + 10 A + 11 X

Z1 11 A + 11 O + 10 O + 01 O + 10 X + 01 X

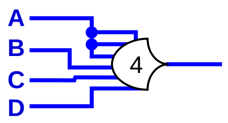
Partitioned output equations

Z0 00 A +00 O + 00 X + 01 A + 10 A + 11 X

Z1 11 A + 11 O + 10 O + 01 O + 10 X + 01 X

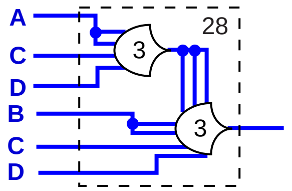
Output subsums — Generic equations

00 A +00 O + 00 X → AB +AC + AD →



17. AB + AC + AD

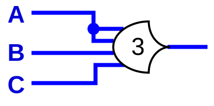
10 O + 01 O + 10 X + 0 X → AC + AD + BC + BD →



28. AC + BC + AD + BD

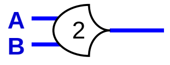
01 A + 10 A → AB + AC

11 A + 11 O → AB + AC



8. AB + AC

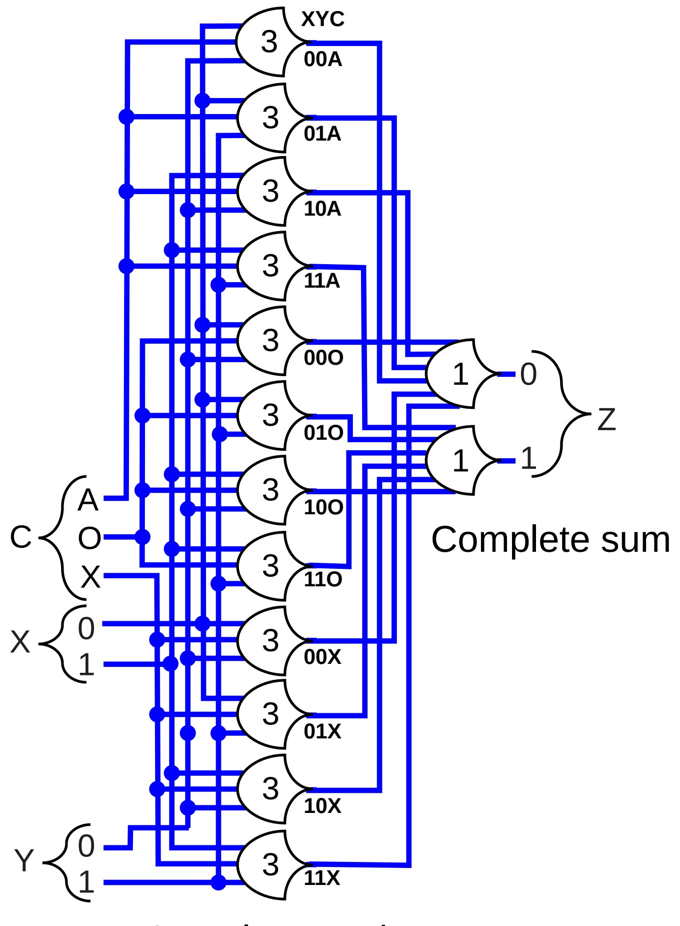
11 X → AB →



3. AB

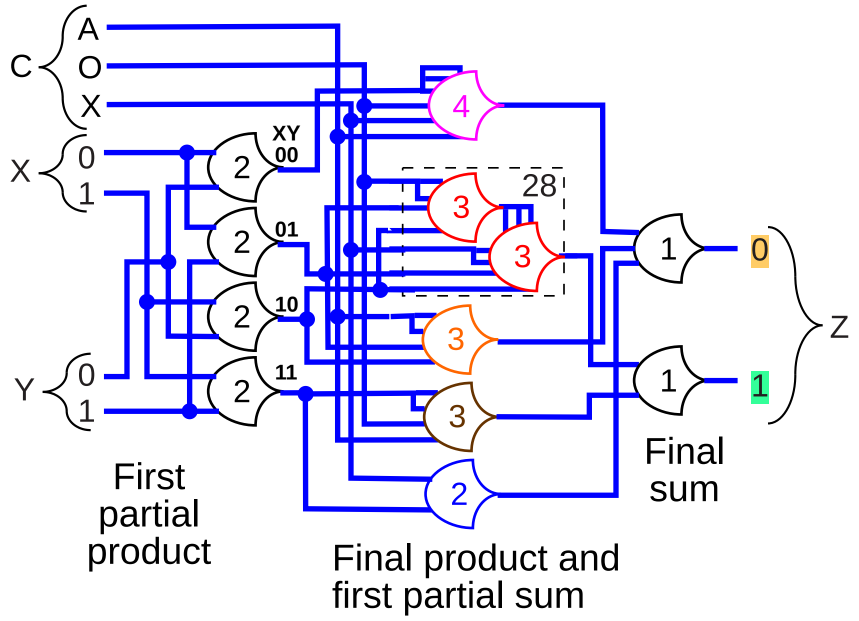
Synthesized Circuits

Minterm of output equations



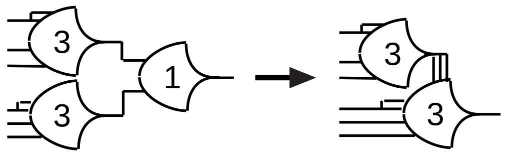
Complete product

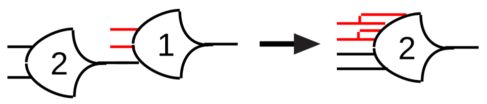
Synthesized from first column

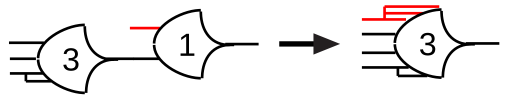


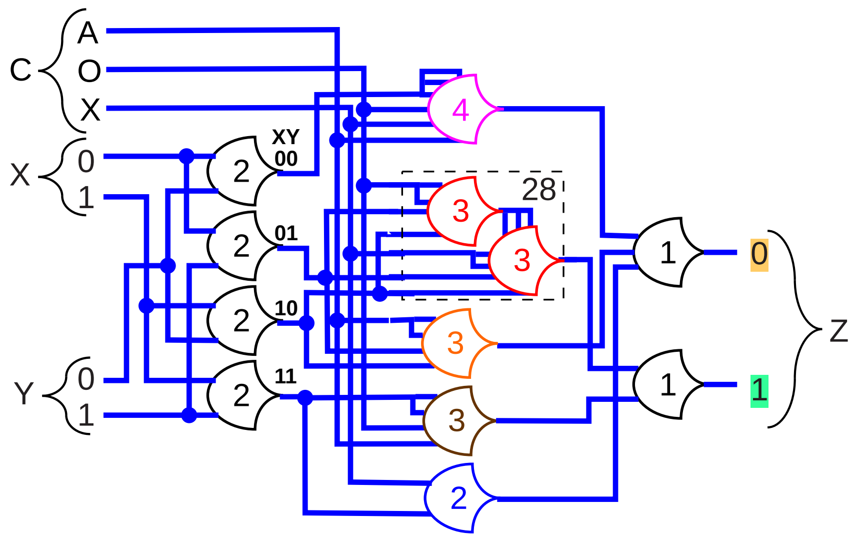
The minterm structure is simple, completeness is fulfilled and orphans are isolated.

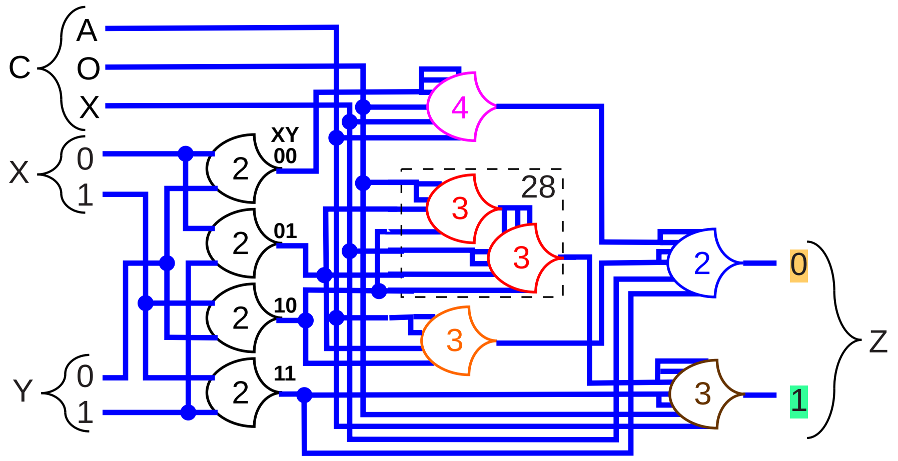
Optimized Circuits





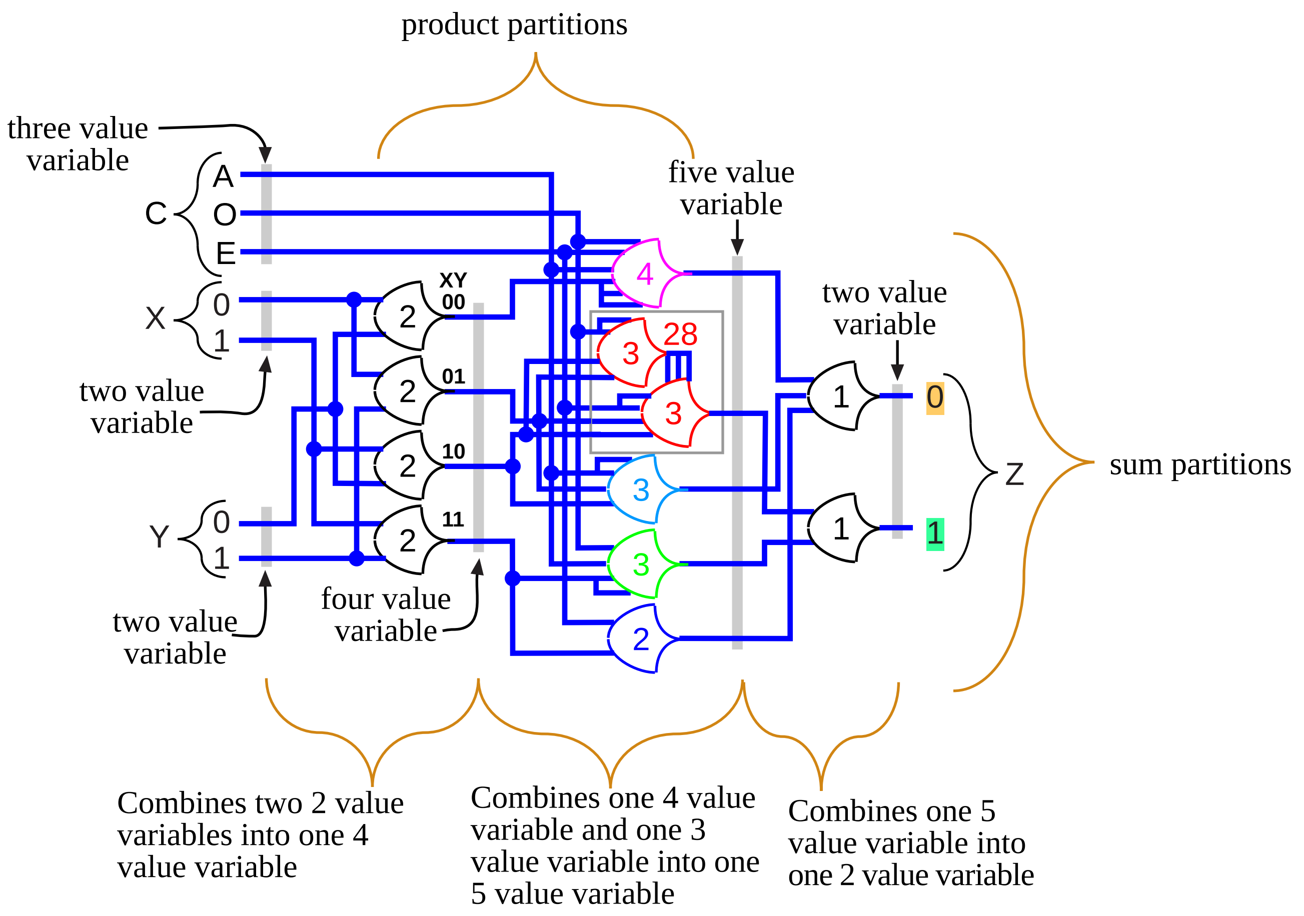






2 less operators, 8 less transistors

Variable Boundaries
Completeness Boundaries



product partitions
three value variable
C
A
O
E
X
0
1
two value variable
Y
0
1
two value variable
XY
00
01
10
11
2
2
2
2
four value variable
4
3
28
3
3
3
2
five value variable
two value variable
1
1
0
1
Z
sum partitions
Combines two 2 value variables into one 4 value variable
Combines one 4 value variable and one 3 value variable into one 5 value variable
Combines one 5 value variable into one 2 value variable

Progressive logically determined variable boundaries. Boundaries of completeness behavior and boudaries isolating orphan paths.

Twos Complement 8 Bit to 6 Bit Clipper

If sign bit 7 is 1 and either bit 5 or bit 6 is 0 then the 8 bit number is a negative number larger that 6 bits. Force to smallest negative 6 bit number (magnitude all zeros).

If sign bit 7 is 0 and either bit 5 or bit 6 is 1 then the 8 bit number is a positive number larger that 6 bits. Force to largest positive 6 bit number (magnitude all ones).

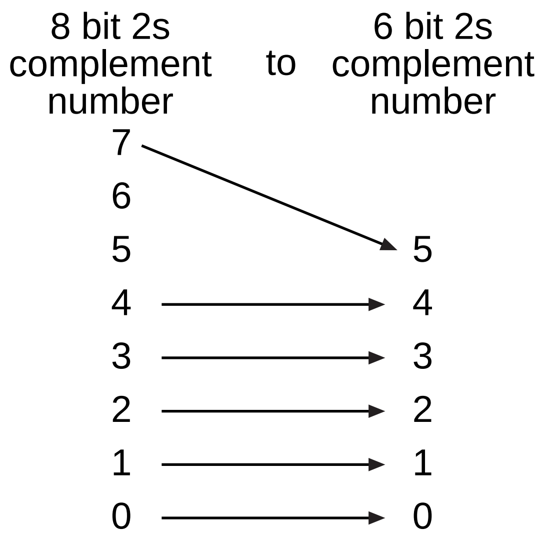
If sign bit 7 and bits 5 and bit 6 match then the magnitude of the number is within 6 bits. Pass the magnitude bits.

3 two value variables to one 3 value variable

	70		71	
	60	61	61	60
50	pass input 50,60,70	force ones 50,61,70	force zeros 50,61,71	force zeros 50,60,71
51	force ones 51,60,70	force ones 51,61,70	pass input 51,61,71	force zeros 51,60,71

Output transition to DATA equations

pass	**706050 + 716151**
force zeros	**715160 + 715060 + 715061**
force ones	**705160 + 705161 + 705061**

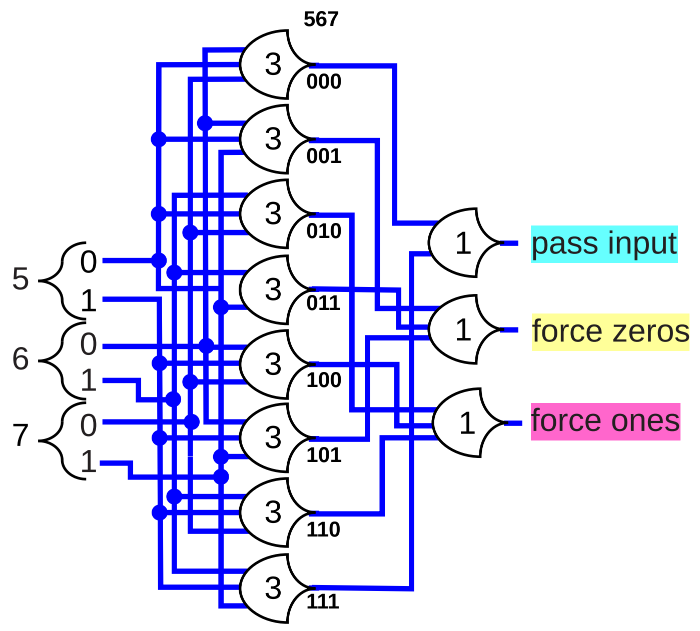


Sum Partitioned Synthesis

	70		71	
50	pass input 50,60,70	force ones 50,61,70	force zeros 50,61,71	force zeros 50,60,71
51	force ones 51,60,70	force ones 51,61,70	pass input 51,61,71	force zeros 51,60,71
	60	61		60

Output transition to DATA equations

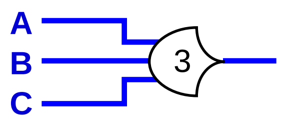
pass	**706050 + 716151**
force zeros	**715160 + 715060 + 715061**
force ones	**705160 + 705161 + 705061**



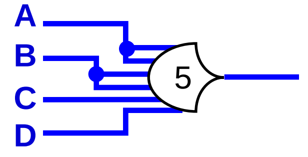
11 operators 150 transistors

Specific equations		Generic equations
716151		**ABC**
706050	→	**ABC**
715160		**ABC**
705061		**ABC**
715060 + 715061	→	**ABC + ABD**
705160 + 705161		**ABC + ABD**

→



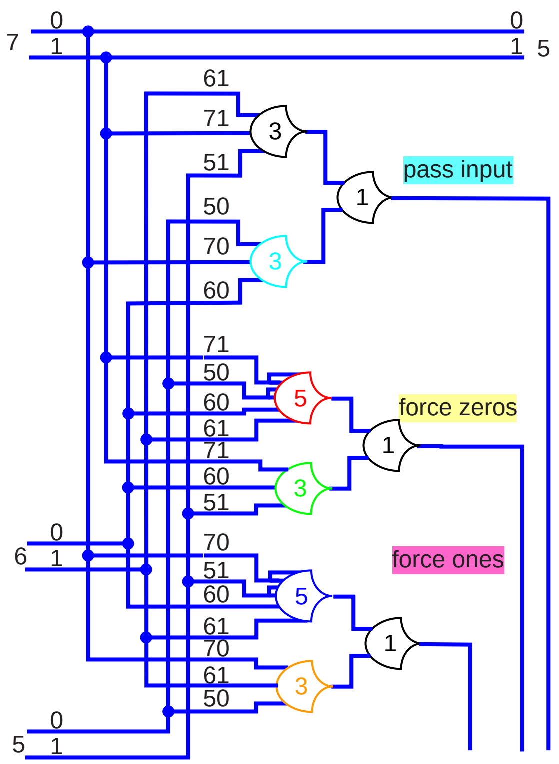
6. ABC



21. ABC + ABD

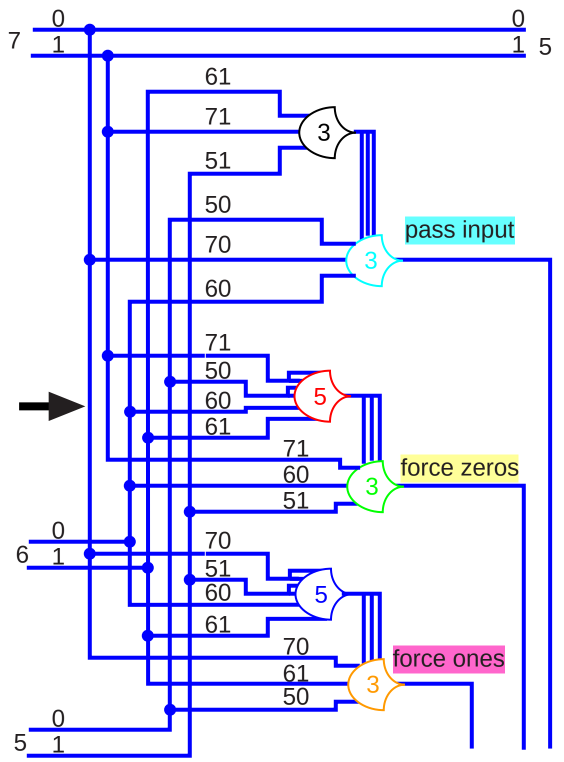
Sum Partitioned Synthesis

pass	706050 + 716151
force zeros	715160 + 715060 + 715061
force ones	705160 + 705161 + 705061



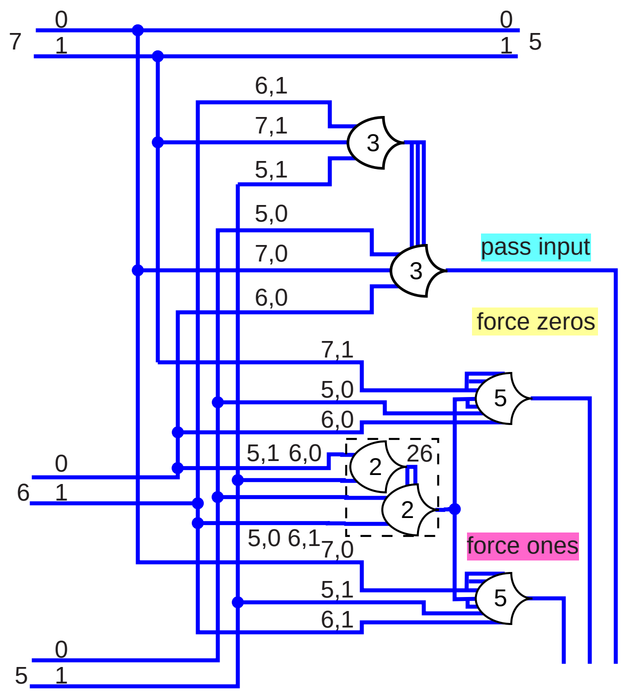
directly mapped expression

9 operators 118 transistors



merged operators

6 operators 100 transistors



mutually exclusive products sharing a single signal

5 operators 109 transistors

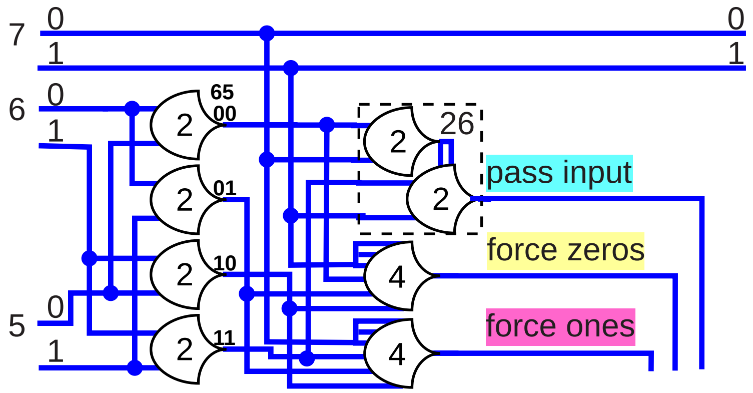
Product Partitioned Expression

Combine bit 5 and bit 6

	60	61
50	(00)	(10)
51	(01)	(11)

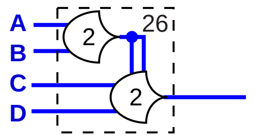
Combine bit 7

	70	71
(00)	pass input	force zeros
(01)	force ones	force zeros
(10)	force ones	force zeros
(11)	force ones	pass input

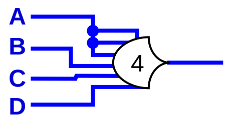


Expression uses
7 operators 104 transistors

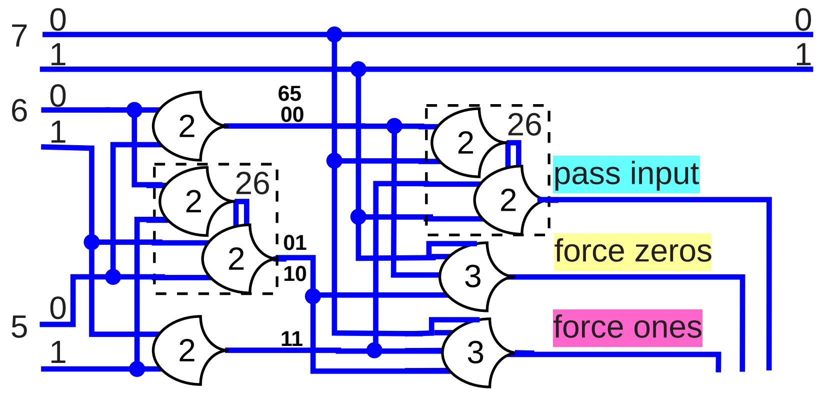
	Specific equations	Generic equations
pass	**70(00) + 71(11)**	**AB + CD**
force zeros	**71(10) + 71(00) + 71(01)**	**AB + AC + AD**
force ones	**70(10) + 70(11) + 70(01)**	**AB + AC + AD**



26. AB + CD



17. AB + AC + AD



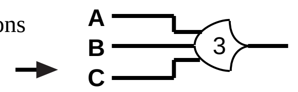
Common factor
6 operators 96 transistors

Do Not Optimize Boolean Expression

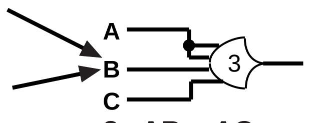
	70		71	
50	pass input 3(70,60,50)	force ones 3(70,61,50)	force zeros 3(71,61,50)	force zeros 3(71,60,50)
51	force ones 3(70,60,51)	force ones 3(70,61,51)	pass input 3(71,61,51)	force zeros 3(71,60,51)
	60	61	61	60

Optimizing the Boolean expression before mapping to NCL destroys the completeness relations and orphan isolation. It can be very difficult fo restore completeness behavior and insure orphan isolation of an expression mapped from the optimized optimized expressions.

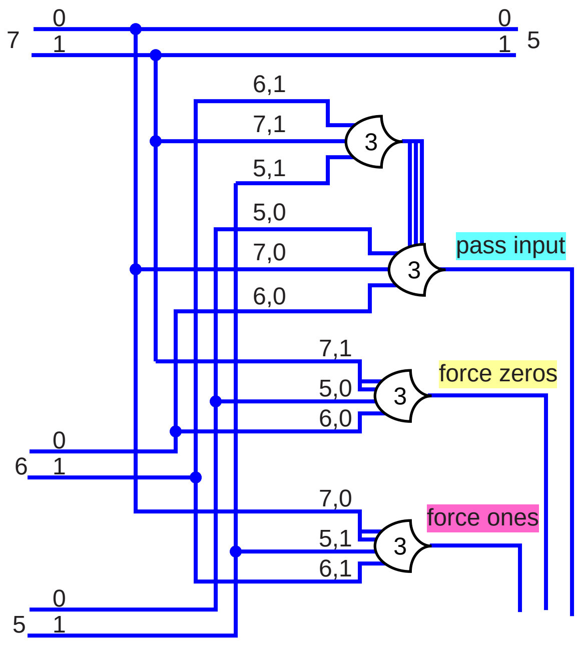
	Specific equations	minimized equations	Generic equations
pass	**706050 + 716151**	**706050 + 716151**	**ABC + ABC**
force zeros	**715160 + 715060 + 715061**	**71(50 +60)**	**A(B + C)**
force ones	**705160 + 705161 + 705061**	**70(51 + 61)**	**A(B + C)**



6. ABC

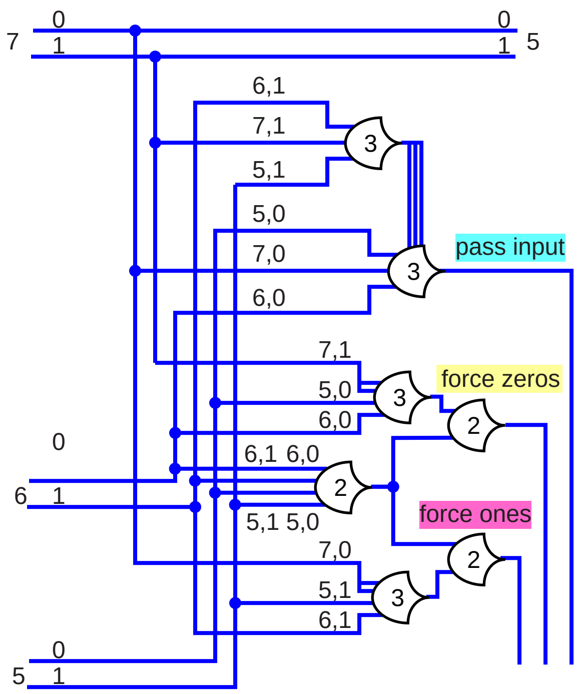


8. AB + AC



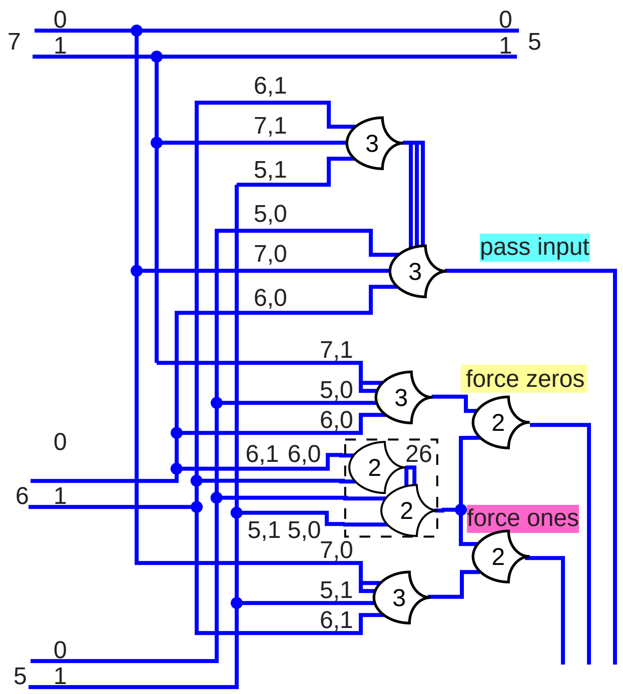
Behavior is not complete.

4 operators 77 transistors



New operator makes behavior complete but will respond to 5,0 6,0 and 5,1 6,1 creating an operator orphan with pass input.

7 operators 118 transistors



Behavior is complete and orphans are isolated.

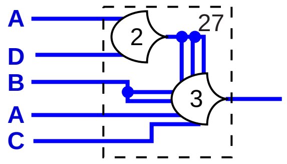
7 operators 111 transistors

Clipper Data Path

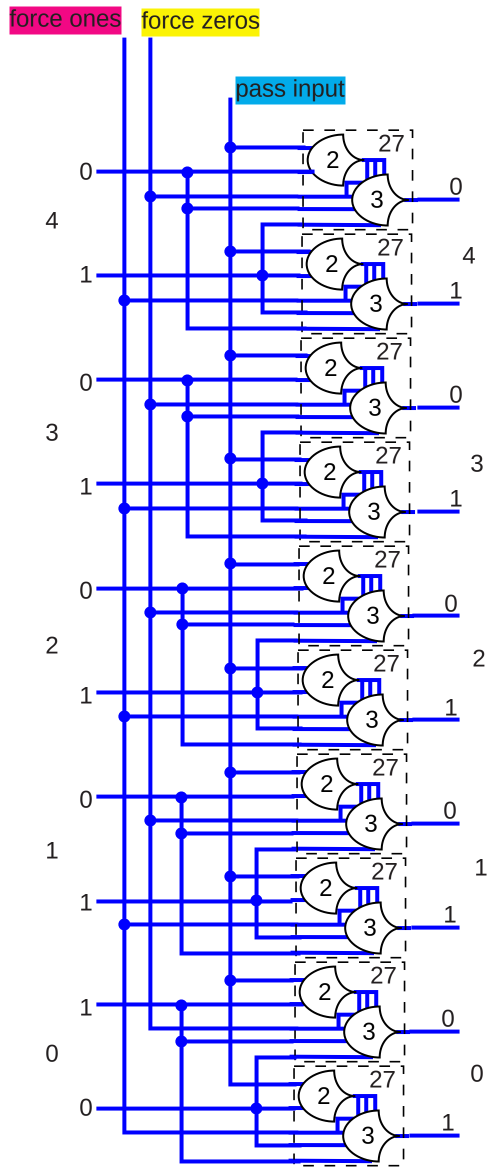
	pass input P	force zeros Z	force ones O
0	0	0	1
1	1	0	1

	Transition to DATA equations	Generic equations
0	Z0 + Z1 + P0	AB + AD + CB
1	O0 + O1 + P1	AB + AD + CB

→



27. AB + BC + AD



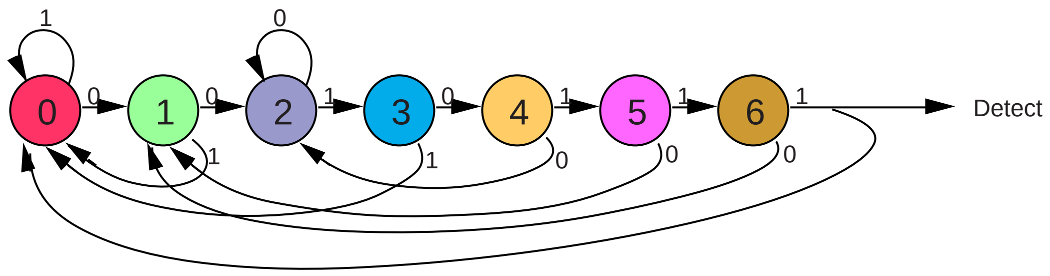
3 value variable and 2 value variable to 2 value variable

10 operators 200 transistors

Code Detector State Machine

Detect the sequence 0010111 in a continuous bit stream

seven states S1 thru S6 all states output No-Detect except S6 which outputs Detect

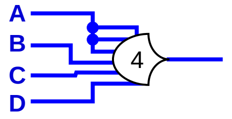


a. state machine diagram

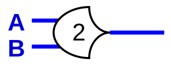
	state 0	state 1	state 2	state 3	state 4	state 5	state 6
0	to state 1	to state 2	to state 2	to state 4	to state 2	to state 1	to state 1
1	to state 0	to state 0	to state 3	to state 0	to state 5	to state 6	to state 0 Detect

b. Next state function map

	Specific equations	Generic equations
state 0	1S0 + 1S1 + 1S3 + 1S6	AB + AC + AD + AB
state 1	0S0 + 0S5 + 0S6	AB + AC + AD
state 2	0S1 + 0S2 + 0S4	AB + AC + AD
state 3	1S2	AB
state 4	0S3	AB
state 5	1S4	AB
state 6	1S5	AB
detect	1S6	
no detect	everything else	



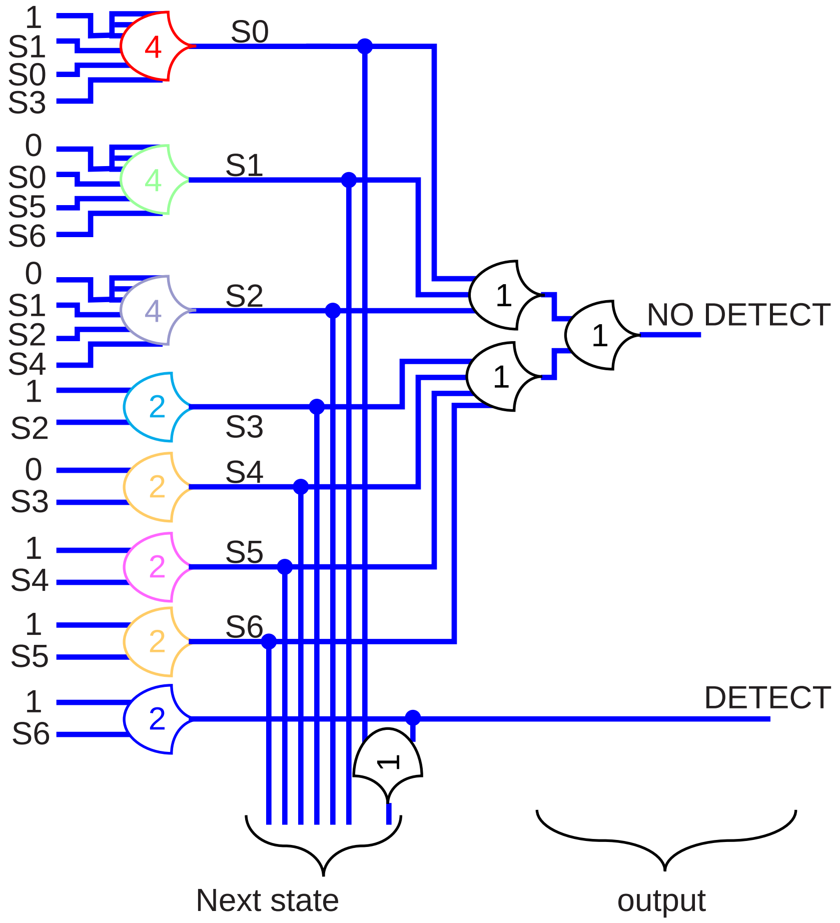
17. AB + AC + AD



3. AB

7 value variable and 2 value variable to 7 value variable and 2 value variable

Code Detector State Machine Circuits

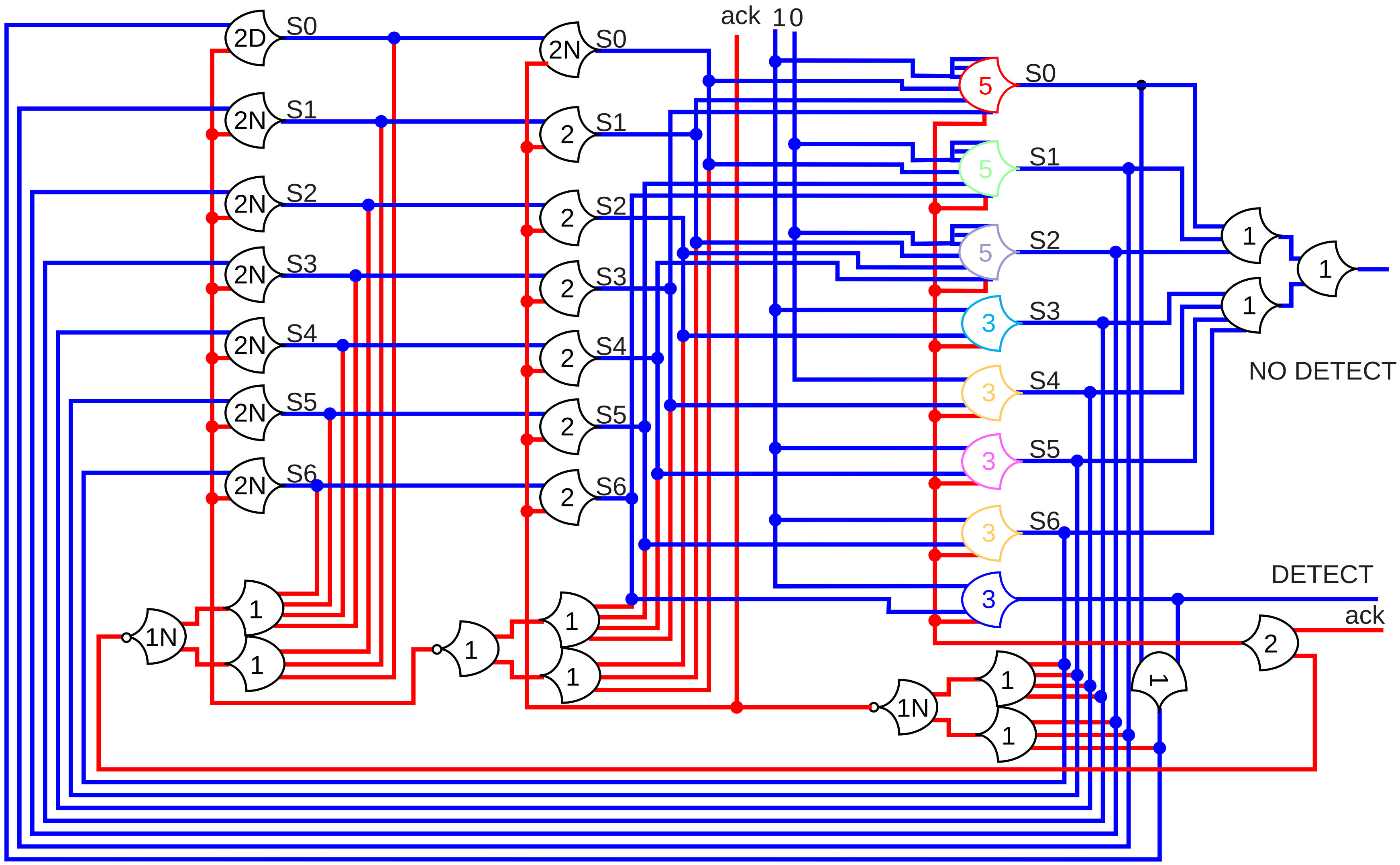


12 operators 138 transistors

Code Detector State Machine

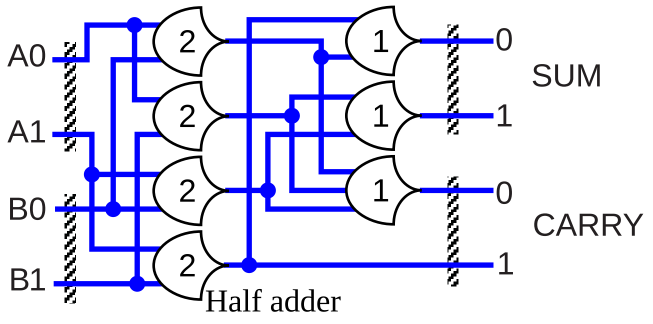
The state is maintained in a three cycle ring. The state update logic is in the rightmost rank of threshold 3 gates with integrated cycle coordination.

The circuit also shows the initialization operators needed to initialize the circuit.

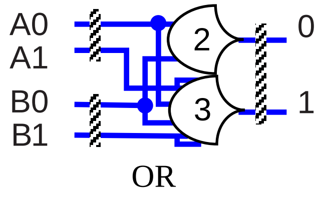


Composition of Combinational Circuits

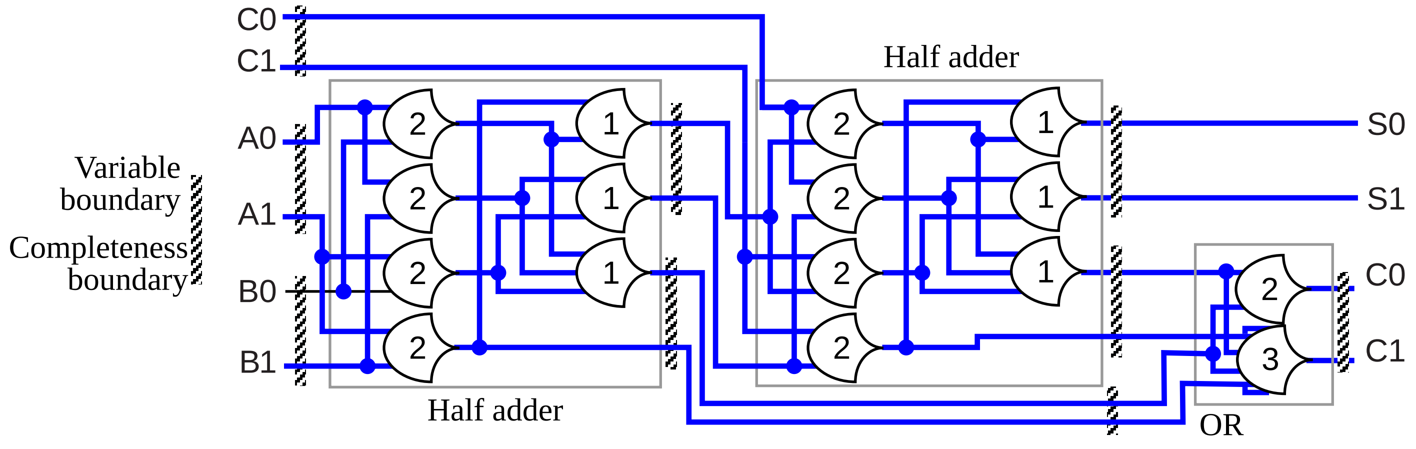
NCL combinational circuits are composed at variable boundaries (completeness boundaries)



2 binary variables to 2 binary variables



2 binary variables to 2 binary variables



Full adder composed of two half adders and OR.

3 binary variables to 2 binary variables

This page intentionally blank

This page intentionally blank

This page intentionally blank