Discrete Structures

Rules of Inference application
Circuits
9/13/2022

Review

Modus Ponens	Modus Tollens	Generalization
$p \Rightarrow q$ p $\therefore \quad q$	$p \Rightarrow q$ $\sim q$ $\therefore \quad \sim p$	p $\therefore \quad p \vee q$
Specialization	**Conjunction**	**Elimination**
$p \wedge q$ $\therefore \quad p$	p q $\therefore \quad p \wedge q$	$p \vee q$ $\sim p$ $\therefore \quad q$
Transitivity	**Cases**	**Contradiction**
$p \Rightarrow q$ $q \Rightarrow r$ $\therefore \quad p \Rightarrow r$	$p \vee q$ $p \Rightarrow r$ $q \Rightarrow r$ $\therefore \quad r$	$\sim p \Rightarrow 0$ $\therefore \quad p$
Dilema		
$(p \Rightarrow q) \wedge (r \Rightarrow s)$ $(p \vee r)$ $\therefore \quad (q \vee s)$		

Argument Valid/Invalid

If Zeke is a cheater, then Zeke sits in the back row.

Zeke sits in the back row.

∴ Zeke is a cheater.

Argument Valid/Invalid

If Zeke is a cheater, then Zeke sits in the back row.	$p \Rightarrow q$
Zeke sits in the back row.	q
∴ Zeke is a cheater.	$\therefore p$

Argument Valid/Invalid

If Zeke is a cheater, then Zeke sits in the back row.	$p \Rightarrow q$
Zeke sits in the back row.	q
∴ Zeke is a cheater.	$\therefore p$
	invalid

Argument Valid/Invalid

If these two vertices are adjacent, then they do not have the same color.

These two vertices are not adjacent.

∴ These two vertices have the same color.

Argument Valid/Invalid

If these two vertices are adjacent, then they do not have the same color.

These two vertices are not adjacent.

∴ These two vertices have the same color.

$p \Rightarrow q$

$\neg p$

$\therefore \neg q$

invalid

$p \Rightarrow q$

p : these two vertices are adjacent

q : they do not have the same color

¬p : these two vertices are not adjacent

¬q : These two vertices have the same color.

Argument Valid/Invalid

If Canada is north of the United States, then temperatures in Canada never rise above freezing.

Canada is north of the United States..

∴ Temperatures in Canada never rise above freezing.

$p \Rightarrow q$

p

$\therefore q$

$p \Rightarrow q$

p : Canada is north of the United States

q : temperatures in Canada never rise above freezing

Argument Valid/Invalid

If New York is a big city, then New York has tall buildings

New York has tall buildings.

∴ New York is a big city.

$p \Rightarrow q$

q

$\therefore p$

p : New York is a big city

q : New York has tall buildings

Soundness of an argument

An argument is called **sound** if, and only if, it is **valid** and all its premises are **true**.

An argument that is not **sound** is called **unsound**.

Validity is a *necessary condition* for **soundness**.

True Premises is a *necessary condition* for **soundness**.

Soundness is a *sufficient condition* for **validity**.

Soundness is a *sufficient condition* for **true premises**.

Example Argument Form

$p \vee q$

$q \Rightarrow r$

$p \wedge s \Rightarrow t$

$\neg r$

$\neg q \Rightarrow u \wedge s$

$\therefore t$

Example Argument Form

$p \lor q$	*premise*
$q \Rightarrow r$	*premise*
$p \land s \Rightarrow t$	*premise*
$\neg r$	*premise*
$\neg q \Rightarrow u \land s$	*premise*

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*

$\therefore t$

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*

$\therefore t$

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)

$\therefore t$

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)

$\therefore t$

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \land s$	Modus ponens (5,6)

$\therefore t$

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \land s$	Modus ponens (5,6)

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \wedge s$	Modus ponens (5,6)
(9)	s	Specialization (8)

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \wedge s$	Modus ponens (5,6)
(9)	s	Specialization (8)

$\therefore t$

Example Argument Form

(1)	$p \vee q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \wedge s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \wedge s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \wedge s$	Modus ponens (5,6)
(9)	s	Specialization (8)
(10)	$p \wedge s$	Conjunction (7,9)
	$\therefore t$	

Example Argument Form

(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \land s$	Modus ponens (5,6)
(9)	s	Specialization (8)
(10)	$p \land s$	Conjunction (7,9)
(11)	$\therefore t$	

Example Argument Form

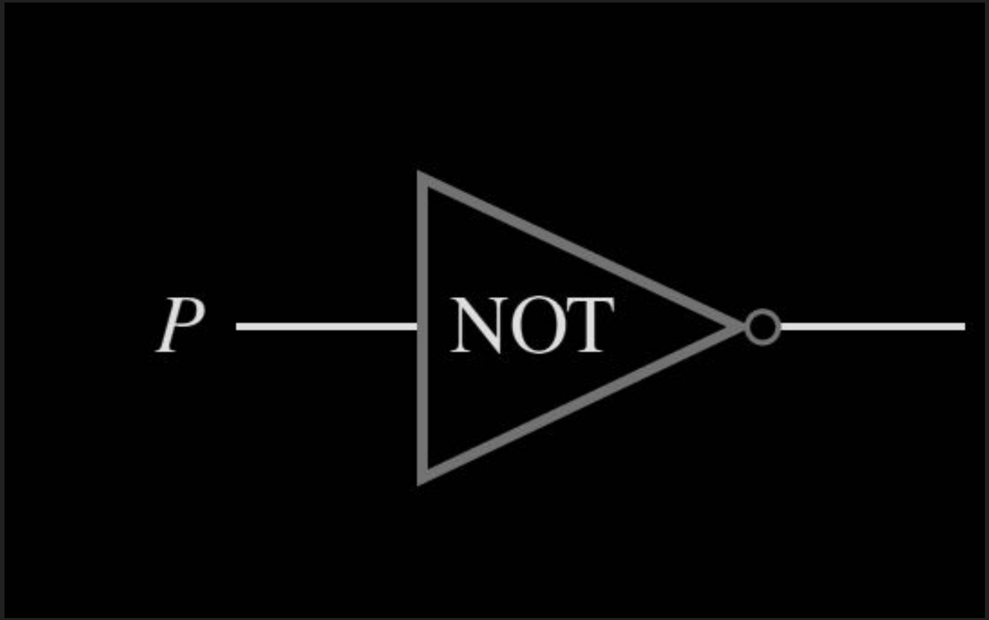
(1)	$p \lor q$	*premise*
(2)	$q \Rightarrow r$	*premise*
(3)	$p \land s \Rightarrow t$	*premise*
(4)	$\neg r$	*premise*
(5)	$\neg q \Rightarrow u \land s$	*premise*
(6)	$\neg q$	Modus tollen (2,4)
(7)	p	Elimination (1,6)
(8)	$u \land s$	Modus ponens (5,6)
(9)	s	Specialization (8)
(10)	$p \land s$	Conjunction (7,9)
(11)	$\therefore t$	Modus Ponens (10,3)

Topics Review

- Arguments vs Argument forms
- Laws of Inference
- Argument Validity
- Argument Soundness

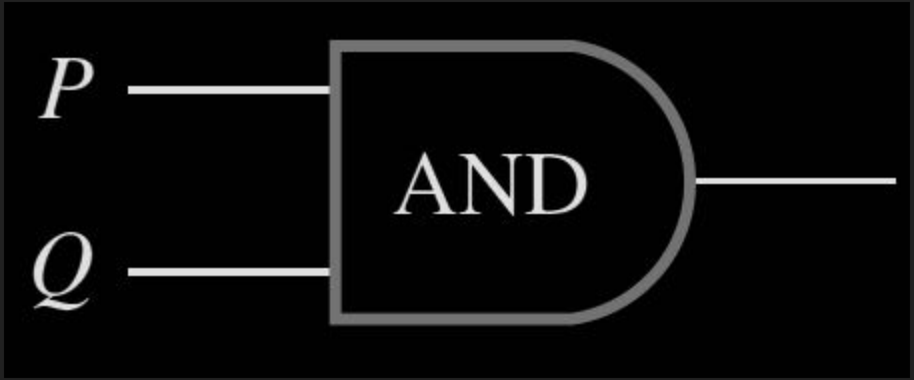
Circuits

p	¬ p
0	1
1	0



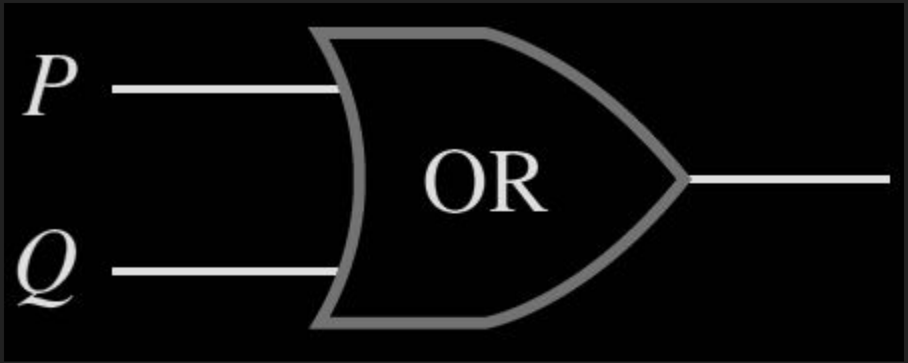
Circuits

p	q	p ∧ q
0	0	0
0	1	0
1	0	0
1	1	1



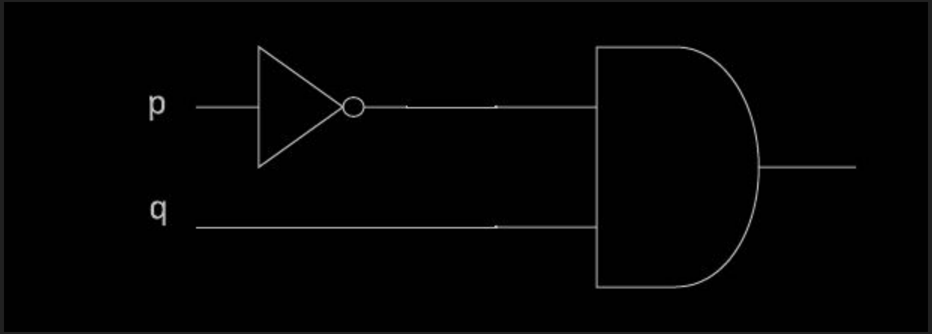
Circuits

p	q	p ∨ q
0	0	0
0	1	1
1	0	1
1	1	1



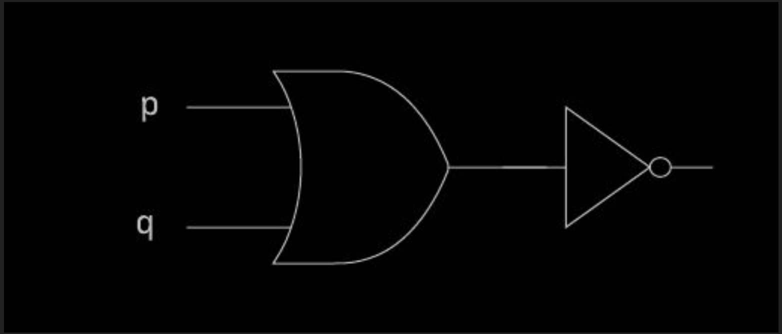
Example Representations

$\neg p \land q$



Example Representations

$\neg(p \lor q)$

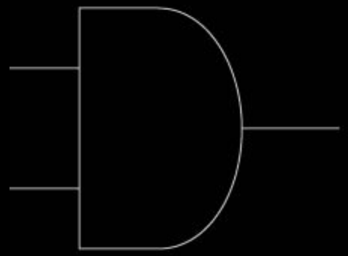


Example Representations

$\neg p \wedge \neg(p \vee q)$

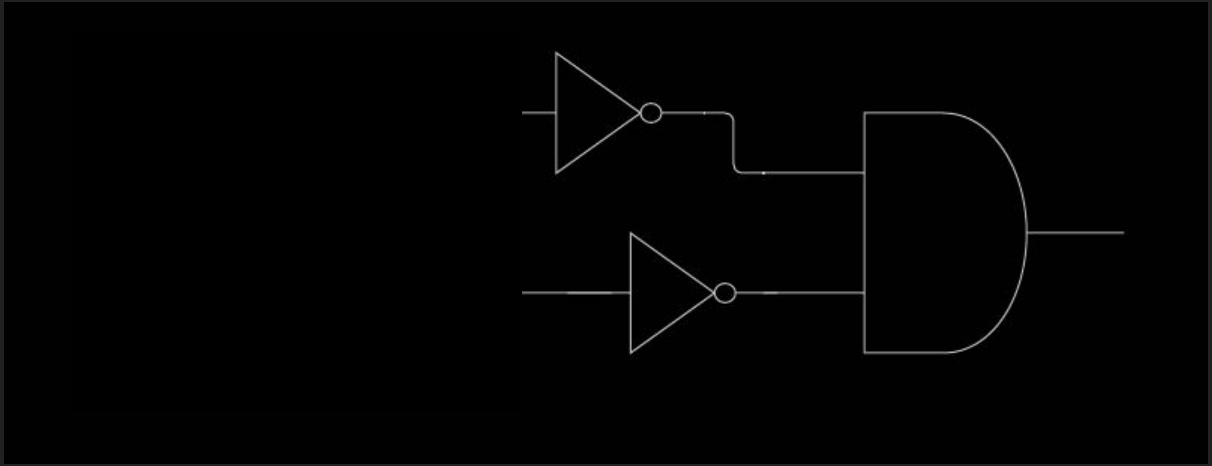
Example Representations

$\neg p \land \neg(p \lor q)$



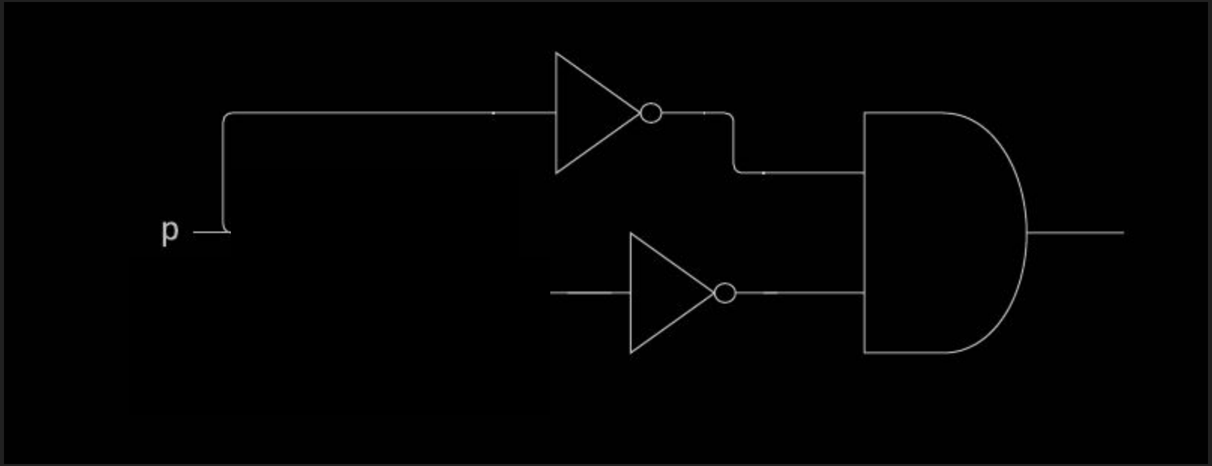
Example Representations

$\neg p \land \neg(p \lor q)$



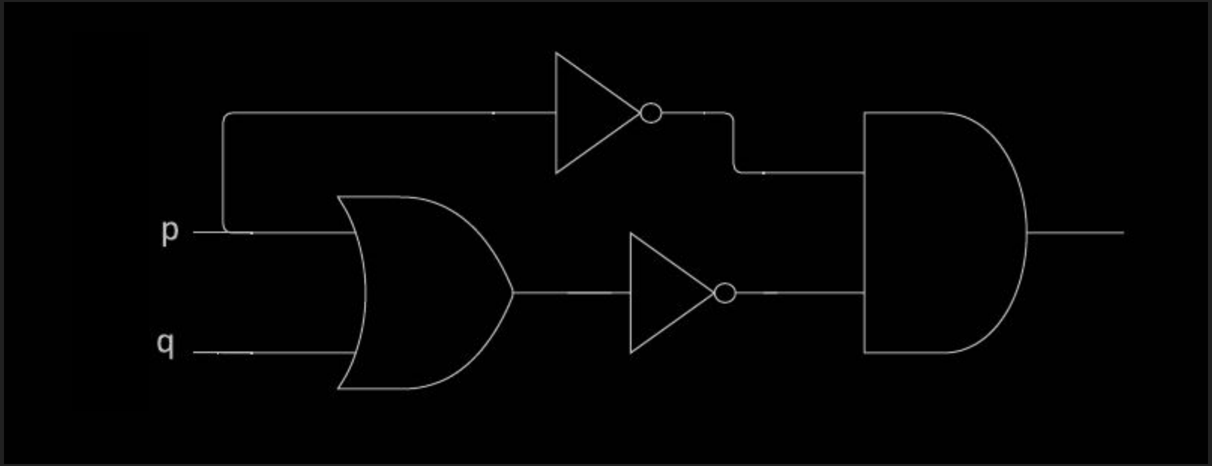
Example Representations

¬p ∧ ¬(p ∨ q)



Example Representations

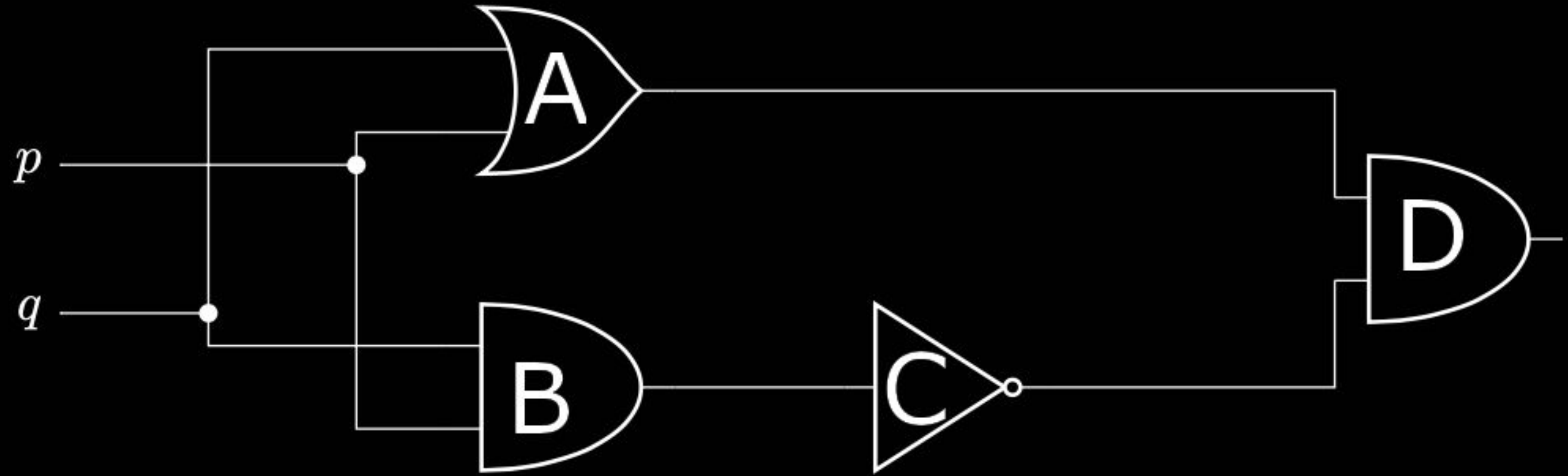
$\neg p \land \neg(p \lor q)$



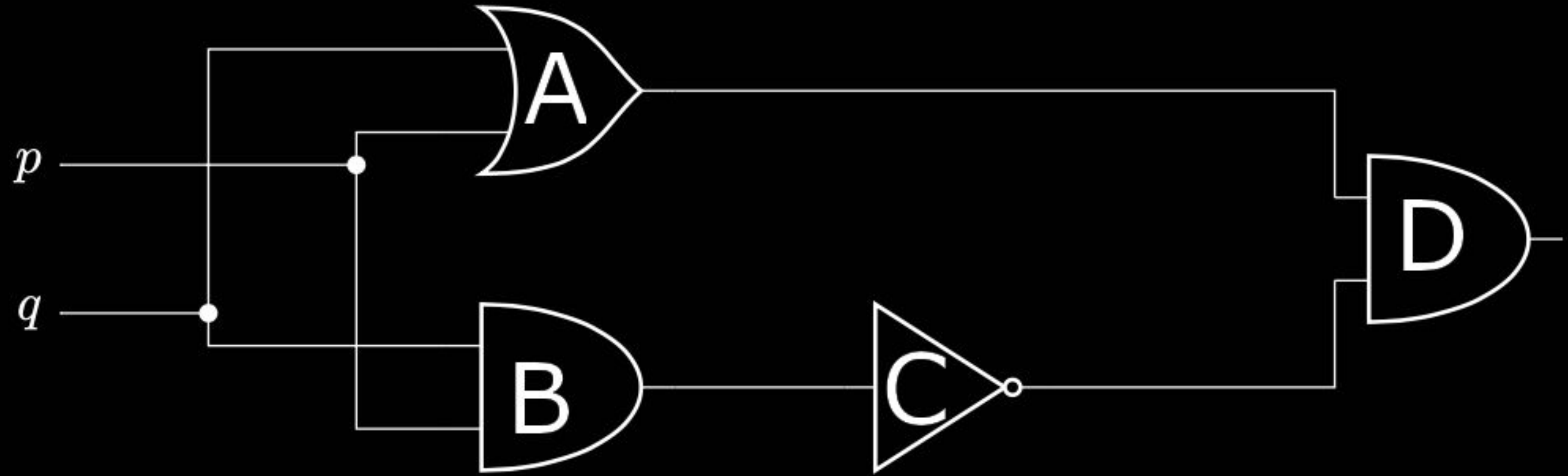
Combinational Circuit Rules

- Don’t combine input wires.
- You can split inputs when needed.
- Output can be input, but not to an earlier circuit (no loops)

Circuit to Statement

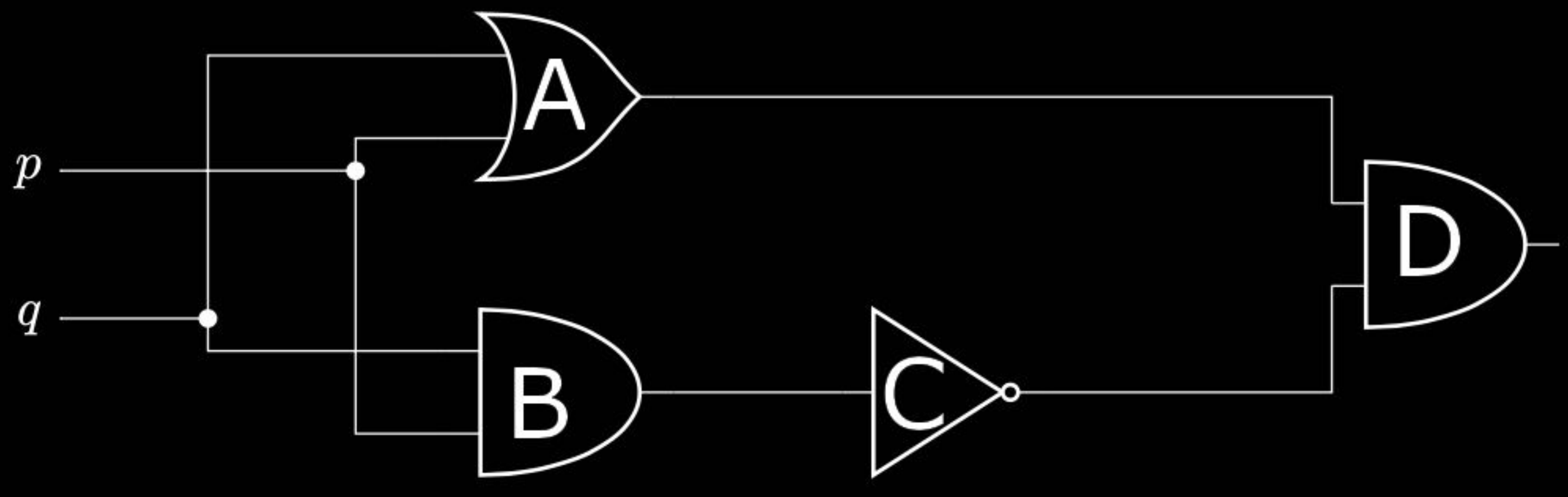


Circuit to Statement



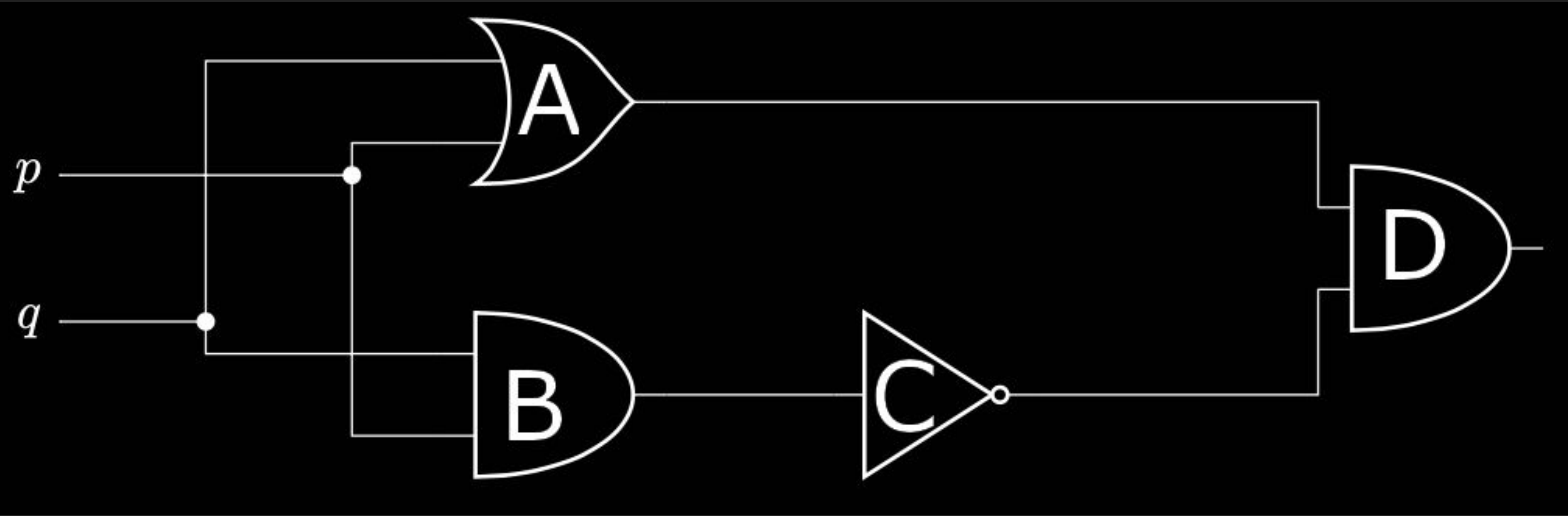
_______________ ∧ _______________

Circuit to Statement



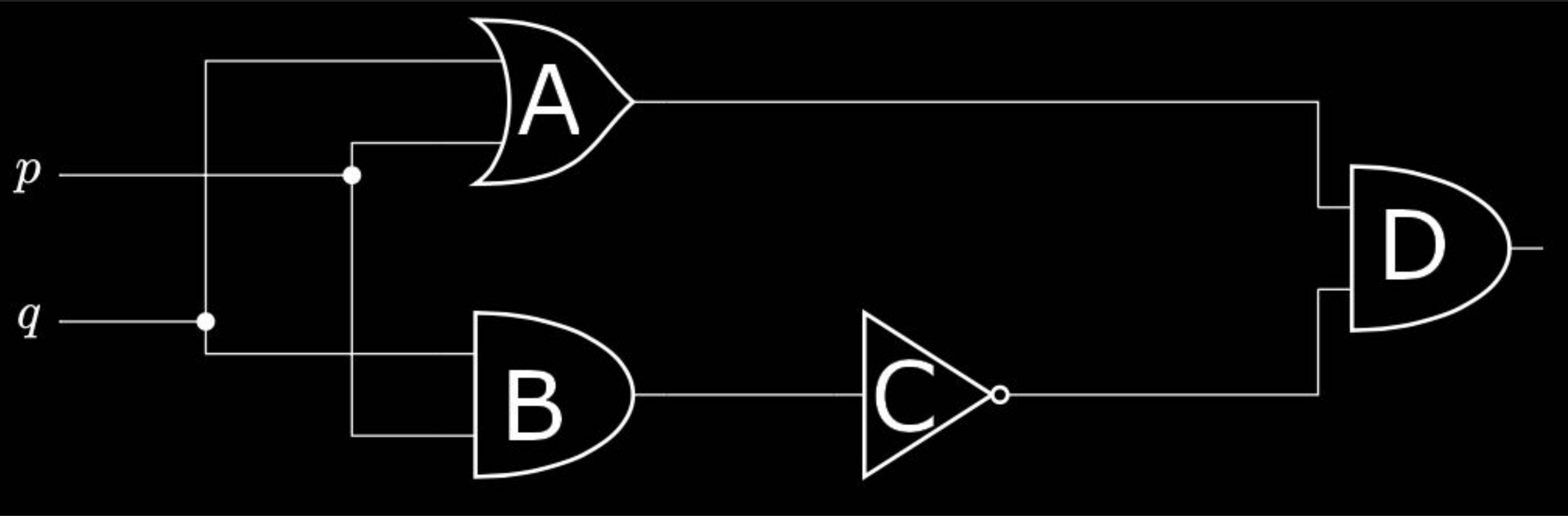
$\_\_\_\_\_\_\_\_\_\_ \wedge \neg \_\_\_\_\_\_\_\_\_\_$

Circuit to Statement



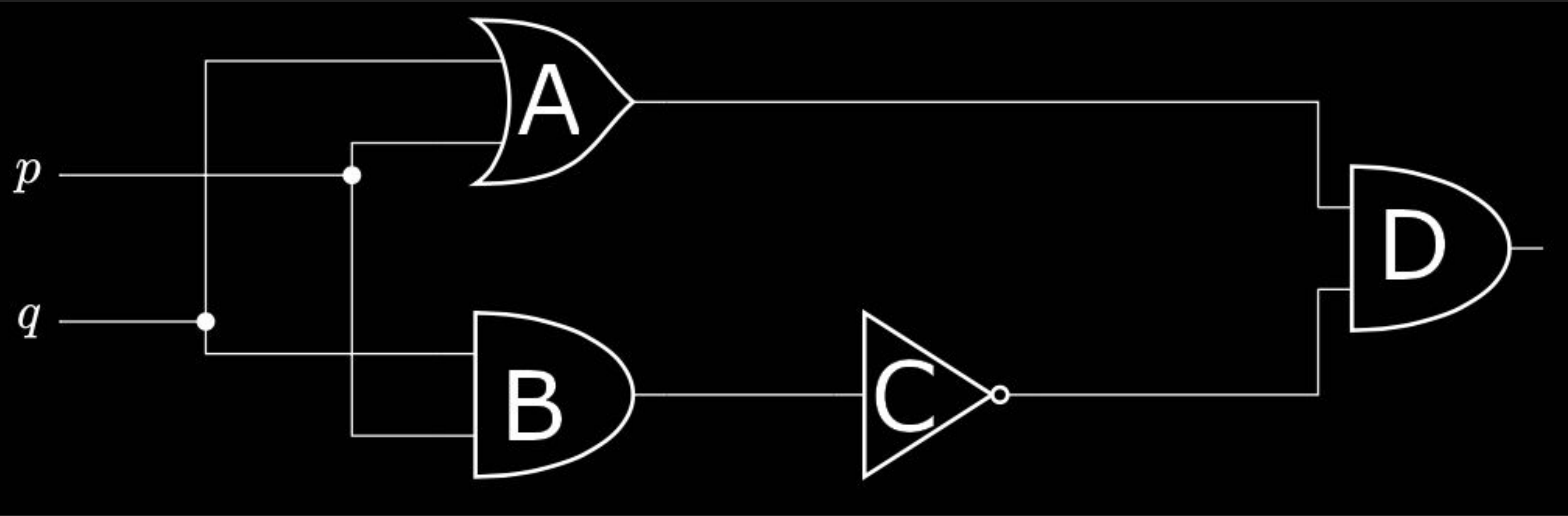
__________ ∧ ¬(__ ∧ __)

Circuit to Statement



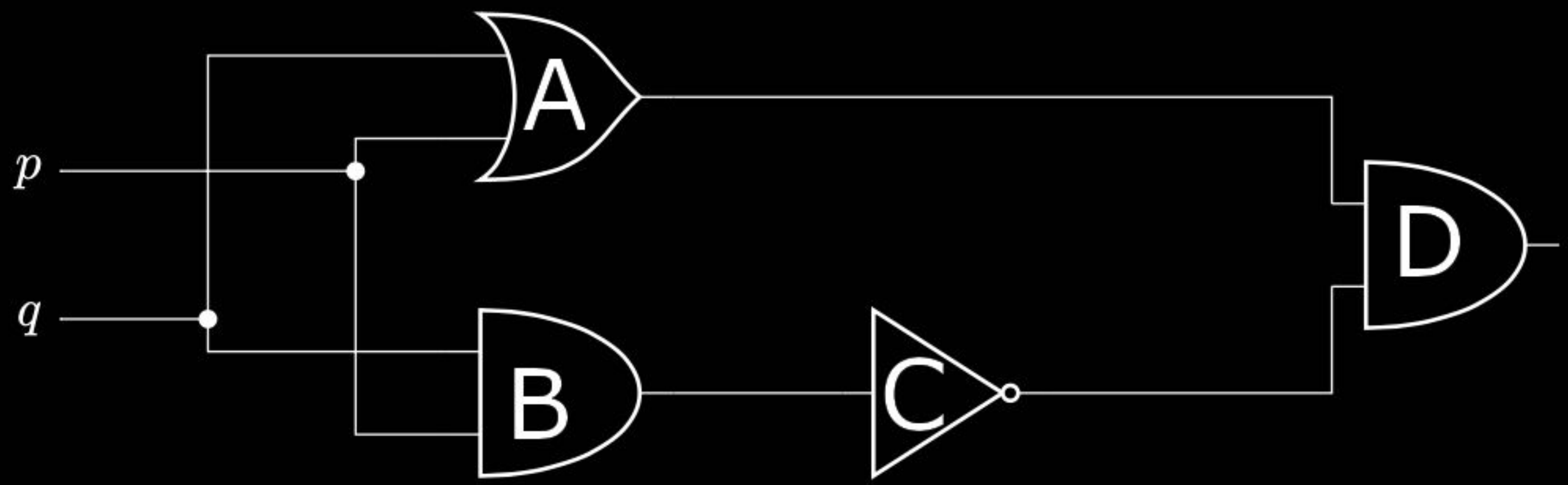
__________ $\wedge$ $\underline{\neg(p \wedge q)}$

Circuit to Statement



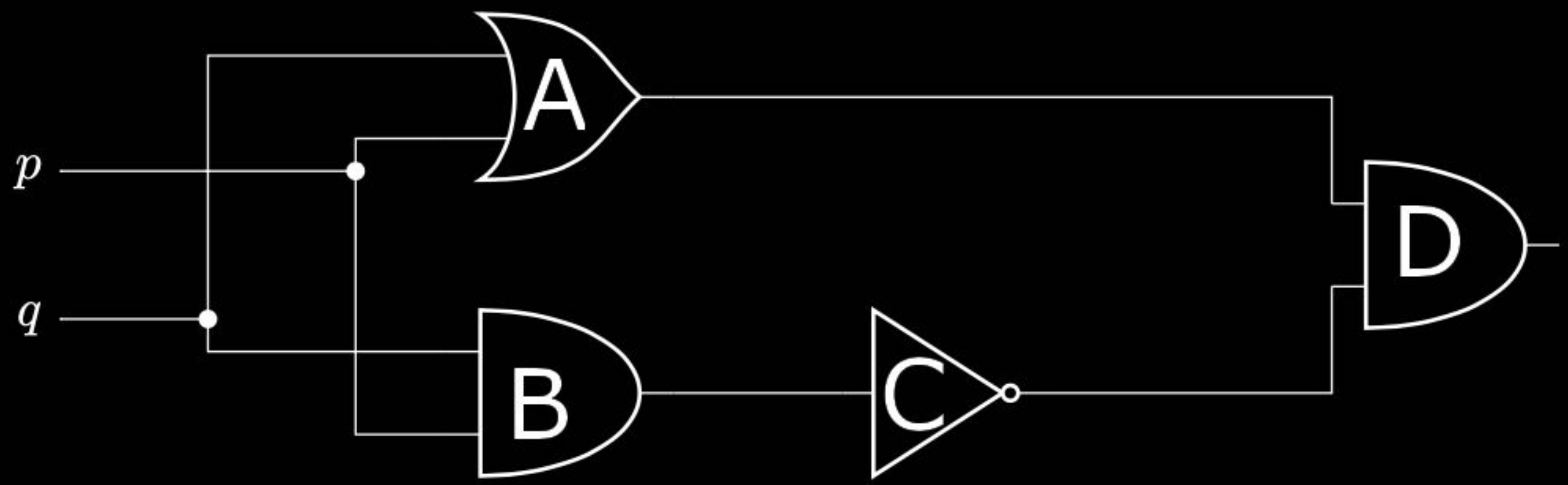
$(\_\_ \vee \_\_) \wedge \neg(p \wedge q)$

Circuit to Statement



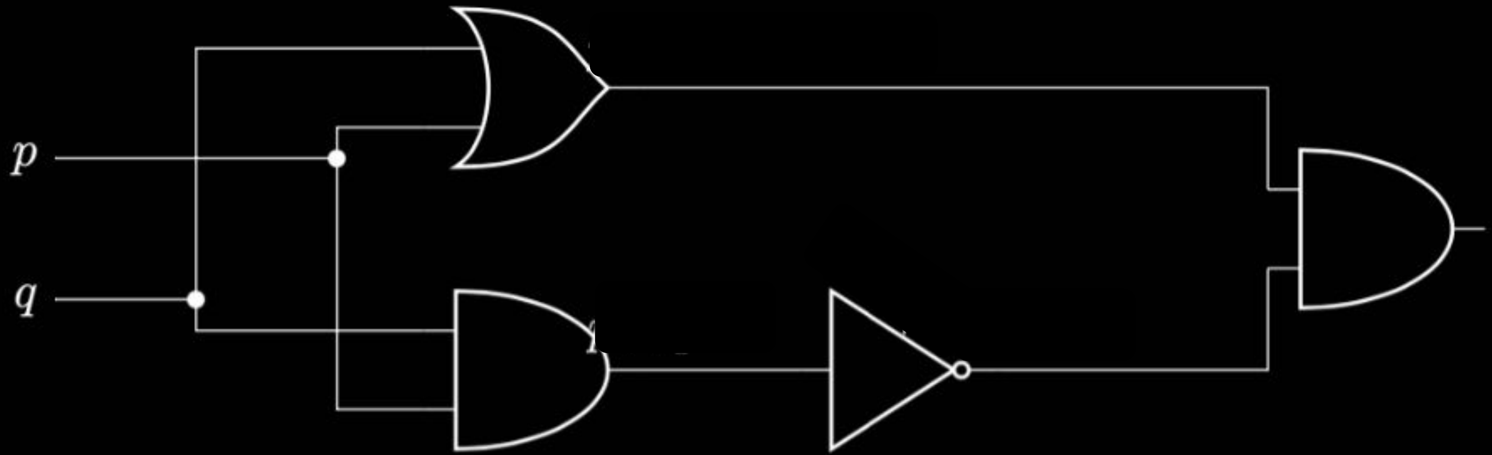
$(q \vee p) \wedge \neg(p \wedge q)$

Circuit to Statement

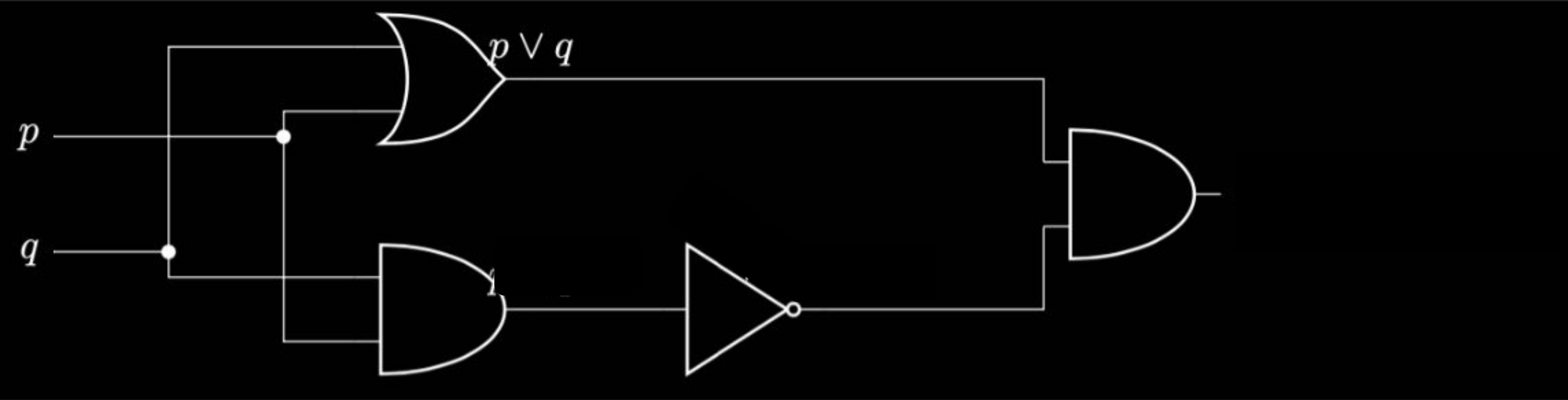


$(q \lor p) \land \neg(p \land q)$

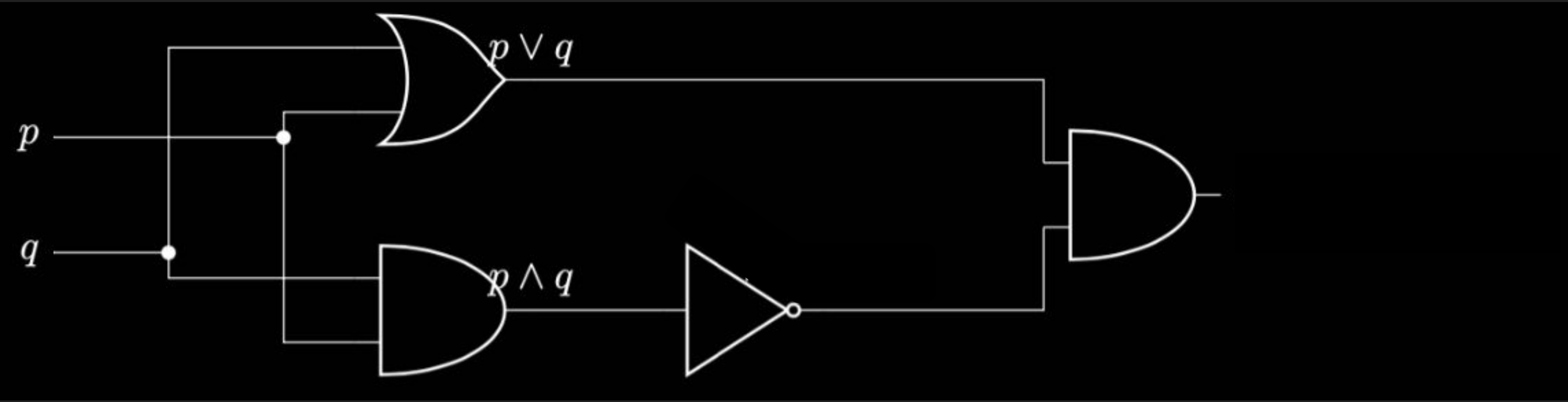
Circuit to Statement



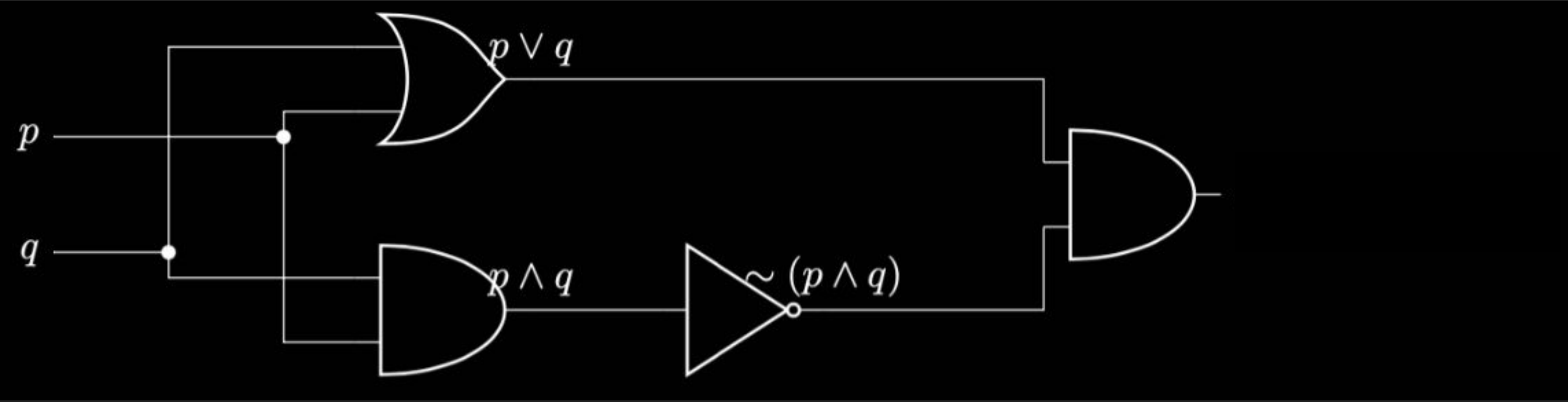
Circuit to Statement



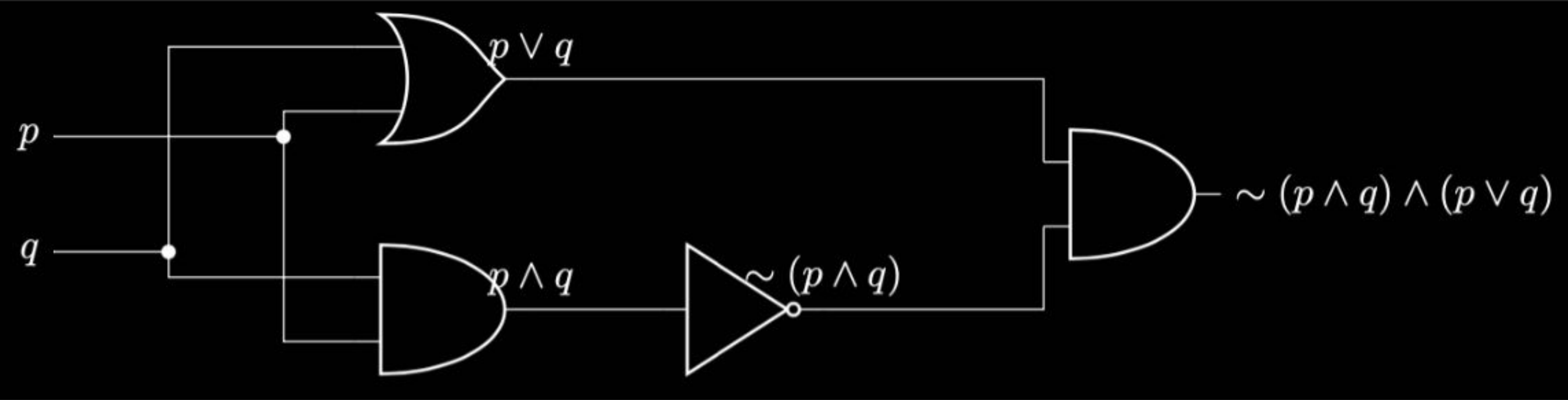
Circuit to Statement



Circuit to Statement



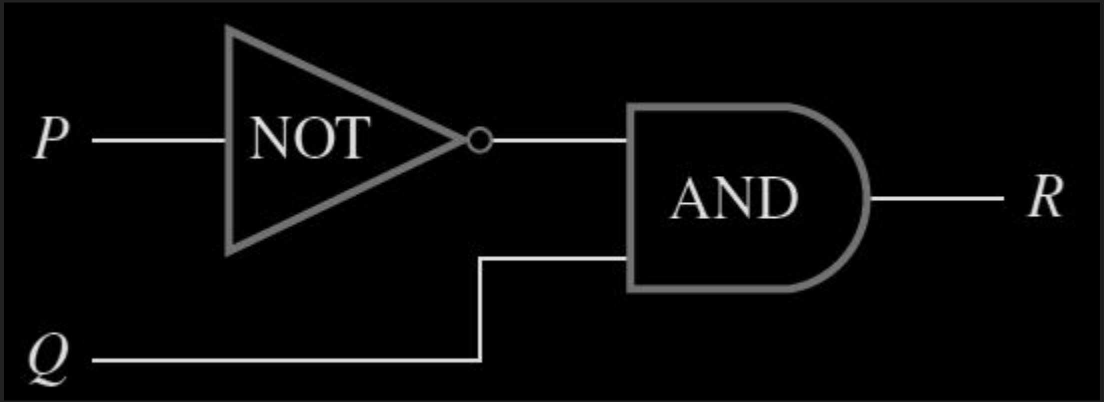
Circuit to Statement



Determining Circuit Outputs

p = 0

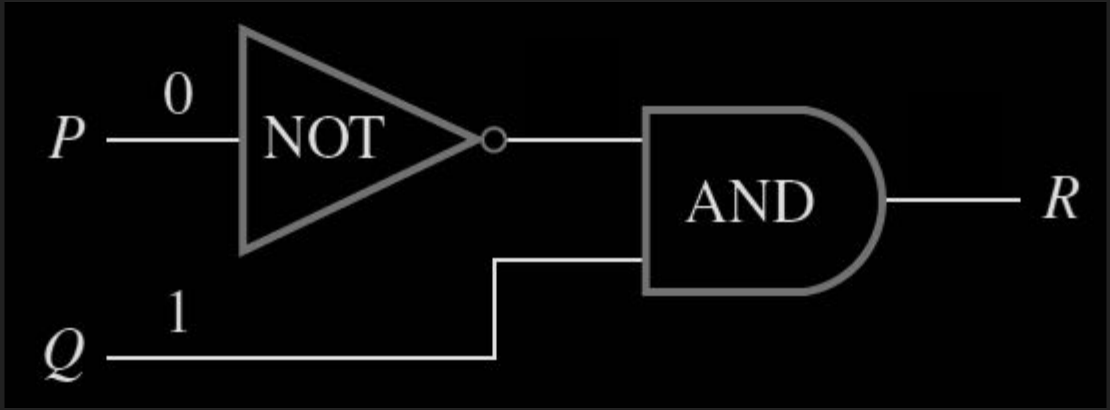
q = 1



Determining Circuit Outputs

p = 0

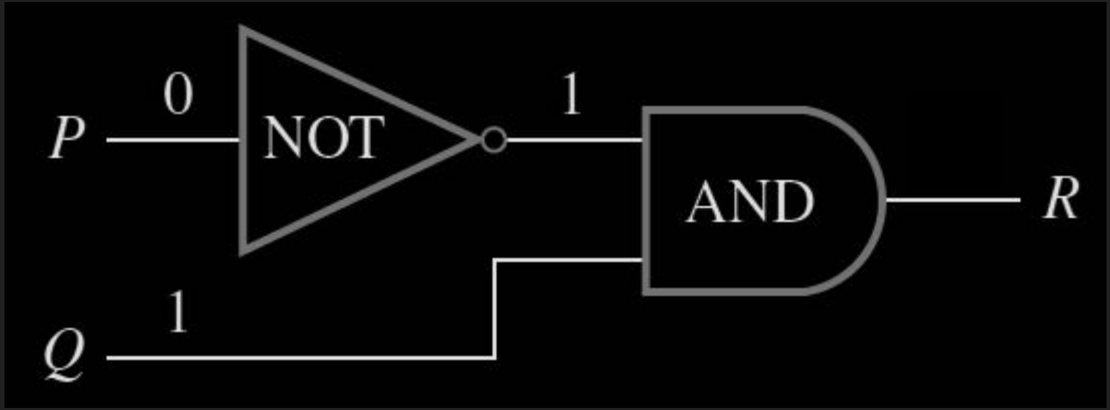
q = 1



Determining Circuit Outputs

p = 0

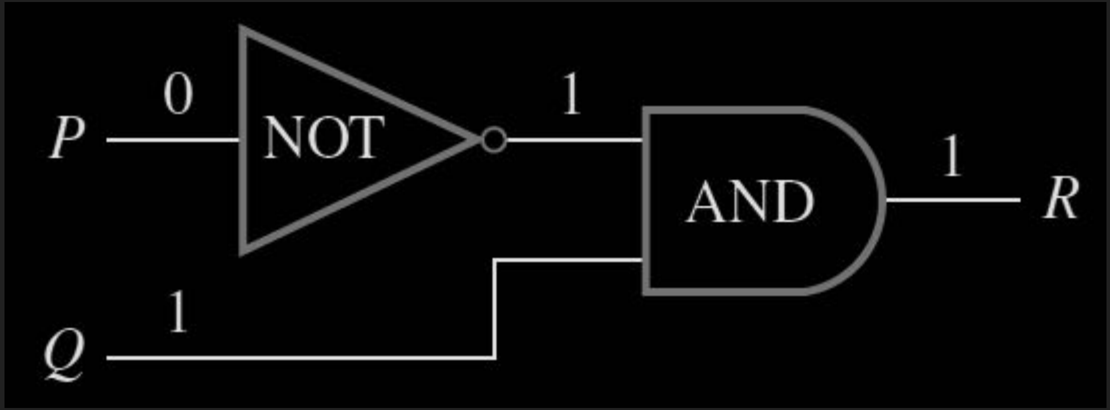
q = 1



Determining Circuit Outputs

p = 0

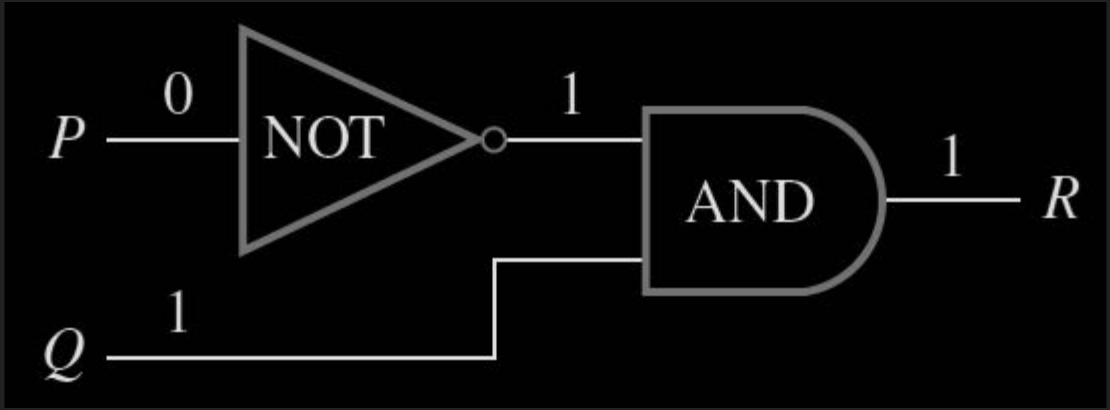
q = 1



Determining Circuit Outputs Statement?

p = 0

q = 1

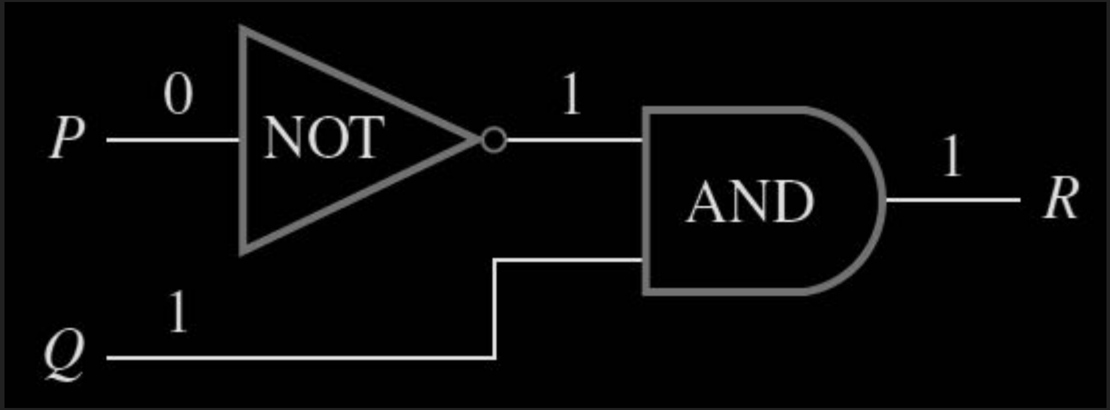


Determining Circuit Outputs Statement?

p = 0

q = 1

¬p ∧ q

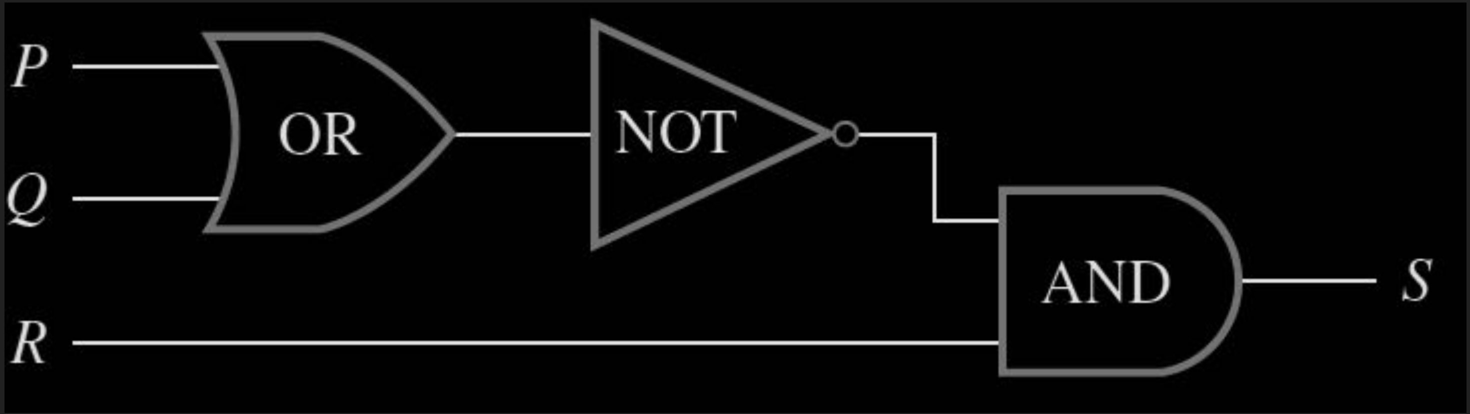


Determining Circuit Outputs

p = 1

q = 0

r = 1

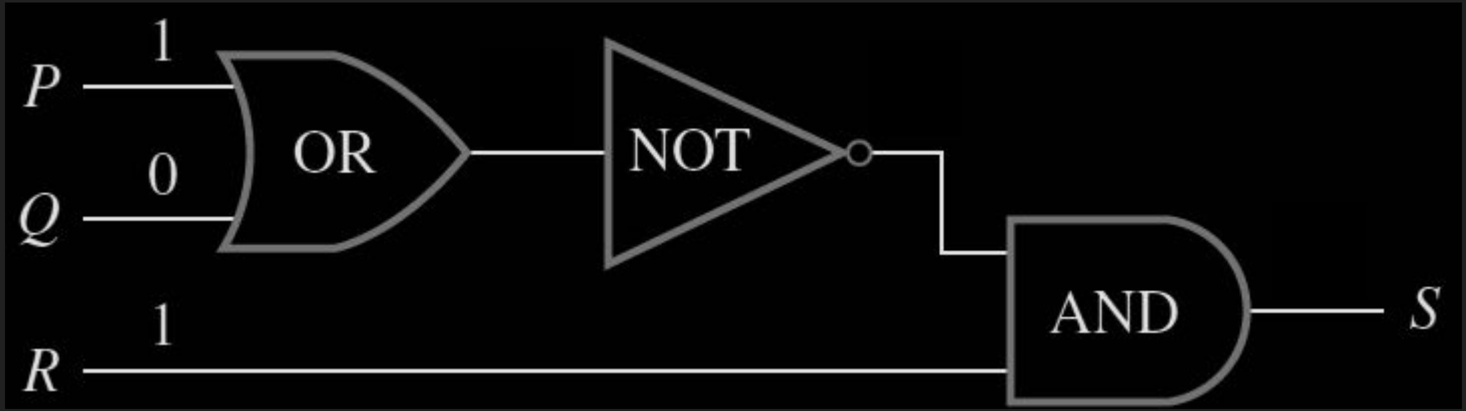


Determining Circuit Outputs

p = 1

q = 0

r = 1

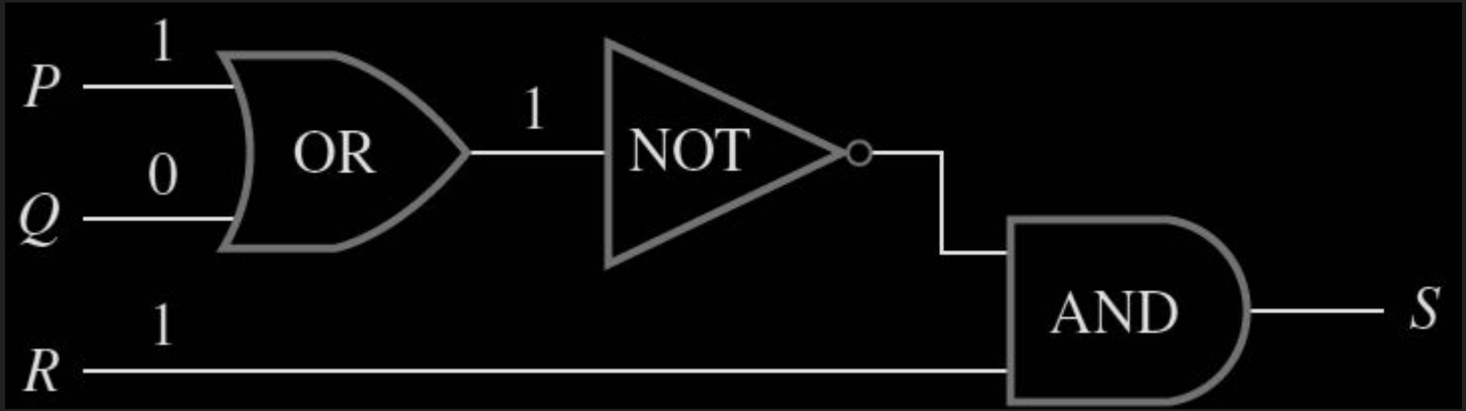


Determining Circuit Outputs

p = 1

q = 0

r = 1

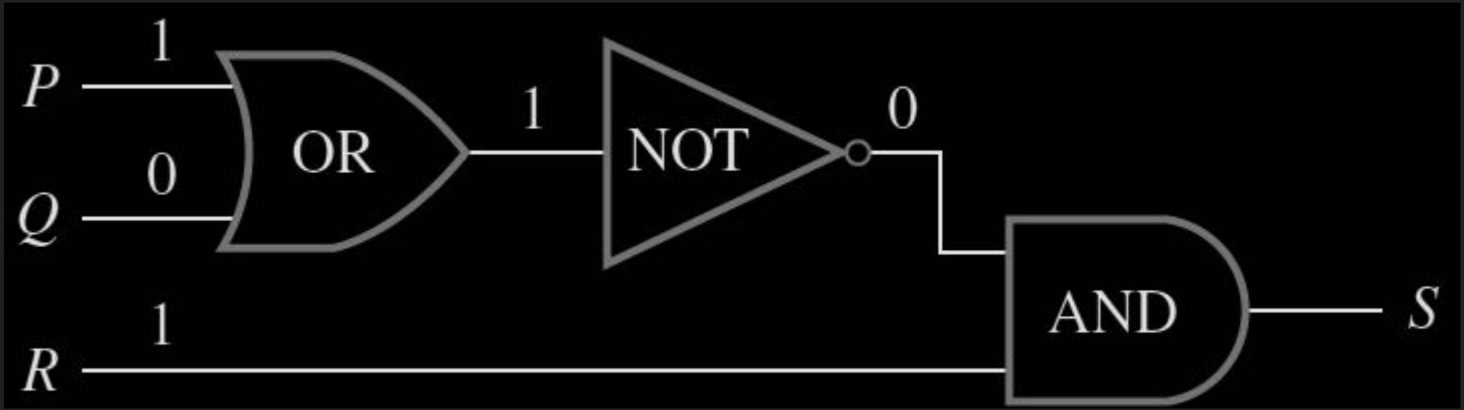


Determining Circuit Outputs

p = 1

q = 0

r = 1

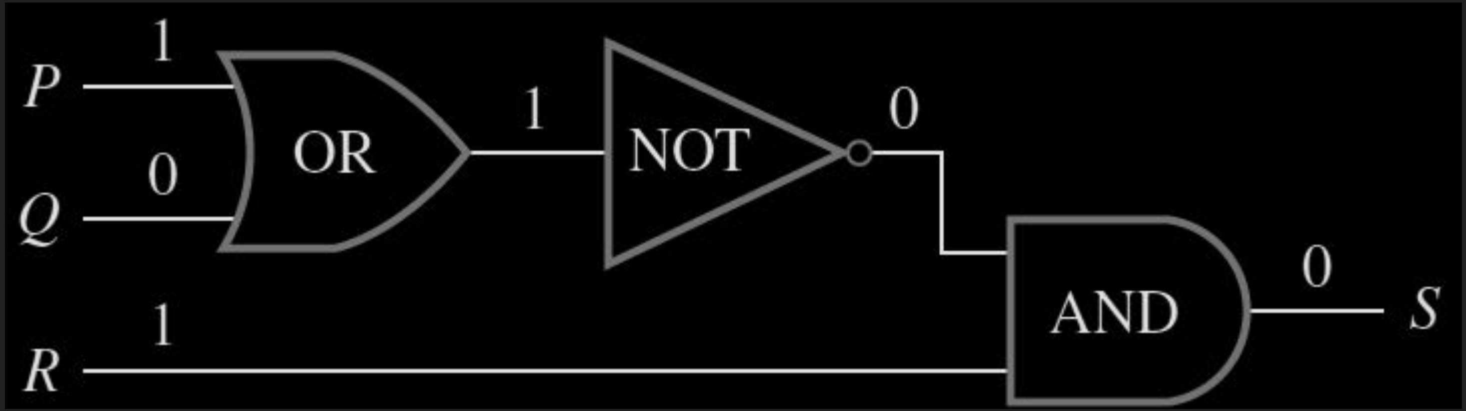


Determining Circuit Outputs

p = 1

q = 0

r = 1

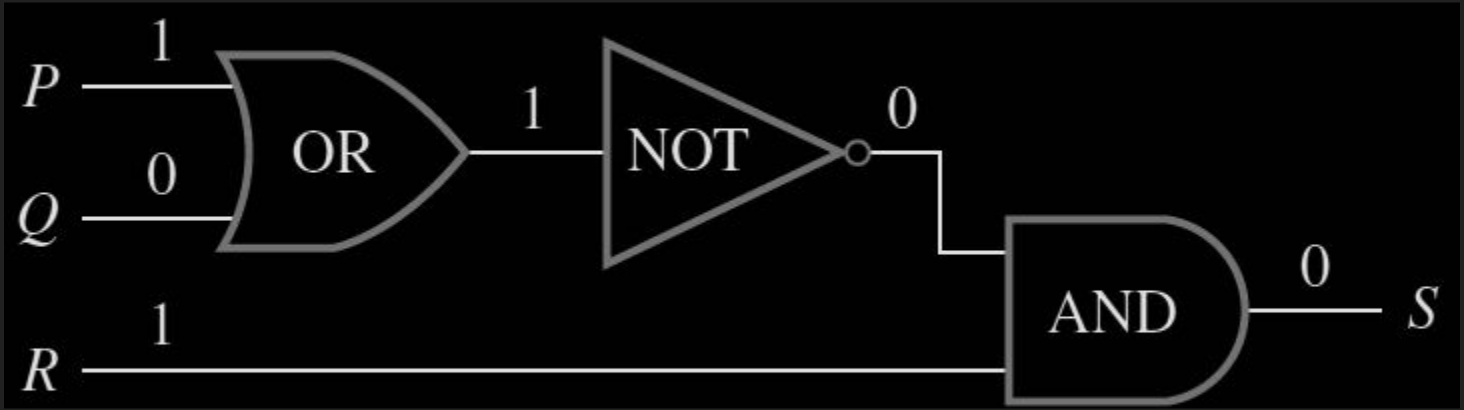


Determining Circuit Outputs

p = 1

q = 0

r = 1



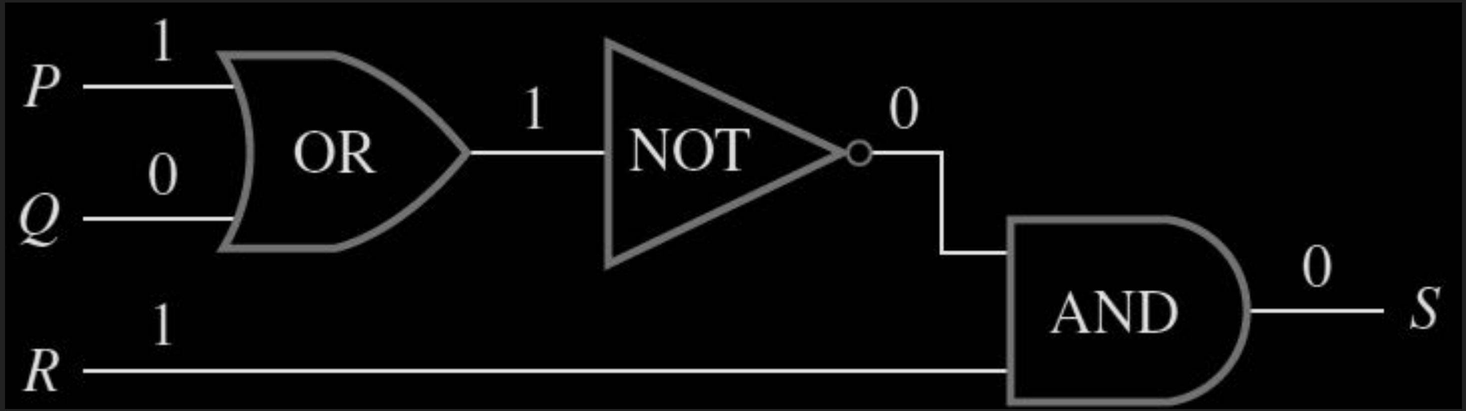
Statement S?

Determining Circuit Outputs

p = 1

q = 0

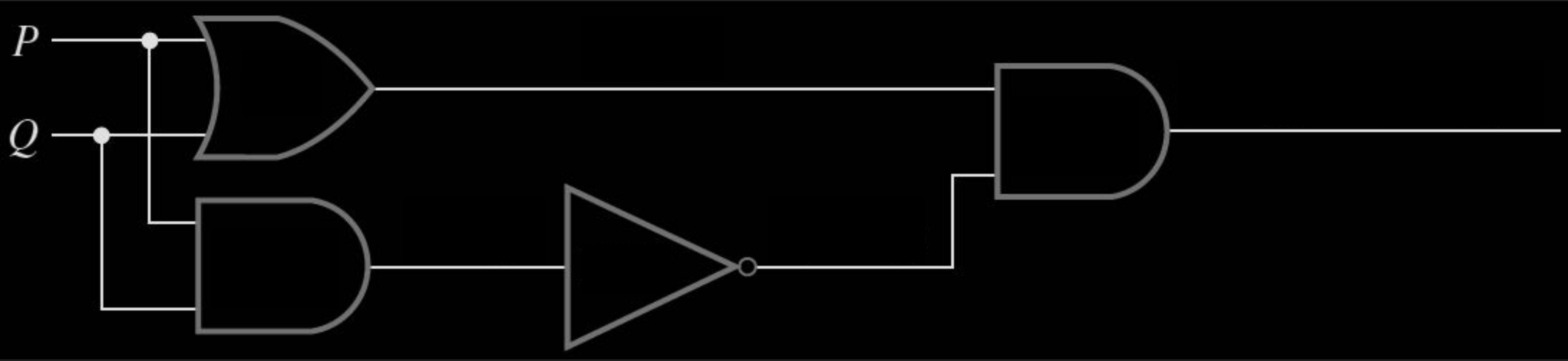
r = 1



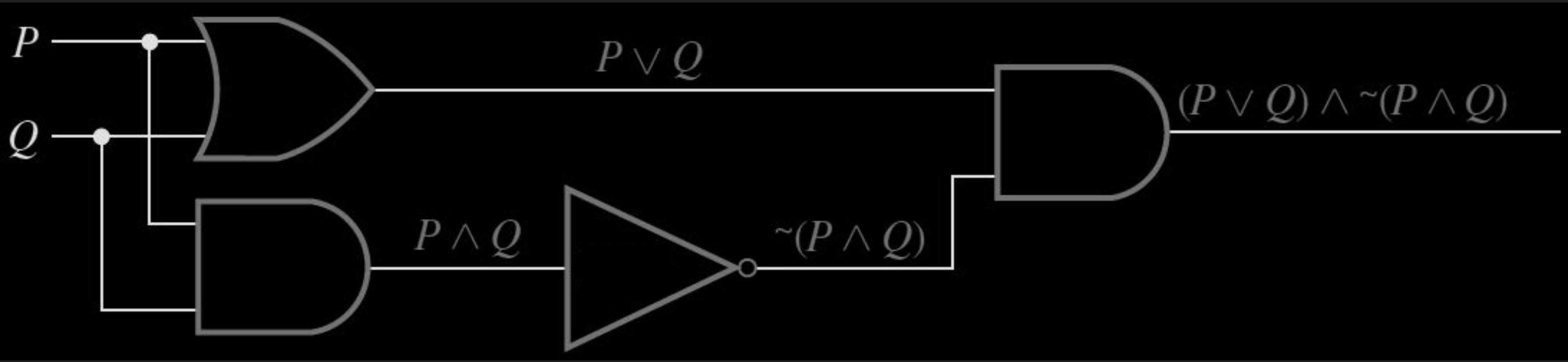
Statement S?

¬(p ∨ q) ∧ r

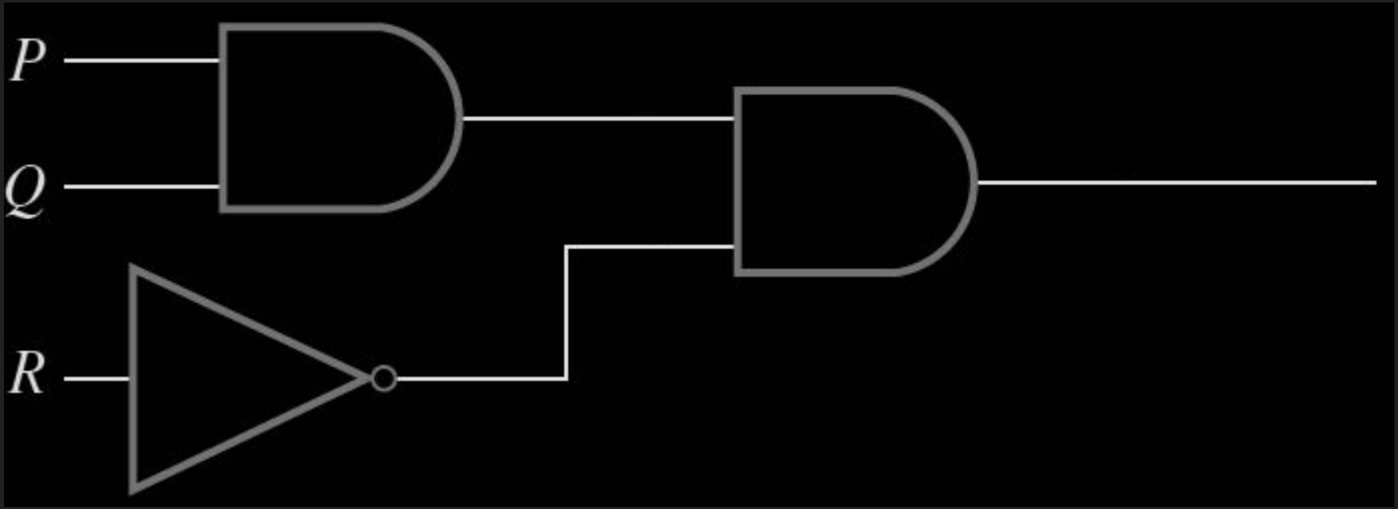
Circuit to Expression (statement)



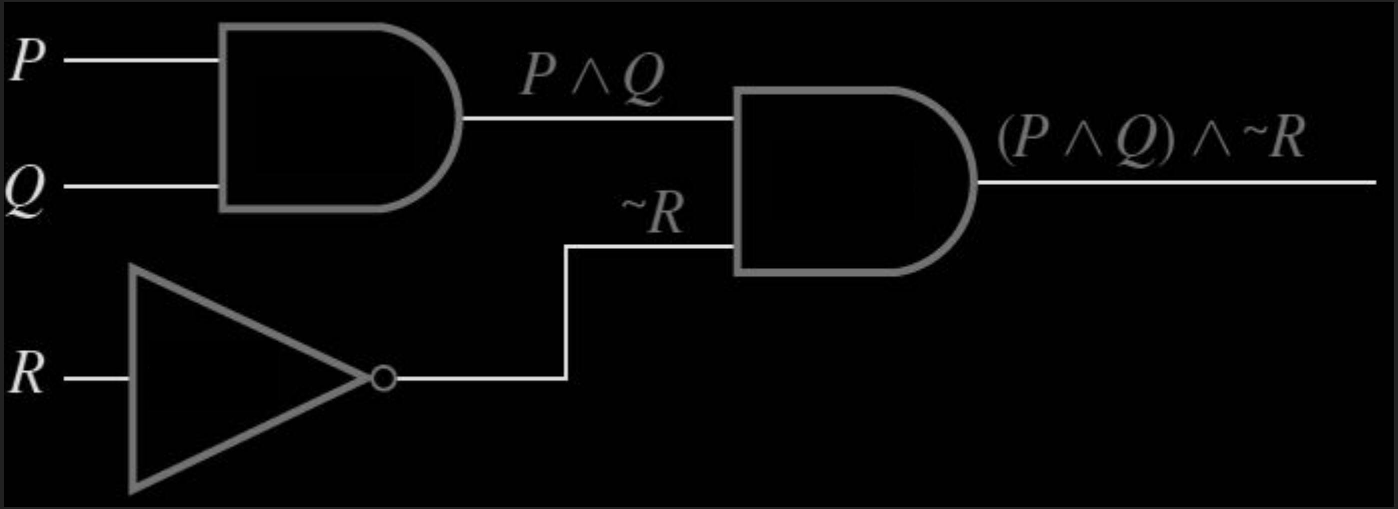
Circuit to Expression (statement)



Circuit to Expression (statement)



Circuit to Expression (statement)



Expression to Circuit

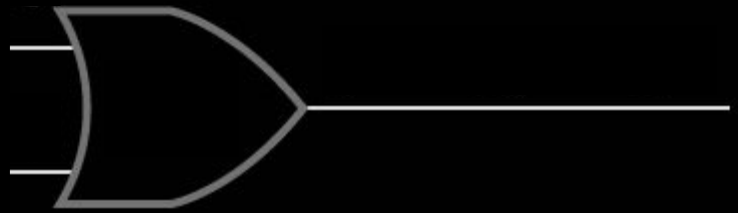
$(\neg p \land q) \lor \neg q$

Expression to Circuit

$(\neg p \land q) \lor \neg q$

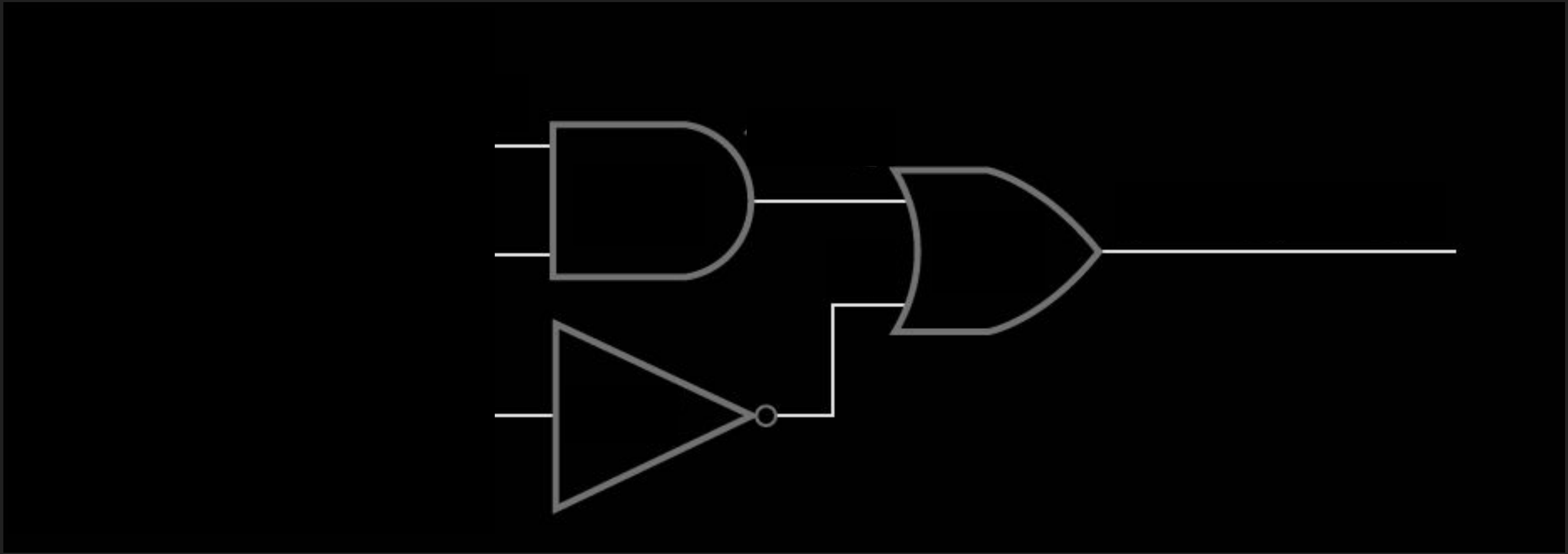
Expression to Circuit

$(\neg p \land q) \lor \neg q$



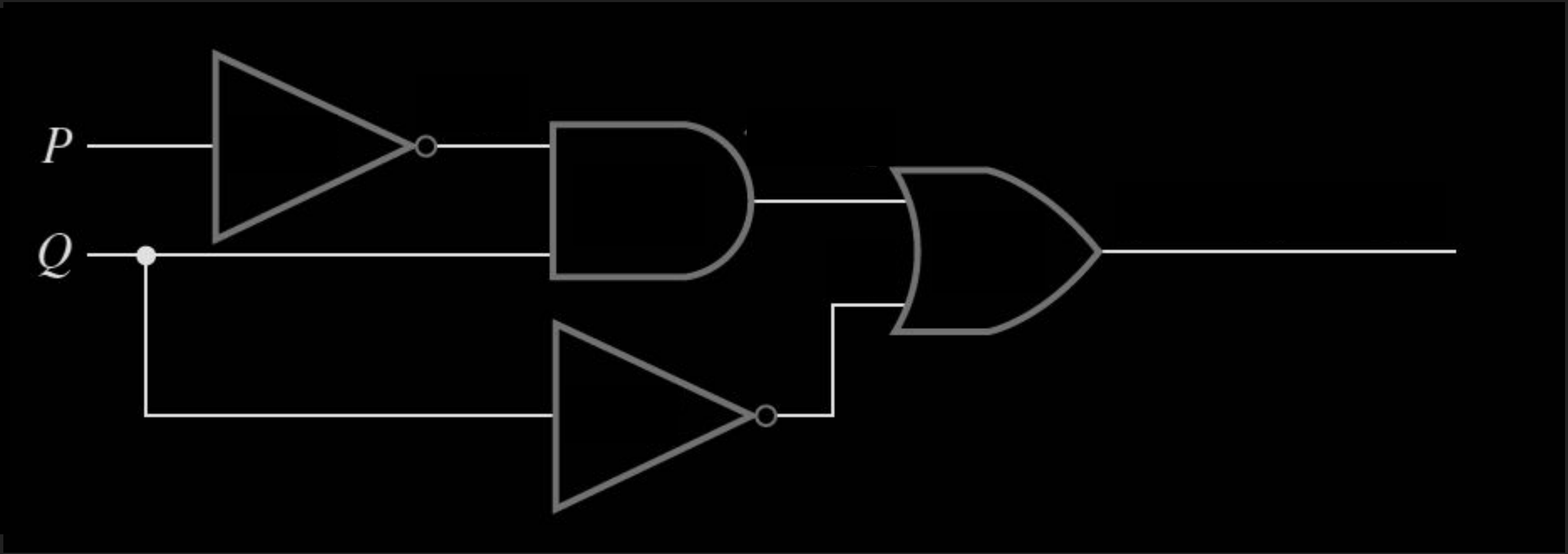
Expression to Circuit

(¬p ∧ q) ∨ ¬q



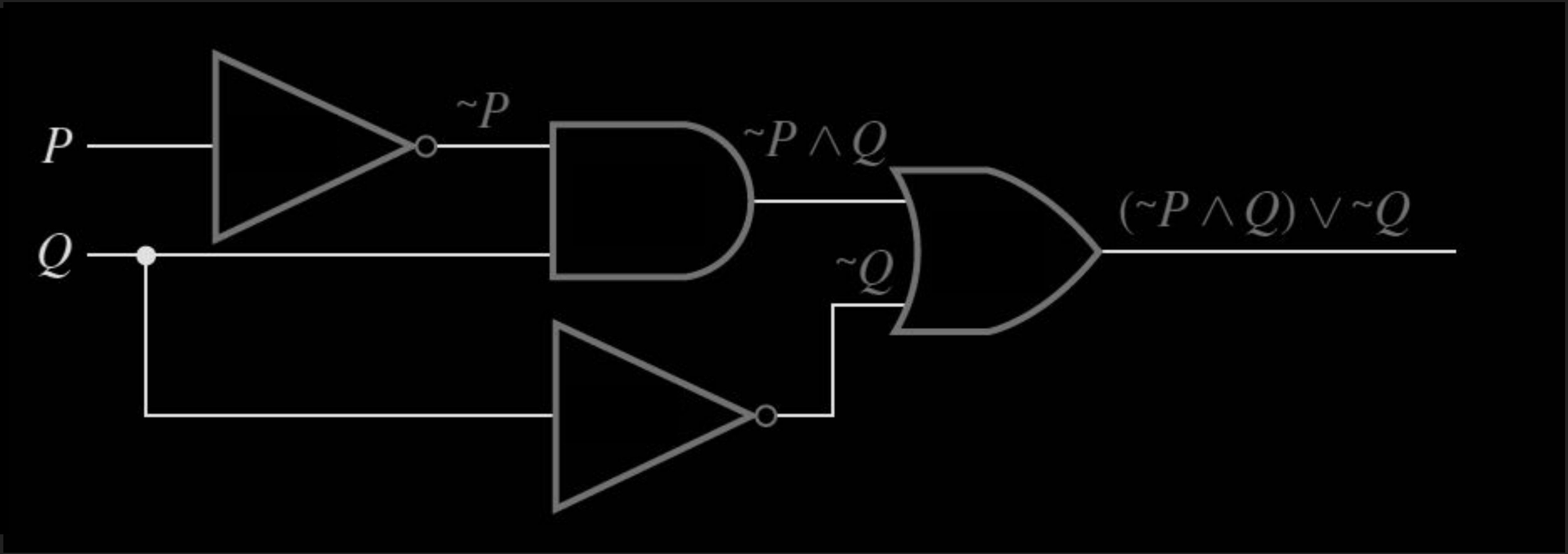
Expression to Circuit

(¬p ∧ q) ∨ ¬q



Expression to Circuit

(¬p ∧ q) ∨ ¬q



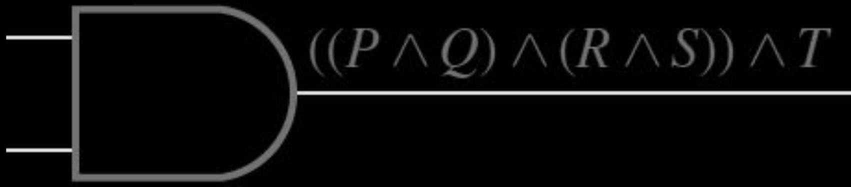
Expression to Circuit

$((p \land q) \land (r \land s)) \land t$

$$((P \land Q) \land (R \land S)) \land T$$

Expression to Circuit

$((p \land q) \land (r \land s)) \land t$

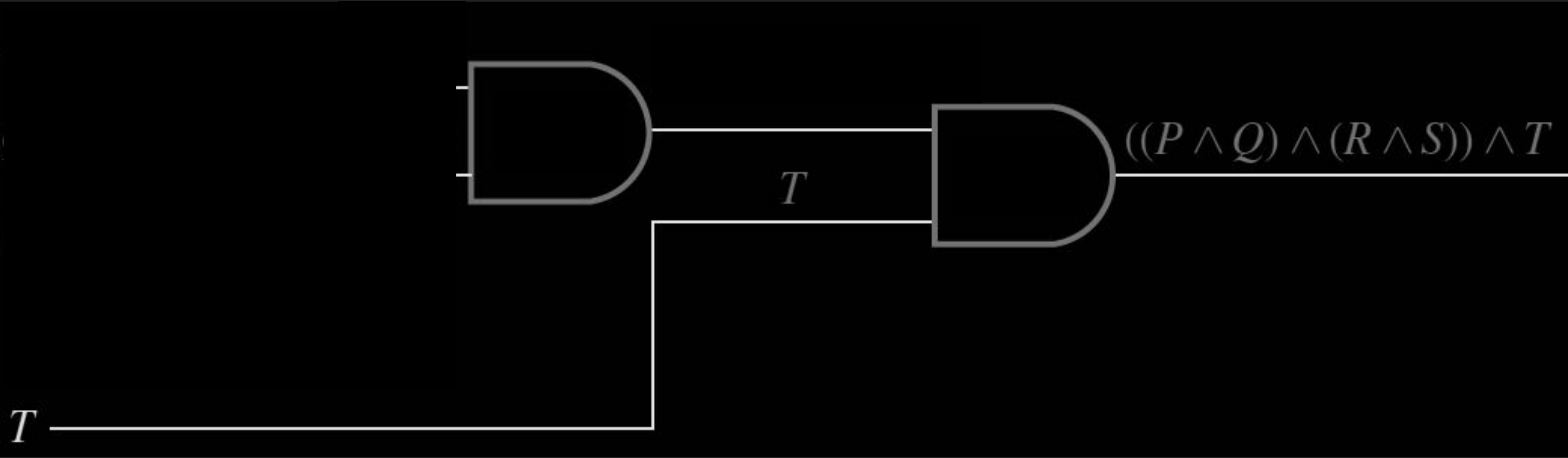


Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t

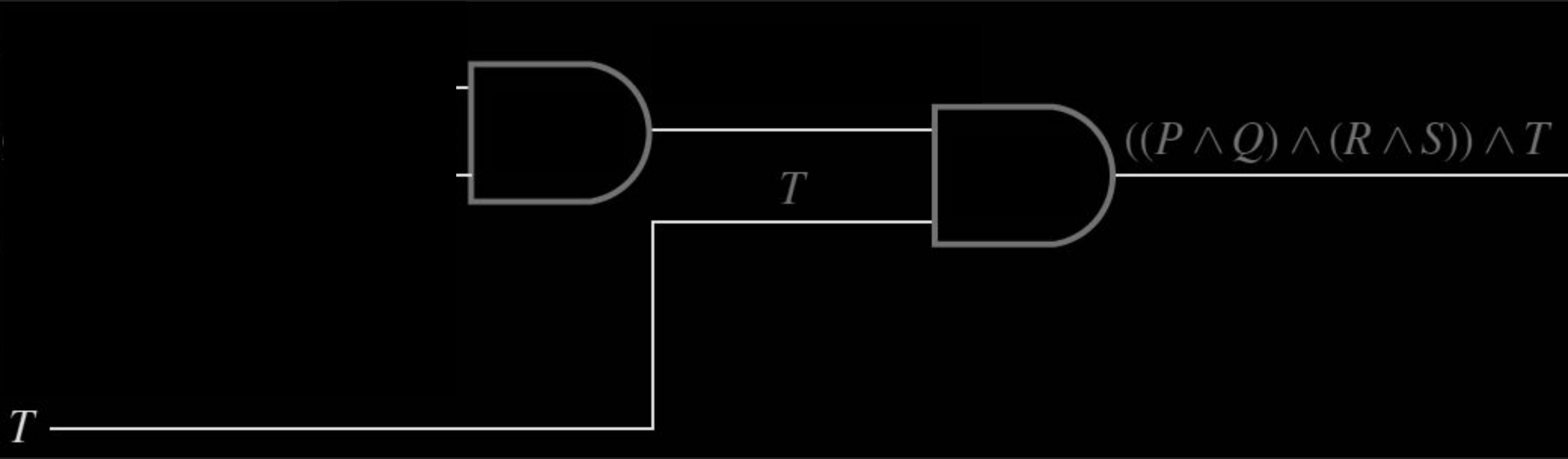
Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t



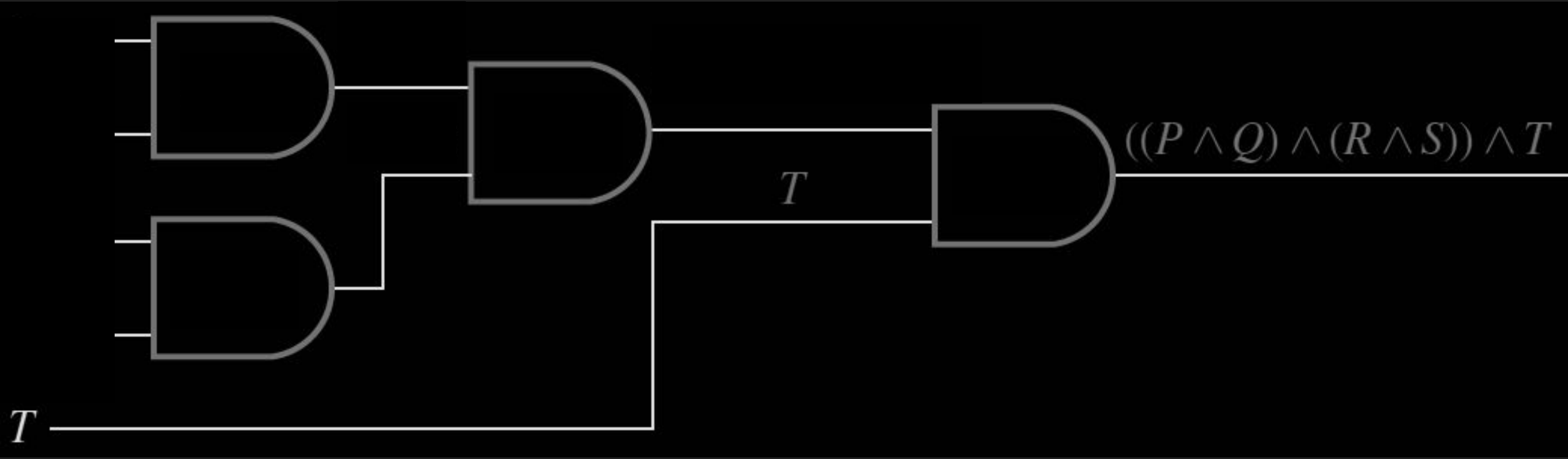
Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t



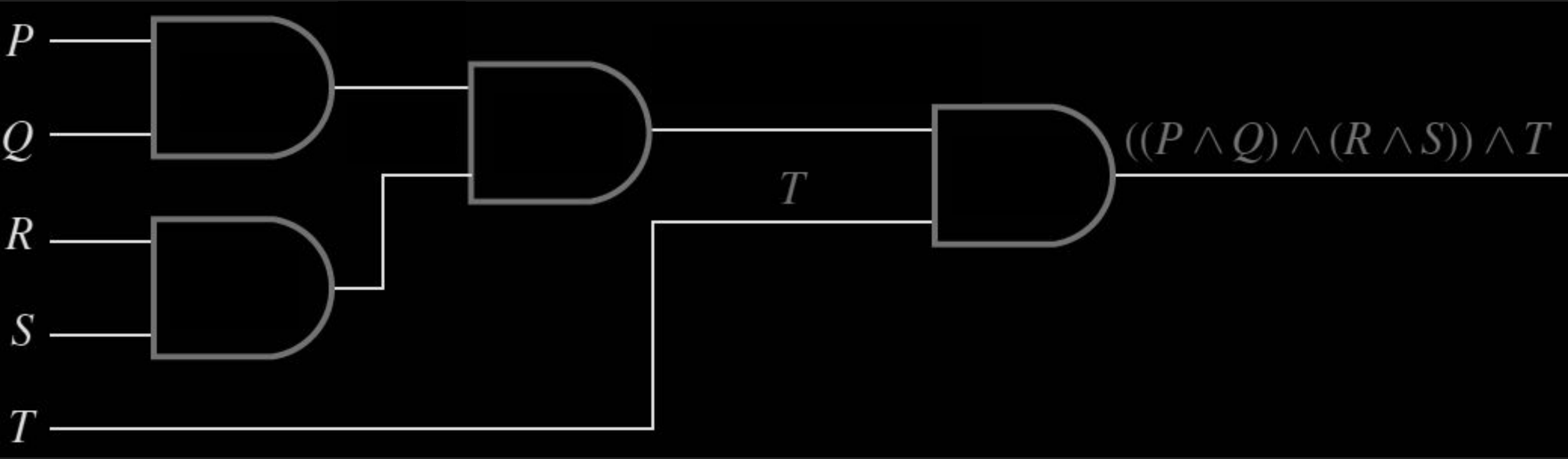
Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t



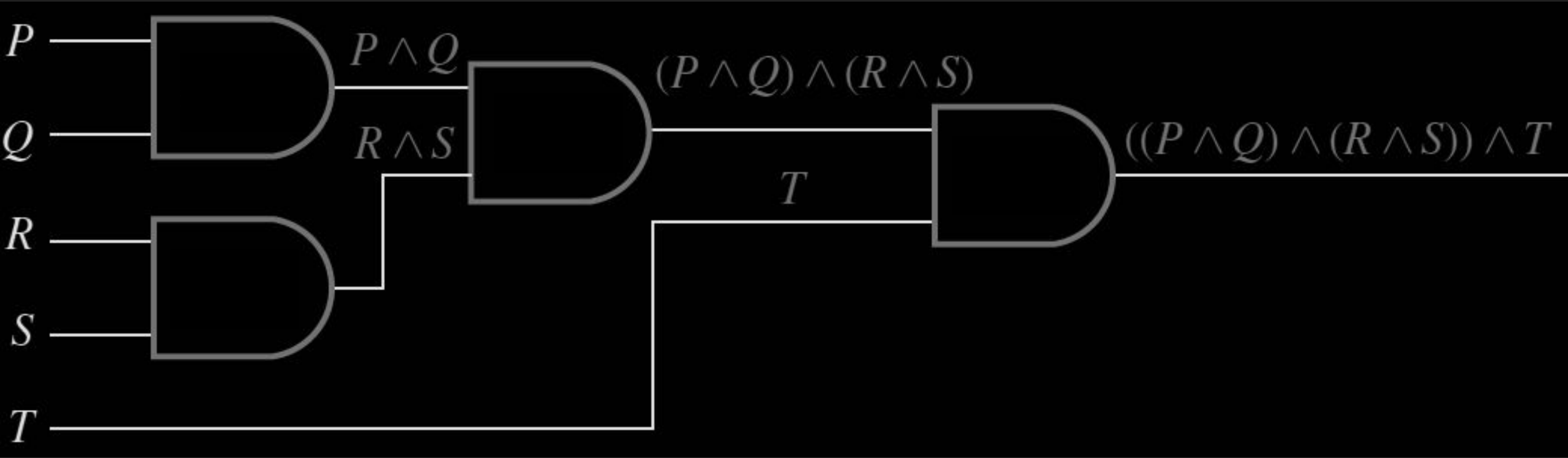
Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t



Expression to Circuit

((p ∧ q) ∧ (r ∧ s)) ∧ t



Constructing an expression of an unknown

Two similar ways.

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 1: **Disjunctive normal form/ sum of products**

1. Identify rows where output is **true**

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 1: **Disjunctive normal form/ sum of products**

1. Identify rows where output is **true**
2. Create the expression that made that row true for each row.

$\neg p \land \neg q \land \neg r$

$\neg p \land q \land \neg r$

$\neg p \land q \land r$

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 1: **Disjunctive normal form/ sum of products**

1. Identify rows where output is **true**
2. Create the expression that made that row true for each row.
3. Since these were the only conditions making s true, or them all together.

$\neg p \land \neg q \land \neg r$

$\neg p \land q \land \neg r$

$\neg p \land q \land r$

$$(\neg p \land \neg q \land \neg r) \lor (\neg p \land q \land \neg r) \lor (\neg p \land q \land r)$$

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 2: **Conjunctive normal form/ product of sums**

1. Identify rows where output is **False**

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 2: **Conjunctive normal form/ product of sums**

1. Identify rows where output is **False**

p	q	r	s
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	0
1	1	1	0

Constructing an expression of an unknown

Method 2: **Conjunctive normal form/ product of sums**

1. Identify rows where output is **False**
2. Create the disjunction s.t. each piece is false.

p	q	r	*s*
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	0
1	1	1	0

$p \lor q \lor \neg r$

$\neg p \lor q \lor r$

$\neg p \lor q \lor \neg r$

$\neg p \lor \neg q \lor r$

$\neg p \lor \neg q \lor \neg r$

Constructing an expression of an unknown

Method 2: **Conjunctive normal form/ product of sums**

1. Identify rows where output is **False**
2. Create the disjunction s.t. each piece is false.
3. Join them all together with conjunctions.

p	q	r	s
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	0
1	1	1	0

$p \lor q \lor \neg r$

$\neg p \lor q \lor r$

$\neg p \lor q \lor \neg r$

$\neg p \lor \neg q \lor r$

$\neg p \lor \neg q \lor \neg r$

$$(p \lor q \lor \neg r) \land (\neg p \lor q \lor r) \land (\neg p \lor q \lor \neg r) \land (\neg p \lor \neg q \lor r) \land (\neg p \lor \neg q \lor \neg r)$$

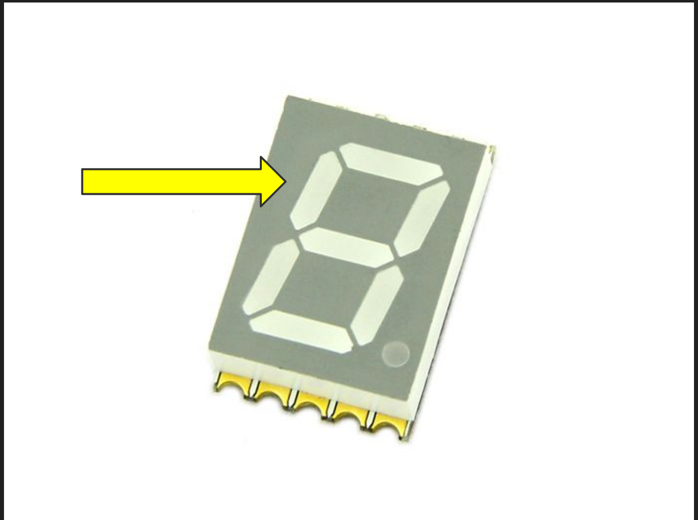
$$(p \lor q \lor \neg r) \land (\neg p \lor q \lor r) \land (\neg p \lor q \lor \neg r) \land (\neg p \lor \neg q \lor r) \land (\neg p \lor \neg q \lor \neg r)$$

$$\equiv$$

$$(\neg p \land \neg q \land \neg r) \lor (\neg p \land q \land \neg r) \lor (\neg p \land q \land r)$$

Constructing an expression of an unknown

Why might you want to do this?



Constructing an expression of an unknown

Why might you want to do this?

1 input:

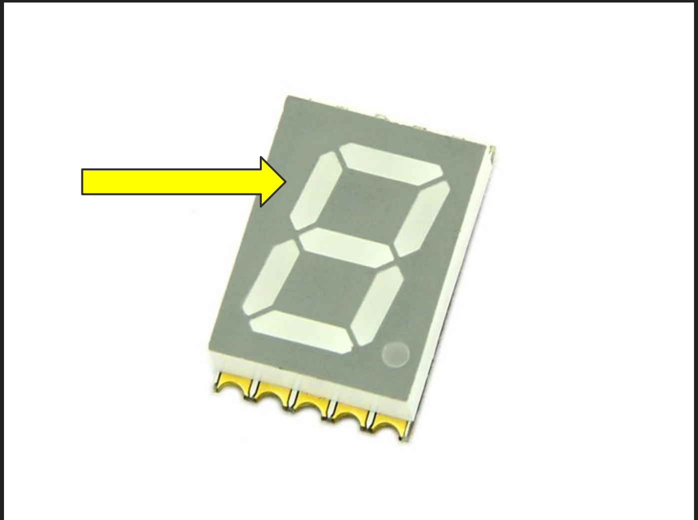
0	1
0	1

2 inputs:

00	01	10	11
0	1	2	3

3 inputs:

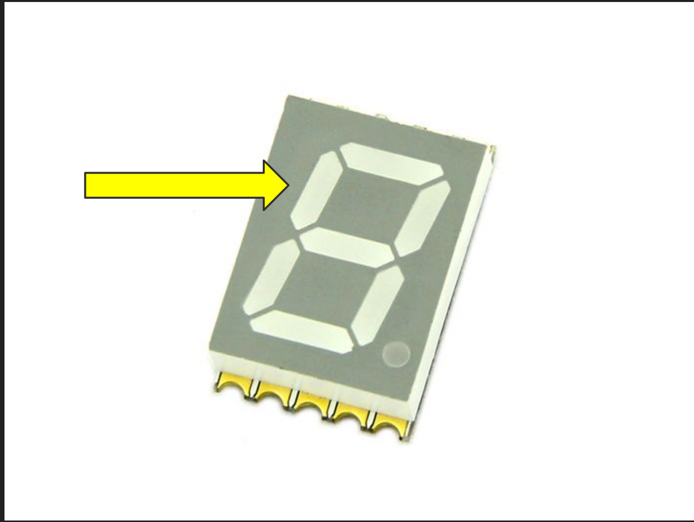
000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7



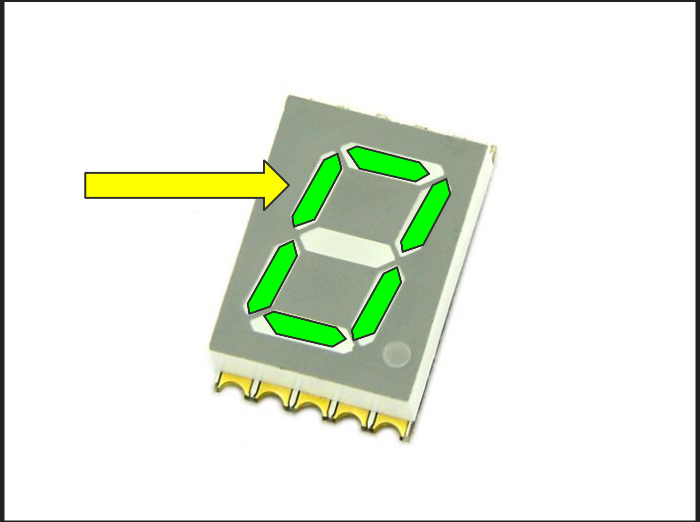
4 inputs:

0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	1	2	3	4	5	6	7	8	9	X	X	X	X	X	X

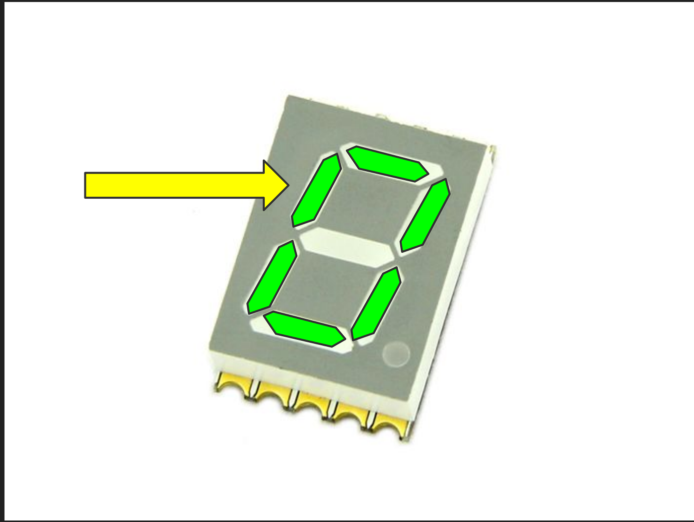
#	A	B	C	D	Y
0	0	0	0	0	
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



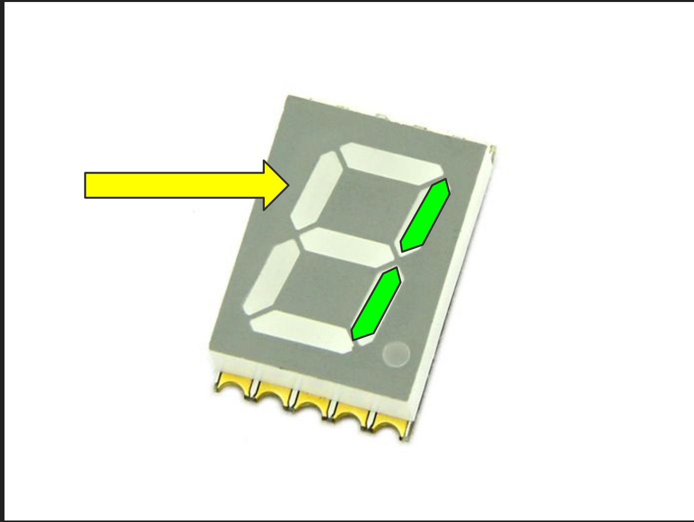
#	A	B	C	D	Y
0	0	0	0	0	
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



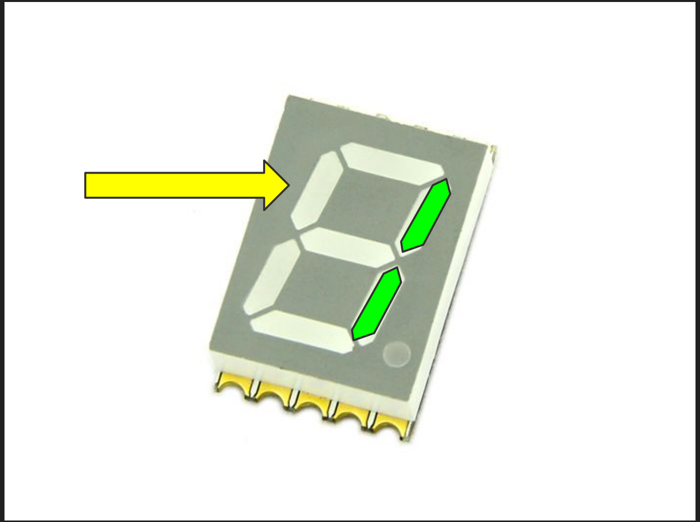
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



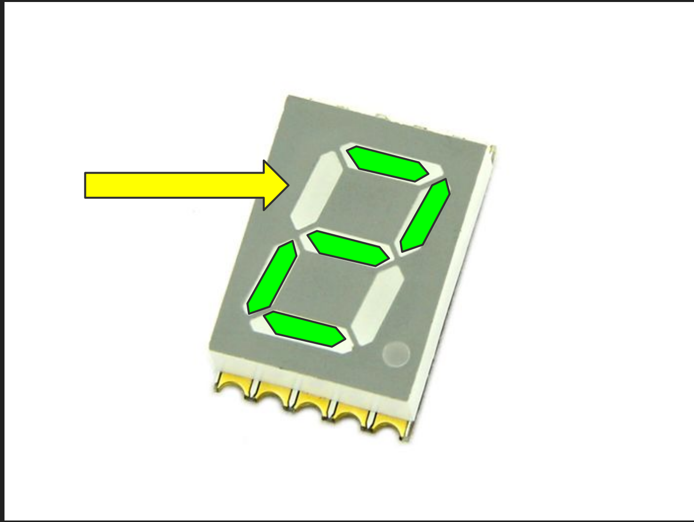
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



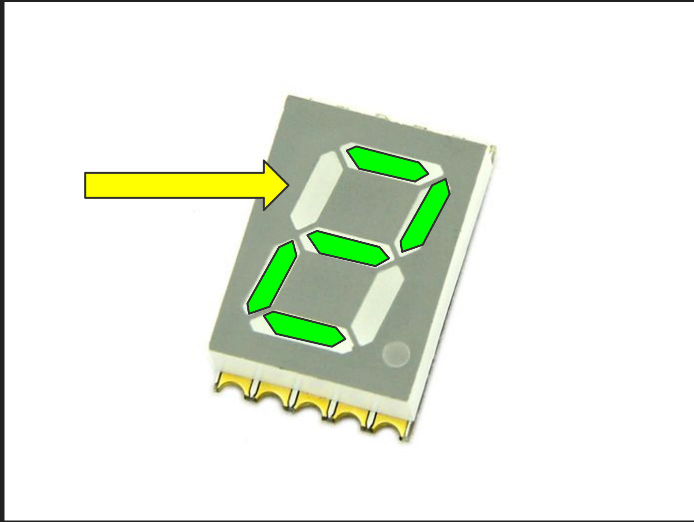
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



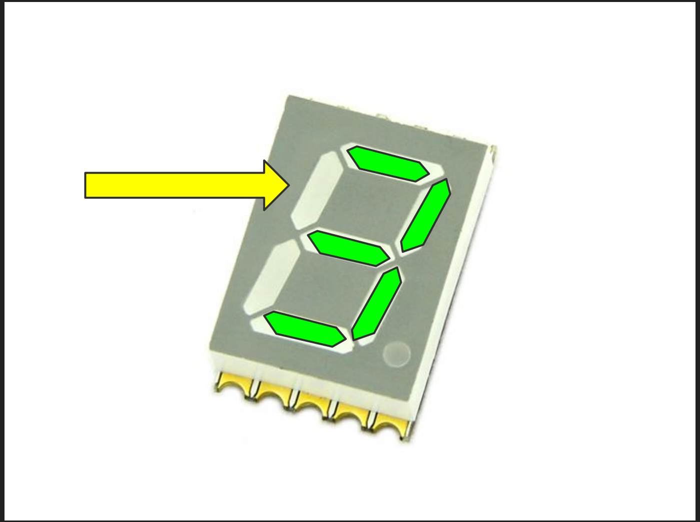
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



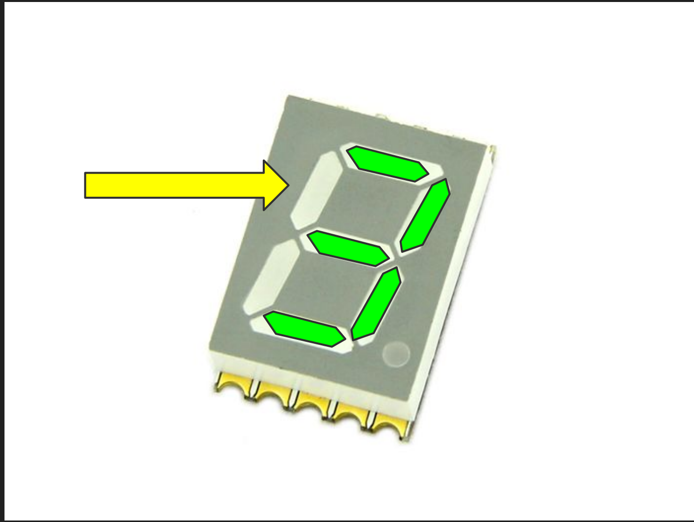
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



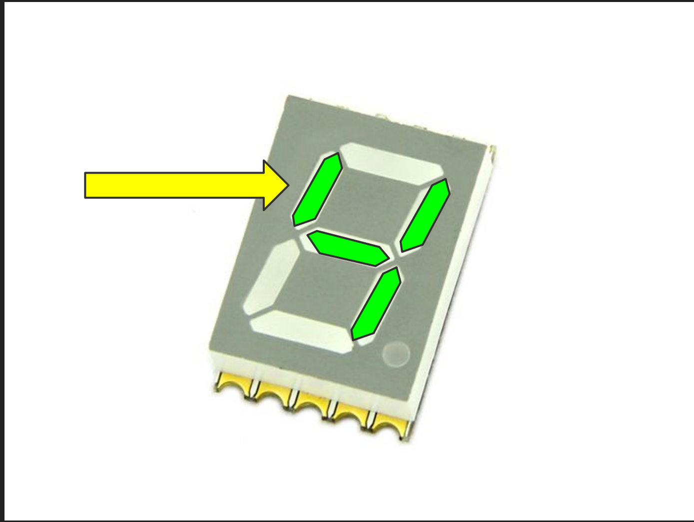
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



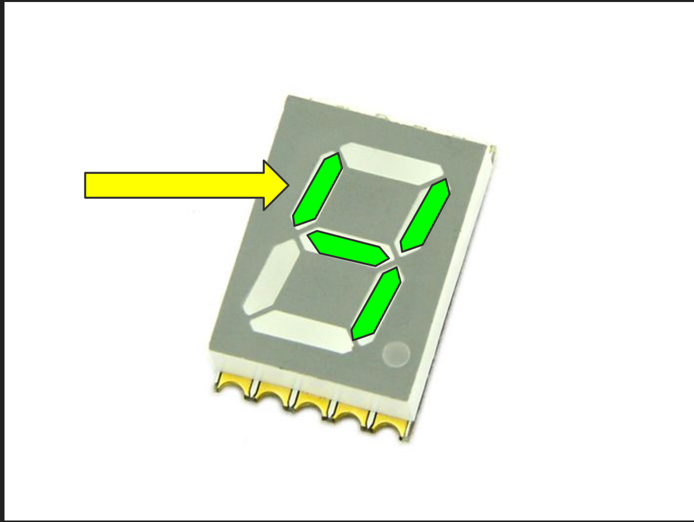
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



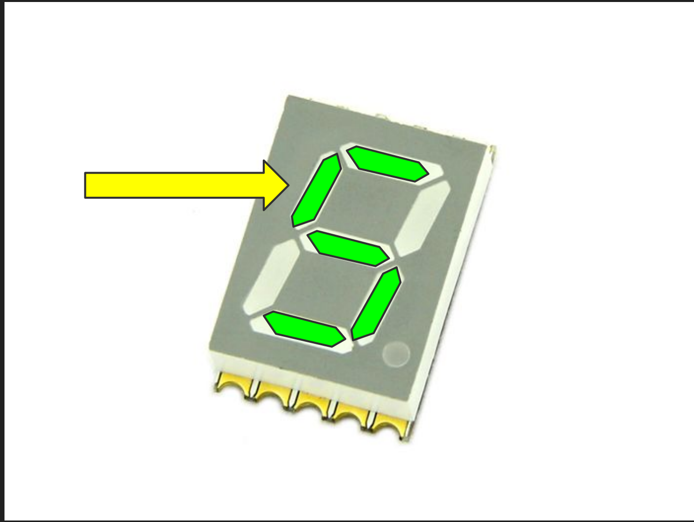
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



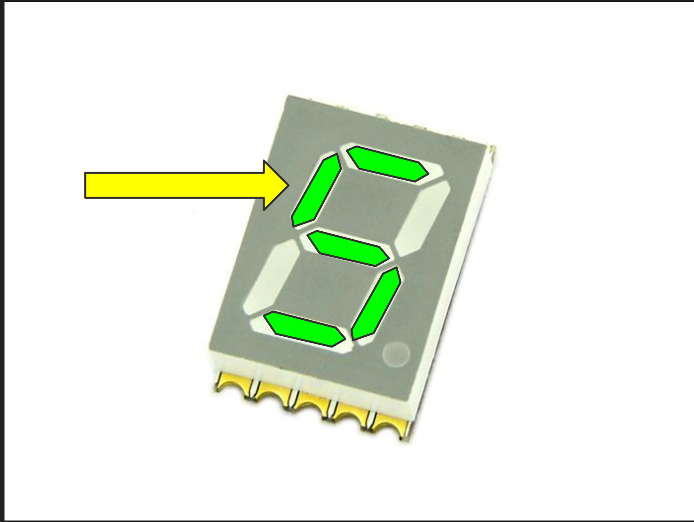
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



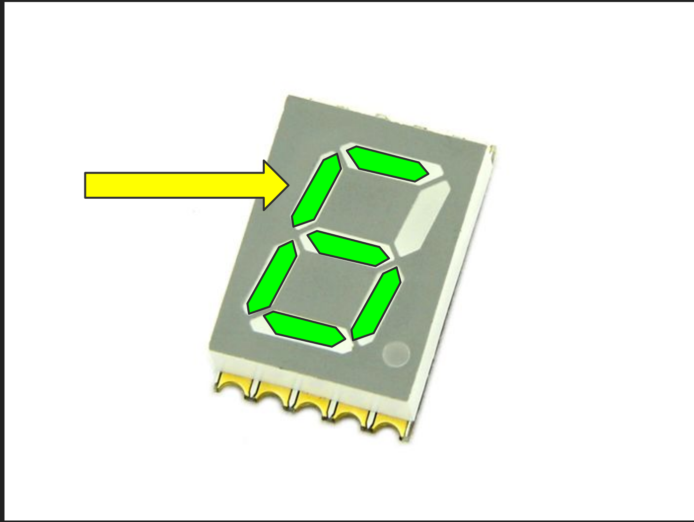
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



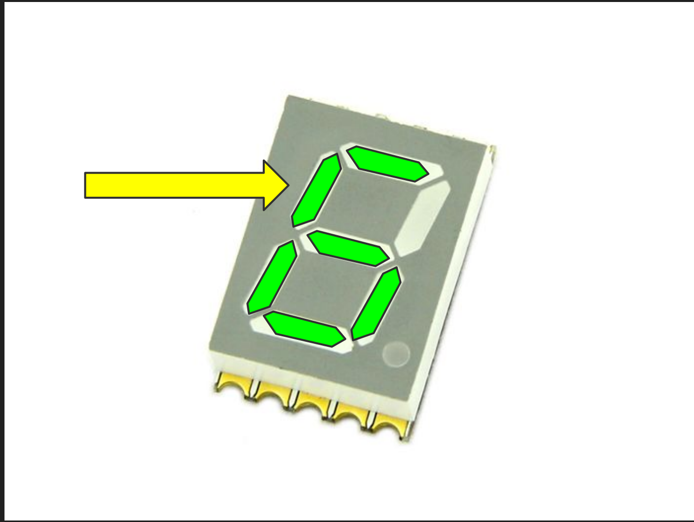
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



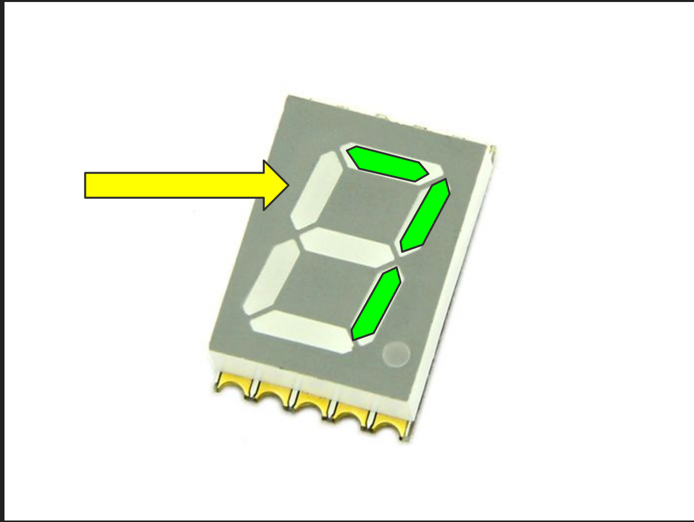
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



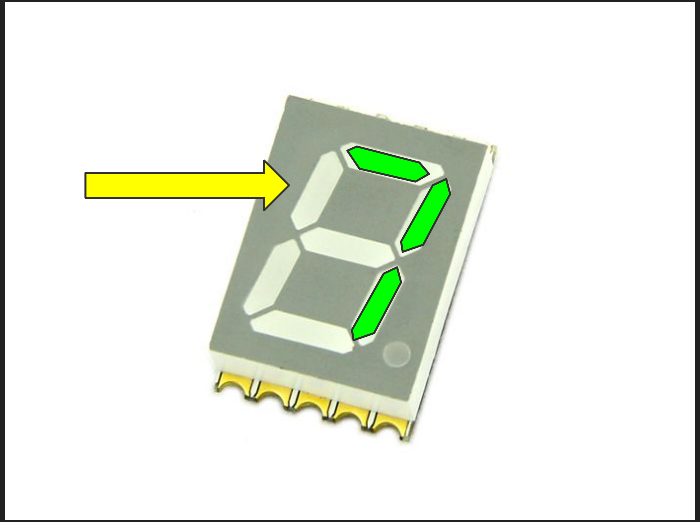
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



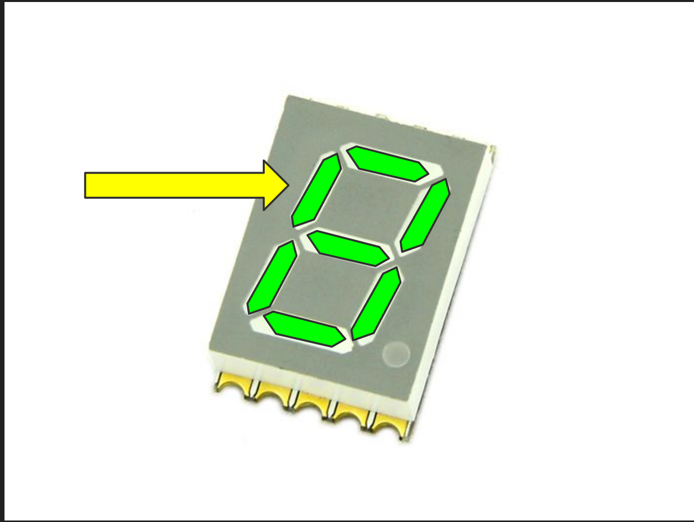
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



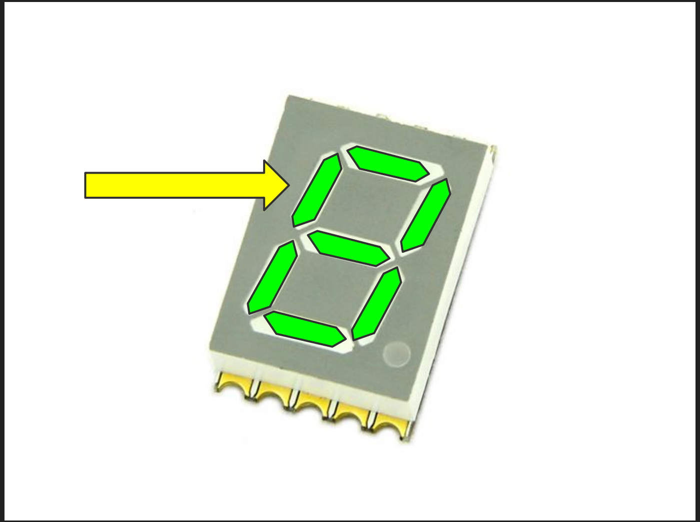
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



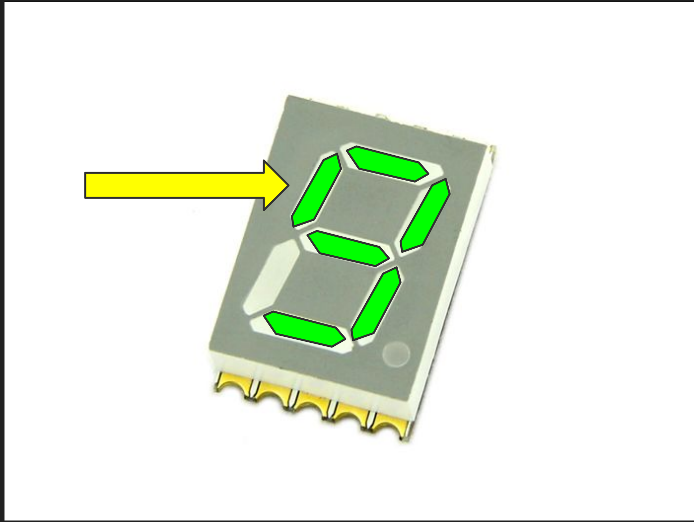
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	
9	1	0	0	1	
	x	x	x	x	



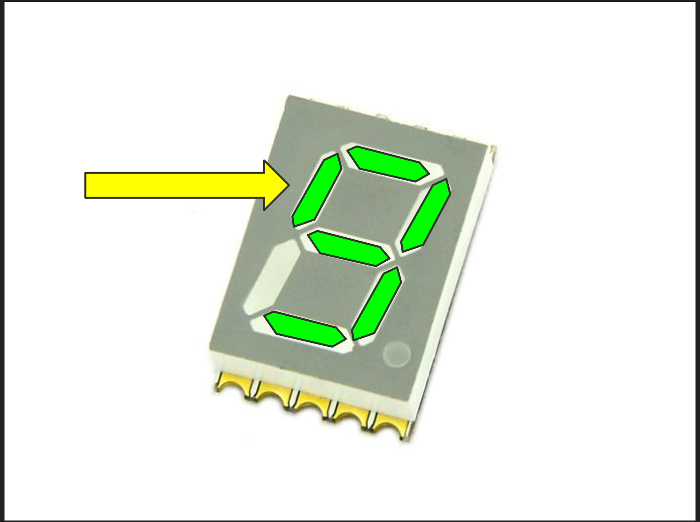
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	
	x	x	x	x	



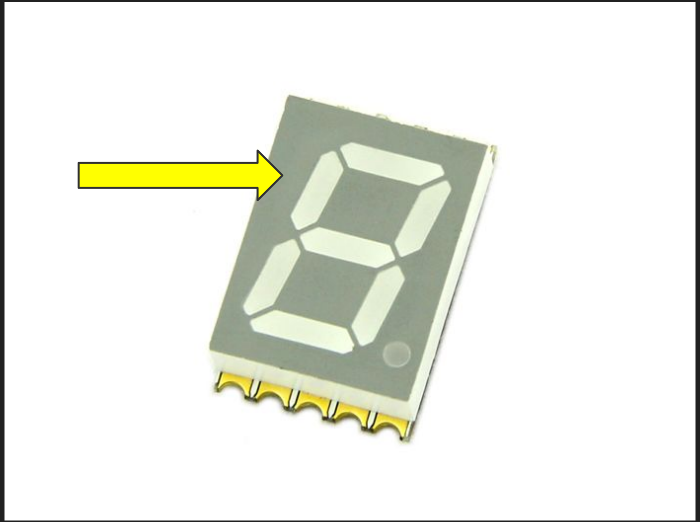
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	
	x	x	x	x	



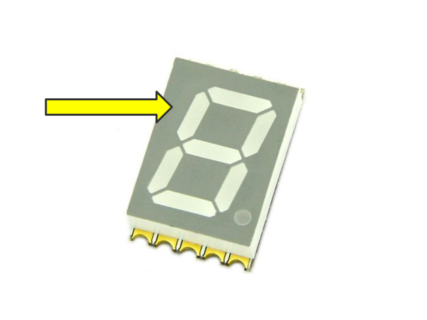
#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
	x	x	x	x	



#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
	x	x	x	x	



#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	1
	x	x	x	x	



$a \vee b \vee c \vee \neg d$

$a \vee b \vee \neg c \vee d$

$a \vee b \vee \neg c \vee \neg d$

$a \vee \neg b \vee \neg c \vee \neg d$

$(a \vee b \vee c \vee \neg d) \wedge (a \vee b \vee \neg c \vee d) \wedge (a \vee b \vee \neg c \vee \neg d) \wedge (a \vee \neg b \vee \neg c \vee \neg d)$

Statement → Circuit Diagram → transistors

Review

- Circuits/ logic gates
  - Statement → Circuit
  - Circuit → Statement
  - Creating expression (statement) from unknown
    - Conjunctive Normal Form (product of sums)
    - Disjunctive Normal Form (sum of products)