Teknik Digital + Praktikum

W3 - Reza Diharja, S.Si., M.T.

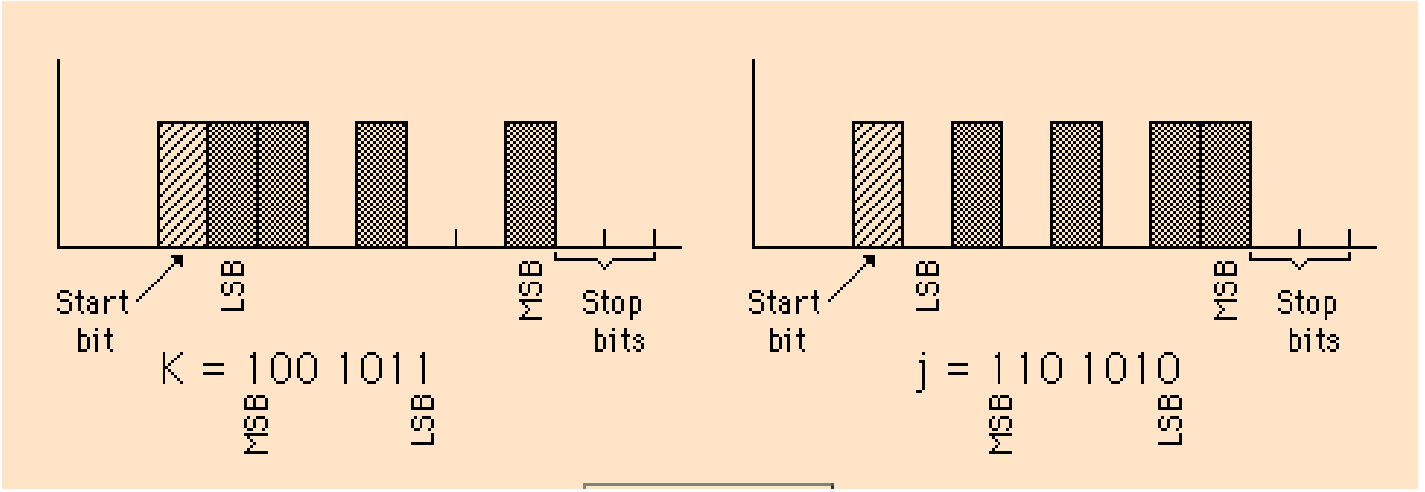
Outlines

Kode Alfanumerik

- Pada dunia komputer, pengguna pastinya akan memasukkan berbagai macam simbol, huruf, angka, dll dalam bentuk angka biner. Kode alfanumerik yang paling umum dipakai adalah kode ASCII dan yang lainnya EBCDIC, kode EBCDIC sering digunakan di aplikasi komunikasi.
- **Kode Alfanumerik/Alphanumeric codes** seringkali disebut dengan kode karakter dikarenakan oleh sifatnya. Kode-kode tersebut berbasis kode biner di mana kita juga bisa menulis data alfanumerik seperti data-data, huruf alfabet, bilangan, simbol matematika dan tanda baca dengan kode ini, sehingga lebih mudah dimengerti dan bisa diproses oleh komputer.

Kode ASCII

- Bentuk utuh kode ASCII adalah American Standard Code for Information Interchange. Merupakan kode 7-bit dan berbasis alfabet Inggris/English.
- Huruf, angka, dan perintah tunggal yang direpresentasikan oleh kata 7-bit. Umumnya, start bit dikirimkan pertama kali dan kemudian diikuti oleh kode dengan LSB. Menggunakan 7-bit kode, menghasilkan setidaknya 128 kode grup yang berbeda.



Char	7 bit ASCII	HEX	Char	7 bit ASCII	HEX	Char	7 bit ASCII	HEX
A	100 0001	41	a	110 0001	61	0	011 0000	30
B	100 0010	42	b	110 0010	62	1	011 0001	31
C	100 0011	43	c	110 0011	63	2	011 0010	32
D	100 0100	44	d	110 0100	64	3	011 0011	33
E	100 0101	45	e	110 0101	65	4	011 0100	34
F	100 0110	46	f	110 0110	66	5	011 0101	35
G	100 0111	47	g	110 0111	67	6	011 0110	36
H	100 1000	48	h	110 1000	68	7	011 0111	37
I	100 1001	49	i	110 1001	69	8	011 1000	38
J	100 1010	4A	j	110 1010	6A	9	011 1001	39
K	100 1011	4B	k	110 1011	6B	blank	010 0000	20
L	100 1100	4C	l	110 1100	6C	.	010 1110	2E
M	100 1101	4D	m	110 1101	6D	(	010 1000	28
N	100 1110	4E	n	110 1110	6E	+	010 1011	2B
O	100 1111	4F	o	110 1111	6F	$	010 0100	24
P	101 0000	50	p	111 0000	70	*	010 1010	2A
Q	101 0001	51	q	111 0001	71	)	010 1001	29
R	101 0010	52	r	111 0010	72	-	010 1101	2D
S	101 0011	53	s	111 0011	73	/	010 1111	2F
T	101 0100	54	t	111 0100	74	,	010 1100	2C
U	101 0101	55	u	111 0101	75	=	011 1101	3D
V	101 0110	56	v	111 0110	76	RETURN	000 1101	0D
W	101 0111	57	w	111 0111	77	LNFEED	000 1010	0A
X	101 1000	58	x	111 1000	78	0	011 0000	30
Y	101 1001	59	y	111 1001	79	0	011 0000	30
Z	101 1010	5A	z	111 1010	7A	0	011 0000	30

Kode EBCDIC

Kemudian terdapat kode Binary Coded Decimal Interchange Code (EBCDIC = ‘eb-si-dik’) merupakan kode alfanumerik 8-bit alphanumeric code yang mana banyak digunakan oleh IBM pada aplikasi *mainframe*-nya

- The 8-bit code in which the numerals (0-9) are represented by the 8421 BCD code preceded by 1111.
- It may be mentioned here that EBCDIC offers no technical advantage over the ASCII code and its variant ISO-8859 or Unicode. Its importance in the earlier days lay in the fact that it made it relatively easier to enter data into larger machines with punch cards.
- It can almost represent 2^3 bit code (= 256) different characters which include both lowercase and uppercase letters in addition to various other symbols and commands.

Char	EBCDIC	HEX	Char	EBCDIC	HEX	Char	EBCDIC	HEX
A	1100 0001	C1	P	1101 0111	D7	4	1111 0100	F4
B	1100 0010	C2	Q	1101 1000	D8	5	1111 0101	F5
C	1100 0011	C3	R	1101 1001	D9	6	1111 0110	F6
D	1100 0100	C4	S	1110 0010	E2	7	1111 0111	F7
E	1100 0101	C5	T	1110 0011	E3	8	1111 1000	F8
F	1100 0110	C6	U	1110 0100	E4	9	1111 1001	F9
G	1100 0111	C7	V	1110 0101	E5	blank	...	...
H	1100 1000	C8	W	1110 0110	E6	.	...	...
I	1100 1001	C9	X	1110 0111	E7	(	...	...
J	1101 0001	D1	Y	1110 1000	E8	+	...	...
K	1101 0010	D2	Z	1110 1001	E9	$	...	...
L	1101 0011	D3	0	1111 0000	F0	*	...	...
M	1101 0100	D4	1	1111 0001	F1	)	...	...
N	1101 0101	D5	2	1111 0010	F2	-	...	...
O	1101 0110	D6	3	1111 0011	F3	/	...	...

Deteksi Error dan Koreksi

- Jika kita berbicara tentang sistem digital, apakah itu komputer digital atau komunikasi digital, maka, isu pendeteksian error dan koreksi menjadi hal yang sangat perlu diperhatikan.
- Error pada *bit stream* menjadikannya terdapat *noise* dan ketidakcocokan selama proses transmisi dari transmitter ke receiver.
- Error yang tidak berhasi dideteksi dan dikoreksi kesalahannya, dapat bersifat merusak karena sistem digital sangat sensitif terhadap error dan akan terjadi malfungsi bilamana bit error rate lebih besar dari *threshold level*.
- Deteksi error dan koreksi diantaranya adalah penambahan (ekstra bit) atau bit cek dan *redundant bits*.

Metode Paritas

- Perpindahan data digital dari satu tempat ke tempat lain dapat menyebabkan eror pada proses transmisinya. Bagian penerima tidak menerima sinyal yang semestinya dikirim karena adanya derau elektrik pada proses transmisi. *Sometimes a noise pulse may be large enough to alter the logic level of the signal*.
- *For example, the transmitted sequence 1001 may be incorrectly received as 1101.*
- *In order to detect such errors* ***a parity bit*** *is often used. A parity bit is an extra 0 or 1 bit attached to a code group at transmissio*n.

Paritas Genap (Even Parity)

- *In the even parity method the value of the bit is chosen so that the total number of 1s in the code group, including the parity bit, is an even number.*
- *Sebagai contoh, dalam mengirikan data 1001, bit paritas yang digunakan adalah 0 sehingga menjadikan data 01001 yang kemudian bilangan genapnya 1. Ketika mengirimkan data 1101, maka bit paritas yang digunakan bisa saja 1, sehingga menjadikan data 11101 yang kemudian bilangan genapnya 1.*
- *For example, in transmitting 1001 the parity bit used would be 0 to give 01001, and thus an even number of 1s. In transmitting 1101 the parity bit used would be 1 to give 11101, and thus an even number of 1s.*

Paritas Ganjil (Odd Parity)

- *With odd parity the parity bit is chosen so that the total number of 1s, including the parity bit, is odd.*
- *Thus if at the receiver the number of 1s in a code group does not give the required parity, the receiver will know that there is an error and can request that the code group be retransmitted.*

Contoh 1

- Dengan menggunakan paritas ganjil, kemudian diterapkan pada **data** 1010, maka seperti apa perubahannya setelah ditambahkan bit paritas?
- Jawab:
  - Terdapat logika 1 dalam jumlah genap
  - Kemudian tambahkan 1 bit tambahan
  - Sehingga kata 1010 menjadi 10101
  - Dimana 1 adalah bit paritasnya

Contoh 2

- Dengan menggunakan paritas genap, kemudian diterapkan pada kata 1010, maka seperti apa perubahannya setelah ditambahkan bit paritas?
- Jawab:
  - Terdapat logika 1 dalam jumlah genap (2 buah)
  - Kemudian tambahkan 1 bit tambahan yakni 0
  - Sehingga kata 1010 menjadi 10100
  - Dimana 0 adalah bit paritasnya

Cek Paritas

- Cek paritas dapat dilakukan pada beberapa titik/lokasi untuk melihat kemungkinan terjadinya eror pada bit-tunggal yang mana hal tersebut bisa mengganggu paritasnya.
- Kode paritas sederhana menderita akibat 2 pembatasan.
- **Pertama**, itu tidak dapat mendeteksi eror jumlah bilangan bitnya mengalami perubahan genap. Eror bit yang sama dengan atau lebih besar dari 4 itu sangat jarang terjadi, penambahan paritas tunggal tidak dapat digunakan untuk mendeteksi eror dua bit, yang mana kemungkinan berbeda dalam media penyimpanan data di pita megnetik.
- **Kedua**, kode paritas tunggal tidak dapat digunakan untuk melokalisasi atau identifikasi bit eror meskipun memang terdapat 1 bit yang eror. Terdapat beberapa kode yang menyediakan pendeteksi mandiri bit eror tunggal beserta mekanisme koreksinya.

Example 2.6

By writing the parity code (even) and threefold repetition code for all possible four-bit straight binary numbers, prove that the Hamming distance in the two cases is at least 2 in the case of the parity code and 3 in the case of the repetition code.

Solution

The generation of codes is shown in Table 2.10. An examination of the parity code numbers reveals that the number of bit disagreements between any pair of code words is not less than 2. It is either 2 or 4. It is 4, for example, between 00000 and 10111, 00000 and 11011, 00000 and 11101, 00000 and 11110 and 00000 and 01111. In the case of the threefold repetition code, it is either 3, 6, 9 or 12 and therefore not less than 3 under any circumstances.

Kode Berulang/Repetisi

- Kode pengulangan memanfaatkan transmisi berulang dari setiap bit data dalam aliran bit. Contoh pada perulangan threefold 1 dan 0, maka akan ditansmisikan menjadi 111 dan 000.
- Pada kasus ini, pada data aliran bit yang diterima, bit terukur jumlahnya sebanyak tiga bit (1 grup), maka kemunculan error bisa dideteksi. Contohnya, jika terdapat error bit tunggal, maka 1 akan diterima 011 atau 101 atau 110 daripada 111. Sedangkan 0 akan diterima 100 atau 010 atau 001 daripada 000.
- Biasanya, aliran data dipecah menjadi beberapa blok bit dan kemudian setiap blok data dikirim beberapa kali yang telah ditentukan sebelumnya. Contohnya, jika ingin mengirimkan data sebanyak 8 bit 11011001, maka akan dipecah menjadi dua blok, masing-masing 4 bit 1101 1001.
- Pada model ***threefold repetition***, maka aliran data bit yang dikirimkan akan menjadi 110111011101100110011001.

Apakah Anda bisa menebak,
di mana kekurangan metode ini?

Aljabar Boolean

- Aljabar Boolean atau logika Boolean pertama kali dikembangkan oleh George Boole pada tahun 1840-an.
- Aljabar Boolean merupakan bentuk dari logika matematika/kalkulus yang merepresentasikan nilai kebenaran (truth values).
- Aljabar Boolean memiliki bentuk menyerupai aljabar untuk bilangan riil, namun pada operasi numeriknya memiliki bentuk yang sedikit berbeda.
- Contoh operasi pada Aljabar Boolean:
  - $xy \rightarrow xoy$
  - $x + y \rightarrow x \vee y$
  - $\sim x \rightarrow -x$

Aturan Dasar Aljabar Boolean

- Pada Aljabar Boolean hanya dikenal dua nilai, benar dan salah atau 1 dan 0.
- Operasi dasar yang terdapat pada Aljabar Boolean adalah AND, OR dan NOT.

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

Operator	Mathematics	Software	Engineering
AND	$\wedge$	&&	A*B
OR	$\vee$	||	A+B
NOT	$\neg$	!	$\overline{A}$

A	B	A ∨ B	A + B
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	2

Hukum dan Aturan-aturan pada Aljabar Boolean

Name of law	Boolean AND	Boolean OR
Commutative law	A * B = B * A	A + B = B + A
Associative law	A * (B * C) = (A * B) * C	A + (B + C) = (A + B) + C
Distributive law	A * (B + C) = (A * B) + (A * C)	A + (B * C) = (A + B) * (A + C)

A * 0 = 0	A + 1 = 1
A * 1 = A	A + 0 = A
A * A = A	A + A = A
A * $\overline{A}$ = 0	A + $\overline{A}$ = 1
A + $\overline{A}$*B = A + B	A * ($\overline{A}$ + B) = A * B
$\overline{\overline{A}}$ = A	

Related identities

$(A + AB) = A$

$(A + B) = A + B$

$(A + B) \bullet (A + C) = (A + BC)$

$A \cdot (A + B) = A$

$A + (A \cdot B) = A$

$A \cdot (\overline{A} + B) = A \cdot B$

$A + (\overline{A} \cdot B) = A + B$

A + (B * C) = (A + B) * (A + C)

Tabel Kebenaran - Contoh

- For example, this is a table for the expression F = A * (B+C)

A	B	C	F = A * (B+C)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

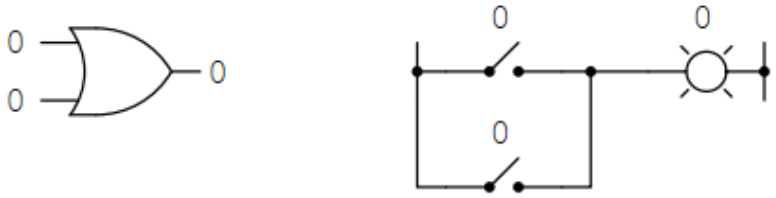
- Sometimes, you want to build a digital circuit with a given functionality that is defined only by a truth table. You can **use the so-called sum-of-products method** to find a Boolean expression that corresponds to a given truth table.

- Each of these expressions is 1 in the corresponding line and 0 in the rest. The final result **is the OR-combination** of all these expressions

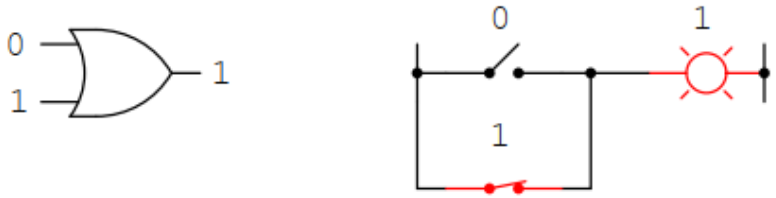
A	B	C	F = A * (B+C)	sum of products expression
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	1	$A*\overline{B}*C$
1	1	0	1	$A*B*\overline{C}$
1	1	1	1	A*B*C

- Now we know that F = + + . This is called a sum of products, even though the * and + actually mean AND and OR.

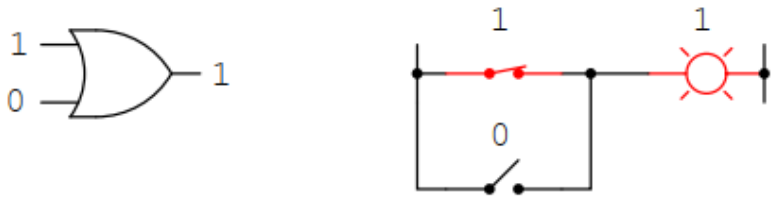
0 + 0 = 0



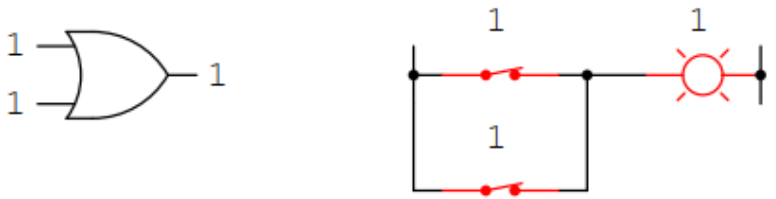
0 + 1 = 1



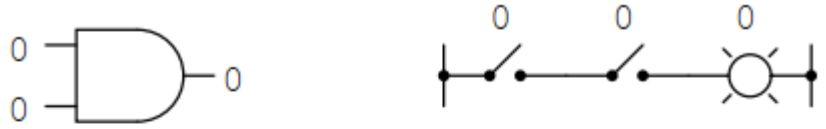
1 + 0 = 1



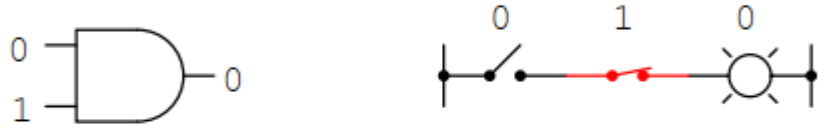
1 + 1 = 1



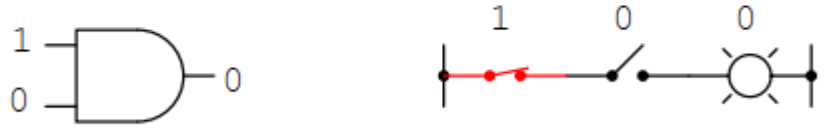
0 × 0 = 0



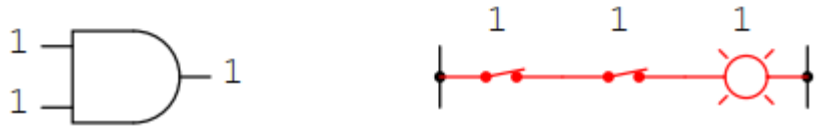
0 × 1 = 0



1 × 0 = 0

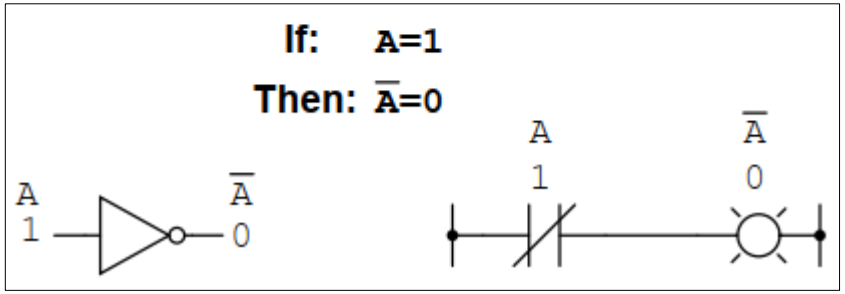


1 × 1 = 1



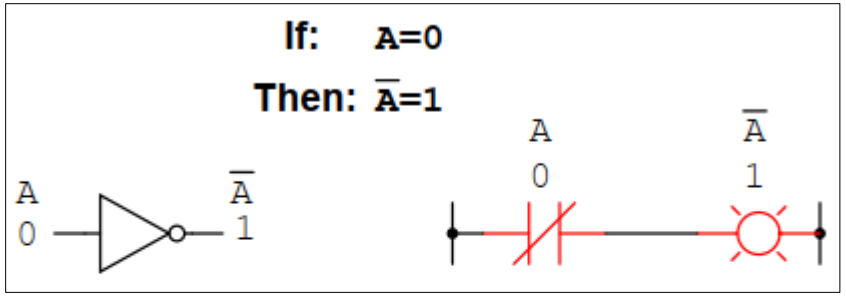
If: A=1

Then: $\overline{A}$=0

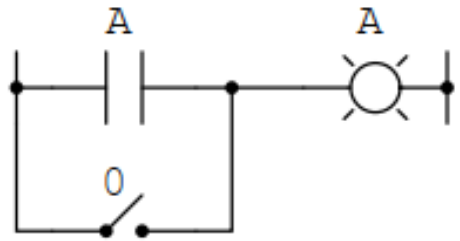


If: A=0

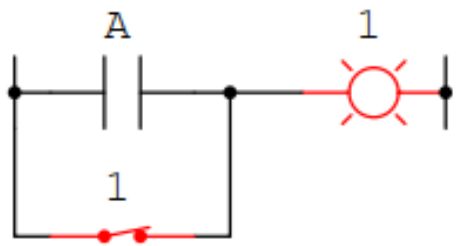
Then: $\overline{A}$=1



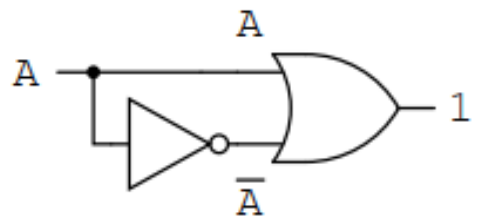
A + 0 = A

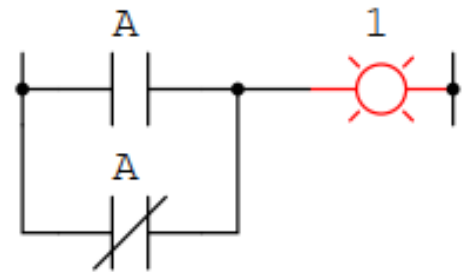


A + 1 = 1

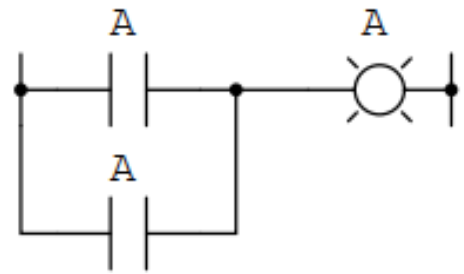


A + $\overline{A}$ = 1

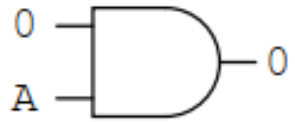


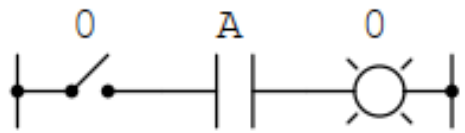


A + A = A

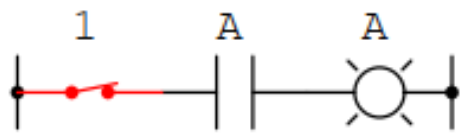


0A = 0

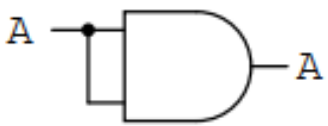


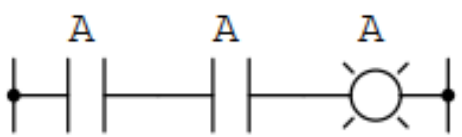


1A = A



AA = A





A$\overline{A}$ = 0