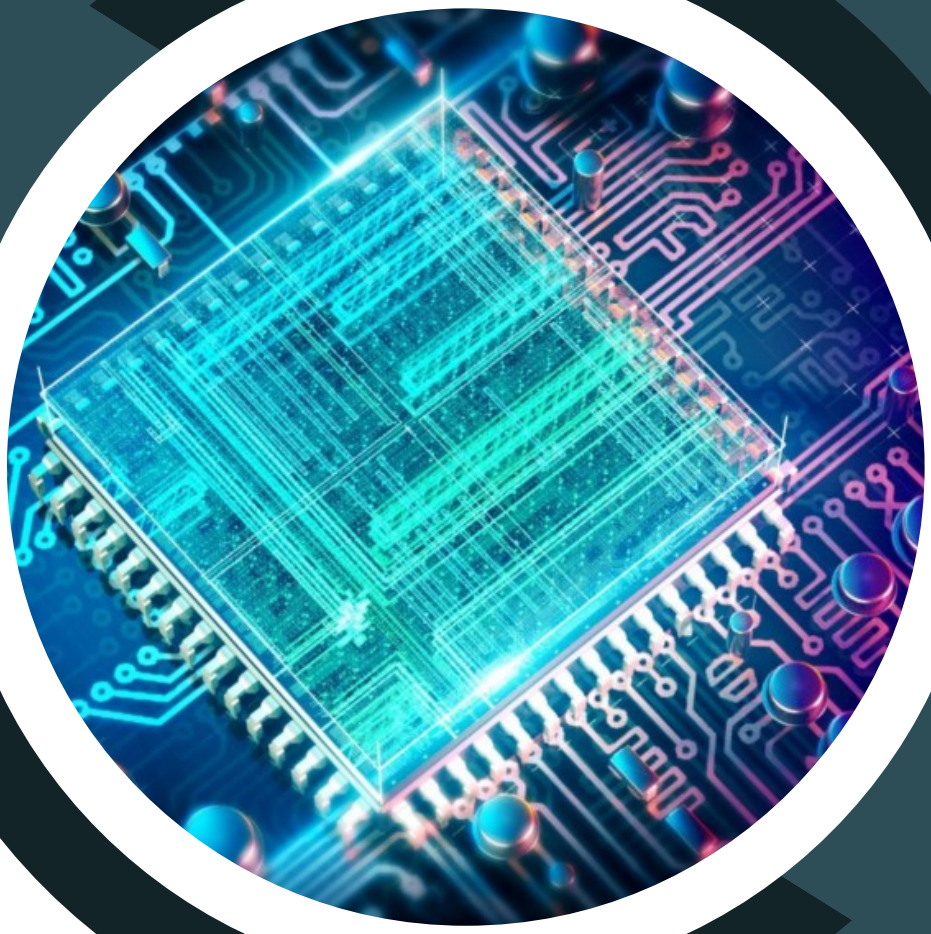
DIGITAL
ELECTRONICS



PREPARED FOR

JKE, PTSS

PREPARED BY

SURAYA BINTI ABU SEMAN

FARIZA BINTI ISHAK

AZMAN BIN MAT HUSSIN

EDITOR

SURAYA BINTI ABU SEMAN
FARIZA BINTI ISHAK
AZMAN BIN MAT HUSSIN

WRITER

SURAYA BINTI ABU SEMAN
FARIZA BINTI ISHAK
AZMAN BIN MAT HUSSIN

DESIGNER

SURAYA BINTI ABU SEMAN
FARIZA BINTI ISHAK
AZMAN BIN MAT HUSSIN

FIRST PUBLISHED 2023

PUBLISHED BY :
POLITEKNIK TUANKU SYED SIRAJUDDIN
PAUH PUTRA
02600 ARAU
PERLIS

ACKNOWLEDGE

Alhamdulillah, by the name of Allah the most gracious and the most merciful. We are thankful to Him that with His mercy and guidance in giving us patient, strength and courage to complete this Digital Electronics e-book.

We would like to express millions of thanks to our Head Department of Electrical Engineering, Mr. Shaffie Bin Husin for all his support in completing this e-book.

Besides that, we would like to express our deep gratitude to our families for their continuous support and understanding.

Last but not least, we would like also to acknowledge our sincere gratitude to all friends who helped us directly or indirectly to finish up this e-book. Only Allah can repay all the kindness. Thank you

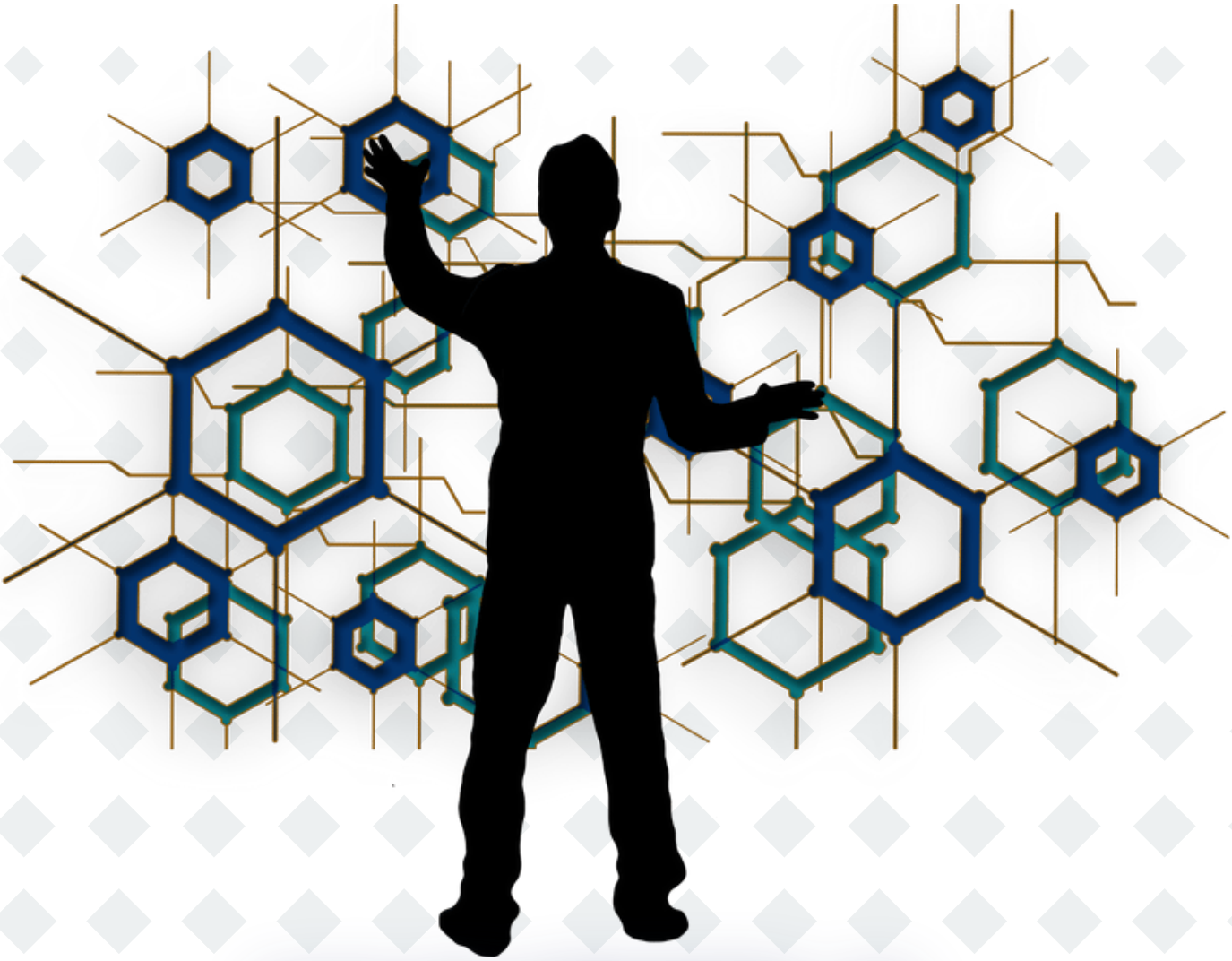
ABSTRACT

This e-book strives to provide students with a comprehension of the fundamental operating principles in digital electronics. The content of this e-book is crafted in accordance with the polytechnic syllabus designed for Electrical and Electronic Engineering students enrolled in the Digital Electronics course.

The e-book is structured around three main topics. The first delves into theoretical explanations, focusing on the basic number system. The second topic delves into the operations of Boolean Algebra, while the third elucidates the functioning of flip-flops. Each topic is complemented with tutorials and associated questions designed to assist students in mastering the concepts presented within each subject area.

TABLE OF CONTENTS

TOPIC 1



NUMBER & CODE SYSTEM

DECIMAL NUMBER SYSTEM (BASE 10)

The decimal numbering system, each position contains 10 different possible digits. The digits are 0,1,2,3,4,5,6,7,8 and 9. The weight in a decimal number is based on powers of 10.

10^5	10^4	10^3	10^2	10^1	10^0	.	10^{-1}	10^{-2}	10^{-3}
100000	10000	1000	100	10	1	.	0.1	0.01	0.001

MSB ***Decimal point*** ***LSB***

Notes:
MSB – most significant bit
LSB – least significant bit

Decimal numbers can be expressed as the sum of the products of each digit times the colum value for that digit. Thus, the number 9240 can be expressed as

$$9240 = (9 \times 1000) + (2 \times 100) + (4 \times 10) + (0 \times 1)$$

BINARY NUMBER SYSTEM (BASE 2)

The binary system with its two digits is base two system. The two binary digits (bits) are 1 and 0. The weight in a binary number is based on powers of 2.

2^0	
0	0_2
1	1_2

2^1	2^0	
0	0	00_2
0	1	01_2
1	0	10_2
1	1	11_2

2^2	2^1	2^0	
0	0	0	000_2
0	0	1	001_2
0	1	0	010_2
0	1	1	011_2
1	0	0	100_2
1	0	1	101_2
1	1	0	110_2
1	1	1	111_2

2^3	2^2	2^1	2^0	
0	0	0	0	0000_2
0	0	0	1	0001_2
0	0	1	0	0010_2
0	0	1	1	0011_2
0	1	0	0	0100_2
0	1	0	1	0101_2
0	1	1	0	0110_2
0	1	1	1	0111_2
1	0	0	0	1000_2
1	0	0	1	1001_2
1	0	1	0	1010_2
1	0	1	1	1011_2
1	1	0	0	1100_2
1	1	0	1	1101_2
1	1	1	0	1110_2
1	1	1	1	1111_2

BINARY NUMBER SYSTEM (BASE 2)

BINARY TO DECIMAL CONVERSION

SUM-OF-WEIGHT-METHODS

One way to find the binary number that is equivalent to a given decimal number is to determine the set of binary weight whose sum is equal to the decimal number.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}	2^{-3}
128	64	32	16	8	4	2	1	.	0.5	0.25	0.125

EXAMPLE 1:

Convert the binary number 1101101 to decimal.

Solution:

Weight	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1
Binary number		1	1	0	1	1	0	1

$\therefore$ 1101101 = 64 + 32 + 8 + 4 + 1 = **109**

BINARY NUMBER SYSTEM (BASE 2)

EXAMPLE 2:

Convert the binary number 0.101 to decimal.

Solution:

	.	2^{-1}	2^{-2}	2^{-3}
0	.	0.5	0.25	0.125
0	**.**	**1**	**0**	**1**

0.101= 0.5 + 0.125=0.625

DECIMAL TO BINARY CONVERSION

REPEATED DIVISION BY 2 METHODS

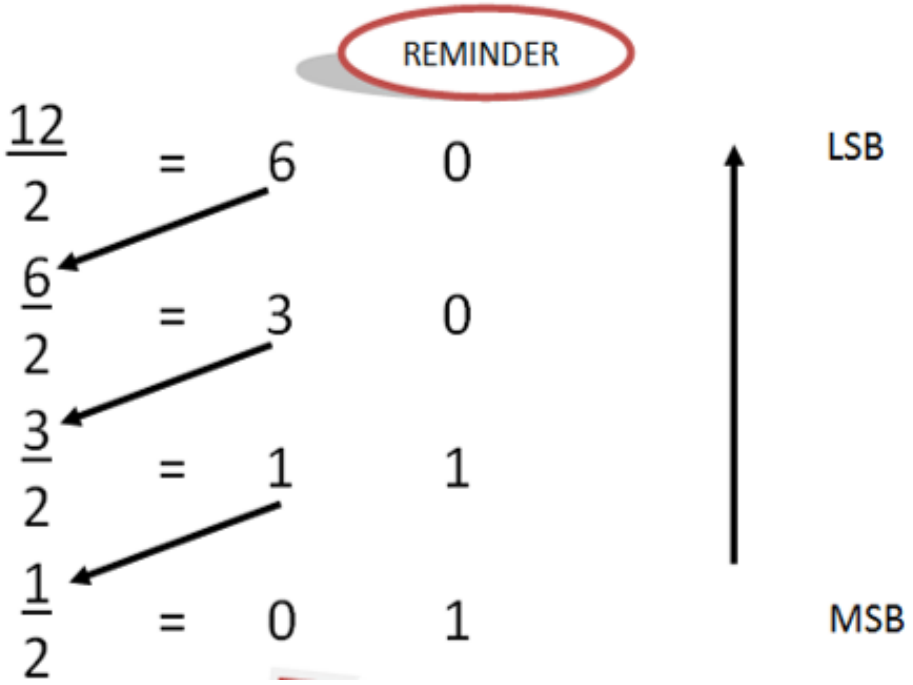
One way to convert the decimal number to binary, begin by dividing decimal number by 2. Then divide each resulting quotient by 2 until there is a 0 whole number quotient. The remainders generated by each division form the binary number.

That is equivalent to a given decimal number is to determine the set of binary weight whose sum is equal to the decimal number.

BINARY NUMBER SYSTEM (BASE 2)

The first remainder to be produced is the **LSB** (least significant bit) in the binary number, and the last remainder to be produced is the **MSB** (Most significant).

EXAMPLE :



BINARY NUMBER SYSTEM (BASE 2)

DECIMAL FRACTION TO BINARY CONVERSION

Decimal fractions can be converted to binary by repeated multiplication by 2.

EXAMPLE :

					Carry
0.3125	X	2	=	0.625	0
0.625	X	2	=	1.25	1
0.25	X	2	=	0.50	0
0.50	X	2	=	1.00	1

MSB

LSB

∴ The decimal fraction 0.3125 to binary is **.0101**

BINARY NUMBER SYSTEM (BASE 2)

BINARY ARITHMETIC (ADDITION)

The four basic rules for adding binary digits (bits) are as follows

0+0 = 0; sum of 0 with a carry of **0**
0+1 = 1; sum of 1 with a carry of **0**
1+0 = 1; sum of 1 with a carry of **0**
1+1 = 10; sum of 1 with a carry of **1**

EXAMPLE 1:

100 + 10

	1	0	0		4
+		1	0	+	2
	1	1	0		6

$100_2 = 4_{10}$
$10_2 = 2_{10}$
$110_2 = 6_{10}$

EXAMPLE 2:

111 + 11

	1	1	1		7
+		1	1	+	3
1	0	1	0		10

$111_2 = 7_{10}$
$11_2 = 3_{10}$
$1010_2 = 10_{10}$

BINARY NUMBER SYSTEM (BASE 2)

BINARY ARITHMETIC (SUBTRACTION)

The four basic rules for adding binary digits (bits) are as follows

0 - 0 = 0
1 - 1 = 0
1 - 0 = 1
10 -1 = 1; 0-1 with a borrow of **1**

EXAMPLE :

Left column:
When a 1 is borrowed, a 0 is left, so 0 – 0 = 0

Middle column:
Borrow 1 from next column to the left, making a 10 in this column, then 10 – 1 = 1

	0	10			
	1	0	1		5
–	0	1	1	–	3
	0	1	0		2

$101_2 = 5_{10}$

$011_2 = 3_{10}$

BINARY NUMBER SYSTEM (BASE 2)

FIRST COMPLIMENT OF BINARY NUMBERS

The first complement of a binary number is found by changing all 1s to 0s and all 0s to 1s.

1	0	1	1	0	0	1	0	**Binary number**
↓	↓	↓	↓	↓	↓	↓	↓	
0	1	0	0	1	1	0	1	**1's complement**

SECOND COMPLIMENT OF BINARY NUMBERS

The second complement of a binary number is found by adding 1 to the LSB of the first complement

2's complement = 1's complement + 1

EXAMPLE :

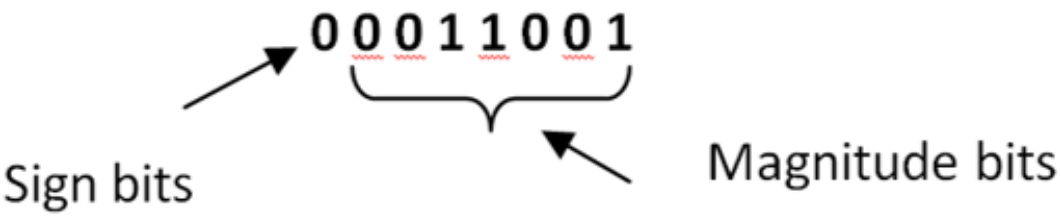
	1	0	1	1	0	0	1	0	**Binary number**
	↓	↓	↓	↓	↓	↓	↓	↓	
	0	1	0	0	1	1	0	1	**1's complement**
+								1	**Add 1**
	0	1	0	0	1	1	1	0	**2's complement**

BINARY NUMBER SYSTEM (BASE 2)

SIGN NUMBERS

The left most bit in a signed binary number is teh sign bit, which tells you whether the number is **positive** or **negative.**

A 0 sign bit indicates a positive number, and a 1 sign bit indicates a negative number



OCTAL NUMBER SYSTEM (BASE 8)

The hexadecimal number system has a base eight. The octal number system is composed of eight digits, which are

0, 1, 2, 3, 4, 5, 6, 7

To count above 7, begin another column and start over

10, 11, 12, 13, 14, 15, 16, 17

OCTAL TO DECIMAL CONVERSION

A octal number can be converted to its decimal equivalent by using the fact that each octal digit position has a weight is a power of 8

Weight	8^6	8^5	8^4	8^3	8^2	8^1	8^0
	262,144	32,768	4096	512	64	8	1

OCTAL NUMBER SYSTEM (BASE 8)

EXAMPLE :

Convert the octal number 2374 to decimal.

Solution:

Weight	8^3	8^2	8^1	8^0
	512	64	8	1
Octal number	2	3	7	4
	2 x 512	3 x 64	7 x 8	4 x 1
	1024	192	56	4
Decimal number	$1024 + 192 + 56 + 4 = 1276_{10}$			

OCTAL NUMBER SYSTEM (BASE 8)

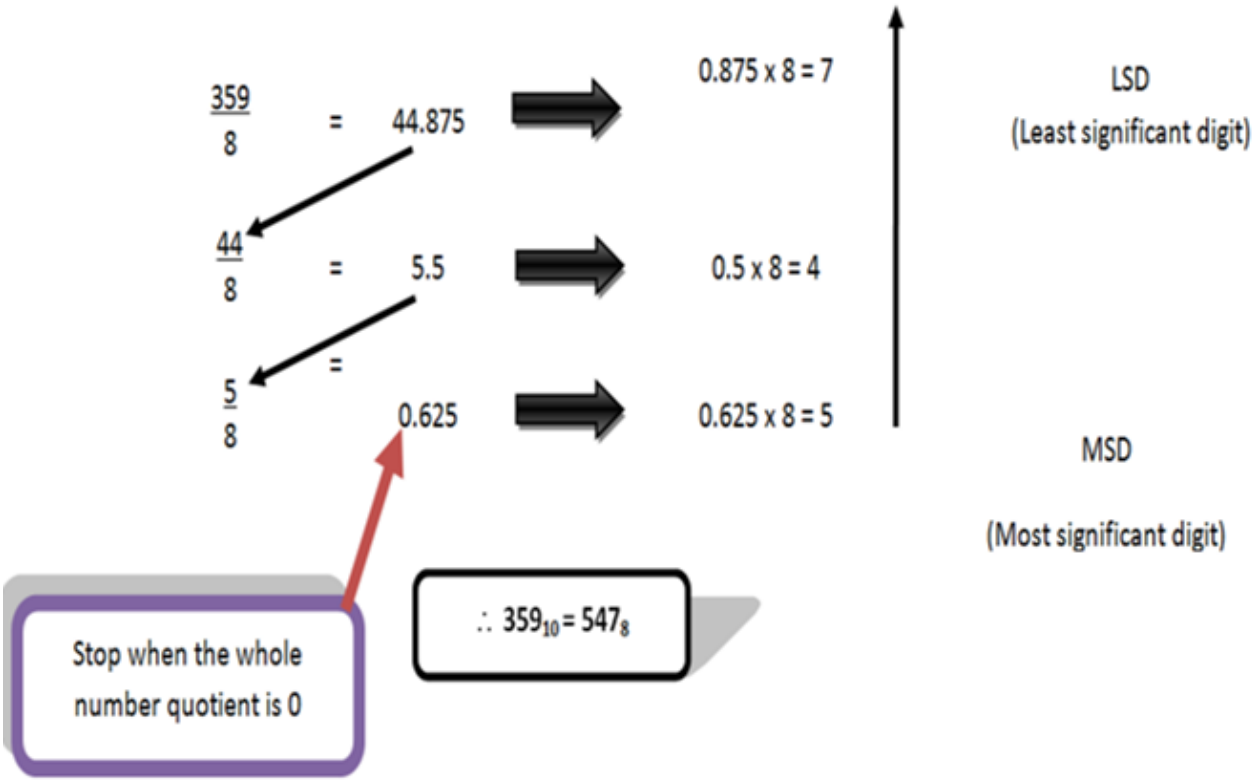
DECIMAL TO OCTAL CONVERSION

A method of converting a decimal number to an octal number is repeated division by 8 method

EXAMPLE :

Convert the decimal number 359 to octal

Solution:



OCTAL NUMBER SYSTEM (BASE 8)

DECIMAL TO OCTAL CONVERSION

To convert an octal number to a binary number, simply replace each digit with the appropriate **three bits**.

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

EXAMPLE 1:

Convert each the following octal number to binary.

Solution:

Octal Digit	140			7526			
	1	4	0	7	5	2	6
Binary	001	100	000	111	101	010	110

OCTAL NUMBER SYSTEM (BASE 8)

BINARY TO OCTAL CONVERSION

The procedure is as follows:

- Start from right, group of three bits and moving to left
- Convert each 3-bit group to octal
- Add zeros to make a complete group

EXAMPLE 1:

a) 110|101	110	101	$= 65_8$
	6	5	

EXAMPLE 2:

b) 101|111|001	101	111	001	$= 571_8$
	5	7	1	

EXAMPLE 3:

c) 100|110|011|010	100	110	011	010	=
	4	6	3	2	4632_8

EXAMPLE 4:

d) 11|010|000|100	011	010	000	100	=
	3	2	0	4	3204_8

HEXADECIMAL NUMBER SYSTEM (BASE 16)

The hexadecimal number system has sixteen characters. The hexadecimal number system has a base of sixteen. The hexadecimal number system is described as a 16 digit number representation of numbers from 0 - 9 and digits from A - F.

In other words, the first 9 numbers or digits are represented as numbers while the next 6 digits are represented as symbols from A - F. Hexadecimal is very similar to the decimal number system that has a base number of 9.

Therefore, after 9 digits, the 10th digit is represented as a symbol - 10 as A, 11 as B, 12 as C, 13 as D, 14 as E, and 15 as F. Hence, the 16 digits are 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

How do you count in hexadecimal once you get to F?

10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1A, 1B, 1C, 1D, 1E, 1F,

20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2A, 2B, 2C, 2D, 2E, 2F, ...

HEXADECIMAL NUMBER SYSTEM (BASE 16)

BINARY TO HEXADECIMAL CONVERSION

Converting a binary number to hexadecimal is a straight forward procedure. Simply breaks the binary number into 4-bit groups, starting at the right-most bit and replace each 4 bit group with the equivalent hexadecimal symbol.

01101110	Original binary number
0110 1110	Grouped binary number
6 E	Hex value of grouped binary number
$(01101110)_2 = (6E)_{16}$	Converted value

HEXADECIMAL NUMBER SYSTEM (BASE 16)

EXAMPLE 1:

Convert the binary number 10101011_2 into hexadecimal.

$10101011_2 = 1010_2 \mid 1011_2$

From the table, we have

$1010_2 = A_{16}$

$1011_2 = B_{16}$

Thus, $10101011_2 = (AB)_{16}$

EXAMPLE 2:

Convert $(00001011)_2$ into a hexadecimal number system by direct method.

Solution:

$(00001011)_2 = (0000) \mid (1011)$

Here,

$0000_2 = 0_{16}$

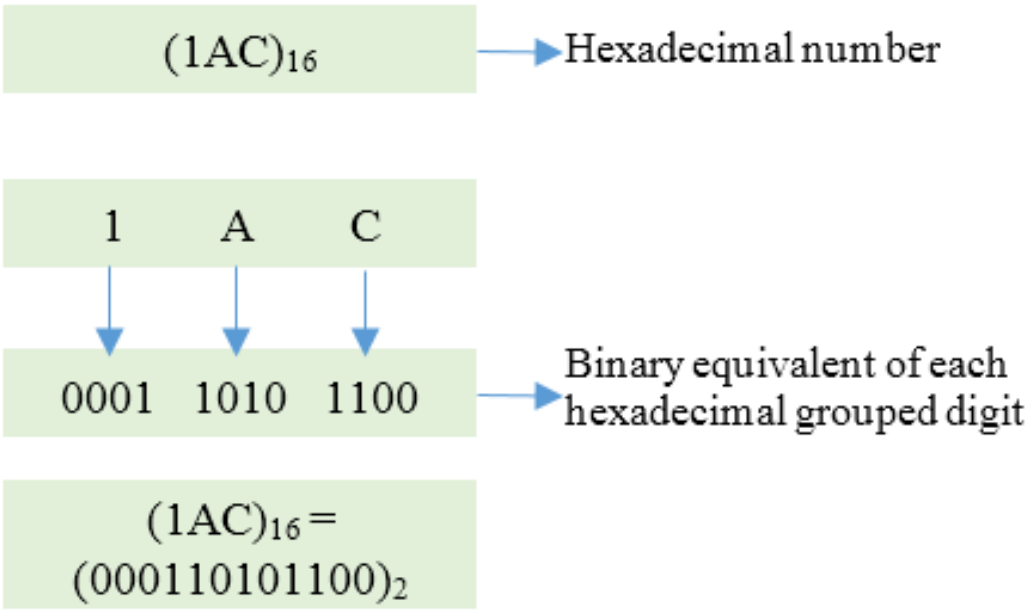
$1011_2 = B_{16}$

So, $(00001011)_2=(0B)_{16}$

HEXADECIMAL NUMBER SYSTEM (BASE 16)

HEXADECIMAL TO BINARY CONVERSION

The hexadecimal to binary conversion can occur in two methods - First, after the hexadecimal is converted to a decimal number, we convert the decimal number by using the division process to obtain the binary number. Second, we can directly use the hexadecimal to decimal to binary conversion table. Let us look at the steps of both methods.



HEXADECIMAL NUMBER SYSTEM (BASE 16)

EXAMPLE 1:

a) $10A4_{16}$	1	0	A	4	= 0001 0000 1010 0100
Binary number	0001	0000	1010	0100	

EXAMPLE 2:

b) $CF8E_{16}$	C	F	8	E	= 1100 1111 1000 1110
Binary number	1100	1111	1000	1110	

EXAMPLE 3:

c) 9742_{16}	9	7	4	2	= 1001 0111 0100 0010
Binary number	1001	0111	0100	0010	

HEXADECIMAL NUMBER SYSTEM (BASE 16)

HEXADECIMAL TO DECIMAL CONVERSION

One way to find the decimal equivalent of a hexadecimal number is to first convert the hexadecimal number to binary and then convert from binary to decimal.

Example:

Convert 7CF (hex) to decimal.

Solution:

Given hexadecimal number is 7CF.

In hexadecimal system,

7 = 7

C = 12

F = 15

To convert this into a decimal number system, multiply each digit with the powers of 16 starting from units place of the number.

$$
\begin{aligned}
7CF &= (7 \times 162) + (12 \times 161) + (15 \times 160) \\
&= (7 \times 256) + (12 \times 16) + (15 \times 1) \\
&= 1792 + 192 + 15 \\
&= 1999
\end{aligned}
$$

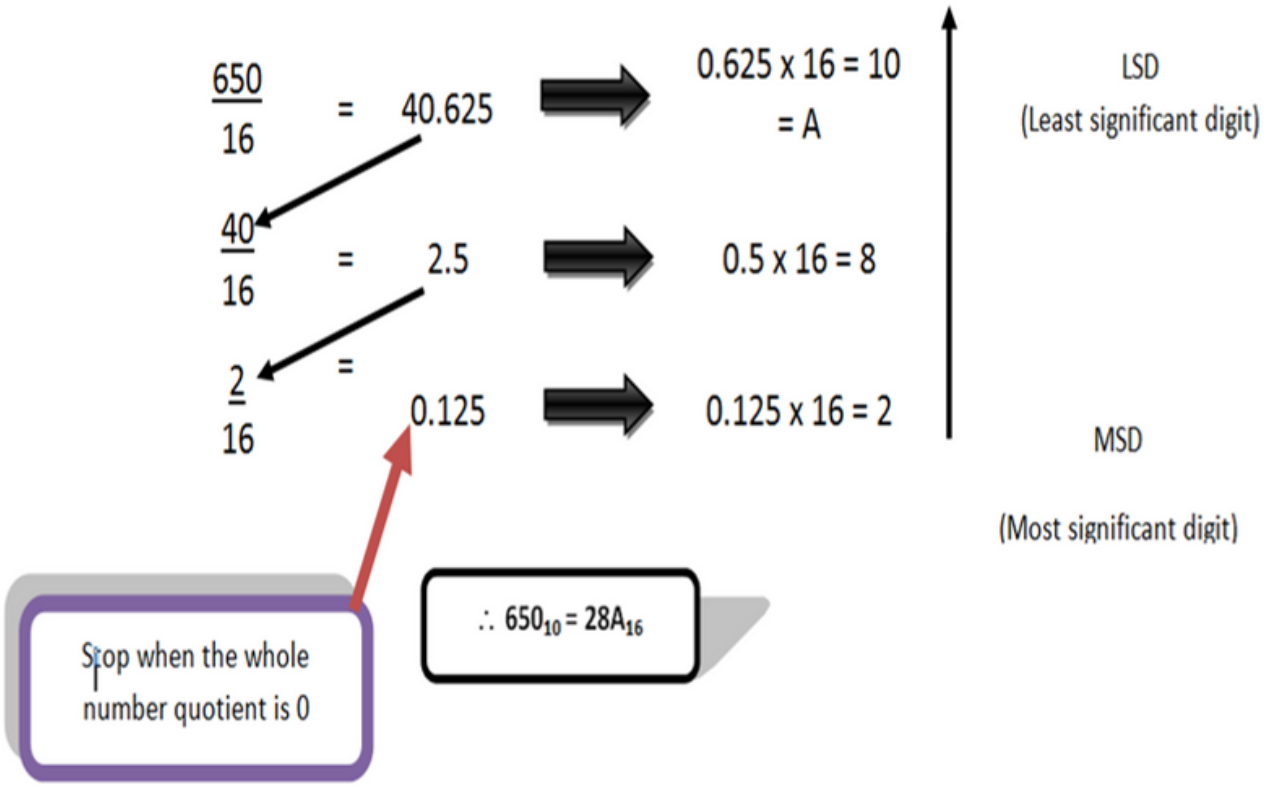
HEXADECIMAL NUMBER SYSTEM (BASE 16)

DECIMAL TO HEXADECIMAL CONVERSION

Repeated division of a decimal number by 16 will produce the equivalent hexadecimal number, formed by remainders of the divisions.

Example: convert the decimal number 650 to hexadecimal by repeated division by 16.

Solution:



HEXADECIMAL NUMBER SYSTEM (BASE 16)

HEXADECIMAL ADDITION

When adding two hexadecimal numbers, use the following rules (Decimal numbers are indicated by a subscript 10)

In any given column of an addition problem, think of the two hexadecimal digits in terms of their decimal values. For instance, 516 = 510 and C16 = 1210.

If the sum of these two digits is 1510 or less, bring down the corresponding hexadecimal digit.

If the sum of these two digits is greater than 1510 bring down the amount of the sum that exceeds 1610 and carry a 1 to the next column.

HEXADECIMAL NUMBER SYSTEM (BASE 16)

Example:
Add the following hexadecimal numbers:

1. $23_{16} + 16_{16}$
2. $58_{16} + 22_{16}$
3. $2B_{16} + 84_{16}$
4. $DF_{16} + AC_{16}$

Solution:

QUESTION	RIGHT COLUMN	LEFT COLUMN
$23_{16} + 16_{16}$ $\begin{array}{r} 2\ 3_{16} \\ +\quad 1\ 6_{16} \\ \hline 3\ 9_{16} \end{array}$	$3_{16} + 6_{16} = 3_{10} + 6_{10} = 9_{10} = 9_{16}$	$2_{16} + 1_{16} = 2_{10} + 1_{10} = 3_{10}$
$58_{16} + 22_{16}$ $\begin{array}{r} 5\ 8_{16} \\ +\quad 2\ 2_{16} \\ \hline 7\ A_{16} \end{array}$	$8_{16} + 2_{16} = 8_{10} + 2_{10} = 10_{10} = A_{16}$	$5_{16} + 2_{16} = 5_{10} + 2_{10} = 7_{10}$
$2B_{16} + 84_{16}$ $\begin{array}{r} 2\ B_{16} \\ +\quad 8\ 4_{16} \\ \hline A\ F_{16} \end{array}$	$B_{16} + 4_{16} = 11_{10} + 4_{10} = 15_{10} = 7_{16}$	$2_{16} + 8_{16} = 2_{10} + 8_{10} = 10_{10} = A_{16}$
$DF_{16} + AC_{16}$ $\begin{array}{r} D\ F_{16} \\ +\quad A\ C_{16} \\ \hline 1\ 8_{16}\quad B_{16} \end{array}$	$F_{16} + C_{16} = 15_{10} + 12_{10} = 27_{10}$ $27_{10} - 16_{10} = 11_{16} = B_{16}$ with a 1 carry	$D_{16} + A_{16} = 13_{10} + 10_{10} + 1 = 24_{10}$ $24_{10} - 16_{10} = 8_{10} = 8_{16}$ with a 1 carry

HEXADECIMAL NUMBER SYSTEM (BASE 16)

HEXADECIMAL SUBSTRACTION

Method 1:

- ✓Convert the hexadecimal number to binary.
- ✓Take 2's complement of the binary number
- ✓Convert the result to hexadecimal

hexadecimal → binary → 2's complement in binary → 2's complement in hexadecimal

Example:

2A → 00101010 → 11010110 → D6

Method 2:

- ✓ Subtract the hexadecimal number from the maximum hexadecimal number and add 1

hexadecimal → subtract from maximum → 1's complement in hexadecimal plus 1 → 2's complement in hexadecimal

HEXADECIMAL NUMBER SYSTEM (BASE 16)

Example:

2A → FF - 2A → D5 + 1 → D6

Method 3:

- ✓Write the sequence of single hexadecimal digits
- ✓Write the sequence in reverse below the forward sequence.
- ✓The 1's compliment of each hex digit directly below it.
- ✓Add 1 to the resulting number to get the 2's complement.

hexadecimal → 0 1 2 3 4 5 6 7 8 9 A B C D E F / F E D C B A 9 8 7 6 5 4 3 2 1 0 → 1's complement in hexadecimal plus 1 → 2's complement in hexadecimal

Example:

2A → 0 1 2 3 4 5 6 7 8 9 A B C D E F / F E D C B A 9 8 7 6 5 4 3 2 1 0 → D5 + 1 → D6

HEXADECIMAL NUMBER SYSTEM (BASE 16)

Example:

2A → FF - 2A → D5 + 1 → D6

Method 3:

- ✓Write the sequence of single hexadecimal digits
- ✓Write the sequence in reverse below the forward sequence.
- ✓The 1's compliment of each hex digit directly below it.
- ✓Add 1 to the resulting number to get the 2's complement.

hexadecimal → 0 1 2 3 4 5 6 7 8 9 A B C D E F / F E D C B A 9 8 7 6 5 4 3 2 1 0 → 1's complement in hexadecimal plus 1 → 2's complement in hexadecimal

Example:

2A → 0 1 2 3 4 5 6 7 8 9 A B C D E F / F E D C B A 9 8 7 6 5 4 3 2 1 0 → D5 + 1 → D6

BINARY CODED DECIMAL (BCD)

The binary coded decimal system is used to represent each of the 10 decimal digits (0 through 9) as a 4 bit binary code.

BCD 4-bit	Digit decimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

BCD TO BINARY NUMBER CONVERSION

Example: Convert BCD number 3906 to Binary number.

Solution:

3	9	0	6
0011	1001	0000	0110

$\therefore 3906_{10} = 0011\ 1001\ 0000\ 0110$

BINARY CODED DECIMAL (BCD)

Example: convert BCD number 0011 1001 0000 0110 to binary

Solution:

BCD 0011 1001 0000 0110 ➔ **Decimal** 3906

decimal 3906 ➔ **Binary** 111101000010

2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
2048	1024	512	256	128	64	32	16	8	4	2	1
1	1	1	1	0	1	0	0	0	0	1	0

∴ 0011 1001 0000 0110 (BCD) = 111101000010_2

ASCII CODE

(AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE) ASCII CODE

To get information into and out of a computer, we need more than just numeric representations; we also have to take care of all the letters and symbols used in day-to-day processing. We need a special code to represent all alphanumeric data (letters, symbols and numbers).
The ASCII code uses 7 bits to represent all the alphanumeric data used in computer I/O.

ASCII TABLE

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

REVIEW QUESTION

QUESTION 1

Change the BCD code 100101111000 to its equivalent binary and octal number.

QUESTION 2

Solve the 8-bits addition of decimal number below using 2's complement.

QUESTION 3

Show TWO (2) arithmetic operations that can be performed by shift register with using original decimal number 22 for each operation.

QUESTION 4

Convert the decimal number 39 to its octal equivalent and binary number 1011000001010101 to its hexadecimal equivalent.

REVIEW QUESTION

QUESTION 5

Carry out the addition for +4 and -6 in 8 bits by using 2's complement method.

QUESTION 6

Determine the decimal values 178 to binary and BCD equivalent.

QUESTION 7

Solve the 8-bits addition of decimal number below using 2's complement.

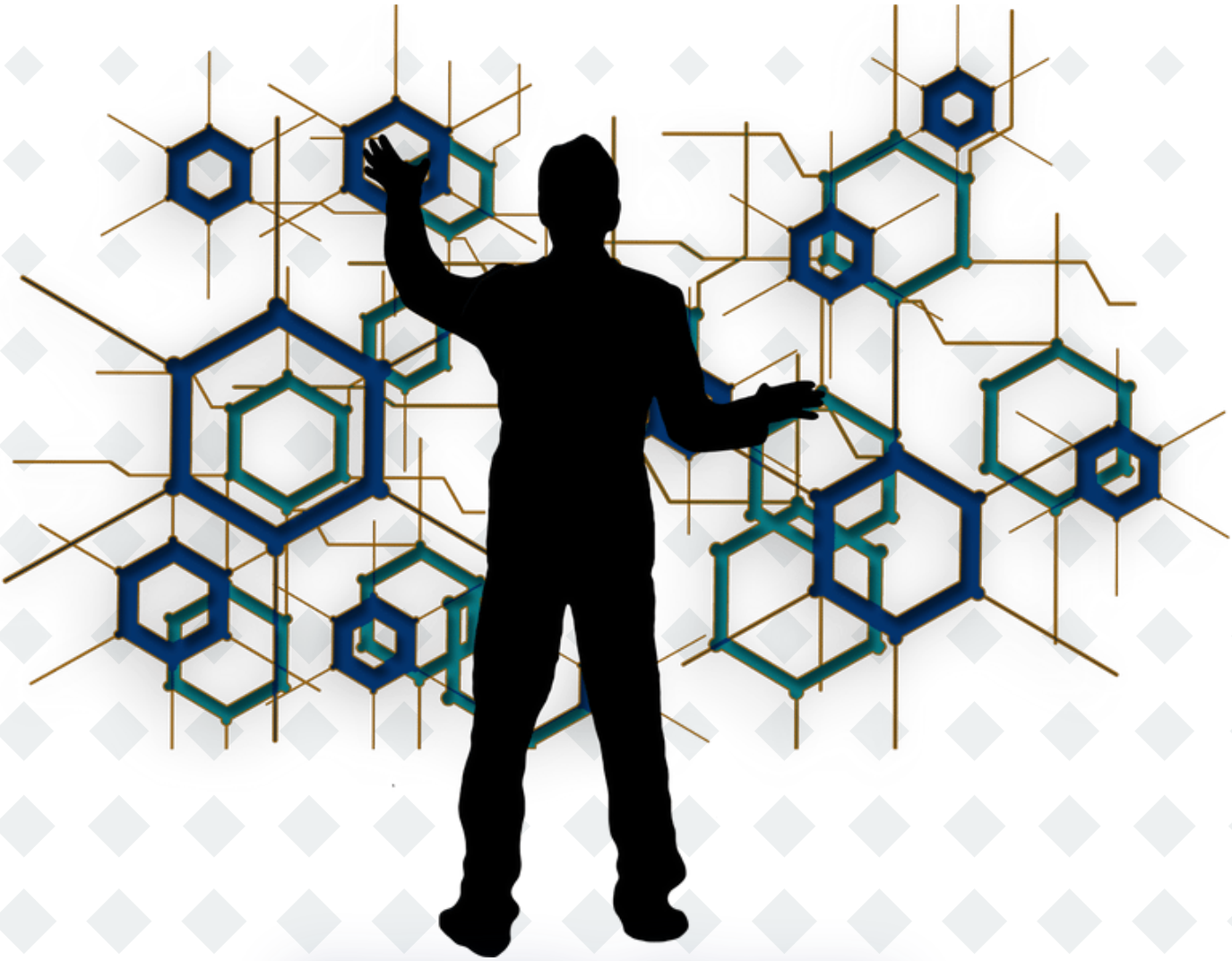
-66 + (-23)

QUESTION 8

Interpret the following ASCII message:

1010011101010010101011000100101100101000001001000100 00011010010100010 0

TOPIC 2



BOOLEAN ALGEBRA

BASIC

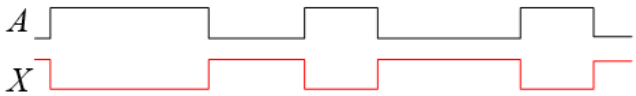
LOGIC GATE

NOT GATE

THE NOT PERFORMS INVERTER OPERATION. THE NOR GATE PRODUCES A HIGH OUTPUT WHEN INPUT IS LOW
BOOLEAN EXPRESSION X=A'

Input	Output
A	X
0	1
1	0

Example waveforms:

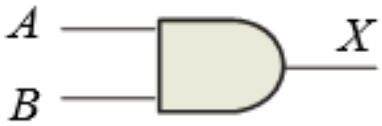


BASIC

LOGIC GATE

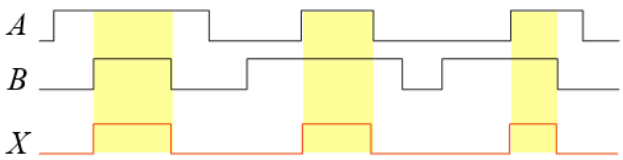
AND GATE

THE AND GATE PRODUCES A HIGH OUTPUT WHEN ALL INPUTS ARE HIGH: OTHERWISE, THE OUTPUT IS LOW.BOOLEAN EXPRESSION X = A.B OR X=AB



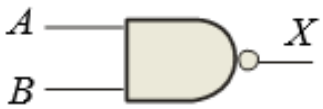
Inputs		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

Example waveforms:



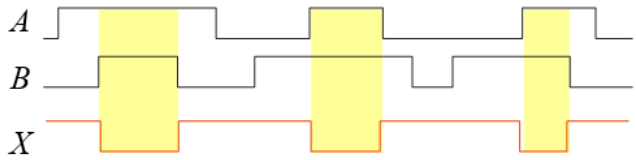
NAND GATE

THE NAND GATE PRODUCES A LOW OUTPUT WHEN ALL INPUTS ARE HIGH: OTHERWISE, THE OUTPUT IS HIGH.BOOLEAN EXPRESSION X = (A.B)' OR X=(AB)'



Inputs		Output
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Example waveforms:

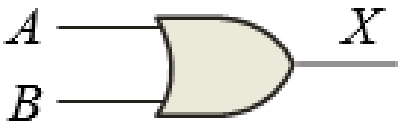


BASIC

LOGIC GATE

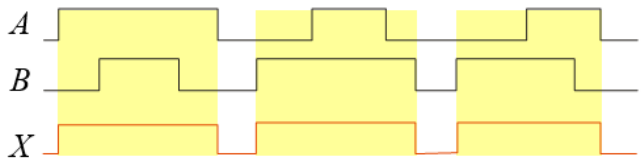
OR GATE

THE OR GATE PRODUCES A HIGH OUTPUT IF ANY INPUT IS HIGH. BOOLEAN EXPRESSION X=A + B



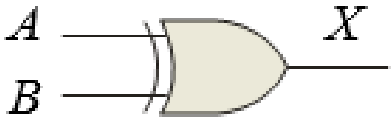
Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Example waveforms:



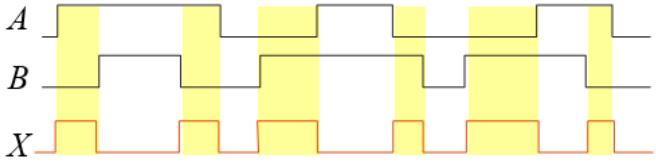
XOR GATE

THE XOR GATE PRODUCES A HIGH OUTPUT WHEN ONE INPUTS IS HIGH: OTHERWISE, THE OUTPUT IS LOW.BOOLEAN EXPRESSION X = A'B + AB'



Inputs		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Example waveforms:

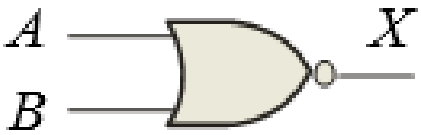


BASIC

LOGIC GATE

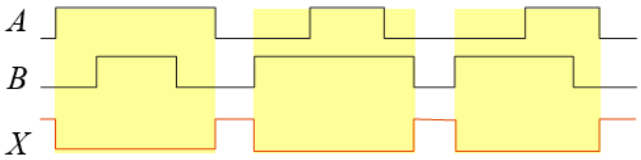
NOR GATE

THE NOR GATE PRODUCES A HIGH OUTPUT IF ALL INPUTS ARE LOW. BOOLEAN EXPRESSION X= (A+B)'



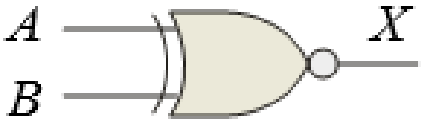
Inputs		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

Example waveforms:



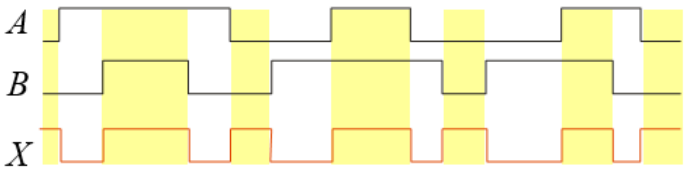
XNOR GATE

THE XNOR GATE PRODUCES A HIGH OUTPUT WHEN ALL INPUTS ARE SAME LOGIC LEVEL: OTHERWISE, THE OUTPUT IS LOW.BOOLEAN EXPRESSION X = A'B' + AB



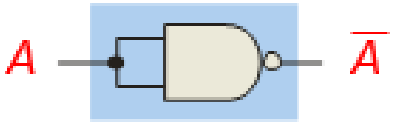
Inputs		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Example waveforms:

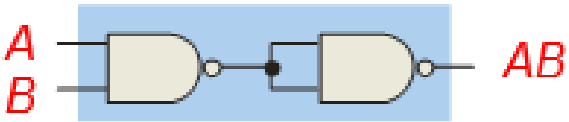


UNIVERSAL GATE

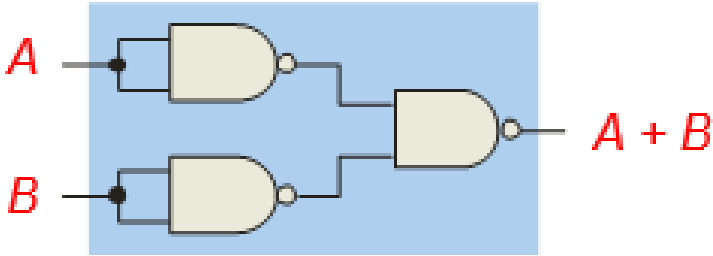
NAND GATES CAN BE USED TO PRODUCE THE OTHER BASIC BOOLEAN FUNCTION



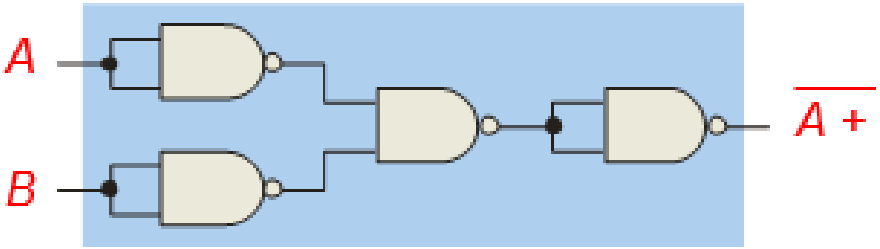
NOT GATE



AND GATE



OR GATE



NOR GATE

BOOLEAN ALGEBRA

BOOLEAN ADDITION

The addition operation of Boolean Algebra is similar to the OR operation. The OR Operation is used to calculate the sum term, without using AND operation.

The examples of 'Sum Term' A+B, A+B', A+B+C'. The value of the Sum Term is true when one or more literals are true and false when all the literals are false.

BOOLEAN MULTIPLICATION

The multiplication operation of Boolean Algebra is similar to the AND operation. The AND Operation is used to calculate the product, without using OR operation.

The examples of 'Product Term' AB, ABC ABCD. The value of the product term is true when all the literals are true and false when any one of the literal is false.

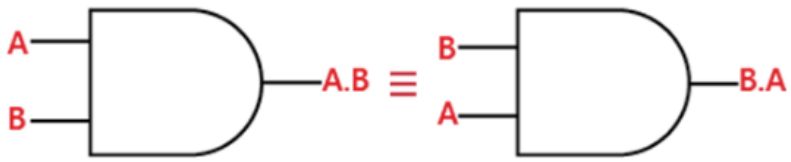
BOOLEAN ALGEBRA

COMMUTATIVE LAW

This law states that the order of the variables doesn't matter at all. The OR and the addition operation are similar. The commutative law of multiplication is written as

$$AB = BA$$

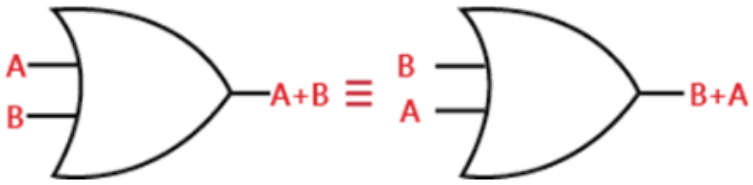
This can be illustrated with equivalent circuits



The Commutative Law of addition is written as

$$A + B = B + A$$

This can be illustrated with equivalent circuits



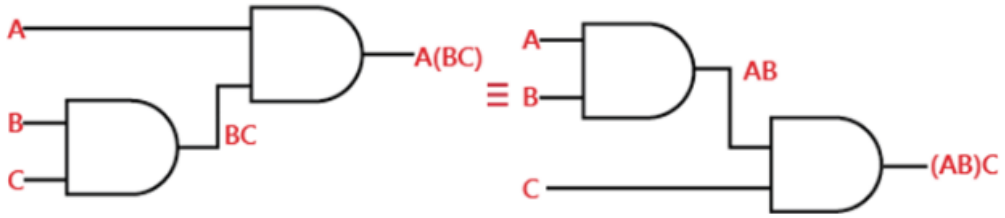
BOOLEAN ALGEBRA

ASSOCIATIVE LAW

This law states that the operation can be performed in any order when the variables priority is same. The associative law of multiplication is written as

$$A(BC) = (AB)C$$

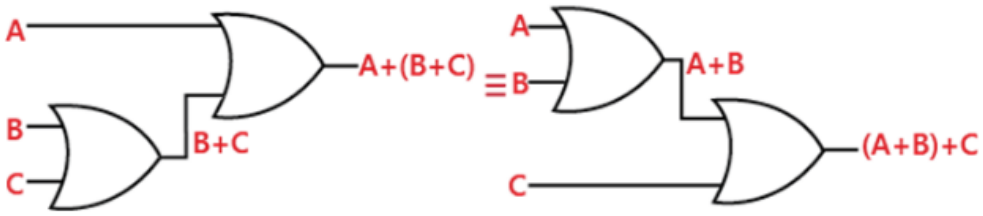
This can be illustrated with equivalent circuits



The associative law of addition is written as

$$A + (B + C) = (A + B) + C$$

This can be illustrated with equivalent circuits



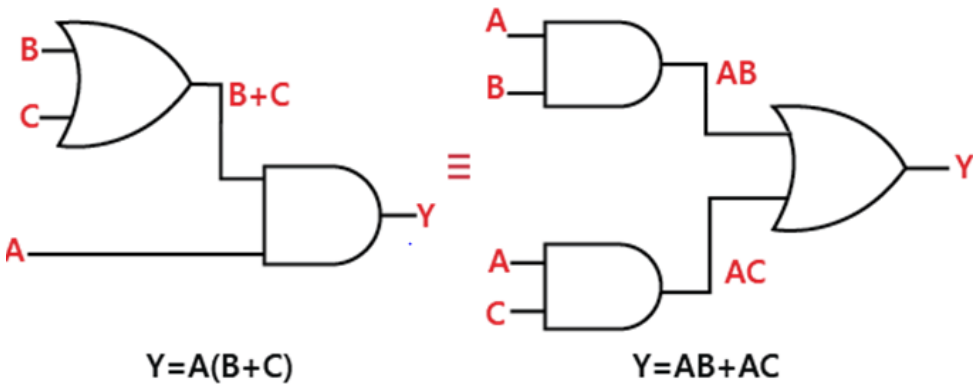
BOOLEAN ALGEBRA

DISTRIBUTIVE LAW

This law is the factoring law. A common variable can be factored from an expression just as in ordinary algebra. That is

$$AB + AC = A(B + C)$$

This can be illustrated with equivalent circuits

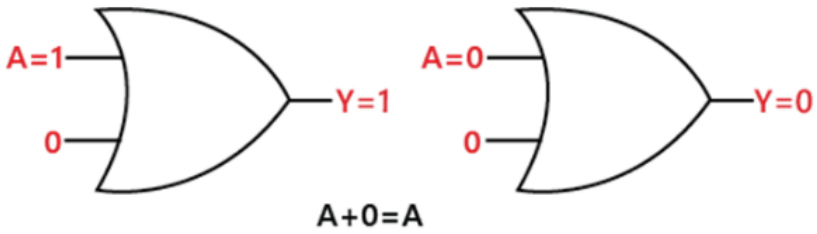


BOOLEAN ALGEBRA

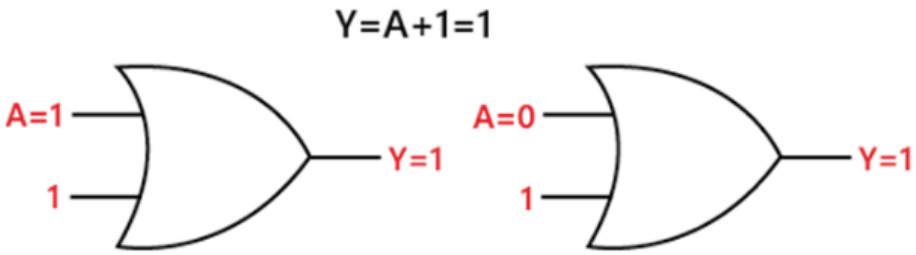
RULES OF BOOLEAN ALGEBRA

The rule are used in manipulating and simplifying boolean expressions.

Rule 1: A + 0 = A



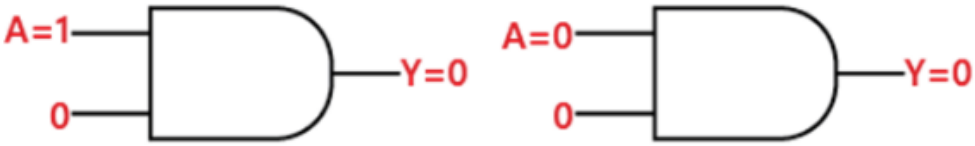
Rule 2: (A + 1) = 1



BOOLEAN ALGEBRA

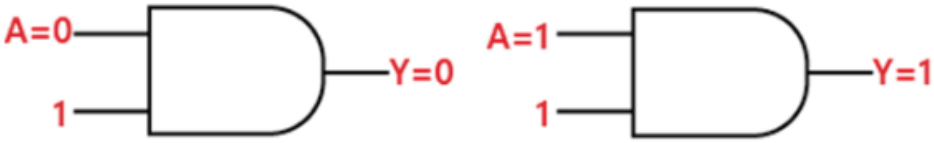
Rule 3: (A.0) = 0

X=A.0=0



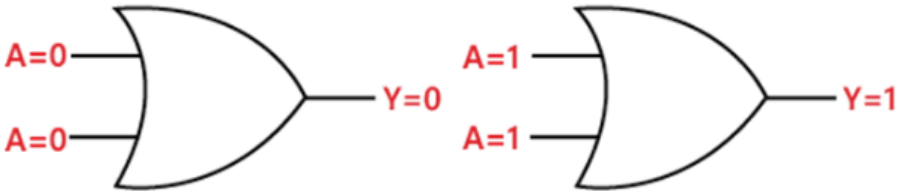
Rule 4: (A.1) = A

(A.1)=A



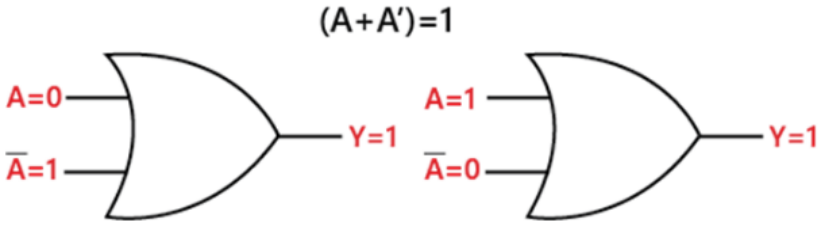
Rule 5: (A + A) = A

(A+A)=A

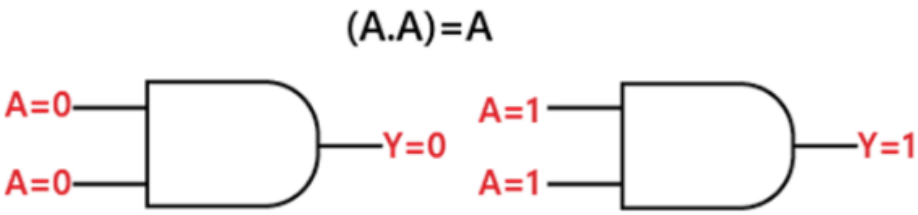


BOOLEAN ALGEBRA

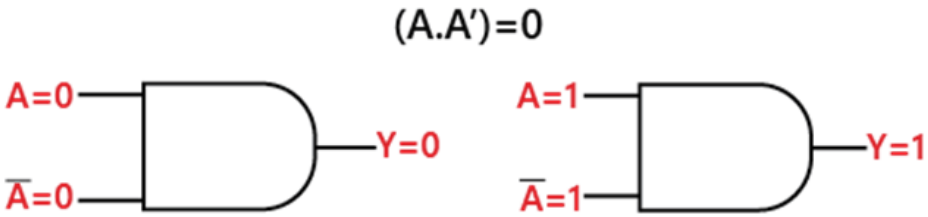
Rule 6: (A + A') = 1



Rule 7: (A.A) = A

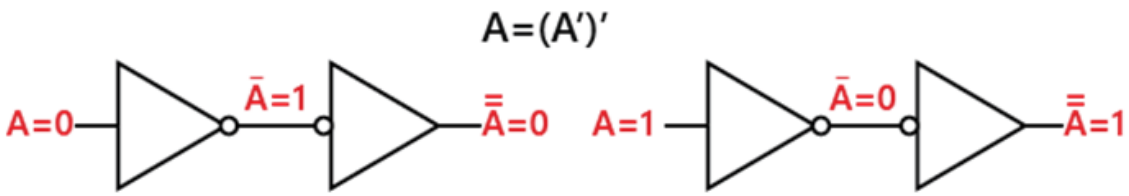


Rule 8: (A.A') = 0



BOOLEAN ALGEBRA

Rule 9: A = (A')'

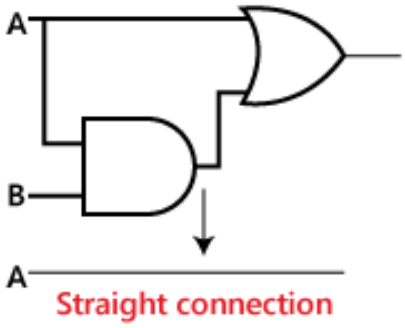


Rule 10: (A + AB) = A

A + AB = A(1 + B)	Factoring (distributive law)
A + AB = A.1	Rule 2: (1 + B)= 1
A + AB = A	Rule 4: A .1 = A

A	B	AB	A+AB
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

Equal



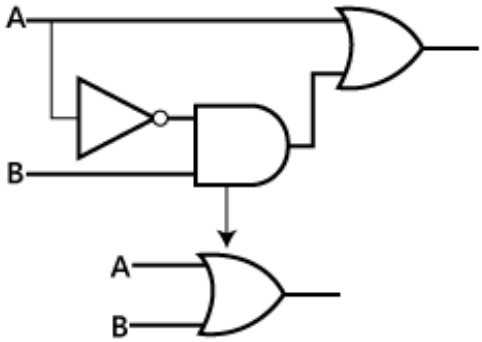
BOOLEAN ALGEBRA

Rule 11: A + AB = A + B

A + AB = (A + AB)+ AB	Rule 10: A = A + AB
A+AB= (AA + AB)+ AB	Rule 7: A = AA
A+AB=AA +AB +AA +AB	Rule 8: adding AA = 0
A+AB= (A + A)(A + B)	Factoring
A+AB= 1.(A + B)	Rule 6: A + A = 1
A+AB=A + B	Rule 4: drop the 1

A	B	$\bar{A}B$	$A+\bar{A}B$	A+B
0	0	0	0	0
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

equal



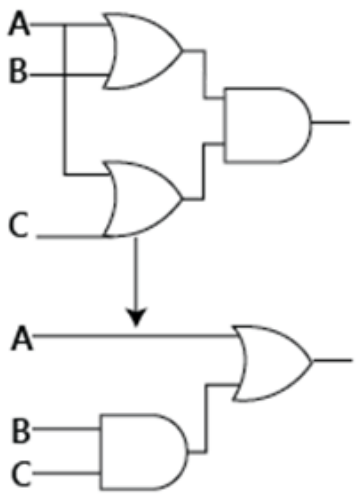
BOOLEAN ALGEBRA

Rule 12: (A + B)(A + C) = A + BC

(A + B)(A + C)= AA + AC + AB + BC	Distributive law
(A + B)(A + C)= A + AC + AB + BC	Rule 7: AA = A
(A + B)(A + C)= A(1 + C)+ AB + BC	Rule 2: 1 + C = 1
(A + B)(A + C)= A.1 + AB + BC	Factoring (distributive law)
(A + B)(A + C)= A(1 + B)+ BC	Rule 2: 1 + B = 1
(A + B)(A + C)= A.1 + BC	Rule 4: A .1 = A
(A + B)(A + C)= A + BC	

A	B	C	A+B	A+C	(A+B)(A+C)	BC	A+BC
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	1	1	0	0	0
0	1	1	1	1	1	1	1
1	0	0	1	1	1	0	1
1	0	1	1	1	1	0	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

equal



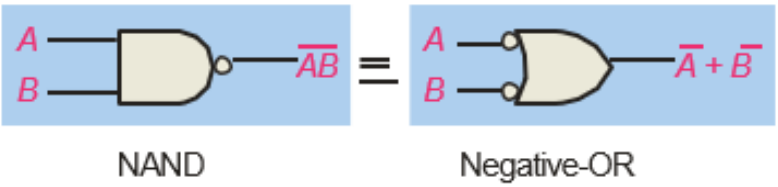
DE MORGAN'S THEOREM

FIRST THEOREM

The complement of a product of variables is equal to the sum of the complemented variables

$$\overline{AB} = \overline{A} + \overline{B}$$

This can be illustrated with equivalent circuits



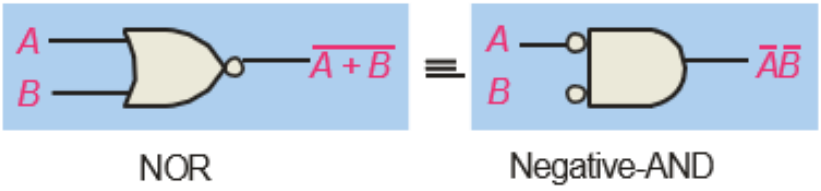
Inputs		Output	
A	B	AB	$\overline{A + B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

SECOND THEOREM

The complement of a sum of variables is equal to the product of the complemented variables

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

This can be illustrated with equivalent circuits



Inputs		Output	
A	B	$A + B$	AB
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

SUM OF PRODUCT (SOP)

An expression is in SOP form when two or more product terms are summed as in the following examples:

$$A'B'C' + AB \qquad ABC' + C'D' \qquad CD + E'$$

PRODUCT OF SUM (POS)

An expression is in POS form when two or more sum terms are multiplied as in the following examples:

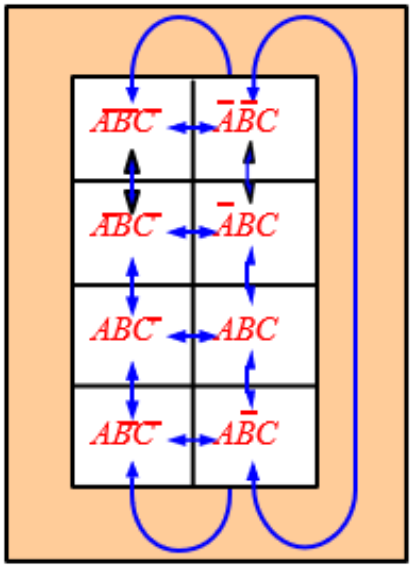
$$(A + B)(A' + C) \qquad (A + B + C')(B + D) \qquad (A' + B)C$$

KARNAUGH MAPS

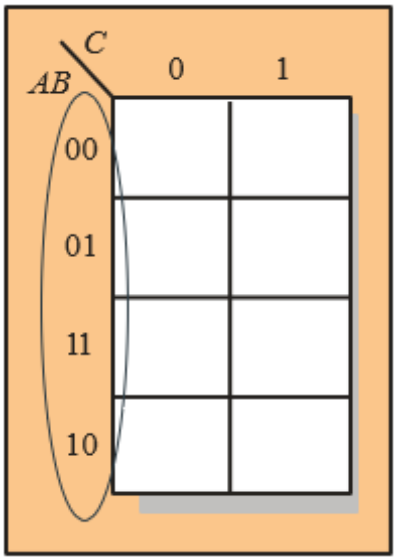
The Karnaugh map (K-map) is a tool for simplifying combinational logic with 3 or 4 wariables.

For 3 variables, 8 cells are required (23). The map shown is for three variables labeled A,B and C. Each cell represents one possible product term.

Each cell differs from an adjacent cell by only one variable.



Cells are usually labeled using 0's and 1's to represent the variable and its complement.



The numbers are entered in gray code, to force adjacent cells to be different by only one variable.

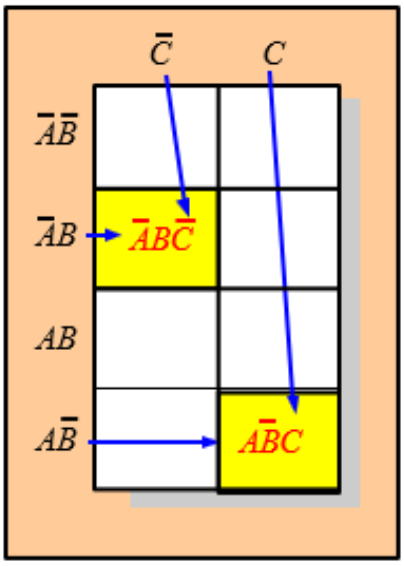
ones are reas as the true variable and zeros are read as the complemented variable.

KARNAUGH MAPS

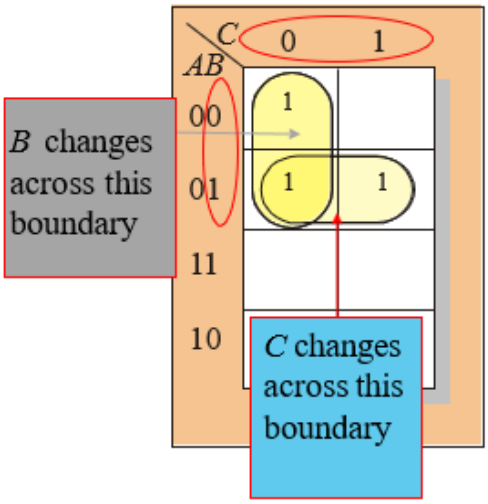
Alternatively, cells can be labeled with the variable letters. This makes it simple to read, but it takes more time preparing the map

Example
Read the terms for the yellow cells

The cells are A'BC' and AB'C



K-maps can simplify combinational logic by grouping cells and eliminating variables that change. Group the 1's on the map and read the minimum logic

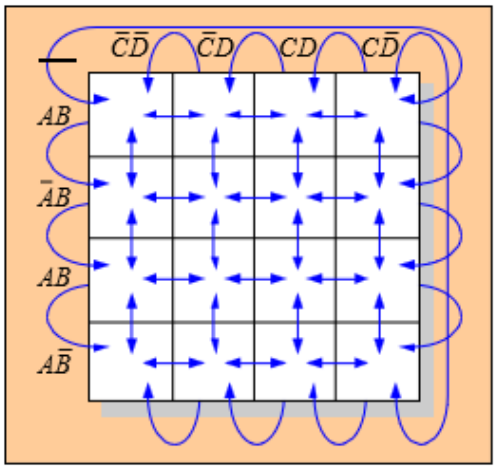


1. Group the 1's into two overlapping groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The vertical group is read A'C'.
4. The horizontal group is read A'B

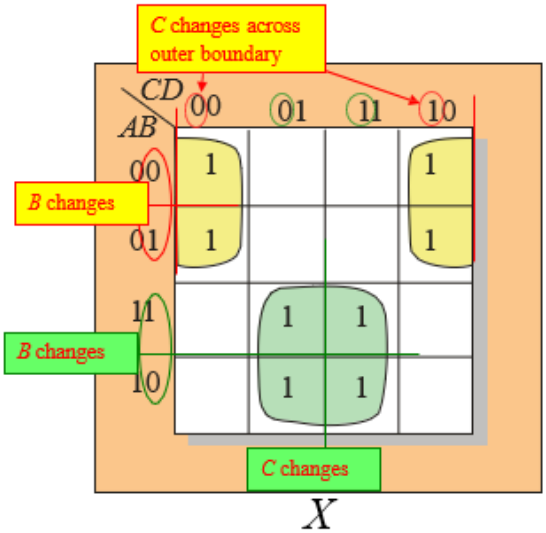
$$X = \bar{A}\bar{C} + \bar{A}B$$

KARNAUGH MAPS

A 4 variables map has an adjacent cell on each of its four boundaries as shown. Each cell is different only by one variable from an adjacent cell.



Group the 1's on the map and read the minimum logic



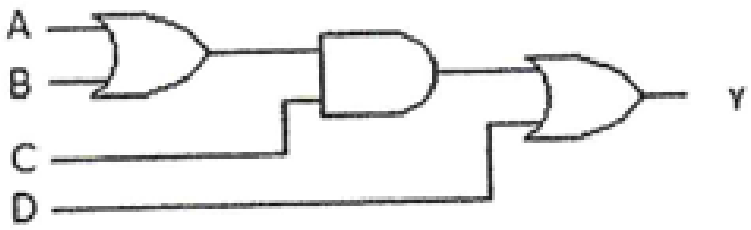
1. Group the 1's into two separate groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The upper (yellow) group is read as A'D'.
4. The lower (green) group is read as AD.

$$X = \overline{AD} + AD$$

REVIEW QUESTION

QUESTION 1

Write the truth table for Boolean expression for the figure below.



QUESTION 2

Simplify this expression using Boolean Algebra technique and construct simplified equation in logic circuit.

$$AB + A(B+C)+B(B+C)$$

QUESTION 3

Use a Karnaugh Map to find the minimum POS for this expression A'B'C' + AB'C'+A'BC'+ABC'

REVIEW QUESTION

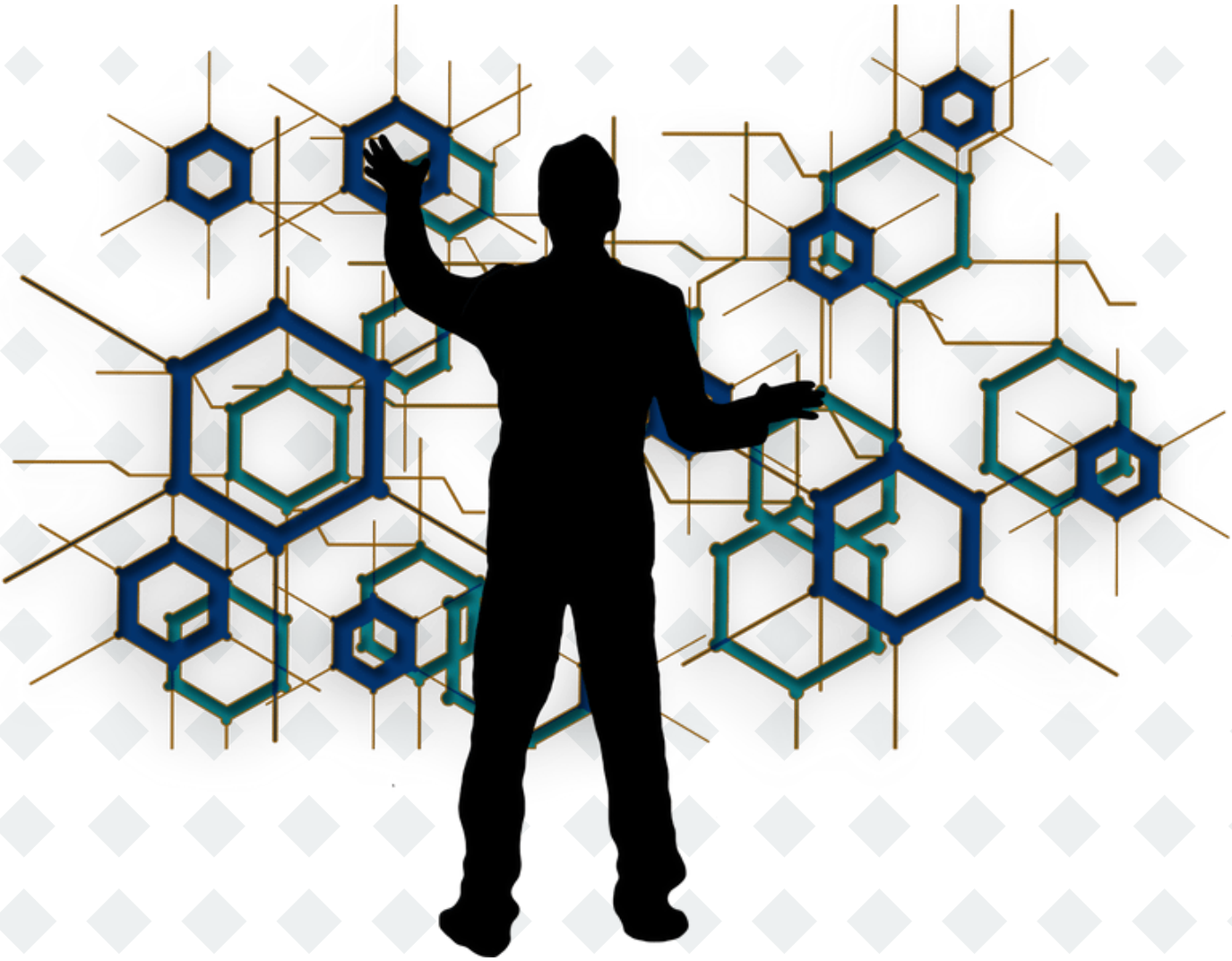
QUESTION 4

Refer to the truth table in Table 1, write logic expression Sum of Product (SOP) for the output. Simplified the output expression by using Karnaugh map and sketch the simplified combination logic circuit.

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Table 1/Jadual 1

TOPIC 3



FLIP-FLOPS

FLIP-FLOPS

LOGIC CIRCUITS

Logic circuits are classified into two groups:

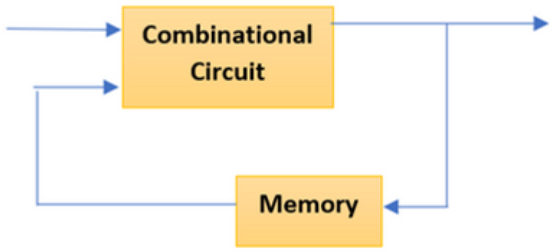
1. Combinational logic circuits
2. Sequential logic circuits

COMBINATIONAL LOGIC CIRCUITS

Whose output is a function of only the present input. The basic building block of combinational circuits is the logic gate. For example AND gates, OR gates, inverters and etc.

SEQUENTIAL LOGIC CIRCUITS

Sequential circuits refer to the combinational logic circuits that consist of input variables and logic gates along with the output variable. For example, flip-flops, counter, register, clocks and etc.

Combinational Circuit	Sequential Circuit
Output only depends on the present input	Output depends on present input and past output
Memory element is absent	Memory element is present
No clock signal is applied	Clock signal is required
	
Example - Half Adder, Full Adder, Multiplexer	Examples - Flipflop, Counters, Registers

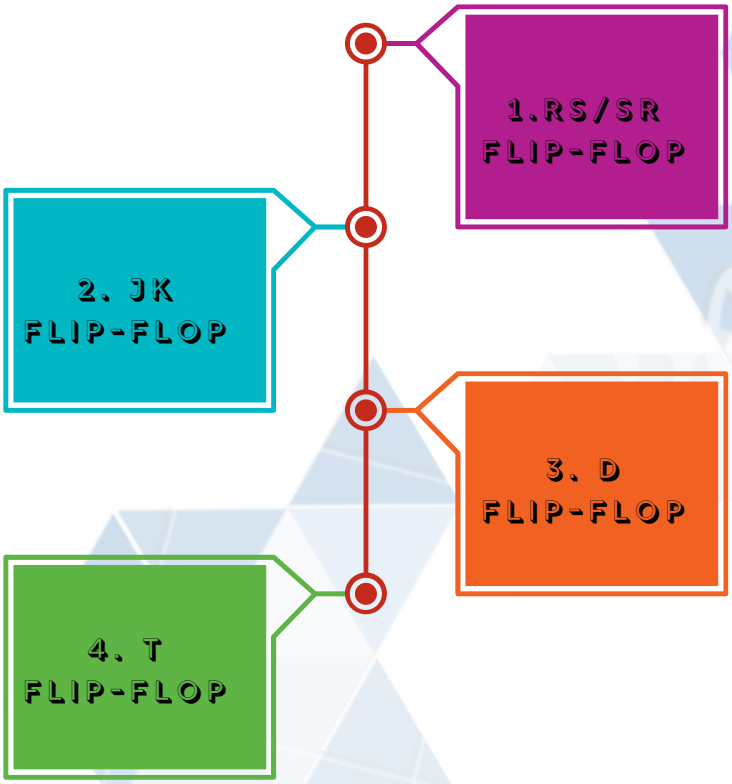
FLIP-FLOPS

DEFINITION OF FLIP-FLOPS

- A flip-flop is a type of circuit that can store and recall a single bit of information.

- A flip-flop is not a specific device but rather a term used to describe a group of sequential logic circuits. These circuits made up of digital logic gates and other components, can be created using different electronic elements like transistors, integrated circuits (ICs), or programmable logic devices (PLDs).

TYPES OF FLIP-FLOPS



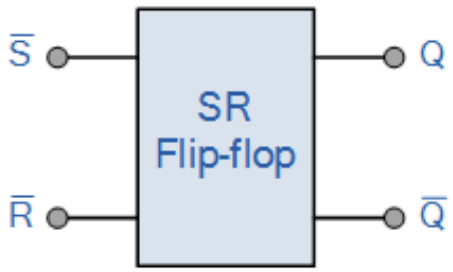
FLIP-FLOPS

RS/SR FLIP-FLOP

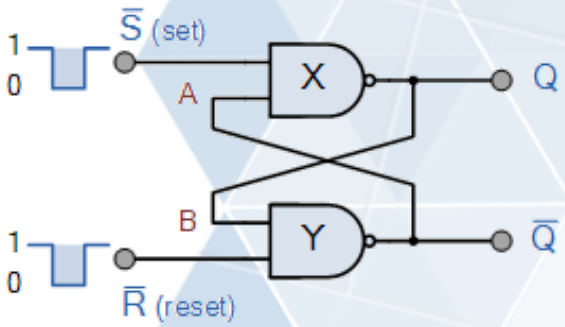
The RS / SR Flip-Flop is also known as the gated or clocked SR latch. The clocked SR latch or SR flip-flop temporarily stores or holds the information until it is needed in digital circuits

RS/SR Flip-Flop (Reset/Set). Set/Clear. Most basic flip flop can be made by cross coupling NAND or NOR gates Activating Set and Reset is invalid

SYMBOL, LOGIC CIRCUIT AND TRUTH TABLE



a) Symbol



b) Logic Circuit

State	S	R	Q	Q̄	Description
Set	1	0	0	1	Set Q̄ » 1
	1	1	0	1	No change
Reset	0	1	1	0	Reset Q̄ » 0
	1	1	1	0	No change
Invalid	0	0	1	1	Invalid Condition

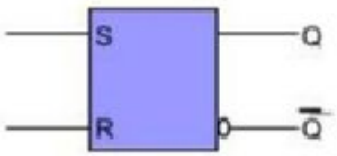
c) Truth Table

FLIP-FLOPS

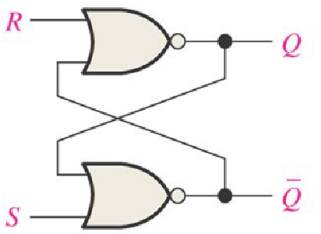
TYPES OF SR FLIP-FLOPS

1. Active high sr (sr nor gate)
2. Active low sr (sr nor gate)
3. Clocked sr

SR ACTIVE HIGH:SYMBOL, LOGIC CIRCUIT AND TRUTH TABLE



a) Symbol



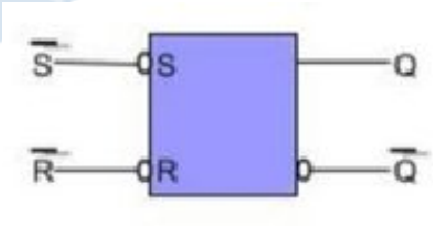
b) Logic Circuit

S	R	Q	Q'	COMMENT
0	0	NC	NC	NO CHANGE (HOLD)
0	1	0	1	RESET (Q = 0)
1	0	1	0	SET (Q = 1)
1	1	0	0	INVALID

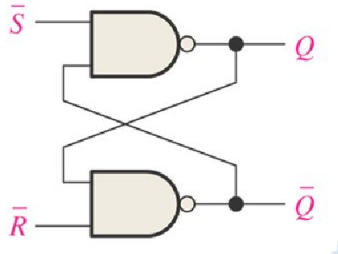
c) Truth Table

FLIP-FLOPS

SR ACTIVE LOW: SYMBOL, LOGIC CIRCUIT AND TRUTH TABLE



a) Symbol



b) Logic Circuit

S	R	Q	Q'	COMMENT
0	0	1	1	INVALID
0	1	1	0	SET (Q = 1)
1	0	0	1	RESET (Q = 0)
1	1	NC	NC	NO CHANGE

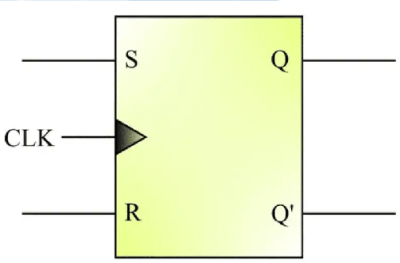
c) Truth Table

CLOCKED SR

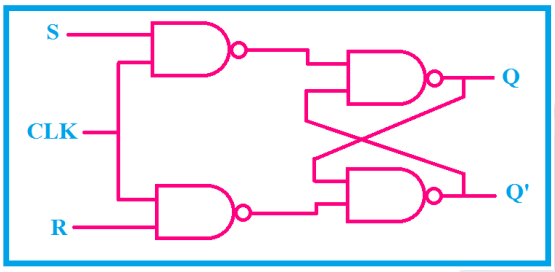
The output is changed (i.e. the stored data is changed) only when you give an active clock signal.

FLIP-FLOPS

CLOCKED SR: SYMBOL, LOGIC CIRCUIT AND TRUTH TABLE



a) Symbol



b) Logic Circuit

S	R	CLK	Q
0	0	↑	NO CHANGE (HOLD)
1	0	↑	SET (Q=1)
0	1	↑	RESET (Q=0)
1	1	↑	INVALID

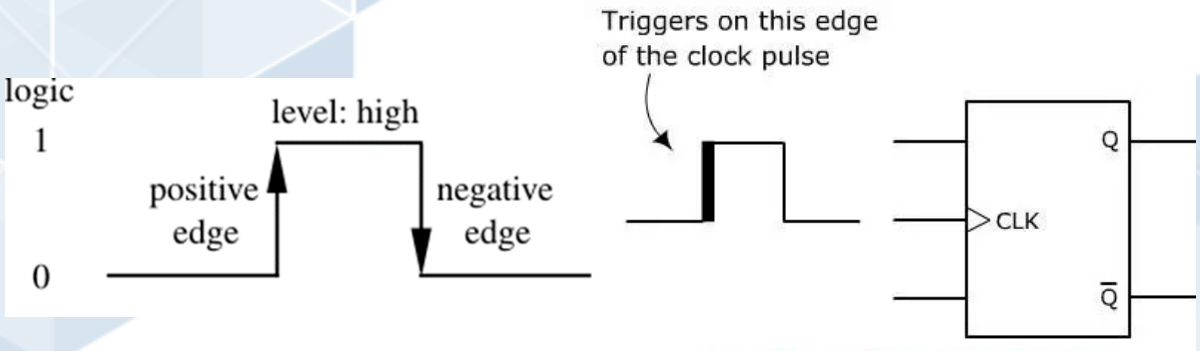
c) Truth Table

EDGE TRIGGERED

1. POSITIVE EDGE-TRIGGERED FLIP-FLOP

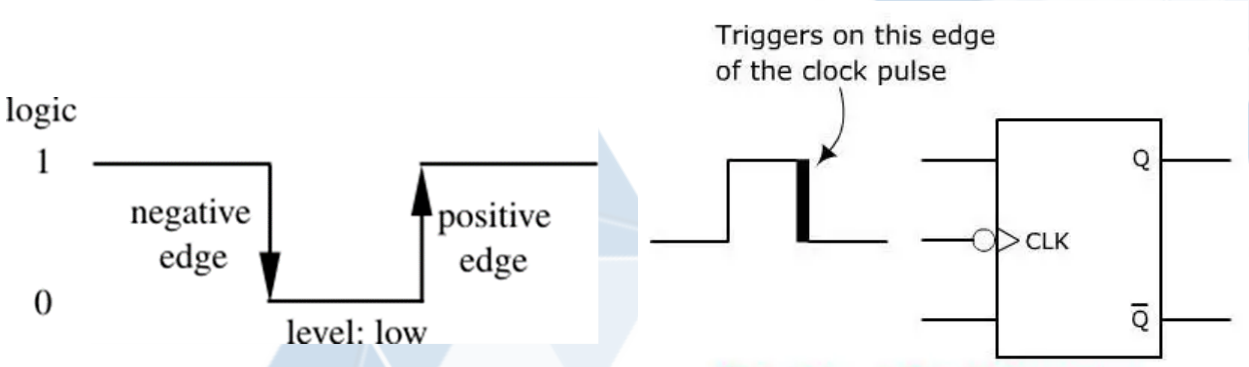
The type of edge-triggered flip-flop whose output changes its state only on the rising edge (edge that goes from low to high) of the clock pulse is called a positive edge-triggered flip-flop. The positive edge triggered flip flop is also called a rising edge-triggered flip-flop.

FLIP-FLOPS



2. NEGATIVE EDGE-TRIGGERED FLIP-FLOP

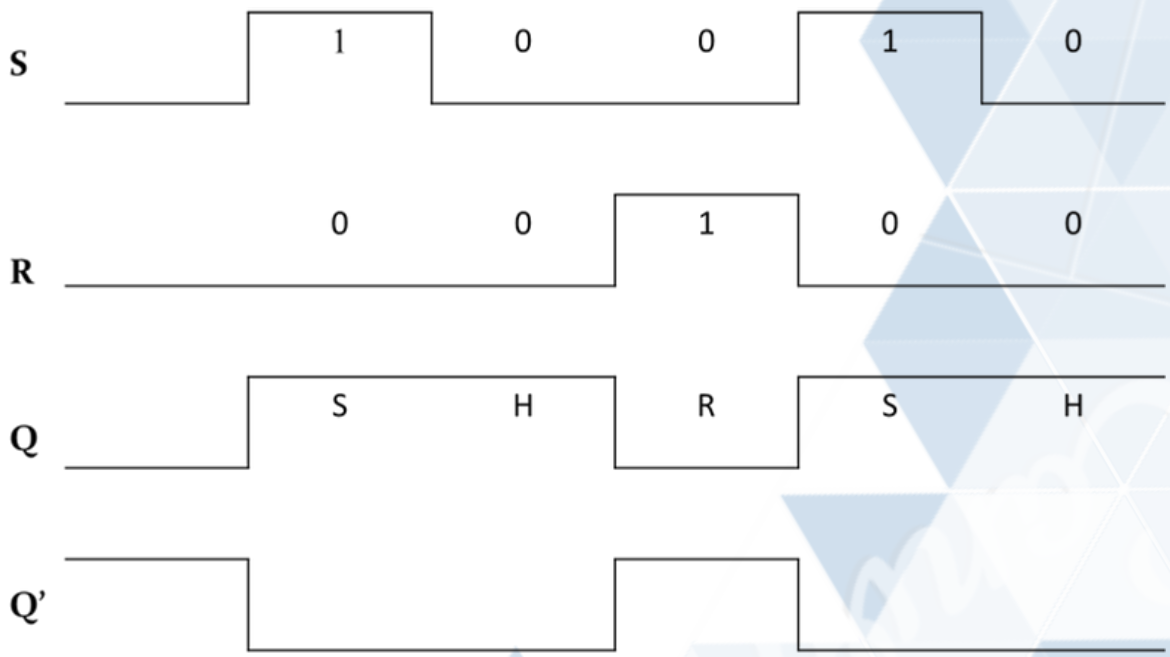
The type of edge-triggered flip flop whose output changes its state only on the falling edge (edge that goes from high to low) of the clock pulse is called a negative edge-triggered flip-flop. The negative edge triggered flip flop is also known as a falling edge-triggered flip-flop.



FLIP-FLOPS

EXAMPLE:

Determine the Q and Q' output for the SR flip-flop active High (SR NOR gate). Assuming that the flip-flop is initially RESET.

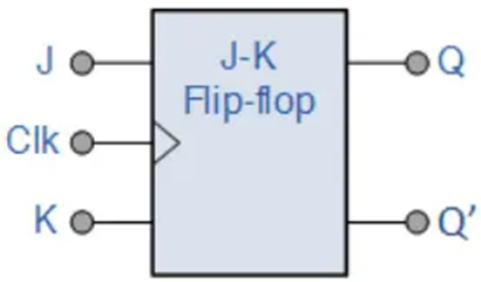


FLIP-FLOPS

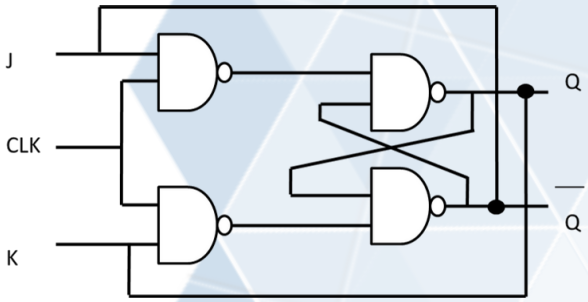
JK FLIP-FLOP

JK Flip Flop is one of the most used flip-flops in digital circuits. The universal flip flop has two inputs, 'J' and 'K.' The JK Flip Flop is a gated SR Flip-Flop with a clock input circuitry that prevents the illegal or invalid output when both inputs S and R are equal to logic level "1."

JK FLIP-FLOP: SYMBOL, LOGIC CIRCUIT AND TRUTH TABLE



a) Symbol



b) Logic Circuit

Inputs			Outputs		
J	K	CLK	Q	$\overline{Q}$	Comments
0	0	↑	Q_0	$\overline{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	$\overline{Q}_0$	Q_0	Toggle

c) Truth Table

FLIP-FLOPS

JK FLIP-FLOP: ASYNCHRONOUS /OVERRIDES INPUTS

Asynchronous Inputs a.k.a. Override Inputs operate independent of the control and clock inputs.

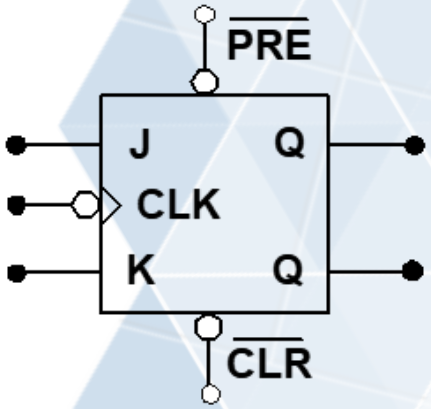
The PRESET and CLEAR inputs of the JK Flip-Flop are asynchronous, which means that they will have an immediate effect on the Q and Q' outputs regardless of the state of the clock and / or the J and K inputs.

CLEAR

Active-low override Q=0 overrides all other inputs

PRESET

Active-low override Q=1 overrides all other inputs



$\overline{PRE}$	$\overline{CLR}$	Q*
1	1	No effect; FF can respond to J, K, and CLK
1	0	Q=0 independent of synchronous inputs
0	1	Q=1 independent of synchronous inputs
0	0	Ambiguous (not used)

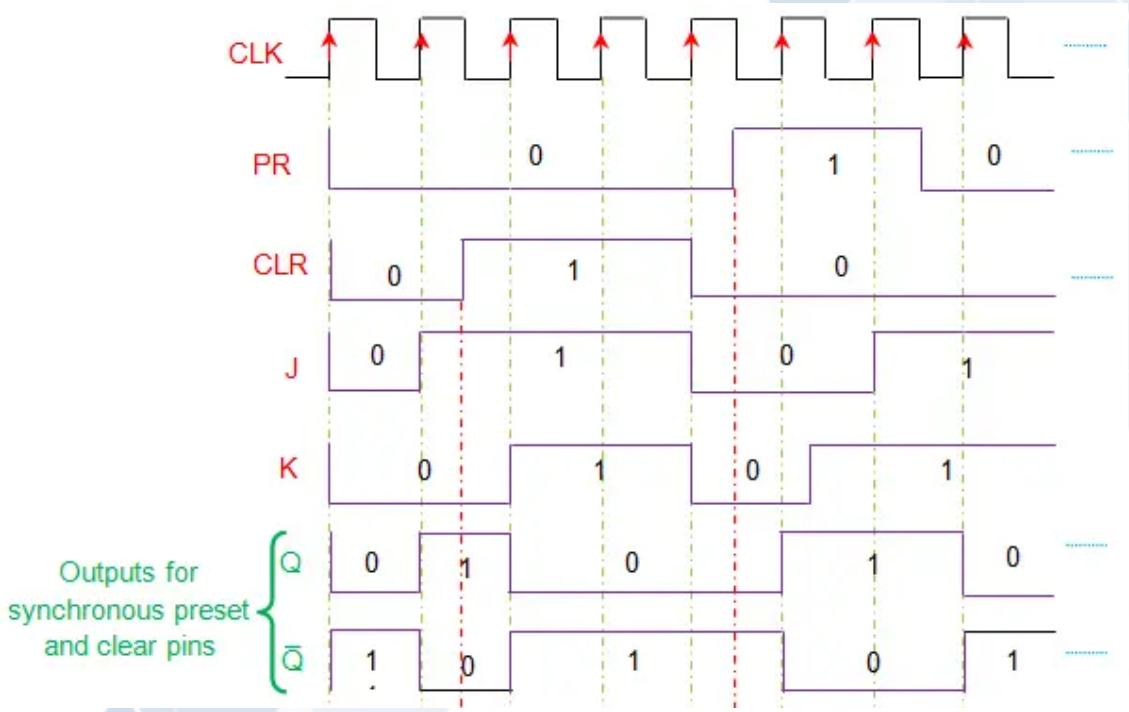
*CLK can be in any state

FLIP-FLOPS

Mode of Operation	Inputs					Outputs	
	PRE	CLR	CLK	J	K	Q	Q'
Asynchronous set	0	1	X	X	X	1	0
Asynchronous reset	1	0	X	X	X	0	1
Prohibited	0	0	X	X	X	1	1
Hold	1	1		0	0	No Change	
Reset	1	1		0	1	0	1
Set	1	1		1	0	1	0
Toggle	1	1		1	1	Toggle	

X = Irrelevant
= H-to-L transition of clock pulse

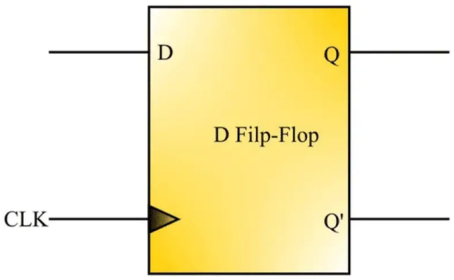
JK FLIP-FLOP TIMING DIAGRAM WITH PRESET AND CLEAR



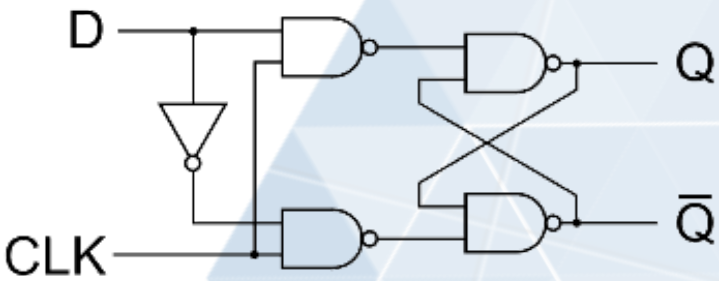
FLIP-FLOPS

D FLIP-FLOP

D flip-flop or Data flip flop is a type of flip Flop that has only one data input that is 'D' and one clock pulse input with two outputs Q and Q bar. This Flip Flop is also called a delay flip flop because when the input data is provided into the d flip-flop, the output follows the input data delay by one clock pulse.



a) Symbol



b) Logic Circuit

Inputs		Outputs		
D	*EN*	*Q*	$\overline{Q}$	Comments
0	1	0	1	RESET
1	1	1	0	SET
X	0	Q_0	$\overline{Q}_0$	No change

c) Truth Table

FLIP-FLOPS

APPLICATION OF D FLIP-FLOP

1. PARALLEL DATA TRANSFER

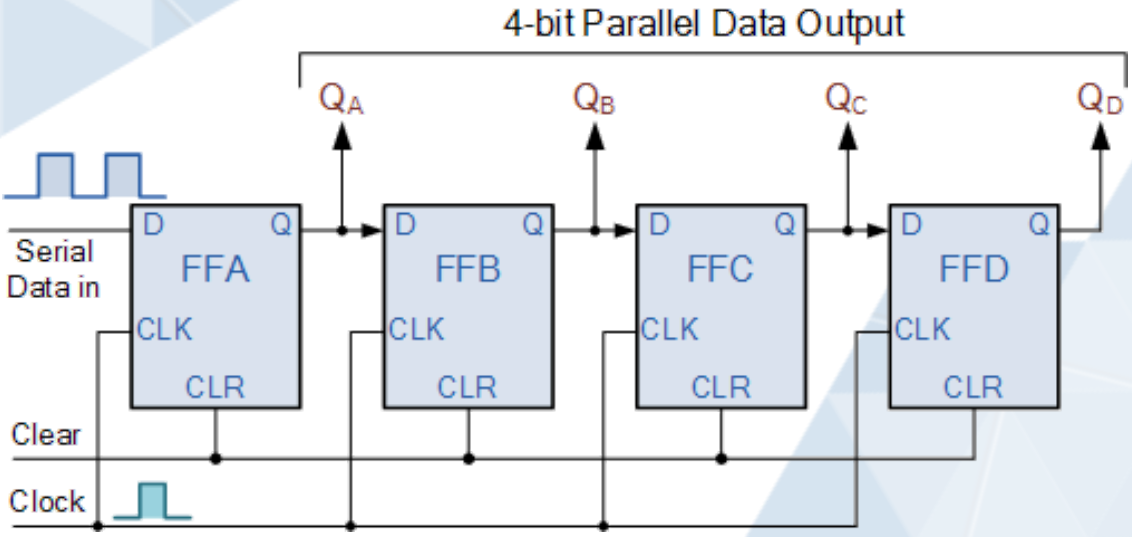


Figure 3.13 shows the parallel transfer of 8 bits of data from the X Register to the Y Register upon application of an enabling transfer pulse.

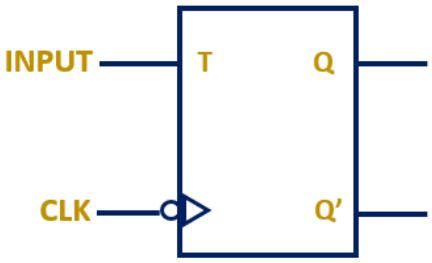
Clearly, parallel data transfer is faster than serial data transfer, but serial transfer has the advantages of requiring less hardware. These registers are made up of D flip-flops, which can serve as memory locations.

The information in the X Register is intact after the transfer to the Y Register, so this process shows a possible scenario for accessing digital information stored in memory.

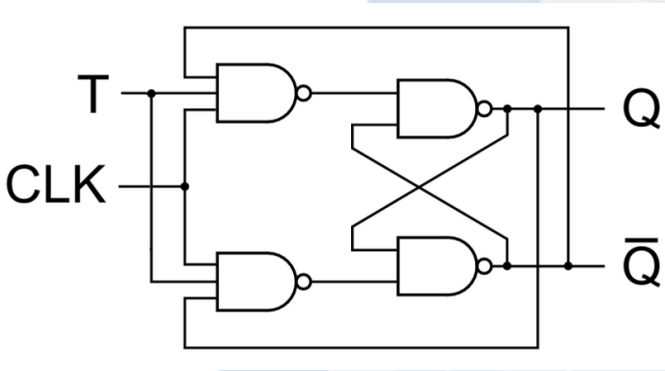
FLIP-FLOPS

T FLIP-FLOP

T flip flop or to be precise it is known as Toggle Flip Flop, because it can able to toggle its output depending upon on the input. T here stands for Toggle. Toggle basically indicates that the bit will be flipped i.e, either from 1 to 0 or from 0 to 1. Here, clock pulse is supplied to operate this flop flop, hence it is clocked flip flop.



a) Symbol



b) Logic Circuit

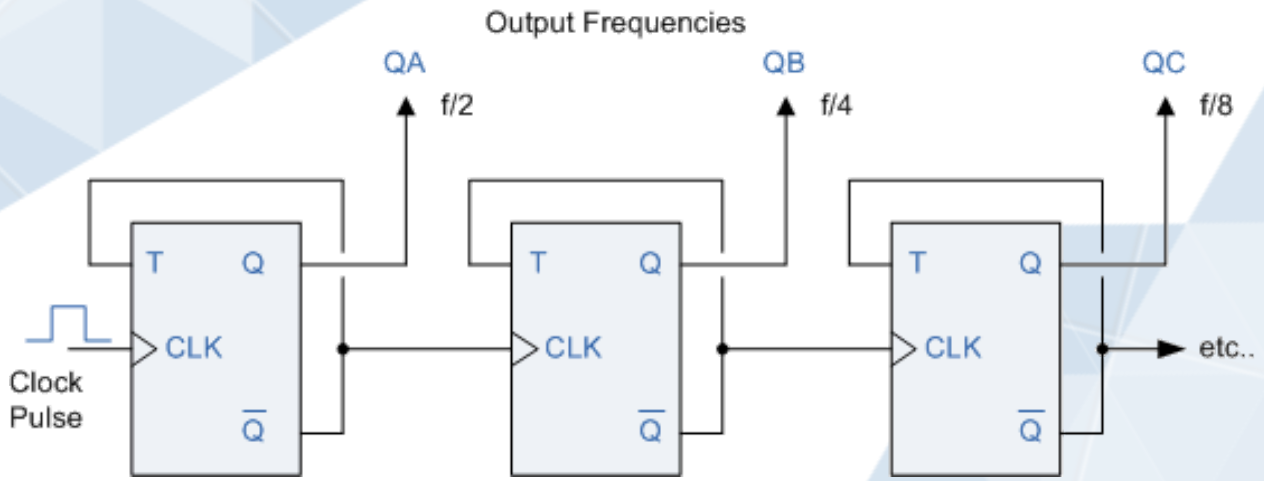
T	CLK	Q(t+1)	Comments
0	↑	Q(t)	No change
1	↑	Q(t)'	Toggle

c) Truth Table

FLIP-FLOPS

APPLICATION OF T FLIP-FLOP

1. FREQUENCY DIVIDER



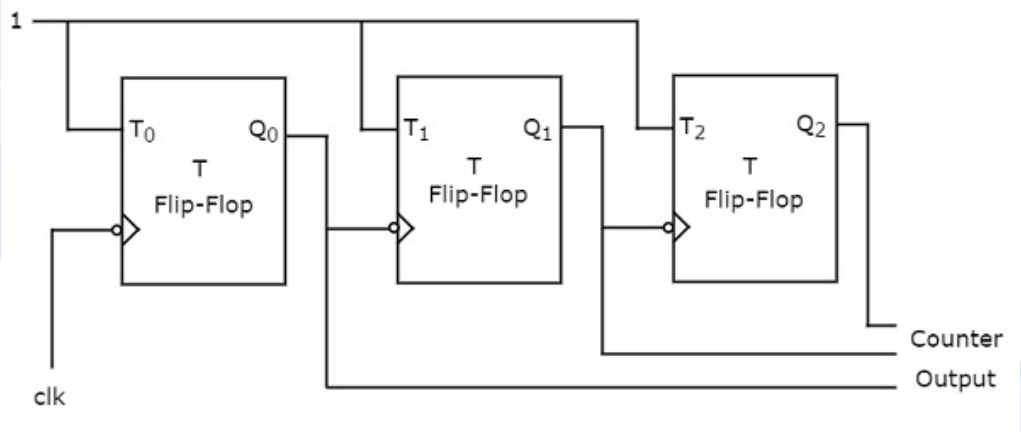
Frequency division, toggle mode flip-flops are used in a chain as a divide by two counter. One flip-flop will divide the clock, *f*in by 2, two flip-flops will divide *f*in by 4 (and so on). One benefit of using toggle flip-flops for frequency division is that the output at any point has an exact 50% duty cycle.

The final output clock signal will have a frequency value equal to the input clock frequency divided by the MOD number of the counter. Such circuits are known as "divide-by-n" counters. Counters can be formed by connecting individual flip-flops together and are classified according to the way they are clocked.

.

FLIP-FLOPS

2. COUNTER



Asynchronous Counters can be made from Toggle or D-type flip-flops. They are called asynchronous counters because the clock input of the flip-flops are not all driven by the same clock signal.

Each output in the chain depends on a change in state from the previous flip-flops output. Asynchronous counters are sometimes called ripple counters because the data appears to "ripple" from the output of one flip-flop to the input of the next. They can be implemented using "divide-by-n" circuits.

REVIEW QUESTION

QUESTION 1

Complete the Table 3.2 below for the output and mode of operation for JK flip-flop.

INPUT		BEFORE CLOCK		AFTER CLOCK		MODE
J	K	Qn	Qn'	Qn+1	Qn+1'	
0	1	1	0			
1	0	0	1			SET
1	1			0	1	TOGGLE
0	0	0	1			
1	1	0	1			
0	1	1	0			

REVIEW QUESTION

QUESTION 2

Based on Figure 3.16, sketch the output Q and Q' for gated JK flip-flop with positive edge triggered. (Assume the initial output Qo=1).

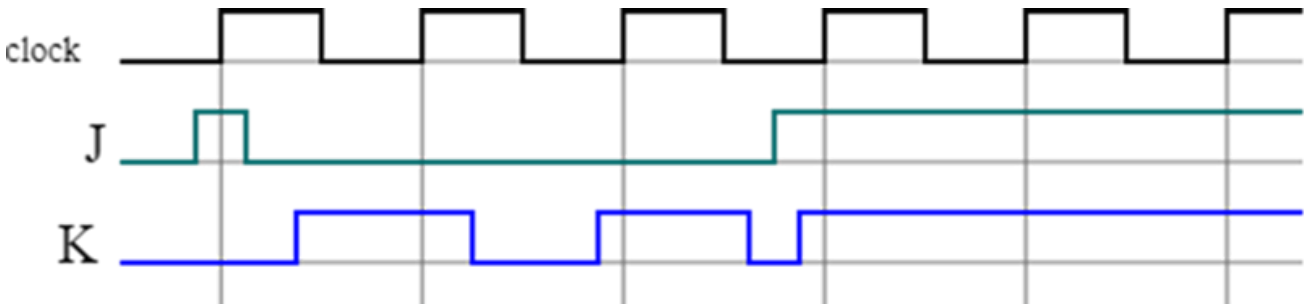


Figure 3.16

QUESTION 3

By using a suitable diagram explain the operation of the JK flip-flop with Preset (PRE) and Clear (CLR).

REVIEW QUESTION

QUESTION 4

State the output for T flip-flop as given in Table 3.3 below.

Table 3.3

INPUT	BEFORE CLOCK		AFTER CLOCK	
T	Qn	Qn’	Qn+1	Qn+1’
1	1	0		
0	0	1	0	1
1	0		1	0
1	1	0		
0		1	0	1
1	0	1		

REVIEW QUESTION

QUESTION 5

State the output for T flip-flop as given in Table 3.3 below.

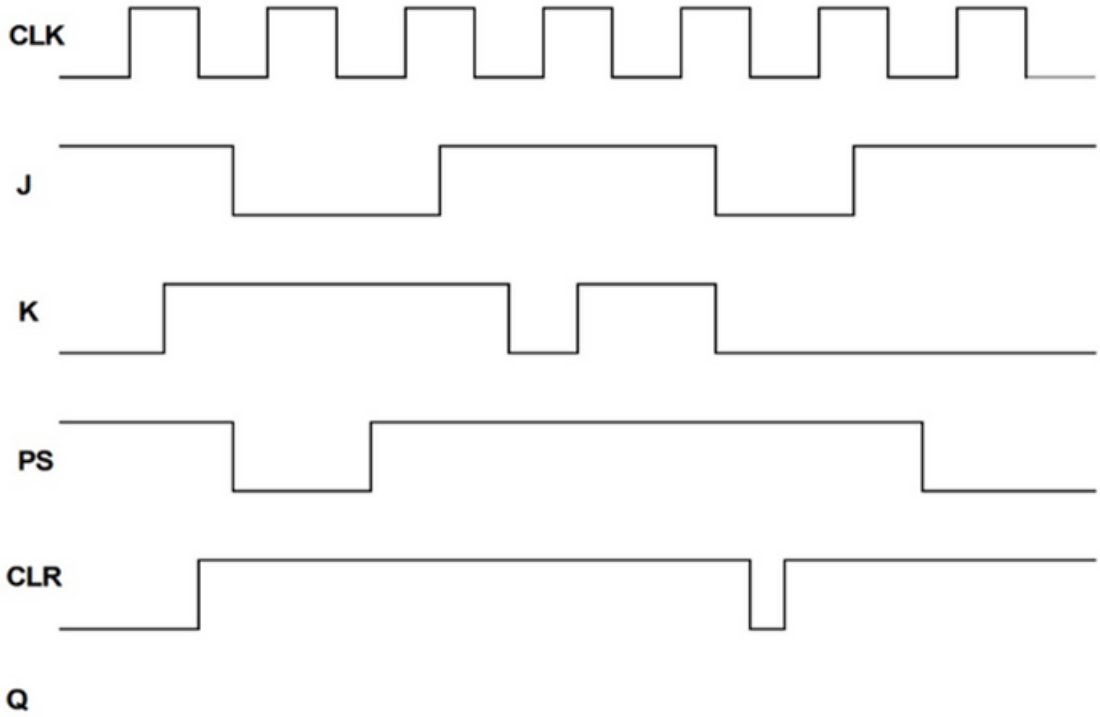


Figure 3.17

QUESTION 6

Draw the logic circuit and truth table of a clocked SR flip-flop.

REFERENCE

Electrical4U. (2020). JK Flip Flop: What is it? (Truth Table & Timing Diagram)Electrical4U Retrieved September 15, 2023, from https://www.electrical4u.com/jk-flip-flop/

Codeforwin. (2017). C Program to Convert Binary to Hexadecimal Number System. Retrieved September 10, 2023, from https://codeforwin.org/c-programming/c-program-to-convert-binary-to-hexadecimal-number-system

Gundala, R. C Program to Convert Hexadecimal to Binary Number System. Computer Programming Language. Retrieved September 12, 2023, from https://www.cprogramcoding.com/p/box-sizing-border-box_730.html

ByJu's. 2023. Hex to Decimal. Retrieved September 12, 2023, from https://byjus.com/maths/hex-to-decimal/

Dheeraj, A. (2020, January 20). JK Flip-Flop Explained in Detail. Retrieved September 19, 2023, from https://dcaclab.com/blog/j-k-flip-flop-explained-in-detail/

DIGITAL ELECTRON I C S

e ISBN 978-629-7514-21-5

9 786297 514215

POLITEKNIK TUANKU SYED SIRAJUDDIN

(online)