INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY, CABLE CENTRATOM TRIESTE

H4.SMR. 403/3

FIFTH COLLEGE ON MICROPROCESSORS: TECHNOLOGY AND APPLICATIONS IN PHYSICS

2- 27 October 1989

Fundamentals of Digital Electronics
Part 1: Boolean Algebra
Part 2: Some Basic Combinational Circuit Chips

S. VENKATARAMAN
Microprocessors Laboratory, ICTP, Trieste

LOGICAL VARIABLES

A LOGICAL VARIABLE A IS A VARIABLE THAT HAS THE FOLLOWING CHARACTERISTICS:

1) A MAY ASSUME ONE OF TWO POSSIBLE VALUES. (0/1, TRUE/FALSE, GREEN/RED, ETC)
2) THESE TWO VALUES MUST BE SUCH THAT, THEY ARE "MUTUALLY EXCLUSIVE" (IF IT IS NOT TAKING ONE OF THE TWO POSSIBLE VALUES IT MUST BE TAKING THE OTHER).

WHEN WE DEFINE A LOGICAL VARIABLE AND ITS MEANING WE HAVE TO FOLLOW THE ABOVE RULES.

EXAMPLE: LET A "REPRESENT PRESSURE" THE ONLY TWO VALUES ARE:

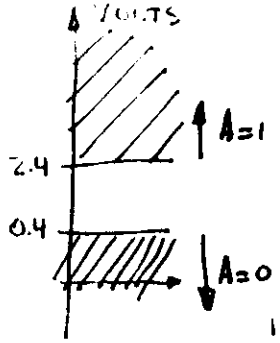
A	MEANING
0	PRESSURE BELOW p_0
1	PRESSURE ABOVE p_0

p_0 = SOME PRESSURE THRESHOLD

WE CAN ALSO CREATE ANOTHER LOGICAL VARIABLE B, FUNCTION OF A, IN THE FOLLOWING WAY:

A	MEANING	B=F(A)	MEANING
0	PRESSURE BELOW p_0	0	DO NOTHING
1	PRESSURE ABOVE p_0	1	SOUND AN ALARM.

[illegible] (TTL ETC.) IN THIS CASE THE VARIABLE "REPRESENTS A VOLTAGE"



A	MEANING	A'
0	VOLTAGE BELOW 0.4V	1
1	VOLTAGE ABOVE 2.4V	0

IN THIS CASE THERE IS A RANGE BETWEEN 0.4 V AND 2.4 V THAT THE VOLTAGE MUST NECESSARILY NOT TAKE. (OF COURSE THIS IS A BREACH OF RULE #2).

THE A' ASSIGNMENT IS USUALLY CALLED NEGATIVE LOGIC, AS OPPOSED TO POSITIVE LOGIC

FUNCTIONS OF ONE VARIABLE

THERE ARE FOUR POSSIBLE FUNCTIONS OUT OF WHICH ONLY TWO ARE INTERESTING.

IDENTITY

A	$B=f_1(A)$
0	0
1	1

INVERSE

A	$B=f_2(A)=\overline{A}$
0	1
1	0

f_1 IS NOT VERY INTERESTING

$B=f_2(A)=\overline{A}$ ← MEANS INVERSE OR COMPLEMENT

A	B	C = F(A,B)
0	0	X ← 0 or 1
0	1	X ← "
1	0	X ← "
1	1	X ←

BEEN C A LOGICAL VARIABLE IT CAN ONLY TAKE THE VALUES 0 OR 1. CONSEQUENTLY THERE ARE ONLY 16 DIFFERENT FUNCTIONS

$2 \times 2 \times 2 \times 2 = 2^4 = 16$

FOR THE TIME BEING LET US CONSIDER THOSE FUNCTIONS THAT ARE OF SPECIAL INTEREST.

THE AND FUNCTION:

$C = A.B$ ← AND OPERATOR IN LOGIC

A	B	C = A.B
0	0	0
0	1	0
1	0	0
1	1	1

THERE IS ONLY ONE CASE (A=1, B=1) THAT GIVES C=1.

IT IS EASY TO SEE THAT THE AND IS COMMUTATIVE

$C = A.B = B.A$

(TO OBTAIN A ZERO IT DOESN'T MATTER WHICH IS ZERO A or B, AS LONG AS ONE OF THE INPUTS IS ZERO).

THE AND FUNCTION IS ALSO ASSOCIATIVE:

$D = (A.B).C = A.(B.C)$

A	B	C	(A.B)	(A.B).C	B.C	A.(B.C)
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

IDENTICAL

WE HAVE PROVED THAT $(A.B).C = A.(B.C)$ SHOWING THAT BOTH GIVE THE SAME RESULT FOR ALL POSSIBLE INPUTS. THIS METHOD OF PROOF IS CALLED "PERFECT INDUCTION".

THE OR FUNCTION:

$C = A + B$

OR OPERATOR IN LOGIC

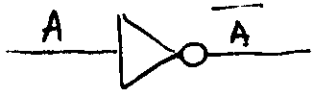
A	B	A+B=C
0	0	0
0	1	1
1	0	1
1	1	1

THERE IS ONLY ONE CASE (A=0, B=0) FOR WHICH C=0.

a) $A+B = B+A$ THE OR FUNCTION IS COMMUTATIVE

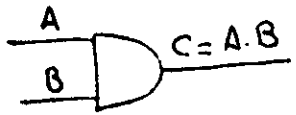
b) $(A+B)+C = A+(B+C)$ THE OR FUNCTION IS ASSOCIATIVE

SYMBOLS FOR THE INVERSION, AND AND OR FUNCTIONS



A	$\overline{A}$
0	1
1	0

INVERSION



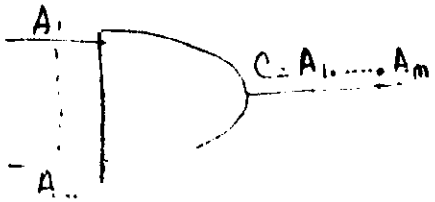
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

TWO INPUT AND

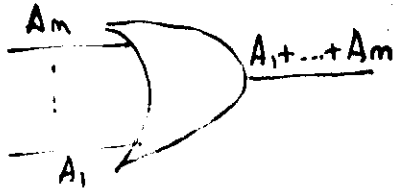
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

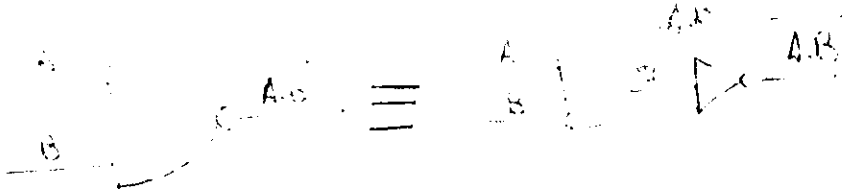
TWO INPUT OR

N-INPUT AND

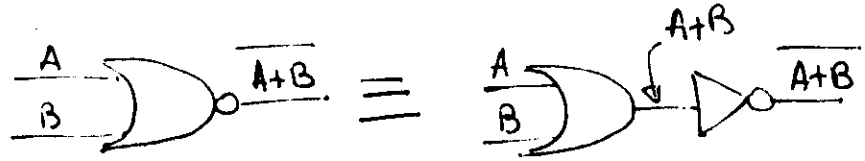


N-INPUT OR



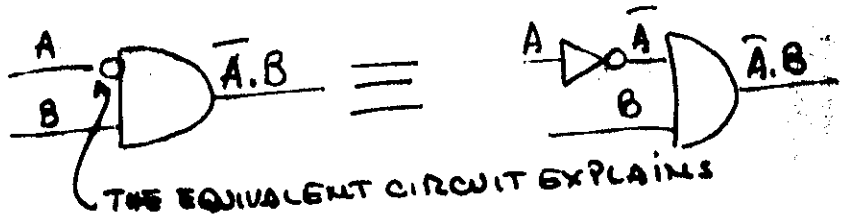


NOT AND OR FOR SHORT NAND.



NOT OR OR FOR SHORT NOR

OTHER SYMBOLS



THE EQUIVALENT CIRCUIT EXPLAINS THE MEANING.

ETC.

BOOLEAN ALGEBRA: THE CONCEPT OF A LOGICAL VARIABLE AND ITS FUNCTIONS WAS INVENTED IN THE MID 1800 BY GEORGE BOOLE IN CONNECTION WITH HIS STUDIES ON THOUGHT PROCESSES.

IN 1938 CLAUDE SHANNON, FROM BELL LABS. CREATED AN ALGEBRA ADAPTED TO ELECTRONIC LOGIC DESIGN USUALLY CALLED "SWITCHING ALGEBRA".

WE HAVE ALREADY ENCOUNTERED MOST THAT IS NEEDED TO UNDERSTAND IT, BUT AS WITH MOST THEORIES, WE NEED TO HAVE A FEW MORE "TOOLS" TO EFFECTIBLY USE IT.

LET US SEE SOME OBVIOUS (AND SOME NOT SO OBVIOUS) CONSEQUENCES OF WHAT WE HAVE ALREADY LEARNT.

THEOREMS:

$$\bar{\bar{A}} = A$$

$A+0=A$ (1)	$A.1=A$ (1')
$A+1=1$ (2)	$A.0=0$ (2')
$A+A=A$ (3)	$A.A=A$ (3')
$A+\bar{A}=1$ (4)	$A.\bar{A}=0$ (4')

$A+(A.B)=A$ (5)	$A.(A+B)=A$ (5')
$AB+A\bar{B}=A$ (6)	$(A+B)(A+\bar{B})=A$ (6')
$A+(\bar{A}.B)=(A+B)$ (7)	$A(\bar{A}+B)=A.B$ (7')
$A+(B.C)=(A+B).(A+C)$ (8)	$A.(B+C)=(A.B)+(A.C)$ (8')
$(A.B)+(\bar{A}.C)=(A+C).(\bar{A}+B)$ (9)	$(A+B).(\bar{A}+C)=(A.C)+(\bar{A}.B)$ (9')
$(A.B)+(\bar{A}C)+(BC)=(A.B)+(\bar{A}.C)$ (10)	$(A+B).(\bar{A}+C).(B+C)=(A+B).(\bar{A}+C)$ (10')

DE MORGAN'S THEOREM.

$$\overline{A.B.\ldots} = \bar{A}+\bar{B}+\ldots \quad (11)$$

$$\overline{A+B+\ldots} = \bar{A}.\bar{B}.\ldots \quad (12)$$

WELL KNOWN TRUTH TABLES

A.B = Z		A+B = Z
0.0 = 0		0+0 = 0
0.1 = 0	(I)	0+1 = 1 (II)
1.0 = 0		1+0 = 1
1.1 = 1		1+1 = 1

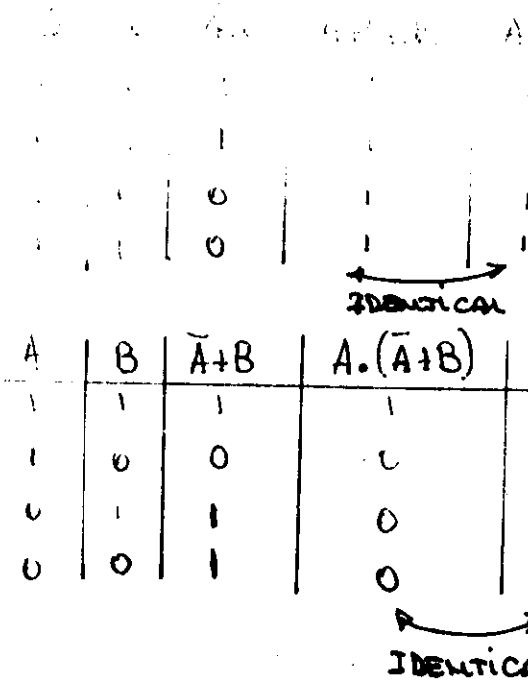
OBSERVE THAT WE GET (II) FROM (I) AND VICE-VERSA IF WE:

a) INTERCHANGE . WITH +

b) INTERCHANGE 0's WITH 1's.

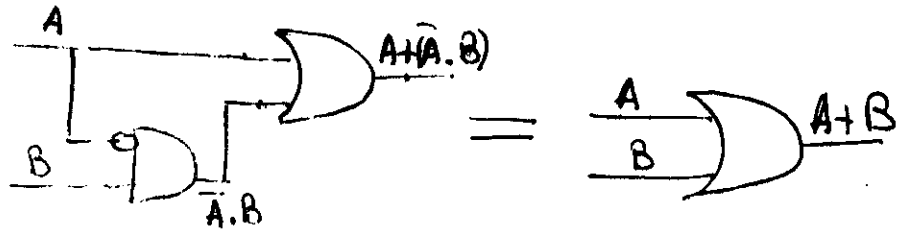
THEN IF WE HAVE A THEOREM RELATING LOGIC VARIABLES WE CAN IMMEDIATELY WRITE DOWN A SECOND THEOREM BY INTERCHANGING + WITH . AND INTERCHANGING 0's WITH 1's. THIS TWO THEOREMS SO RELATED ARE CALLED DUAL THEOREMS.

EXAMPLE: LET US OBTAIN, BY DUALITY, 7' FROM 7.



THEOREMS DRAWN AS CIRCUITS

WE CAN REPRESENT (DRAW) THEOREM 7 AS



WE SEE HERE THAT THE SAME LOGICAL FUNCTION CAN BE IMPLEMENTED BY USING PHYSICALLY DIFFERENT CIRCUITS.

[illegible] A LOGICAL FUNCTION ?

FIRST LET US ASSUME (OF COURSE IT IS TRUE) THAT AND, NOT AND OR ARE SUFFICIENT TO EXPRESS ANY LOGICAL FUNCTION

BUT IT IS EASILY SHOWN THAT WE COULD DISPENSE OF EITHER AND OR OR. FOR EXAMPLE: WE WRITE AND AS A FUNTION OF OR AND NOT

$$A.B = \overline{\overline{A}+\overline{B}}$$ DE MORGAN'S

EXAMPLE: OR AS A FUNCTION OF AND AND NOT

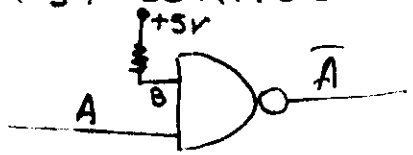
$$A+B = \overline{\overline{A}.\overline{B}}$$ DE MORGAN'S

WHAT IS MORE, ALL LOGICAL FUNCTIONS SYNTHESIZE USING ONLY NAND OR NOR GATES. BUT DESIGNING CIRCUITS ONLY WITH ONE GATE, BE IT NAND OR NOR IS INCONVENIENT AT BEST. THE INCONVENIENCE RESIDES IN THE FACT THAT, NEITHER NAND NOR NOR ARE ASSOCIATIVE.

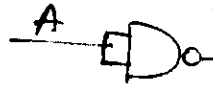
THE USUAL PROCEDURE IS TO USE AND, OR AND NOT. OBTAIN THE SIMPLEST GATE STRUCTURE AND TRANSFORM IT INTO ANOTHER STRUCTURE USING ONLY NANDs OR NORs (IF YOU WANT).

HOW TO OBTAIN A NOT FROM A NAND

JUST CONNECT

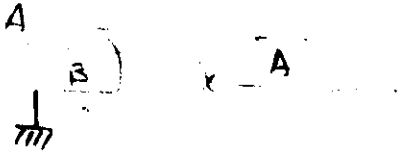


A	B	$\overline{A.B}=\overline{A}$
0	0	1
0	1	1
1	0	1
1	1	0

; WHY THE ABOVE CONNECTION IS BETTER THAN? 

HOW TO OBTAIN A NOT FROM A NOR.

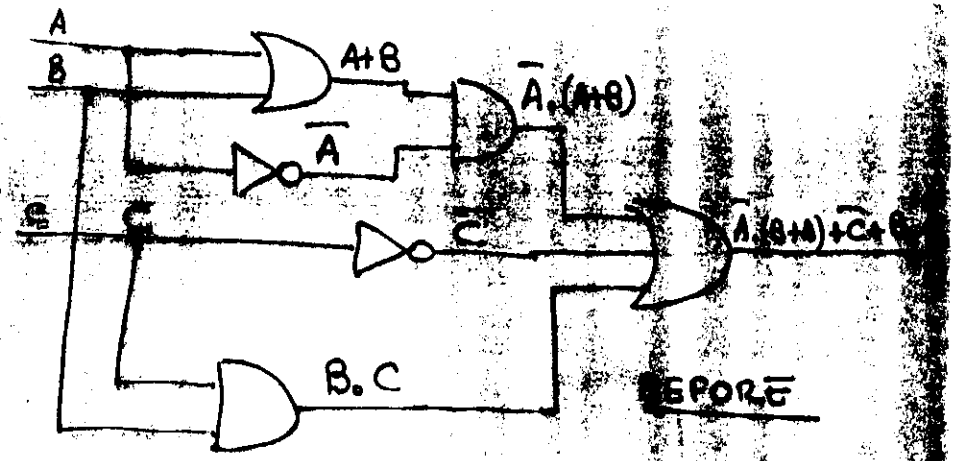
JUST CONNECT



A	B	$\overline{A+B}=\overline{A}$
0	0	1
0	1	0
1	0	0
1	1	0

[illegible] IS THE ABOVE CONNECTION [illegible] THAN? 

EXAMPLE: $W = \bar{A}.(A+B) + \bar{C} + (B.C)$



BEFORE

$\bar{A}.(A+B)+\bar{C}+B.C = \bar{A}.(A+B)+\bar{C}+B$

USING $A+\bar{A}.B = A+B$

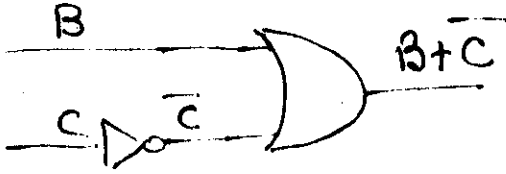
$\bar{A}.(A+B)+\bar{C}+B = \bar{A}.A+\bar{A}.B+\bar{C}+B$

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

$A.\bar{A} = 0$

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

$A+A.B = A$

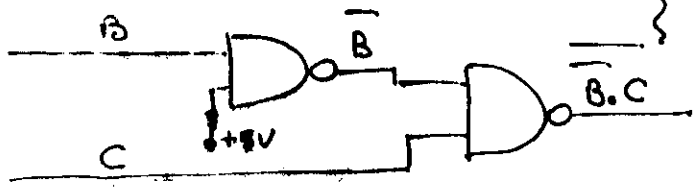


(12)

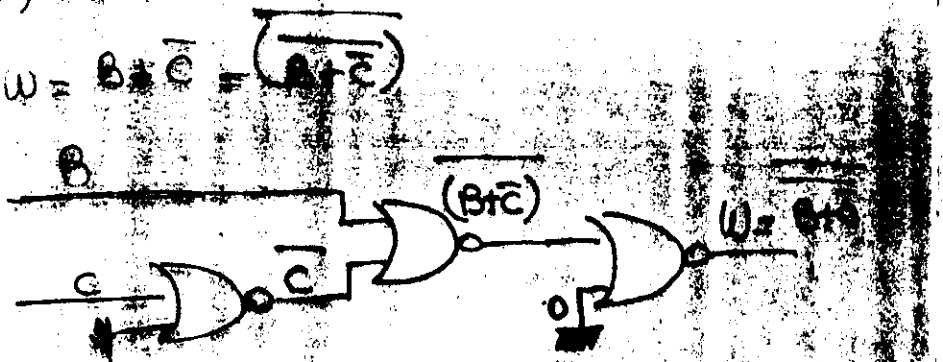
IMPLEMENTATE $W = B+\bar{C}$ WITH NAND'S

USING $A+B = \overline{\bar{A}.\bar{B}}$

$W = B+\bar{C} = \overline{\bar{B}.\bar{\bar{C}}} = \overline{\bar{B}.C}$



B) NOW USING NOR's



PROBLEM: SIMPLIFY

$W = A.[B+C.(D+\bar{A})]$

NOW LET US SHOW A GRAPHIC PROCEDURE FOR THE SIMPLIFICATION WORK. THIS PROCEDURE ALLOWS A VISUAL APPRECIATION THAT REQUIRES LITTLE MEMORIZATION

→ 13

...A KARNAUGH MAP METHOD

...IN OUR WAY TO USING THE GRAPHIC PROCEDURE WE INTRODUCE THE DEFINITION OF "FUNDAMENTAL PRODUCTS", ALSO CALLED "MINTERMS".

WE ALREADY SAW THAT AN AND SUCH AS

$$A.\bar{B}.C$$

WILL OUTPUT A ONE ONLY IF

$A = 1$ $\quad B = 0 \Rightarrow \bar{B} = 1$

$C = 1$

AND ONLY IN THIS CASE

GIVEN N LOGICAL VARIABLES

$$A_1, A_2, \ldots A_N$$

WE CALL A FUNDAMENTAL PRODUCT OR MINTERM TO AN AND TERM IN WHICH ALL N VARIABLES PARTICIPATE, COMPLEMENTED OR NOT.

EXAMPLE: GIVEN A, B AND C

$A.\bar{B}.C$, $\bar{A}.\bar{B}.C$ } ARE F. PRODUCTS

$A.B$, $A.\bar{C}$ } ARE NOT

...USING A TRUTH TABLE AS "SUM OF FUNDAMENTAL PRODUCTS"

SUPPOSE WE ARE GIVEN THE FOLLOWING TRUTH TABLE

A	B	C	F(A,B,C)	
0	0	0	0	
0	0	1	1	$\leftarrow \bar{A}.\bar{B}.C$
0	1	0	0	
0	1	1	1	$\leftarrow \bar{A}.B.C$
1	0	0	1	$\leftarrow A.\bar{B}.\bar{C}$
1	0	1	0	
1	1	0	0	
1	1	1	0	

WE SAW THAT THE AND FUNCTION GIVES A "1" AS AN OUTPUT ONLY WHEN EACH OF ITS VARIABLES ARE EQUAL TO "1"

SO THE PRODUCTS $\bar{A}.\bar{B}.C$ a)

$\bar{A}.B.C$ b)

$A.\bar{B}.\bar{C}$ c)

GIVE THE "1's" OF F. THEN F(A,B,C) OUTPUTS A "1" IF a) OR b) OR c) ARE "1"

$$F(A,B,C) = (\bar{A}.\bar{B}.C) + (\bar{A}.B.C) + (A.\bar{B}.\bar{C})$$

THEN, WRITING A TRUTH TABLE IN ALGEBRAIC FORM NATURALLY LEADS US TO A "SUM OF FUNDA

... MINTERM ELEMENTS OS WITH AN AND TERM OF THREE VARIABLES WHERE ONE OR TWO VARIABLES ARE MISSING WE CAN EASILY CREATE "FUNDAMENTAL PRODUCTS".

EXAMPLE:

$$A.B = A.B.(C+\bar{C}) = ABC + AB\bar{C}$$

BECAUSE $C+\bar{C}=1$

EXAMPLE:

$$\bar{A} = \bar{A}.(B+\bar{B}) = \bar{A}.B + \bar{A}.\bar{B} = \bar{A}B.(C+\bar{C}) + \bar{A}.\bar{B}(C+\bar{C})$$

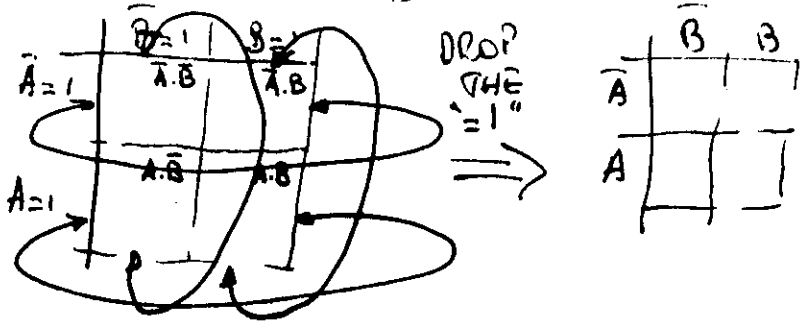
$$\bar{A} = \bar{A}BC + \bar{A}B\bar{C} + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C}$$

SO, NOW WE KNOW THAT ANY LOGICAL FUNCTION CAN BE WRITTEN AS A "SUM OF FUNDAMENTAL PRODUCTS".

DRAWING KARNAUGH MAPS

A KARNAUGH MAP IS A TABLE THAT HAS ONE SQUARE FOR EACH POSSIBLE "FUNDAMENTAL PRODUCT". NEIGHBORING SQUARES THAT HAVE A SIDE IN COMMON DIFFER IN A VARIABLE THAT IS THE COMPLEMENT OF THE ORIGINAL

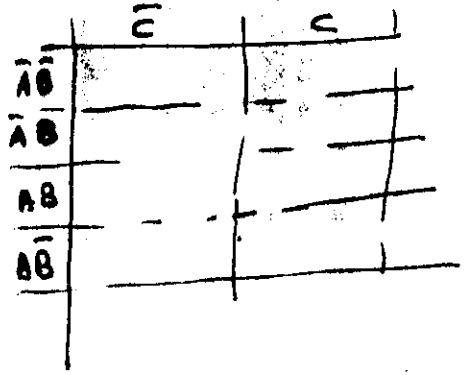
A) TWO VARIABLES



ALSO SEE THAT $\bar{A}B$ IS ALSO A NEIGHBOR OF $\bar{A}\bar{B}$, ETC

IT IS NOT SEEN INMEDIATLY BECAUSE WE CHOSE TO START WITH $\bar{A}.\bar{B}$ IN THE UPPER LEFT SQUARE.

B) THREE VARIABLES



	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$				
$\bar{A}B$				
AB				
$A\bar{B}$				

FROM TRUTH TABLES TO KARNAUGH MAPS.

LET US SUPPOSE WE HAVE THE FOLLOWING TRUTH TABLE

A	B	C	D	Z(A,B,C,D)	
0	0	0	0	0	
0	0	0	1	1	$\bar{A}\bar{B}\bar{C}D$
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	0	
0	1	1	0	1	$\bar{A}BC\bar{D}$
0	1	1	1	1	$\bar{A}BCD$
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	0	
1	1	1	0	1	$ABC\bar{D}$
1	1	1	1	0	

NOW WRITE A "1" FOR EACH MINTERM ON THE CORRESPONDING SQUARE OF THE MAP, ELSE WRITE A ZERO

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	0	0
$\bar{A}B$	0	0	1	1
AB	0	0	0	1
$A\bar{B}$	0	0	0	0

(10)

THIS TABLE SHOWS THE OUTPUT FOR EACH MINTERM CONDITION WHILE A KARNAUGH MAP ON THE OTHER HAND SHOWS THE FUNDAMENTAL PRODUCTS NEEDED TO PRODUCE THE OUTPUT "1s".

PAIRS, QUADS AND OCTETS

TWO 1's ADJACENT EITHER HORIZONTALLY OR VERTICALLY ARE CALLED A PAIR

	$\bar{C}\bar{B}$	$\bar{C}B$	CB	$C\bar{B}$
$\bar{A}\bar{B}$	0	1	0	0
$\bar{A}B$	0	0	1	0
AB	0	0	0	1
$A\bar{B}$	0	0	0	0

2

3

IF YOU SEE A PAIR, YOU KNOW YOU CAN ELIMINATE FROM THE PAIR OF F. PRODUCTS THE VARIABLE THAT CHANGES FORM.

PAIR 1:

$$y = \bar{A}BC\bar{D} + ABC\bar{D} = (A+\bar{A})BC\bar{D}$$

$$= BC\bar{D}$$

PAIR 2:

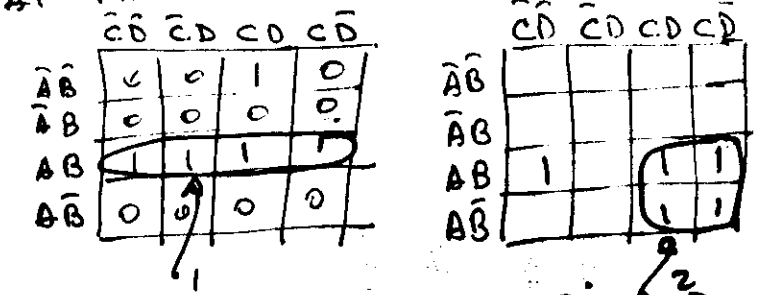
$$y = \bar{A}BCD + \bar{A}BC\bar{D} = \bar{A}BC(D+\bar{D}) = \bar{A}BC$$

PAIR 3:

$$Z = \bar{A}\bar{B}\bar{C}D + A\bar{B}\bar{C}D = \bar{B}\bar{C}D$$

QUAD

IT DEFINES ITSELF BY JUST LOOKING AT THE DRAWING



WHEN YOU SEE A QUAD TWO VARIABLES AND ITS COMPLEMENTS DROP FROM THE BOOLEAN EQUATION THAT REPRESENTS THE QUAD.

QUAD 1:

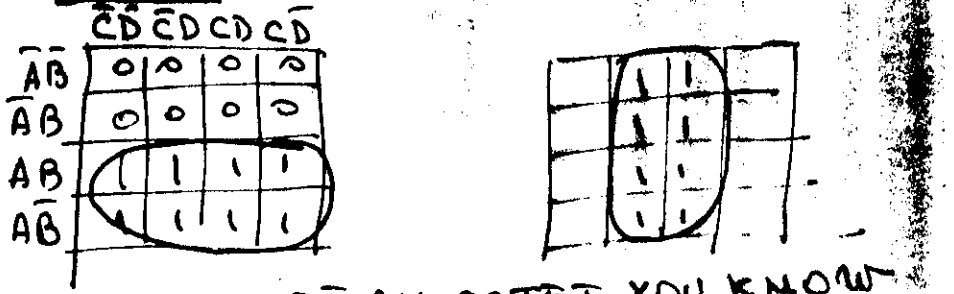
$y = AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + AB\bar{C}\bar{D}$

$= AB(\bar{C}\bar{D} + \bar{C}D + CD + C\bar{D})$

$= AB(\bar{C}(\bar{D}+D) + C(D+\bar{D})) = AB(\bar{C}+C) = \underline{AB}$

HOW NICE!!

OCTET

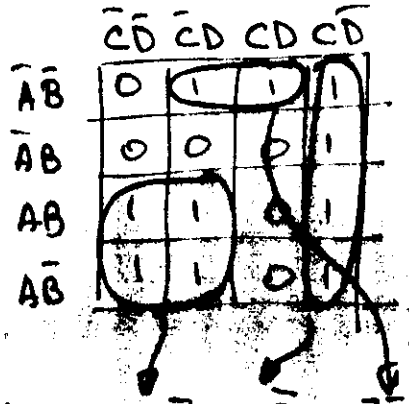


WHEN YOU SEE AN OCTET YOU KNOW THREE VARIABLES DROP FROM THE BOOLEAN EXPRESSION OF THE OCTET

SIMPLIFYING A KARNAUGH MAP

CIRCLE THE OCTETS FIRST
QUADS SECOND
PAIRS LAST

EXAMPLE:



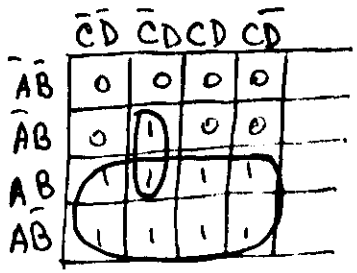
2 QUADS.
1 PAIR

$F(A,B,C,D) = A\bar{C} + C\bar{D} + \bar{A}\bar{B}D$ DONE

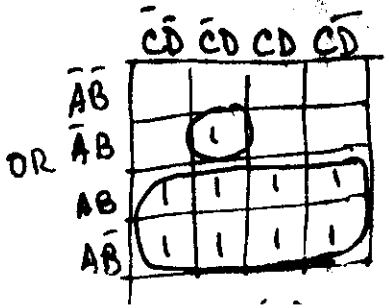
OVERLAPPING GROUPS.

WHEN YOU ENCIRCLE GROUPS YOU ARE ALLOWED TO USE THE SAME "1" MORE THAN ONCE. WHY? BECAUSE (AB) AND $(AB+AB)$ ARE EQUAL FROM A LOGICAL POINT OF VIEW

TAKE FOR EXAMPLE:



OR



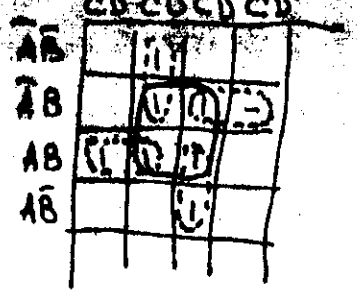
5. [illegible] THE SAME FUNCTION

1) $F(A,B,C,D) = \overbrace{A}^{\text{OCTET}} + \overbrace{B\bar{C}D}^{\text{PAIR}}$

2) $F(A,B,C,D) = \overbrace{A}^{\text{OCTET}} + \overbrace{\bar{A}B\bar{C}D}^{\text{LONELY "1"}}$

THERE ARE LIMITS ABOUT USING "1's" MORE THAN ONCE. ONE HAS TO ELIMINATE GROUPS WHOSE "1's" ARE COMPLETELY OVERLAPPED BY OTHER GROUPS.

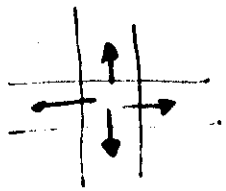
EXAMPLE:



IN THIS CASE IT IS BETTER NOT TO CONSIDER THE QUAD.

A WORD ABOUT NEIGHBORS:

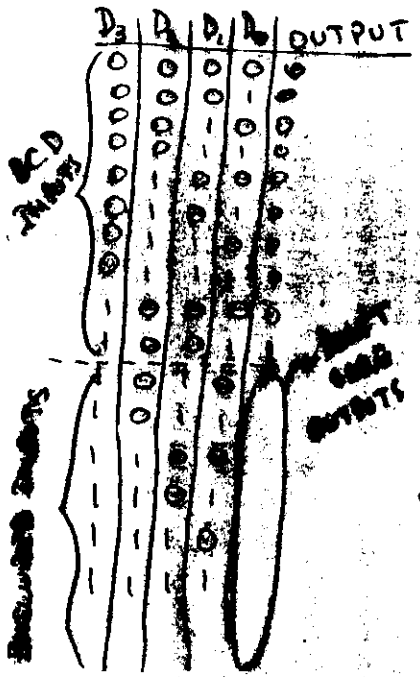
WE LIVE IN A 3-D WORLD BUT WE HAVE TO RELINQUISH ONE DIMENSION WHEN WE WORK ON A FLAT SURFACE AS THIS PAGE. A SQUARE ON A CHECKER BOARD HAS, AT THE MOST, FOUR NEIGHBORS. SO IF WE ARE DEALING WITH FIVE OR MORE VARIABLE YOU HAVE TO SEE WHERE THE OTHER NEIGHBORS ARE. (PLAY WITH IT)



[illegible] CONDITIONS:

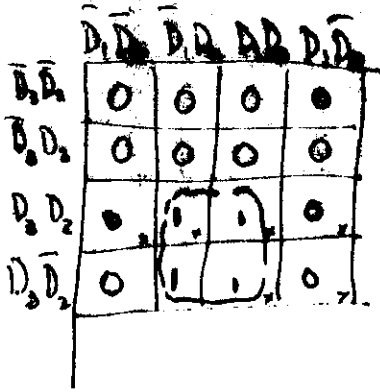
PROBLEM: A BCD INPUT DRIVES A DECODER. THE DECODER IS TO PRODUCE A "1" OUTPUT ONLY FOR A INPUT OF $9_{10} \rightarrow 1001$

THE TRUTH TABLE IS:



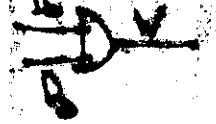
THE EXCLUDED INPUTS CAN NEVER OCCUR AS WE SAID THAT THE INPUT WILL BE A BCD NUMBER.

SO FOR THOSE [illegible] WE DON'T CARE [illegible] OUTPUTS ARE. WE ARE FREE TO USE THEM AS ADVANTAGE TO SIMPLIFY THE KARNAUGH MAP



ENCIRCLE THE "1's" FROM THE TRUTH TABLE WITH AS MANY "1's" FROM DON'T CARE TO FORM THE LARGEST POSSIBLE GROUP.

$W = D_3D_0$



STEPS TO FOLLOW WHEN USING THE KARNAUGH METHOD FOR SIMPLIFYING BOOLEAN EQUATIONS

STEP 1: BUILD YOUR TRUTH TABLE ACCORDING TO THE REQUIREMETS OF YOUR APPLICATION.

STEP 2: DRAW "1"s ON THE KARNAUGH MAP FOR EACH FUNDAMENTAL PRODUCT NECESSARY TO PRODUCE THE "1s" AT THE OUTPUTS.

STEP 3: ENCIRCLE THE OCTETS, QUADS AND PAIRS. ENCIRCLE THE "ISOLATED" "1"s.

STEP 4: ELIMINATE REDUNDANT GROUPS.

STEP 5: WRITE THE BOOLEAN EQUATION BY ORING THE PRODUCTS CORRESPONDING TO THE GROUPS.

STEP 6: IF NEEDED REWRITE USING ONLY NANDs OR NORs.

ELECTRONIC CIRCUITS

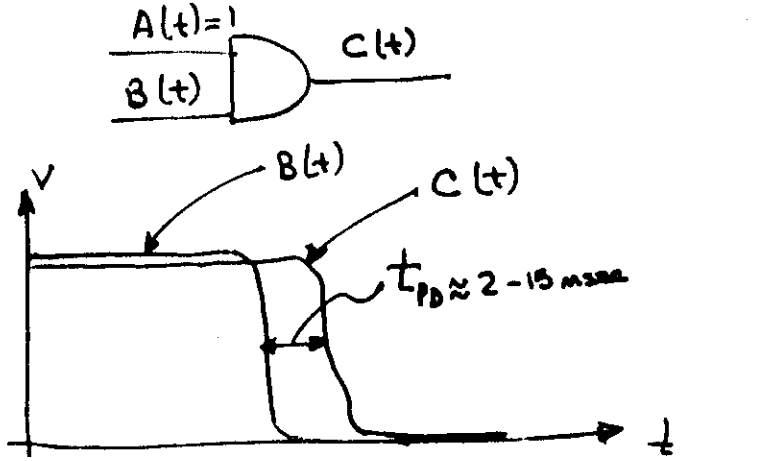
IN THIS SHORT AND BASIC INTRODUCTORY SET OF LECTURES WE DELIBERATELY AVOID ANY DISCUSSION OF THE ELECTRONIC CIRCUITRY USED TO IMPLEMENT PHYSICALLY OUR DIGITAL CIRCUITS. WE ARE MAINLY CONCERNED WITH THE LOGIC THAN THE ELECTRONICS. NEVER THE LESS SOME CONSIDERATIONS ARE IN ORDER.

SPEED, PROPAGATION TIME

ANY DIGITAL SYSTEM CAN BE ASSEMBLED FROM GATES. THESE GATES CHANGE THEIR OUTPUTS AS A RESPONSE TO CHANGES ON THEIR INPUTS. HOW FAST A DIGITAL SYSTEM IS, DEPENDS MAINLY ON HOW FAST THE COMPONENT GATES ARE

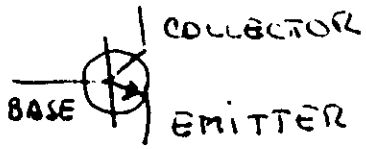
THE MOST IMPORTANT FACTOR RELATED TO THE SPEED OF A GATE IS THE PROPAGATION TIME.

THE PROPAGATION TIME IS THE TIME REQUIRED FOR A GATE OUTPUT TO RESPOND TO A CHANGE ON THE GATE INPUTS.

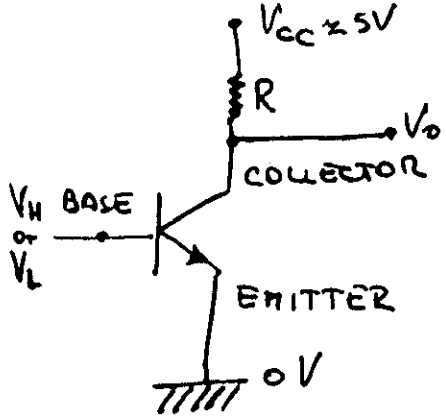


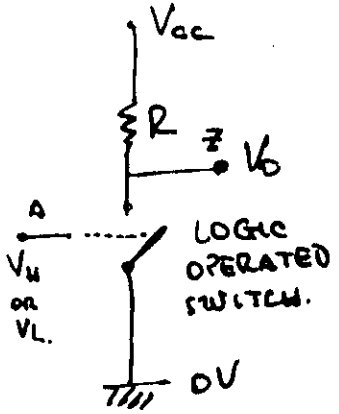
THE SPEED OF THE GATES WILL DEPEND ON THE TECHNOLOGY CHOSEN TO IMPLEMENT THE LOGIC.

A TRANSISTOR IS A SEMICONDUCTOR DEVICE WITH THREE TERMINALS



WITH IT WE CAN CREATE A SWITCH



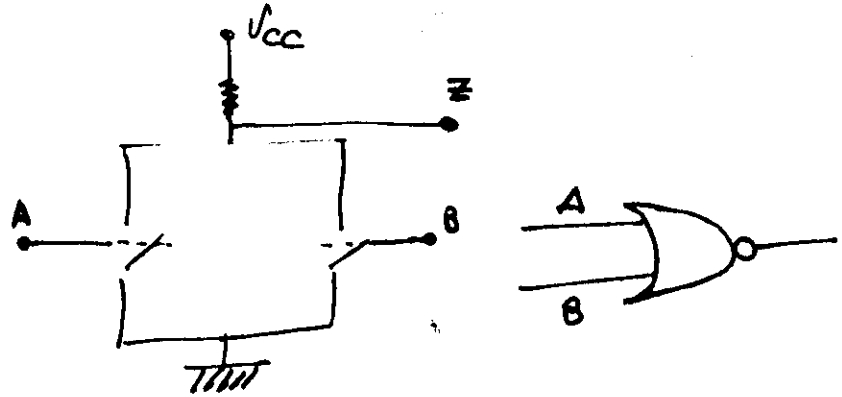


WE APPLY TO THE BASE ONE OF TWO VOLTAGE VALUES $V_H \approx 5V$ OR $V_L \approx 0V$. ALL WE NEED TO KNOW ABOUT THE CIRCUIT IS THAT WHEN WE APPLY V_L THE TRANSISTOR DOES NOT CONDUCT. CURRENT, THERE IS NO DROP ON R AND CONSEQUENTLY $V_o = V_{cc}$. ON THE OTHER HAND IF WE APPLY V_H THEN $V_o \approx 0V$

V_A	POSICION DEL INTERRUPTOR	V_Z
V_L	OPEN	V_{CC}
V_H	CLOSED	0

SO WE HAVE CREATED.

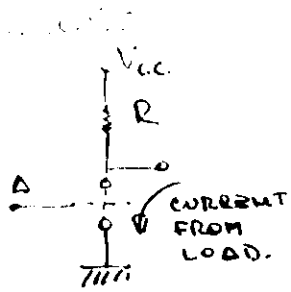
WE CAN EASILY WIRE A NOR GATE.



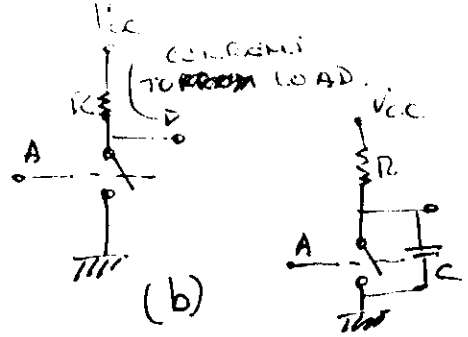
V_A	V_B	V_Z
V_L	V_L	V_{CC}
V_L	V_H	V_L
V_H	V_L	V_L
V_H	V_H	V_L

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

NOR



(a)



(b)

IT WOULD BE A GOOD CIRCUIT BECAUSE:

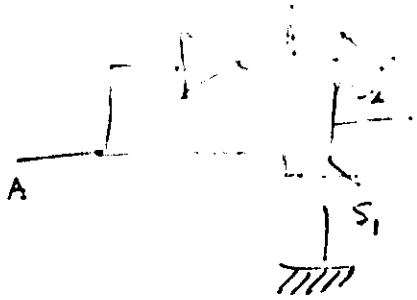
a) WHEN THE SWITCH IS OPEN (FIG b) THE DRIVING GATE SOURCES CURRENT TO ITS LOAD. IT FLOWS THRU R. THIS CURRENT IS SMALL DUE TO THE PRESENCE OF R. CAUSES NO IMPORTANT LOSS OF VOLTAGE.

b) WHEN CLOSED, IT CALLS THE LOAD TO SINK CURRENT TO GROUND. THE RESISTANCE OF THE TRANSISTOR IS LOW. THIS SITUATION IS FORTUNATE BECAUSE THE CURRENT TO SINK IS RELATIVELY LARGE.

IT IS NOT SUCH A GOOD CIRCUIT BECAUSE:

THERE IS ALWAYS A CAPACITOR PRESENT. IT CAN NOT AVOIDED BECAUSE THE SWITCH IS MADE WITH A TRANSISTOR. THE CAPACITY (THAT ALSO IS INCREASED WITH FAN-OUT) MUST BE CHARGED EVERY TIME THERE IS A TRANSITION FROM 0 → 1. IT CHARGES SLOWLY BUT DISCHARGES FAST.

TO SPEED THE CHARGING, THE RESISTOR CAN BE REPLACED BY ANOTHER SWITCH, THE NEW CIRCUIT IS CALLED "TOTEM POLE OUTPUT."



THE TOTEM POLE CONFIGURATION

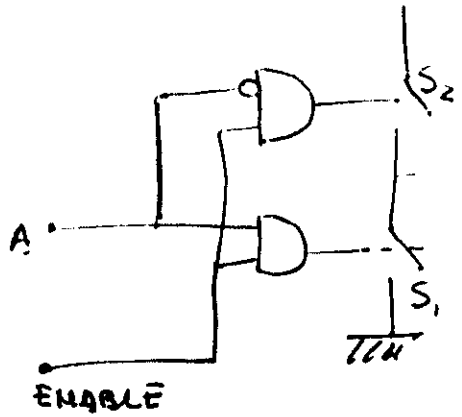
WHEN A IS LOW S_1 IS OPEN AND S_2 IS CLOSED SO THAT Z IS HIGH. WHEN A IS HIGH THE SWITCHES REVERSE AND Z IS LOW.

	A	S_2	S_1	Z	
0	V_L	CLOSED	OPEN	V_{CC}	1
1	V_H	OPEN	CLOSED	0	0

IN THIS CONFIGURATION BOTH SWITCHES CAN NOW BE OPENED AT THE SAME TIME.

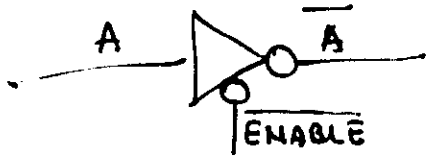
THE EFFECT IS THE SAME, BUT IT CHANGES FASTER.

A DIGITAL MODIFICATION OF THE TOTEM-POLE OUTPUT IS.



$Z=\bar{A}$

THIS CIRCUITS ALLOW BOTH. S_1 AND S_2 TO BE OPENED AT THE SAME TIME PROVIDING AN OUT PUT WITH VERY (1) HIGH IMPEDANCE.



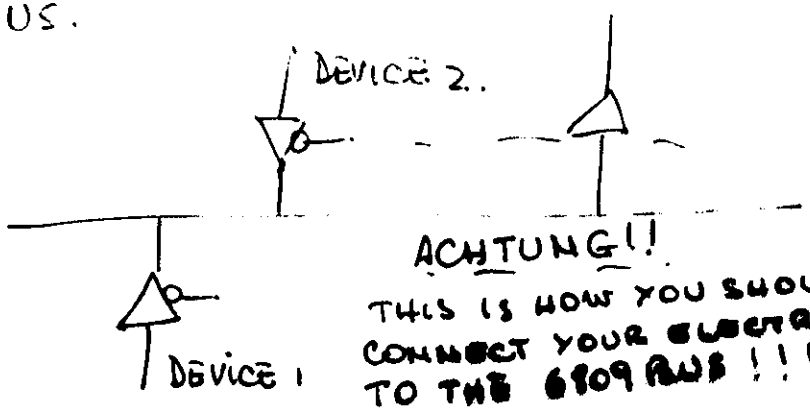
(2)

THE CIRCUIT IN (1) OPERATES AS A REGULAR INVERTER AS LONG AS LONG AS ENABLE IS HIGH, BUT IF ENABLE IS LOW IT PRESENTS ITS OUTPUT AT HIGH IMPEDANCE (CIRCUIT IS DISABLED).

GENERALLY THE LOGIC OF THE ENABLE IS REVERSED (FIG (2)) AND THE DRIVER IS ENABLED WHEN ENABLE IS LOW.

EXTREMELY IMPORTANT FOR THE

COMMON LINE (WIRE), THE SO CALLED. BUS.

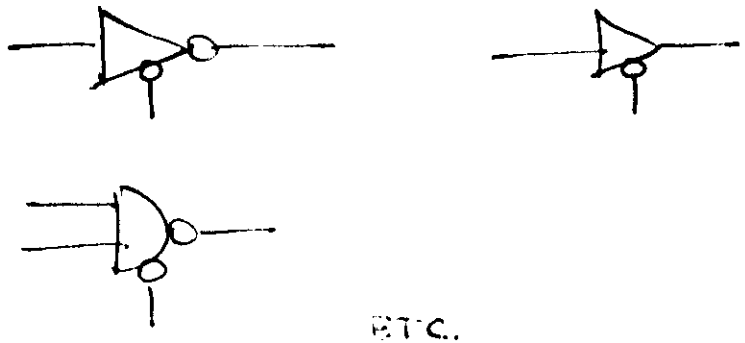


ACHTUNG!!

THIS IS HOW YOU SHOULD CONNECT YOUR ELECTRONIC TO THE 6809 BUS !!!

ONLY ONE DEVICE SHOULD BE CONNECTED TO THE BUS. IF THIS RULE IS NOT FOLLOWED THE SYSTEM WILL NOT WORK OR WHAT IS WORSE CHIPS MIGHT GET DAMAGED.

NOT ONLY THE INVERTER COMES IN THREE-STATE.



ETC.

CIRCUIT CHIPS (FAMILIES)

THE 7400 SERIES OF TTL (TRANSISTOR-TRANSISTOR LOGIC) CHIPS ARE THE LEAST EXPENSIVE AND THE MOST WIDELY USED LOGIC CIRCUIT CHIPS.

SOME TYPICAL CHARACTERISTICS

SERIES	POWER TRANSISTOR TYPE	t_{PD} ns	POWER DISSIPATION mW
74LS...	LOW POWER SCHOTTKY	9.5	2
74L....	LOW POWER STANDARD	33	1
74S...	STANDARD POWER. SCHOTTKY	3	19
74	STANDARD POWER STANDARD	10	10
74H...	HIGH POWER STANDARD	6	22
74F...	FAST	3	2

IN GENERAL HIGHER SPEED MEANS ALSO HIGHER POWER DISSIPATION.

[illegible] FAMILIES OF CMOS

THE CMOS FAMILY:

THE DISTINCTIVE FEATURE OF THIS FAMILY IS THAT THE INPUT CURRENT REQUIRED ~~AS~~ IS INSIGNIFICANT. GENERALLY IN THE ORDER OF 1pA OR LESS. IN GENERAL THIS FAMILY IS SLOWER THAN THE TTL FAMILY (≈100 ns VERSUS ≈ 20 ns)

THE ECL FAMILY:

THE PRINCIPAL MERIT OF THIS FAMILY IS ITS HIGH SPEED (TYPICALLY 2 ns). THIS SPEED GOES TOGETHER WITH A LARGE DISSIPATION (≈ 25 mW per GATE).

THERE ARE ALSO THE ECL II SERIES (t_{PD} ≈ 4 ns) AND ECL III (t_{PD} ≈ 1 ns).

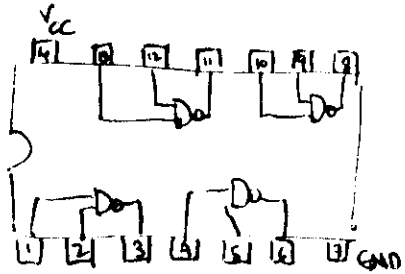
CMOS: COMPLEMENTARY SYMMETRY METAL-OXIDE

[illegible] GATES:

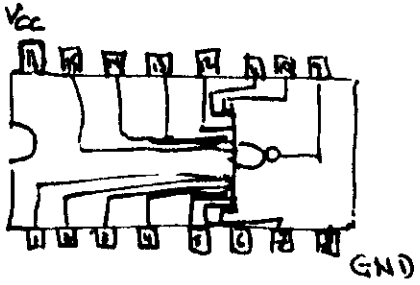
[illegible] QUAD 2-INPUT NAND GATE

← THE NAND HAS TWO INPUTS

THERE FOUR ON THE CHIP



74 133 13-INPUT NAND GATE



ETC.

AT THE LABORATORY YOU WILL FIND SEVERAL MANUAL WHERE THE AVAILABLE CHIPS ARE LISTED AS WELL AS ITS CHARACTERISTIC PARAMETERS.

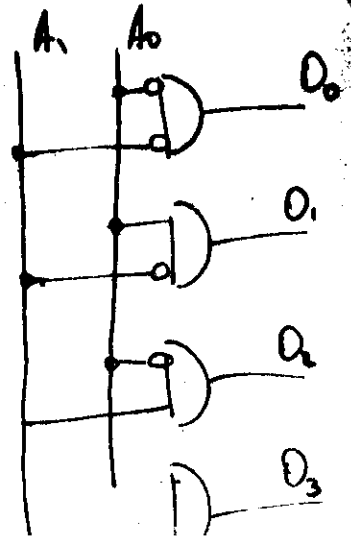
BUT LET US SEE SOME SPECIALLY IMPORTANT ONES. (THOSE THAT WILL BE USED AT THE LABORATORY)

DECODERS: A DECODER IS A CHIPS THAT GIVES YOU AN OUTPUT THAT SHOWS YOU EXPLICITY WHAT IS IMPLICIT AT THE INPUT THAT IS TO SAY IT DECODES WHAT IS ENCODED IN THE INPUT SIGNAL.

LET AS DESIGN A CHIP THAT WILL SHOW AT ITS OUTPUTS WHICH NUMBER HAS BEEN ENCODED IN BINARY FORM IN ITS TWO INPUTS

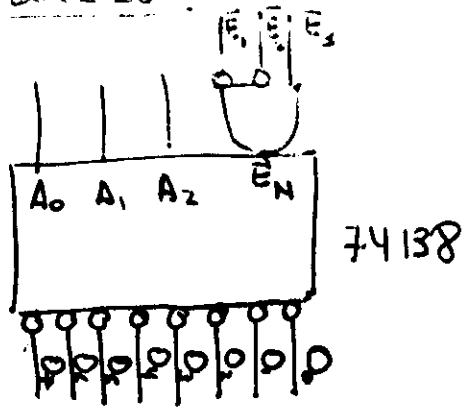
TRUTH TABLE

#	A_1	A_0	O_0	O_1	O_2	O_3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
2	1	0	0	0	1	0
3	1	1	0	0	0	1



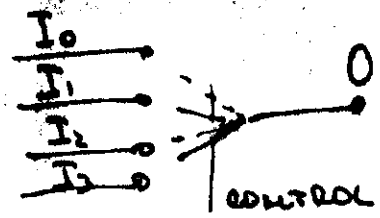
THIS CIRCUIT WOULD DO IT

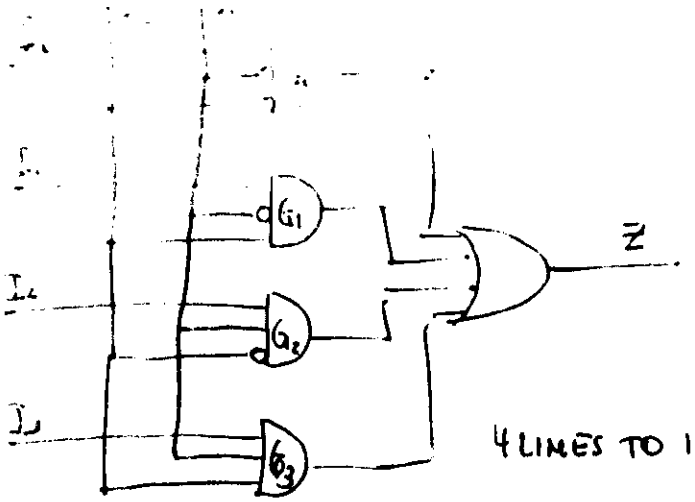
74138 DECODER



ALL THE OUTPUT LINES ARE HIGH WHEN E_N IS LOW. ELSE THE LINE WHOSE SUBINDEX CORRESPONDS TO THE BINARY NUMBER PRESENT AT THE INPUT GOES LOW

MULTIPLEXERS: A MULTIPLEXER CHIP PERFORMS THE OPERATION OF GIVEN AT THE OUTPUT A SIGNAL EQUAL OR DEPENDENT ON ONE INPUT CHOSEN AMONG SEVERAL





4 LINES TO 1 LINE MULTIPLEXER

TYPICAL EXAMPLE 74151 8-INPUT MULTIPLEXER. THIS MULTIPLEXER CAN PERFORMS ALSO OTHER FUNCTIONS.

ENCODER: IT PRODUCES THE INVERSE FUNTION OF A DECODER. GENERALLY TO A CERTAIN INPUT LINE (WHOSE LEVEL IS DIFFERENT TO ALL OF THE OTHERS) CORRESPOND A CERTAIN CODE WORD. WITH BITS $A_0, A_1, \ldots$ GIVING THIS WORD.

LET US WRITE THE TRUTH TABLE FOR A PRIORITY ENCODER. LINE A HAS A HIGHER PRIORITY THAN B

A	B	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	0	x	x	x	x	x	x	x	x	← DON'T CARE
0	1	0	0	0	1	0	0	0	0	~ $1Φ
1	0	0	1	0	0	0	0	0	0	~ $4Φ
1	1	0	1	0	0	0	0	0	0	~ $4Φ

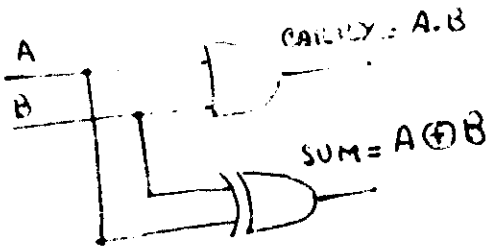
SERVICE ROUTINE FOR EACH OUTPUT LINE B AT $10
" " " " " A AT $40

(THE FULL ADDER ALSO ADDS TWO BINARY DIGITS AT A TIME BUT TAKING)

TRUTH TABLE

A	B	SUM	CARRY	
0	0	0	0	
0	1	1	0	
1	0	1	0	
1	1	0	1	← AND

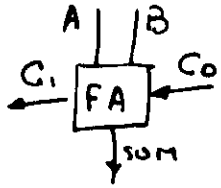
← EXCLUSIVE OR or XOR



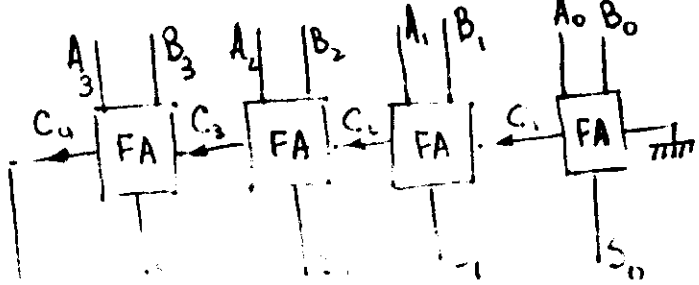
THE FULL ADDER: THE FULL ADDER ALSO ADDS TWO BINARY DIGITS AT A TIME BUT TAKING INTO ACCOUNT A CARRY AS AN INPUT.

A	B	C_0	SUM	C_1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

SYNTHESICE THE CIRCUIT AS A PROBLEM.



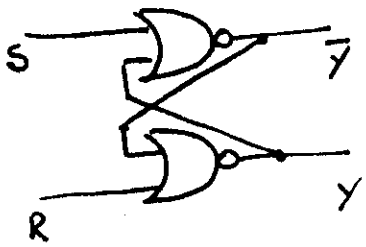
YOU CAN PUT TOGETHER A 4-BIT ADDER.



FLIP-FLOP & BI-STABLE LATCHES.

FLIP-FLOP IS ANOTHER NAME FOR A BISTABLE MULTIVIBRATOR, WHOSE OUTPUT IS EITHER LOW OR HIGH. THIS OUTPUT STAYS LOW OR HIGH; TO CHANGE IT, THE CIRCUIT MUST BE DRIVEN BY AN INPUT CALLED A TRIGGER. UNTIL THE TRIGGER ARRIVES IT REMAINS LOW OR HIGH.

THE RS FLIP FLOP :

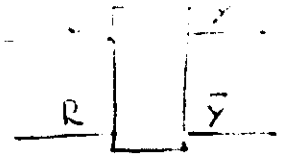


RS FLIP FLOP

R	S	Y
0	0	LAST VALUE
0	1	1
1	0	0
1	1	FORBIDDEN

THE OUTPUT ON EACH NOR GATE DRIVES ONE OF THE INPUTS ON THE OTHER NOR GATE. ALSO THE S AND R INPUTS ALLOW US TO SET OR RESET THE Y OUTPUT. IF R=S=0 THE OUTPUT REMAINS LATCHED OR STUCK IN ITS LAST STATE. THE CONTRADICTORY* CONDITION OF R AND S BOTH HIGH AT THE SAME TIME IS FORBIDDEN.

*AFTER YOU LEAVE THE R=S=1 CONDITION YOU CAN NOT PREDICT THE FINAL STATE OF Y.



SYMBOL WE WILL USE FOR A R-S FLIP-FLOP.

1 R=S=0 MEANS THE Y OUTPUT REMAINS IN IT LAST STATE INDEFINITELY BECAUSE OF THE INTERNAL LATCHING ACTION.

2 A HIGH S INPUT SETS THE Y OUTPUT TO A 1, UNLESS THIS OUTPUT IS ALREADY IN THE HIGH STATE. HERE THE OUTPUT REMAINS EVEN THOUGH THE S INPUT RETURNS TO THE LOW STATE.

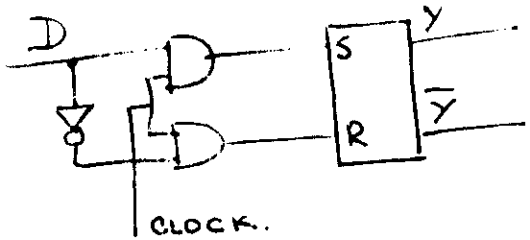
3 A HIGH R INPUT RESETS THE Y OUTPUT TO A 0 UNLESS THE OUTPUT IS ALREADY LOW. THEY OUTPUT THEN REMAINS IN THE LOW STATE EVEN THOUGH THE R INPUT GOES BACK TO THE LOW STATE.

4 HAVING BOTH R AND S HIGH AT THE SAME TIME IS FORBIDDEN BECAUSE IT IS THE STATE AT WHICH Y WILL BE LEFT AFTER R AND S ARE RETURNED TO LOW. (A RACE PROBLEM).

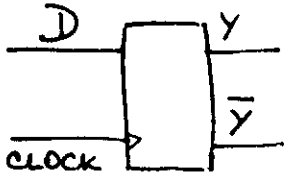
THE D-FLIP FLOP : GENERATING TWO SIGNALS

TO DRIVE A F/F (THE R AND S SIGNALS) IS A DISADVANTAGE IN MANY APPLICATIONS, ALL THIS COMPOUNDED BY THE FACT THAT A FORBIDDEN CONDITION EXISTS. THIS CONDITION MAY OCCUR

IMPLEMENTATION OF THE 4-BIT LATCH USING THE D-TYPE F/F (74174 OR 7475 YOU WILL USE QUITE OFTEN).



D-FLIP FLOP BUILT USING AN R-S F/F



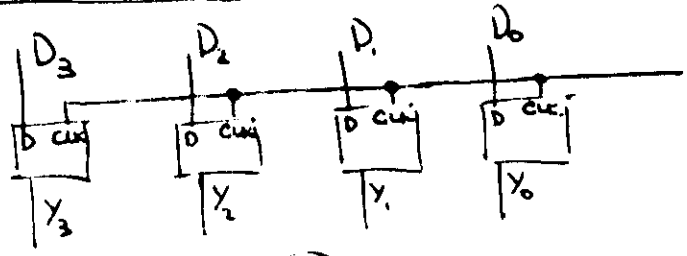
USUAL LOGIC SYMBOL

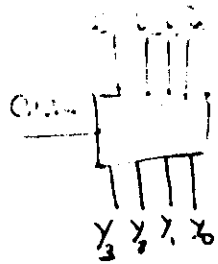
CLK	D	Y
0	X	LAST STATE
1	1	1
1	0	0

X = DON'T CARE

REMEMBER IT STORES THE VALUE OF D EXISTING WHEN CLK GOES LOW. ☺

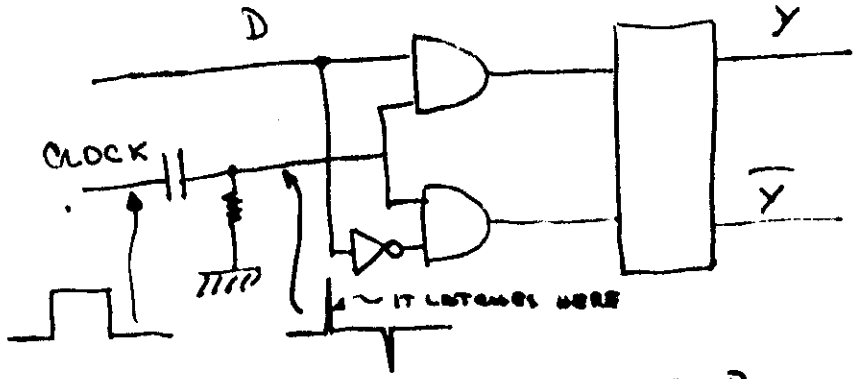
STORING 4-BITS:





QUAD D-TYPE BISTABLE LATCH.

YOU CAN ALSO PUT TOGETHER A POSITIVE EDGE TRIGGERED FLIP FLOP.



POSITIVE EDGE TRIGGERED FLIP-FLOP

EXERCISE: FIND THE TRUTH TABLE FOR THE RS FLIPFLOP BUILT USING NAND GATES