Digital Design

“TTL - CMOS ”

Dr. Cahit Karakuş, February-2018

Digital integrated circuits

Logic families of digital integrated circuits

- Many different logic families of digital integrated circuits have been introduced commercially. The following are the most popular:
  TTL transistor–transistor logic;
  ECL emitter-coupled logic;
  MOS metal-oxide semiconductor;
  CMOS complementary metal-oxide semiconductor.
- TTL, 50 yıldır kullanımda olan ve standart olarak kabul edilen bir lojik ailesidir.
- ECL, yüksek hızlı çalışma gerektiren sistemlerde bir avantaja sahiptir.
- MOS, yüksek bileşenli yoğunluğa ihtiyaç duyan devreler için uygundur ve CMOS, dijital kameralar, kişisel medya oynatıcılar ve diğer taşınabilir taşınabilir aygıtlar gibi düşük güç tüketimi gerektiren sistemlerde tercih edilir.
- Düşük güç tüketimi VLSI tasarımı için gereklidir; Bu nedenle, TTL ve ECL kullanımda düşmeye devam ederken CMOS, baskın mantık ailesi haline geldi

Logic Chips

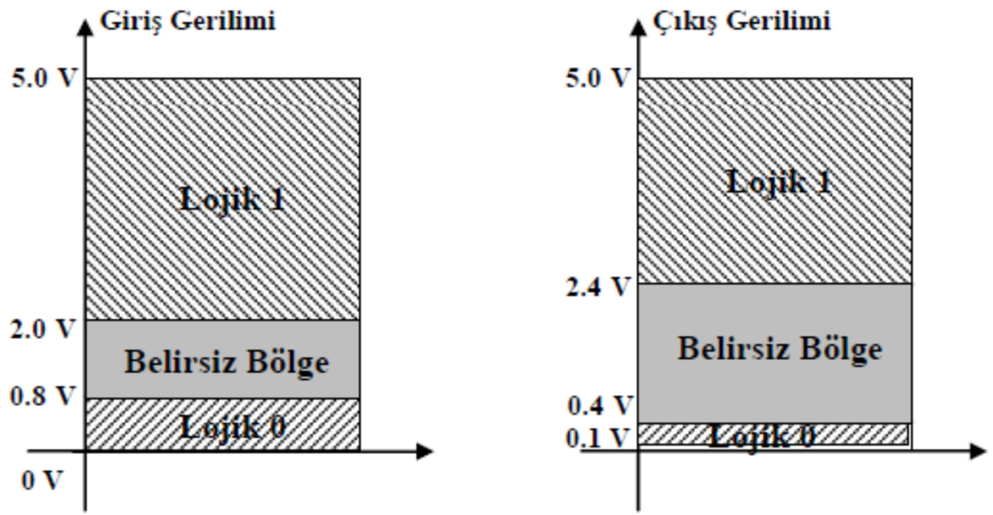
- Integration levels
  - SSI (small scale integration)
    - Introduced in late 1960s
    - 1-10 gates (previous examples)
  - MSI (medium scale integration)
    - Introduced in late 1960s
    - 10-100 gates
  - LSI (large scale integration)
    - Introduced in early 1970s
    - 100-10,000 gates
  - VLSI (very large scale integration)
    - Introduced in late 1970s
    - More than 10,000 gates

Üretim Teknolojisine Göre Tümdevrelerin Lojik Kodları

Üretim Teknolojisi	**Kodu**
Standart TTL	74
Yüksek Hızlı TTL	74H
Düşük Güç Tüketen TTL	74L
Shottky TTL	74S
Düşük Güç Tüketen Shottky TTL	74LS
İleri Shottky TTL	74AS
Düşük Güç Tüketen İleri Shottky TTL	74ALS
CMOS	40
TTL ile Bağlantı Uyumlu CMOS	74C
Yüksek Hızlı ve TTL Bağlantı Uyumlu CMOS	74HC
Yüksek Hızlı ve TTL Elektriksel Uyumlu CMOS	74HCT

TTL

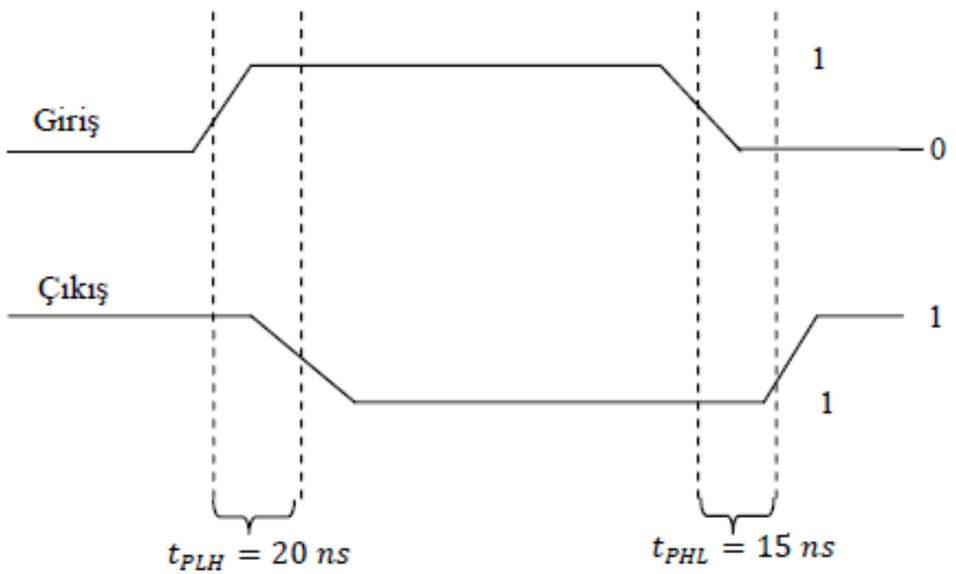
- TTL tümdevrelerin besleme gerilimi +5 V dur. Bu aileye ait bütün elemanların lojik giriş seviyeleri lojik çıkış seviyeleri ile aynıdır. Örneğin TTL teknolojisi ile üretilmiş 7404 entegresinde giriş gerilimi0 V−0.8 V arasındaysa Lojik 0; 2.0 V−5.0 V arasında ise Lojik 1seviyesindedir.



TTL tümdevrelerin giriş ve çıkış gerilimlerine ilişkin özellikler

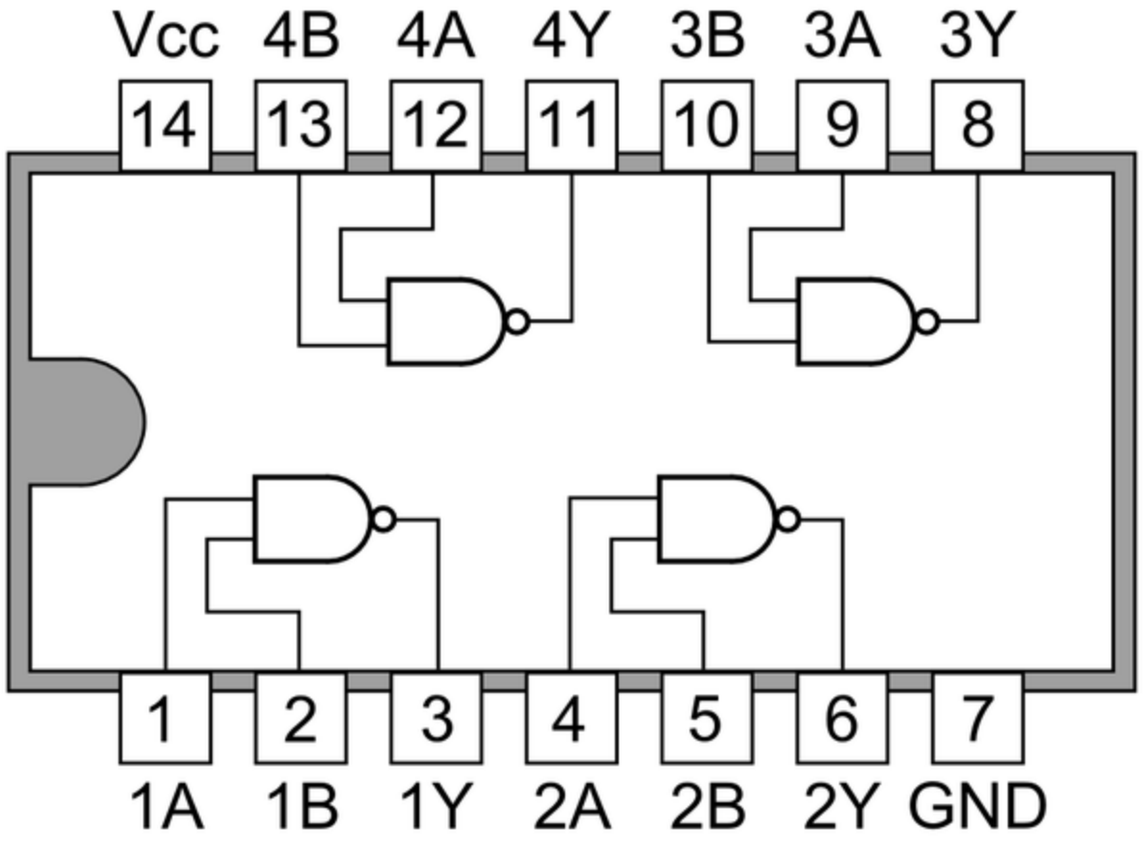
Propagasyon gecikmesi

- Bir elemanın girişinde oluşan bir işaret değişiminin çıkışta görülmesi için geçen süreye “propagasyon gecikmesi” denir. Bir NOT lojik kapısının girişine uygulanan 0−1−0 geçişli bir işarete karşılık çıkıĢında oluşan 1−0−1 Ģeklindeki işaret değişimi arasındaki gecikmeler değerlendirildiğinde şu sonuçla karşılaşılır. Lojik kapının 1-0 geçiĢine (15 ns‟ lik propagasyon gecikmesi) oranla 0-1 geçişinde (20 ns‟ lik propagasyon gecikmesi) daha büyük bir gecikme söz konusudur. TTL elemanların güç harcama miktarı 10 mW civarındadır. Fakat tümdevre içindeki kapıların kullanımları güç tüketimini etkiler.

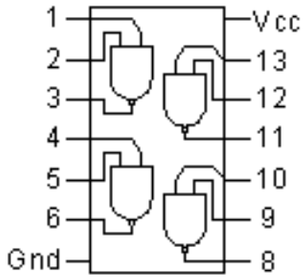


TTL tümdevrelerin propagasyon gecikmesi

7400 Quad 2-input NAND Gates

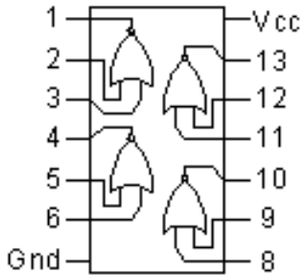


74LS00/03



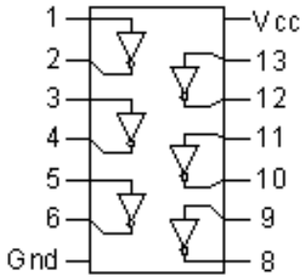
Quad 2-input NAND

74LS02



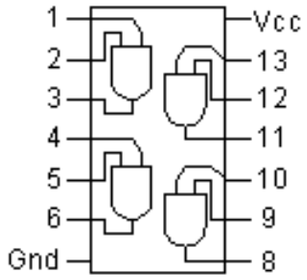
Quad 2-input NOR

74LS04/05



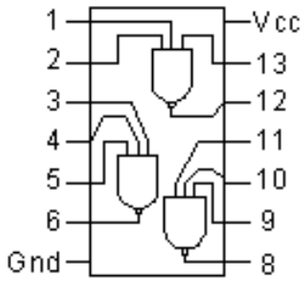
Hex INVERTER

74LS08/09



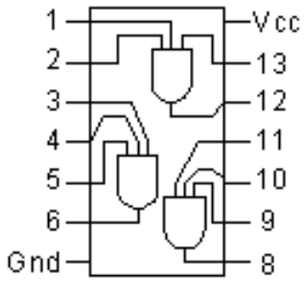
Quad 2-input AND

74LS10



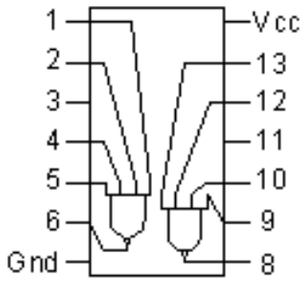
Triple 3-input NAND

74LS11



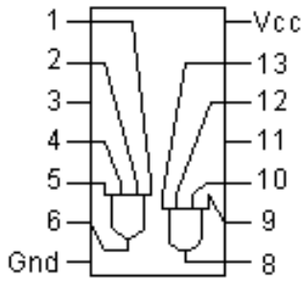
Triple 3-input AND

74LS20



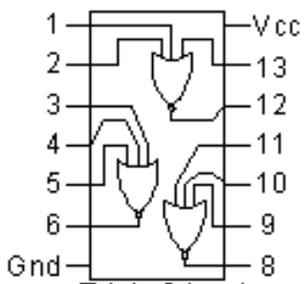
Dual 4-input NAND

74LS21



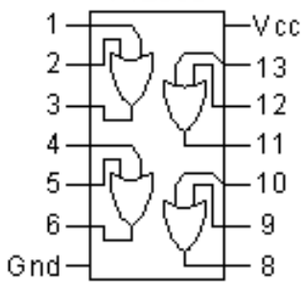
Dual 4-input AND

74LS27



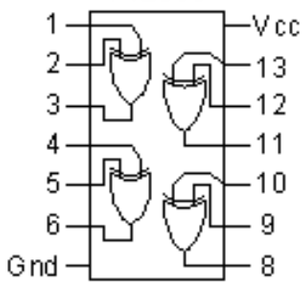
Triple 3-input NOR

74LS32



Quad 2-input OR

74LS86



Quad 2-input XOR

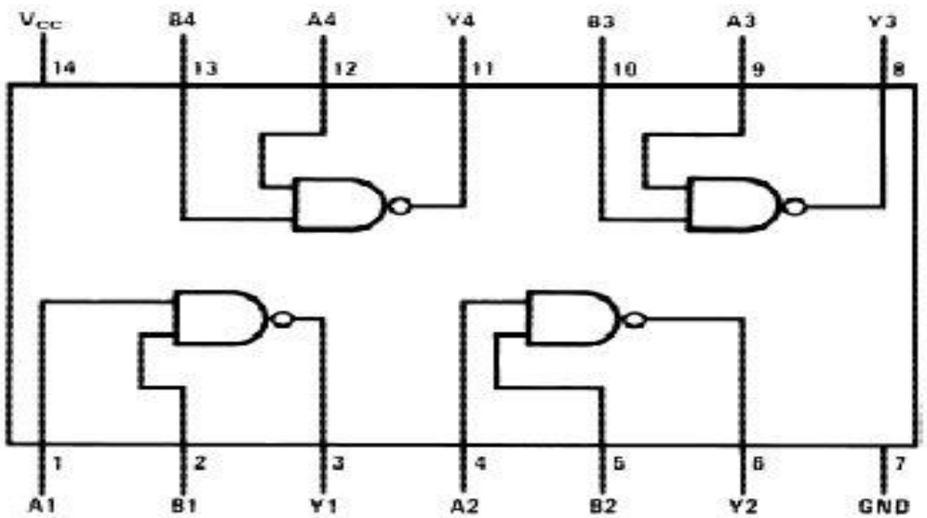
Logic Gates: The NAND Gate

- The NAND Gate

- NAND gate is self-sufficient (can build any logic circuit with it).
- Can be used to implement AND/OR/NOT.
- Implementing an inverter using NAND gate:

A	B	(A.B)'
0	0	1
0	1	1
1	0	1
1	1	0

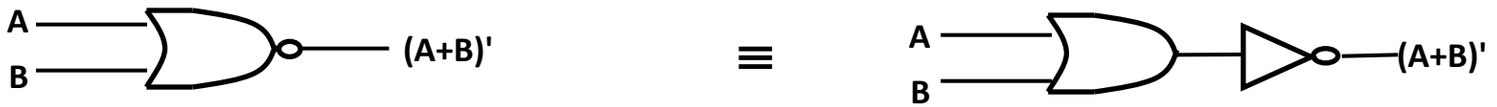
Truth table



Top View of a TTL 74LS family 74LS00 Quad 2-input NAND Gate IC Package

Logic Gates: The NOR Gate

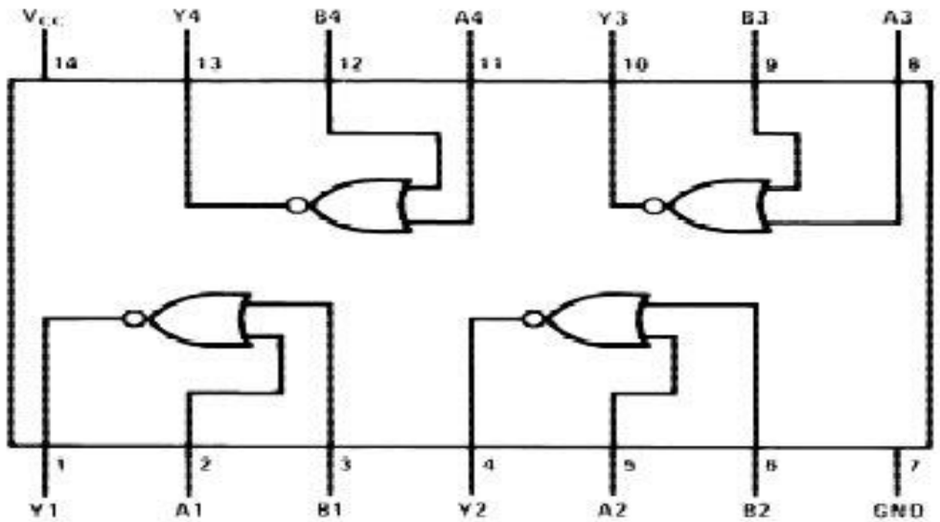
- The NOR Gate



- NOR gate is also self-sufficient (can build any logic circuit with it).
- Can be used to implement AND/OR/NOT.
- Implementing an inverter using NOR gate:

A	B	(A+B)'
0	0	1
0	1	0
1	0	0
1	1	0

Truth table



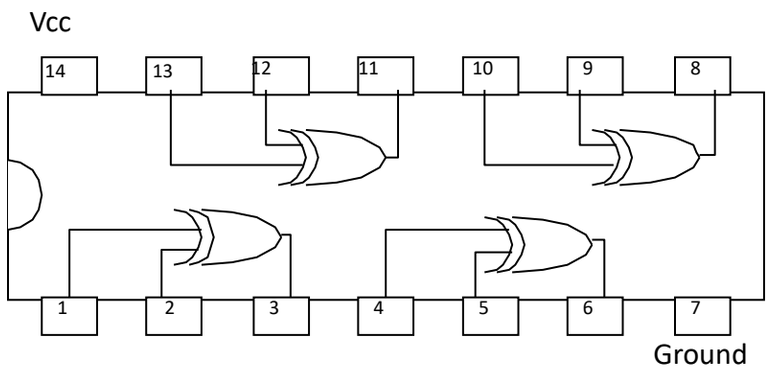
Top View of a TTL 74LS family 74LS02 Quad 2-input NOR Gate IC Package

Logic Gates: The XOR Gate

- The XOR Gate

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

Truth table

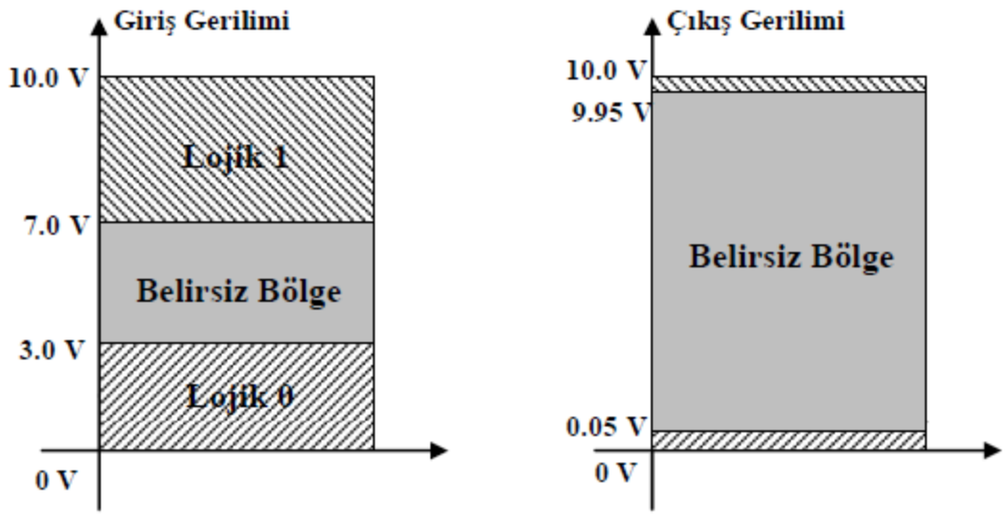


Top View of a TTL 74LS family 74LS86 Quad 2-input XOR Gate IC Package

CMOS Logic Gates

CMOS tümdevre özellikleri

- CMOS tümdevrelerin besleme gerilimi +10 V dur.
- CMOS teknolojisi ile üretilmiş entegreslerde giriş gerilimi 0 V−3.0 V arasındaysa Lojik 0, 7.0 V−10.0 V arasında ise Lojik 1seviyesindedir. Ayrıca gürültü filtreleme konusunda CMOS entegrelerde TTL'lerden daha iyidir.
- Standart CMOS tümdevrelerde propagasyon gecikmesi 25−100 ns arasında değiĢir. Fakat, yeni nesil yüksek hızlı CMOS devrelerde (HC serisi) bu gecikme 8 ns mertebesine düĢmektedir. CMOS elemanların güç harcama miktarı 0.01−1 mW mertebesindedir.



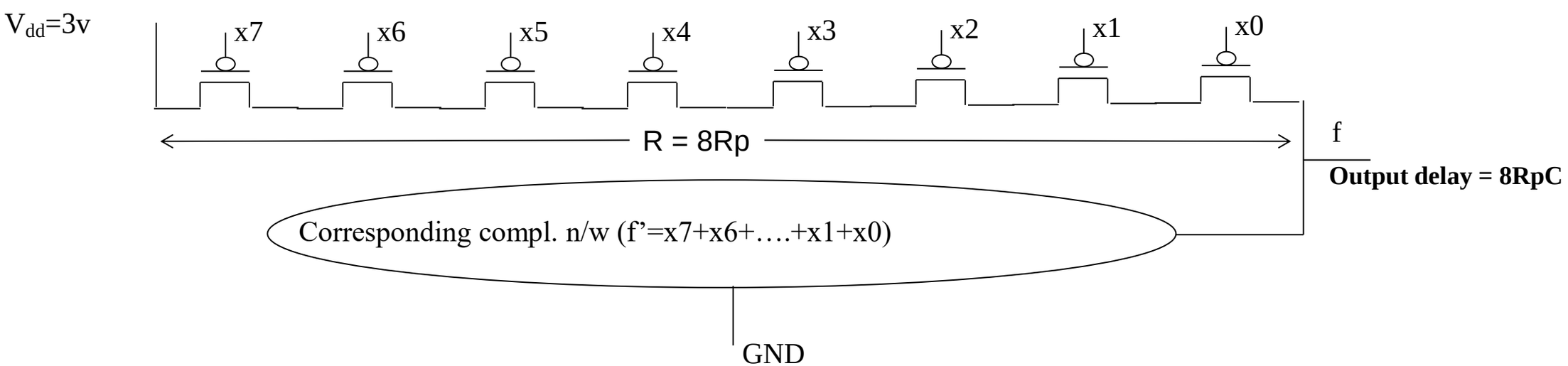
CMOS tümdevrelerin giriş ve çıkış gerilimlerine ilişkin özellikler

CMOS LOGIC GATES

- Thus an nMOS transistor passes a ***strong*** 0 and a ***weak*** 1.
- A similar analysis (for pMOS, gate to source voltage has to be < the (negative) threhold voltage V_T for transistor to conduct) shows that a pMOS transistor passes a ***strong*** 1 and a ***weak*** 0.
- This is the basis of CMOS logic gates, where pMOS transistors are used in the “top” n/w connected to Vdd to conduct a ***strong*** or ***good*** 1, and nMOS transistors are used in the “bottom” or complementary n/w to conduct a ***strong*** 0.
- Also, can Combine the two to make a CMOS pass gate, called a *transmission gate*, which will pass a ***strong*** 0 and a ***strong*** 1.

Problem w/ Large Switching Networks

• Even though pMOS conducts a good 1, a long series of pMOS transistors for a many-input gate can lead to excessive resistance R and thus a large output delay RC, where C is the load capacitance driven by the gate.
•Example: Consider an 8-variable NOR function f = (x7+x6+x5+x4+x3+x2+x1+x0)'. Its implementation using a single n/w is given below; we assume that a pMOS transistor has an on-resistance of Rp. Note that f = x7'x6' ….. x1'x0'

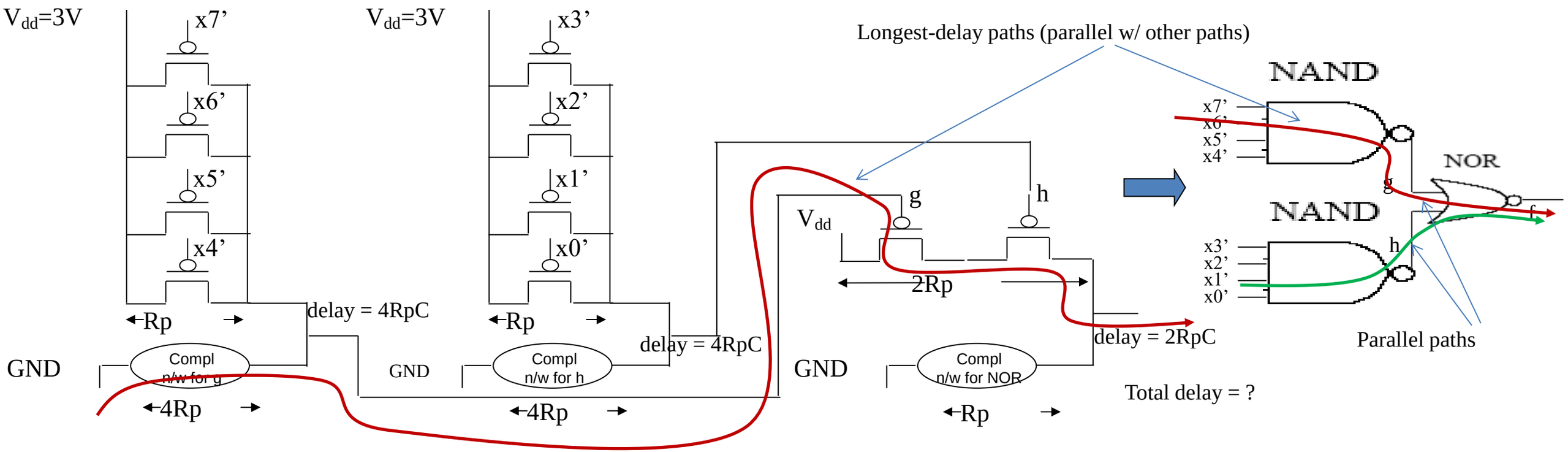


Problem with Large Switching Networks (contd)

- The solution for avoiding such excessive delay is using a number of smaller switching n/ws over “parallel” paths [otherwise, if all the smaller n/ws are on one sequential path, there will be no or little delay improvement].
- Thus we need to break down a large function (function w/ many variables—generally > 6) into smaller ones that can each be implemented using smaller n/ws. This happens to a large extent when a function is represented as an SOP or POS expression (it is lready broken down into ANDs and ORs) but not always (e.g., an AND or OR term may have a large # of vars).
- E.g., the 8-i/p NOR function f can be decomposed as (and then impl as below):
  - f = [(x7+x6+x5+x4) + (x3+x2+x1+x0)]’ = [(x7’x6’x5’x4’)’ + (x3’x2’x1’x0’)’]’ = NOR(NAND(x7’,x6’,x5’,x4’), NAND(x3’,x2’,x1’,x0’)).
  - Alternatively, f = (x7+..+ x4)’ (x3+..+x0)’ = AND(NOR(x7,..,x4), NOR(x3,..x0)) = NOT(NAND(NOR(x7,..,x4), NOR(x3,..x0)))

Problem with Large Switching Networks (contd)

- The 8-i/p NOR function f can be decomposed as (and then impl as below):
  - f = [(x7+x6+x5+x4) + (x3+x2+x1+x0)]' = [(x7'x6'x5'x4')' + (x3'x2'x1'x0')']' = NOR(NAND(x7',x6',x5',x4'), NAND(x3',x2',x1',x0')).

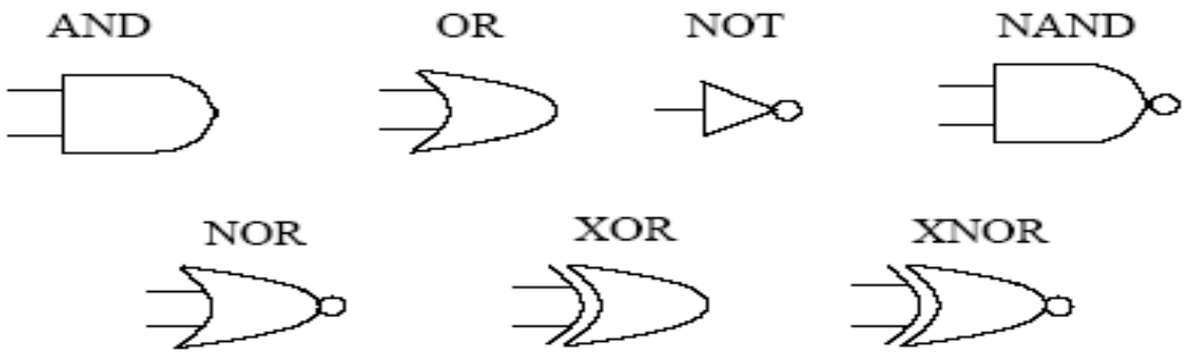


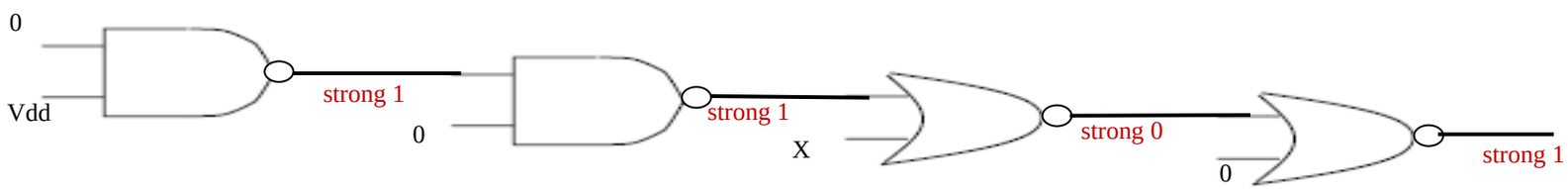
Note: Delay of a circuit = delay of its longest-delay path from input [i/p] to output [o/p]

Problem with Large Switching Networks (contd)

- These small switching networks are called *gates*
- Thus need to use small to medium-size (<= 4 inputs) gates to implement large logic functions

- Examples of typical gates are AND, OR, NOT, NAND, NOR, EXOR and XNOR





A cascade or series of NAND/NOR gates will produce strong 1's as well as strong 0's as well as smaller delay than a large switching n/w for the corresponding logic expression.

Circuit Delay—Definition & Computation

- Assume R is the on-resistance of a single nMOS or pMOS transistor, and C its i/p or gate capacitance.
- Then the worst-case "top" network resistance R_{top} of a gate gi is the k*R, where k = max. # of transistors in series in the top n/w of gi. Similarly, for the resistance R_{bot} of the "bottom" or complementary n/w of gi.
- If C_L is the capacitive load seen by a gate gi (generally = the sum of gate capacitances C of the transistors of the gate(s) that gi drives), then the delay in gi driving its output from 0 → 1 is R_{top}* C_L and the delay in gi driving its output from 1 → 0 is R_{bot}* C_L . In general, we define gate res. R_g = max(R_{top} , R_{bot}), and the delay of its output signal as R_g* C_L
- Example: For the 2 i/p NAND gate in Fig. 1, R_{top} = R (note that in the worst-case only 1 pMOS transistor is on, so the res. then is R, and *not* R/2), R_{bot} = 2R. Thus R_g = 2R, and the gate's output delay = R_g*C_L = 2R*C_L . If the gate is driving a 2-input NAND/NOR/AND/OR gate, then = C_L= 2C.
- The *delay of a path* = Σ (output delays of gates on the path). The delay of the path shown in Fig. 2 = [d(g1) + R_g(g1)*C_L(g2)] + [d(g2) + R_g(g2)*C_L(g3)] + [d(g3) + R_g(g3)*C_L(g4)] + [d(g4) + R_g(g4)*C_L(op)], where C_L(op) is the load at the output of the path and d(gi) is the "intrinsic" delay of a gate gi to switch from off to on.

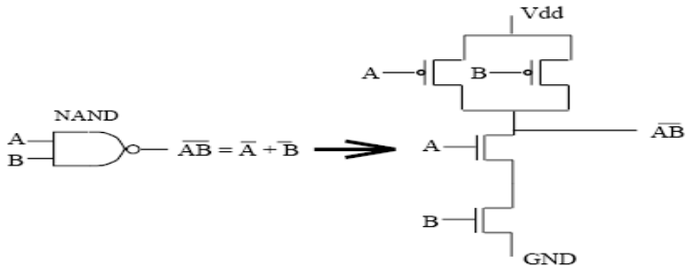


Fig. 1: CMOS realization of a 2-i/p NAND gate

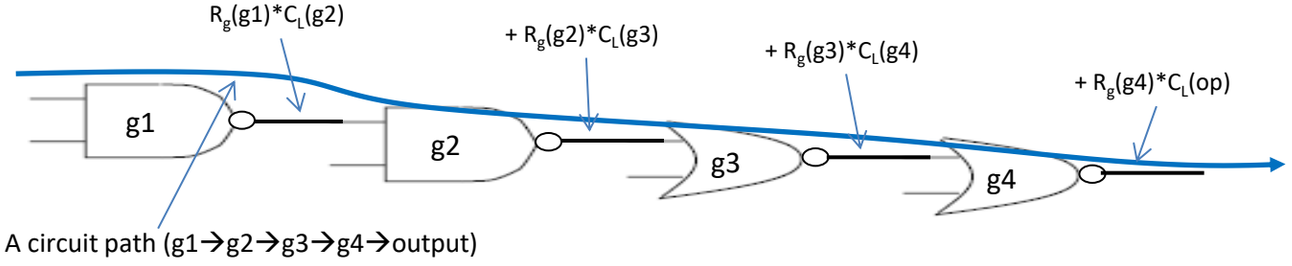


Fig. 2: A path of a circuit and its delay

Circuit Delay (cont'd)

- The *delay of a path* = Σ (output delays of gates on the path). The delay of the path shown in Fig. 2 = [d(g1) + $R_g(g1)*C_L(g2)$] + [d(g2) + $R_g(g2)*C_L(g3)$] + [d(g3) + $R_g(g3)*C_L(g4)$] + [d(g4) + $R_g(g4)*C_L(op)$], where $C_L(op)$ is the load at the output of the path and d(gi) is the "intrinsic" delay of a gate gi to switch from off to on.
- Thus, assumimg that the d(gi) for all 2-i/p gates is the same and = d(g), the path delay = 4*d(g) + 2R*2C + 2R*2C + 2R*2C + 2R* $C_L(op)$ = 4*d(g) + 12RC + 2R* $C_L(op)$ = 4*d(g) + 16RC if $C_L(op)$ = 2C.
- If we ignore the d(gi)'s (which are typically small compared to the RC delays), the rest of the delay is the RC delay, which for this ex. = 16RC = 4*(2R*2C)
- The 2C part of the delay expression will remain unchanged (for nand/nor/and/or gates) irrespective if the gate sizes # of i/ps). However the 2R part in each term will change to kR where k = # of i/ps (for nand/nor/and/or gates)
- If the gates in Fig. 2 were all 3-i/p gates, the RC delay expression will be 4*(3R*2C) = 24RC = (3/2)*(16RC) (as the # of i/ps change from 2 to 3, delay increases proportionately by a factor of 3/2).
- Thus the delay is proportional to the sum of the # of each inputs along a path (8 for the path w/ 2-i/p gates and 12 if the gates are 3 i/ps).
- **Thus a simple delay model we will use is that the delay of a gate w/ k i/ps = k, and add up this simplified gate delay units along a path to get the path's delay.**
- The *delay of a circuit* is the delay in the longest (max-delay) path of the circuit from primary inputs to an output

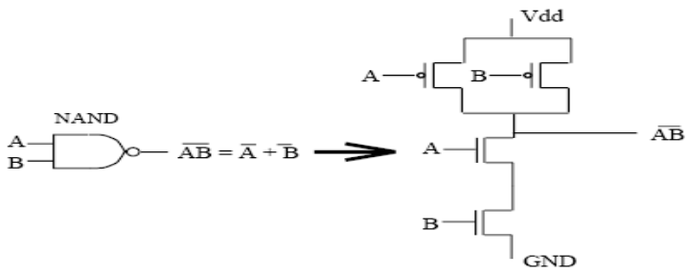


Fig. 1: CMOS realization of a 2-i/p NAND gate

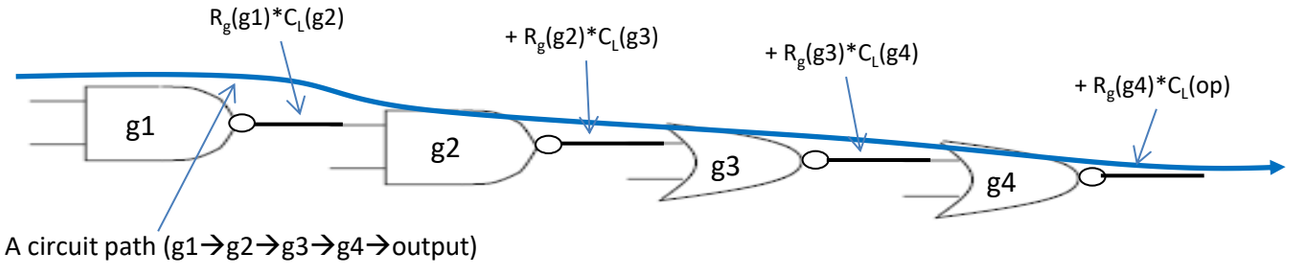
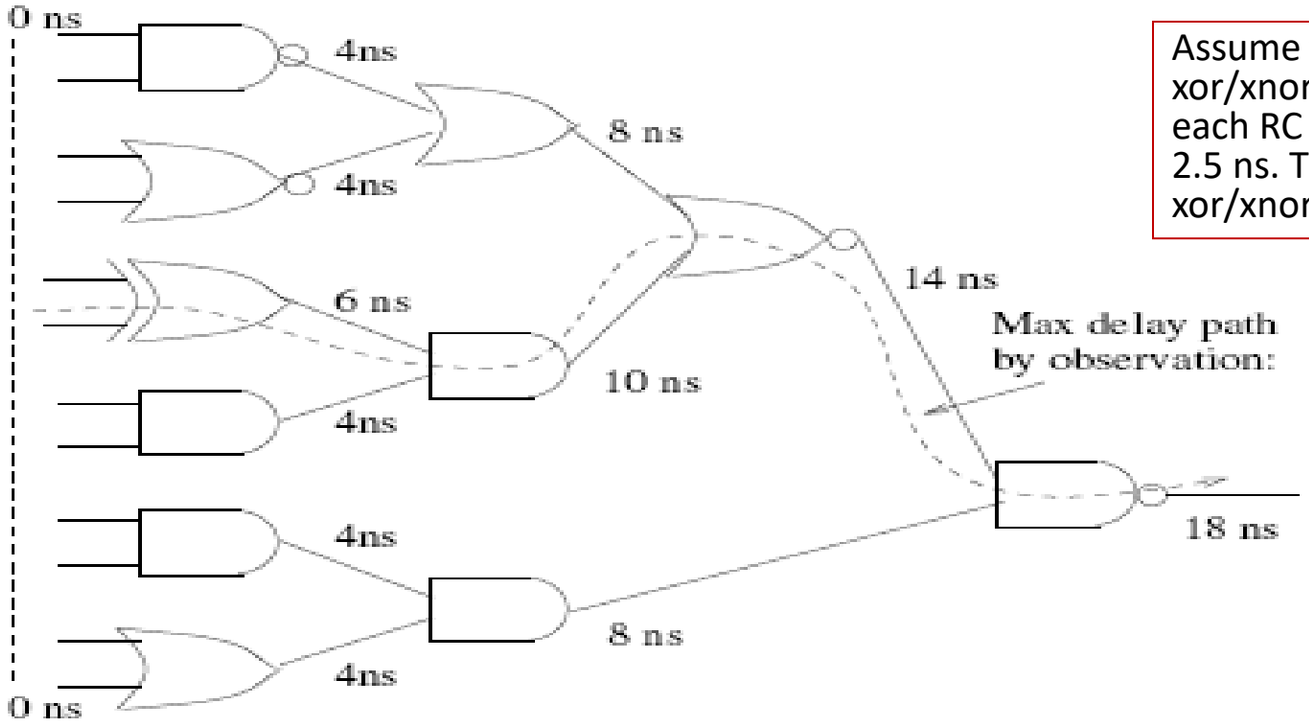


Fig. 2: A path of a circuit and its delay

Determining Circuit Delay



Assume that the intrinsic delay d(gi) of each gate except xor/xnor = 1.5 ns, that of xor/xnor gates is 3.5 ns, and each RC delay between a driving gate and driven i/p is 2.5 ns. Thus i/p -> o/p sink delay for each gate except xor/xnor = 4 ns, while that for xor/xnor is 6 ns

Delays: 2-i/p gates, AND, OR, NAND, NOR: 4ns, XOR, XNOR: 6ns

Determining the max. delay of a circuuit:
(1) Path tracing by observation -- prone to human error in large circuits

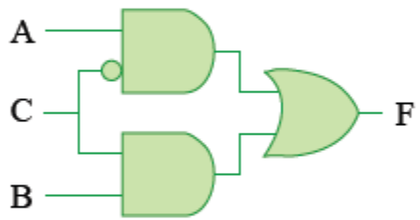
(2) Recursive formulation: Delay at o/p of gate g_i = max{delay of all i/ps to g_i}
+ delay of gate gi(intrinsic gate delay) + RC delay at gi's o/p

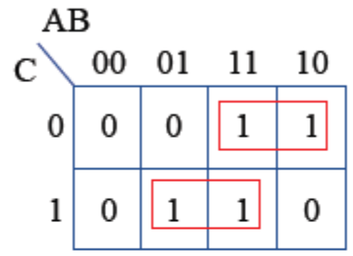
Delay of a circuit = max{delay at all o/ps}

Hazards

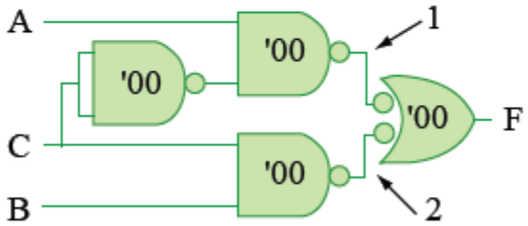
Static hazards: Consider this function:

$F = A * \overline{C} + B * C$

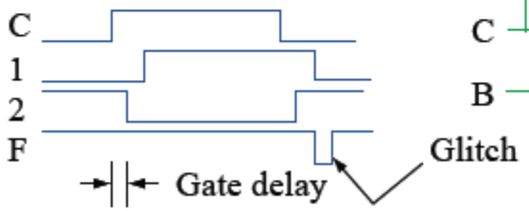




Implemented with MSI gates:

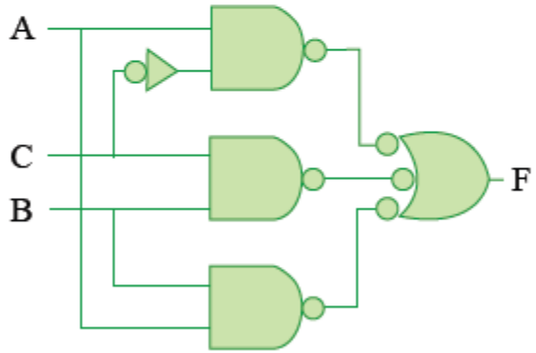


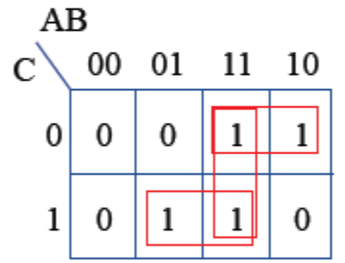
A = B = 1



Fixing Hazards

The glitch is the result of timing differences in parallel data paths. It is associated with the function jumping between groupings or product terms on the K-map. To fix it, cover it up with another grouping or product term!





$F = A * \overline{C} + B * C + A * B$

- **In general, it is difficult to avoid hazards – need a robust design methodology to deal with hazards.**

Kaynakça

- https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-111-introductory-digital-systems-laboratory-spring-2006/lecture-notes/
- http://web.ee.nchu.edu.tw/~cpfan/FY92b-digital/Chapter-4.ppt
- http://www.cs.nccu.edu.tw/~whliao/ds2003/ds4.ppt
- http://www.just.edu.jo/~tawalbeh/cpe252/slides/CH1_2.ppt
- Lessons In Electric Circuits, Volume IV { Digital By Tony R. Kuphaldt Fourth Edition, last update July 30, 2004.
- Digital Electronics Part I – Combinational and Sequential Logic Dr. I. J. Wassell.
- Digital Design With an Introduction to the Verilog HDL, M. Morris Mano Emeritus Professor of Computer Engineering California State University, Los Angeles; Michael D. Ciletti Emeritus Professor of Electrical and Computer Engineering University of Colorado at Colorado Springs.
- Digital Logic Design Basics, Combinational Circuits, Sequential Circuits, Pu-Jen Cheng.