# Switching Circuits \＆ Logic Design 

Jie－Hong Roland Jiang
江介宏
Department of Electrical Engineering National Taiwan University


Fall 2014

## §7 Multi－Level Gate Circuits



## Outline

$\square$ Multi-level gate circuits
$\square$ NAND and NOR gates
$\square$ Design of two-level circuits using NAND and NOR gates
$\square$ Design of multi-level NAND- and NOR-gate circuits
$\square$ Circuit conversion using alternative gate symbols
$\square$ Design of two-level, multiple-output circuits
$\square$ Multiple-output NAND and NOR circuits

## Multi-Level Gate Circuits

The number of levels of gates- The maximum number of gates cascaded in series between a circuit input and output
$\square$ I nverters which are connected directly to input variables do not count (assume variables and their complements are available as circuit inputs)

SOP (POS) correspond to AND-OR (OR-AND) two-level gate circuits
$\square$ AND-OR

- 2-level circuit composed of a level of AND gates followed by an OR gate at the outputOR-AND
- 2-level circuit composed of a level of OR gates followed by an AND gate
at the output
$\square$ OR-AND-OR
- 3-level circuit composed of a level of OR gates followed by a level of AND gates followed by an OR gate at the output
$\square$ Circuit of AND and OR gates
- No particular order of the gates


## Multi-Level Gate Circuits



## Multi-Level Gate Circuits


$\square$ OR-AND-OR 3-level realization of Z

- Partially multiplying out $Z=(A B+C)[(D+E)+F G]+H$
- 3 levels, 6 gates, 19 gate inputs


## Multi-Level Gate Circuits

$\square$ Drawing the tree diagram of an expression helps determine the realization costs
\#level $\rightarrow$ circuit delay
■ \#gates, \#gate inputs $\rightarrow$ circuit area
$\square$ Different expressions of a Boolean function provide different tradeoffs between delay and area

## Multi-Level Gate Circuits

$\square$ Find a circuit of AND and OR gates realizing $\mathrm{f}(\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d})=$ $\sum \mathrm{m}(1,5,6,10,13,14)$
$f=a ' c ' d+b c^{\prime} d+b c d^{\prime}+a c d '$

| ${ }_{c d} a b$ |  | 01 | 11 | 10 |
| :---: | :---: | :---: | :---: | :---: |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 1 | 1 | 1 | 0 |
| 11 | 0 | 0 | 0 | 0 |
| 10 | 0 | 1 | 1 | 1 |



2 levels, 5 gates, 16 gate inputs

## Multi-Level Gate Circuits

Example (cont'd)

$$
\begin{aligned}
f & =a^{\prime} c^{\prime} d+b c^{\prime} d+b c d^{\prime}+a c d^{\prime} \\
& =c^{\prime} d\left(a^{\prime}+b\right)+c d^{\prime}(a+b)
\end{aligned}
$$




3 levels, 5 gates, 12 gate inputs

## Multi-Level Gate Circuits

Example (cont'd)

$$
\begin{aligned}
& f^{\prime}=c^{\prime} d^{\prime}+a b b^{\prime}+c d+a^{\prime} b^{\prime} c \\
& f=(c+d)\left(a^{\prime}+b+c\right)\left(c^{\prime}+d^{\prime}\right)\left(a+b+c^{\prime}\right)
\end{aligned}
$$



2 levels, 5 gates, 14 gate inputs

## Multi-Level Gate Circuits

Example (cont'd)

$$
\begin{aligned}
f & =(c+d)\left(a^{\prime}+b+c\right)\left(c^{\prime}+d^{\prime}\right)\left(a+b+c^{\prime}\right) \\
& =\left(c+a^{\prime} d+b d\right)\left(c^{\prime}+a d^{\prime}+b d^{\prime}\right)
\end{aligned}
$$




3 levels, 7 gates, 16 gate inputs

## Multi-Level Gate Circuits

$\square$ To be sure of obtaining a minimum solution, we have to find both the circuit with the AND-gate output and the one with the OR-gate output
$\square$ If the expression for $f$ ' has $n$ levels with an ANDgate (OR-gate) output, its complement is an nlevel expression for $f$ with an OR-gate (AND-gate) output

## NAND and NOR Gates

$\square$ NAND
AND-NOT gate

- 3-input NAND: $F=(A B C)^{\prime}=A^{\prime}+B^{\prime}+C^{\prime}$

■ n-input NAND: $F=\left(X_{1} X_{2} \ldots X_{n}\right)^{\prime}=X_{1}{ }^{\prime}+X_{2}{ }^{\prime}+\ldots+X_{n}{ }^{\prime}$
A



$\square$ NOR

- OR-NOT gate
- 3-input NOR: $F=(A+B+C)^{\prime}=A^{\prime} B^{\prime} C^{\prime}$
- n-input NOR: $F=\left(X_{1}+X_{2}+\ldots+X_{n}\right)^{\prime}=X_{1}{ }^{\prime} X_{2}{ }^{\prime} \ldots X_{n}{ }^{\prime}$





## NAND and NOR Gates

$\square$ A set of logic operations is functionally complete if any Boolean function can be expressed in terms of the set of operations

- \{AND, OR, NOT\} is functionally complete
$\square$ Any Boolean function can be expressed in SOP form, which uses only the AND, OR, NOT operations
- Any set of logic operations that can realize AND, OR, NOT is also functionally complete
$\square\{$ AND, NOT $\}$ is functionally complete since OR can be realized using AND and NOT as shown below



## NAND and NOR Gates

$\square\{N A N D\}$ is functionally complete
NOT: (X•X)' = X'


■ AND: ((A•B)')' = A•B

■OR: $\left(A^{\prime} \cdot B^{\prime}\right)^{\prime}=A+B$


## NAND and NOR Gates

- How to show whether or not a set of logic operations is functionally complete?

1. Write out a minimum SOP expression for the function realized by each gate
2. If no complement appears in any of these expressions, then NOT cannot be realized
3. Otherwise, NOT can be realized by an appropriate choice of inputs to the corresponding gate (assume 0 and 1 are available as gate inputs)
4. Try to realize AND or OR (now with NOT available)

## NAND and NOR Gates

- Exercises

Show that the sets \{NOR\} and \{OR, NOT\} are functionally complete

- Is the majority gate major functionally complete?
$\square$ major $(A, B, C)=1$ iff at least two of $A, B, C$ are 1
$\square$ Is the minority gate minor functionally complete?
$\square$ minor $(A, B, C)=1$ iff at most one of $A, B, C$ is 1
$\square$ Is $\{\rightarrow\}$ functionally complete?
$\square A \rightarrow B$ is true iff $A$ is false ( 0 ), or both $A$ and $B$ are true (1)
-Does the assumption " 0 and 1 are available as gate inputs" make a difference?


## Two-Level Circuit Design Using NAND and NOR Gates

$\square$ A 2-level circuit composed of AND, OR gates can be converted to a circuit composed of NAND, NOR gates
■ By using $F=\left(F^{\prime}\right)$ ' and DeMorgan's laws

## Example 1

$$
\begin{align*}
F & =A+B C^{\prime}+B^{\prime} C D  \tag{AND-OR}\\
& =\left[\left(A+B C^{\prime}+B^{\prime} C D\right)^{\prime}\right]^{\prime} \\
& =\left[A^{\prime}\left(B C^{\prime}\right)^{\prime}\left(B^{\prime} C D\right)^{\prime}\right]^{\prime} \\
& =\left[A^{\prime}\left(B^{\prime}+C\right)\left(B+C^{\prime}+D^{\prime}\right)\right]^{\prime} \\
& =A+\left(B^{\prime}+C\right)^{\prime}+\left(B+C^{\prime}+D^{\prime}\right)^{\prime} \tag{NOR-OR}
\end{align*}
$$

## Two-Level Circuit Design Using NAND and NOR Gates



## Two-Level Circuit Design Using NAND and NOR Gates

Example 2

| $F$ | $=(A+B+C)\left(A+B^{\prime}+C^{\prime}\right)\left(A+C^{\prime}+D\right)$ |  | (OR-AND) |
| ---: | :--- | ---: | :--- |
|  | $=\left\{\left[(A+B+C)\left(A+B^{\prime}+C^{\prime}\right)\left(A+C^{\prime}+D\right)\right]^{\prime}\right\}^{\prime}$ |  |  |
|  | $=\left[(A+B+C)^{\prime}+\left(A+B^{\prime}+C^{\prime}\right)^{\prime}+\left(A+C^{\prime}+D\right)^{\prime}\right]^{\prime}$ |  | $($ NOR-NOR) |
|  | $=\left[\left(A^{\prime} B^{\prime} C^{\prime}\right)+\left(A^{\prime} B C\right)+\left(A^{\prime} C D^{\prime}\right)\right]^{\prime}$ |  | (AND-NOR) |
|  | $=\left(A^{\prime} B^{\prime} C^{\prime}\right)^{\prime}\left(A^{\prime} B C\right)^{\prime}\left(A^{\prime} C D^{\prime}\right)^{\prime}$ |  | (NAND-AND) |

## Two-Level Circuit Design Using NAND and NOR Gates

$F=\left(A^{\prime} B^{\prime} C^{\prime}\right)^{\prime}$.
( $\left.A^{\prime} B C\right)^{\prime} \cdot\left(A^{\prime} C D^{\prime}\right)^{\prime}$


## Two-Level Circuit Design Using NAND and NOR Gates

Among the 16 two-level forms:

- The following 8 are generic (can realize all switching functions): AND-OR, AND-NOR, OR-AND, OR-NAND, NAND-AND, NANDNAND, NOR-OR, NOR-NOR
- The following 8 are degenerate (cannot realize all functions): AND-AND, AND-NAND, OR-OR, OR-NOR, NAND-OR, NANDNOR, NOR-AND, NOR-NAND
E.g.,

NAND-NOR form can realize only a product of literals and not a sum of products


## Two-Level Circuit Design Using NAND and NOR Gates

$\square \quad$ NAND-NAND and NOR-NOR are the most widely used forms in integrated circuits
$\square \quad$ Procedure for minimum NAND-NAND (NOR-NOR) implementation

1. Find a minimum SOP (POS) expression of $F$
2. Draw the corresponding two-level AND-OR (OR-AND) circuit
3. Replace all gates with NAND (NOR) gates with interconnections unchanged. Complement the single-literal inputs of the output gate

$$
F=I_{1}+I_{2}+\cdots+P_{1}+P_{2}+\cdots
$$

$$
F=\left(I_{1} I_{2}{ }^{\prime} \ldots P_{1} P_{2}{ }^{\prime} \ldots\right)^{\prime}
$$



## Design of Multi-Level NAND- and NOR-Gate Circuits

$\square$ Procedure for designing multi-level NAND-gate circuits

1. Simplify F
2. Design a multi-level circuit of AND and OR gates with output gate being OR

- AND-gate (OR-gate) outputs cannot be used as AND-gate (OR-gate) inputs

3. Number the levels starting with the output gate as level 1. Replace all gates with NAND gates, leaving interconnections unchanged. Invert any literals which appear as inputs to levels $1,3,5, \ldots$

## Design of Multi-Level NAND- and NOR-Gate Circuits

## Example

$$
F_{1}=a^{\prime}\left[b^{\prime}+c\left(d+e^{\prime}\right)+f^{\prime} g^{\prime}\right]+h i ' j+k
$$

Level 5 Level 4 Level 3 Level 2 Level 1


## Circuit Conversion Using Alternative Gate Symbols

$\square$ Alternative gate symbols
■ Useful for circuit analysis and design


NOT


AND


OR


NAND


NOR

## Circuit Conversion Using Alternative Gate Symbols

$\square$ NAND gate circuit conversion


## Circuit Conversion Using Alternative Gate Symbols

- Conversion to NOR gates



## Circuit Conversion Using Alternative Gate Symbols

- A circuit composed of AND and OR gates can be converted to a circuit composed of NAND and NOR gates, and vice versa

By properly adding inverters, removing canceling inverter pairs, and/or negating inputs, we can change a gate type to a desired one

Circuit Conversion Using Alternative Gate Symbols
$\square$ Conversion to NAND gates (even if AND and OR gates do not alternate)




## Two-Level, Multiple-Output Circuit Design

- In circuit design, often we need several Boolean functions rather than one
- Although every function can be implemented separately, recognizing common gates among these functions can achieve logic sharing and thus reduce area
■ When designing multiple-output circuits, we should try to first minimize the total number of gates required and then minimize gate inputs


## Two-Level, Multiple-Output Circuit Design

## Example

$$
\begin{aligned}
& \mathrm{F}_{1}(\mathrm{~A}, \mathrm{~B}, \mathrm{C}, \mathrm{D})=\sum \mathrm{m}(11,12,13,14,15) \\
& \mathrm{F}_{2}(\mathrm{~A}, \mathrm{~B}, \mathrm{C}, \mathrm{D})=\sum \mathrm{m}(3,7,11,12,13,15) \\
& \mathrm{F}_{3}(\mathrm{~A}, \mathrm{~B}, \mathrm{C}, \mathrm{D})=\sum \mathrm{m}(3,7,12,13,14,15)
\end{aligned}
$$



## Two-Level, Multiple-Output Circuit Design

Example (cont'd)
Separate vs. multiple-output realization



7 gates, 18 gate inputs
Note that $F_{2}$ in the multipleoutput realization is not a minimum SOP

## Two-Level, Multiple-Output Circuit Design

## Example

$$
\begin{aligned}
& f_{1}(a, b, c, d)=\sum m(2,3,5,7,8,9,10,11,13,15) \\
& f_{2}(a, b, c, d)=\sum m(2,3,5,6,7,10,11,14,15) \\
& f_{3}(a, b, c, d)=\sum m(6,7,8,9,13,14,15)
\end{aligned}
$$



## Two-Level, Multiple-Output Circuit Design

Example (cont'd)

Separate realization
$f_{1}=b d+b^{\prime} c+a b^{\prime}$
$f_{2}=c+a ' b d$
$f_{3}=b c+a b^{\prime} c^{\prime}+a b d$
10 gates, 25 gate inputs

Multi-output realization
$f_{1}=a^{\prime} b d+a b d+a b '^{\prime}+b^{\prime} c$
$f_{2}=c+a ' b d$
$f_{3}=b c+a b^{\prime} c^{\prime}+a b d$
8 gates, 22 gate inputs

## Two-Level, Multiple-Output Circuit Design

## Example

$\square$ When designing multiple-output circuits, it is sometimes best not to combine a 1 with its adjacent 1's

Best solution:

|  |  | 01 | 11 | 10 |
| :---: | :---: | :---: | :---: | :---: |
| 00 |  |  |  |  |
| 01 | 1 | 1 | 1 | 1 |
| 11 |  |  | 1 |  |
| 10 |  |  |  |  |



Solution requires an extra gate:

| $d^{a b}$ | $00$ | 01 | 11 | 10 |
| :---: | :---: | :---: | :---: | :---: |
| 00 |  |  |  |  |
| 01 | 1 | 1 | 1 | 1 |
| 11 |  |  | 1 |  |
| 10 |  |  |  |  |



## Two-Level, Multiple-Output Circuit Design

## Example

$\square$ The solution with the maximum number of common terms is not necessarily best

Solution with maximum \# of common terms:
(8 gates, 26 inputs)


ab



## Two-Level, Multiple-Output Circuit Design

$\square$ Procedure of SOP minimization applies for multiple-output realization with slight modification of the determination of essential prime implicants (EPIs)

- We find prime implicants that are essential to one of the functions and to the multiple-output realization
$\square$ Some of the prime implicants essential to an individual function may not be essential to the multiple-output realization
- Recall an EPI is a prime implicant that covers some minterm $m_{i}$ that is not covered by any other prime implicant
- The minterm $\mathrm{m}_{\mathrm{i}}$ may appear and be covered by a (different?) prime implicant in another function
E.g.,

Slide 34:
bd is essential to $f_{1}$, but not for the multiple-output realization
Slide 36:
$c^{\prime} d$ is essential to $f_{1}$ for the multiple-output realization abd is essential to $f_{1}$, but not for the multiple-output realization
Slide 37:
a'd', a'bc' are essential to $f_{1}$ for the multiple-output realization bd' is essential to $f_{2}$ for the multiple-output realization

## Multiple-Output NAND and NOR

 Circuits$\square$ Procedure for designing single-output, multi-level NAND- and NOR-gate circuits applies to multiple-output circuits

- If all of the output gates are OR (AND), direct conversion to a NAND-gate (NOR-gate) circuit is possible


## Multiple-Output NAND and NOR

 Circuits$\square$ Multi-level circuit conversion to NOR gates Level 4 Level 3 Level $2 \quad$ Level 1


