Sintesi di reti combinatorie

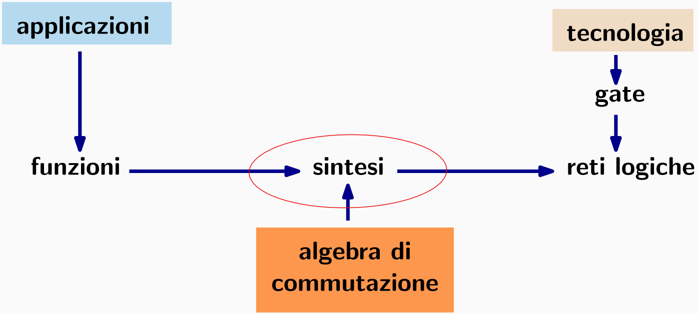
Funzioni $\Rightarrow$ Espressioni

M. Favalli

Engineering Department in Ferrara

Sommario

Obbiettivo



- **Siamo al punto di prima ?**
- **No, abbiamo due strumenti per arrivare a un risultato: l'algebra di commutazione e il caclolo delle proposizioni**

Sommario

Motivazioni

- **Si deve trovare una metodologia in grado di ottenere un espressione equivalente a una funzione di partenza**
- **Il numero di tali espressioni é peró infinito**
- **Si puó utilizzare come punto di partenza una forma canonica, ovvero un espressione che data la funzione é unica**
- **Questa espressione é poi il punto di partenza per possibili strategie di ottimizzazione che data una specifica (costo....) sfruttano le proprietá dell'algebra di commutazione per ottenere un espressione migliore di quella di partenza**
- **Le forme canoniche sono utili anche per la verifica**

Approccio

- **Sono possibili due diversi approcci al problema**
  1. **utilizzo di alcune proprietá dell'algebra di commutazione (Teorema di Espansione di Shannon)**
  2. **utilizzo del calcolo delle proposizioni**
- **Utilizzermo il secondo approccio perché piú intuitivo, in seguito utilizzeremo l'algebra**

Termine prodotto $\Leftrightarrow$ configurazione binaria

- **Si consideri una funzione binaria** $f(x_1, x_2, ..., x_n) : \mathbb{B}^n \to \mathbb{B}$
- **Una riga della tabella di veritá é caratterizzata da una configurazione binaria degli ingressi**
- **Supponiamo di avere** $n = 3$ **e consideriamo la riga con la configurazione binaria** $x_1 x_2 x_3 = 011$
- **Siamo in questa configurazione se** $x_1 = 0$ *e* $x_2 = 1$ *e* $x_3 = 1$
- **Trasformando tale proposizione congiuntiva (**$\neg x_1 \wedge x_2 \wedge x_3$**) in un termine prodotto** p**, si ha** $p = x_1' x_2 x_3$
- **Se** $p = 1$**, allora la configurazione di ingresso é quella considerata**

Termine somma $\Leftrightarrow$ configurazione binaria

- **Si puó associare anche un termine somma a una configurazione binaria**
- **Supponiamo di avere $n = 3$ e consideriamo la riga con la configurazione binaria $x_1 x_2 x_3 = 011$**
- **Non siamo in questa configurazione se $x_1 = 1$ *o* $x_2 = 0$ *o* $x_3 = 0$**
- **Trasformando tale proposizione disgiuntiva in un termine somma s si ha $s = x_1 + x_2' + x_3'$**
- **Se $s = 0$, allora la configurazione di ingresso é quella considerata**

Mintermini

Data una funzione $f : \mathbb{B}^n \to \mathbb{B}$, si definiscono *mintermini*, quei termini prodotto in cui compaiono n variabili diverse (in forma vera o complementata) che corrispondono a righe della tabella di veritá dove la funzione vale 1

indice	$x_1 x_2 x_3$	f	termine prodotto	mintermini
0	000	0	$x_1' x_2' x_3'$	
1	001	0	$x_1' x_2' x_3$	
2	010	1	$x_1' x_2 x_3'$	m_2
3	011	1	$x_1' x_2 x_3$	m_3
4	100	0	$x_1 x_2' x_3'$	
5	101	1	$x_1 x_2' x_3$	m_5
6	110	1	$x_1 x_2 x_3'$	m_6
7	111	0	$x_1 x_2 x_3$	

Prima forma canonica (SP)

- **La funzione f puó essere descritta dalla seguente proposizione:** *la funzione vale 1 se il valore delle variabili di ingresso rende vero il primo mintermine, o il secondo, ... , o il k-mo*
    - **dove k é il numero di righe dove f vale 1**
- **Trasformando tale proposizione in un espressione dell'algebra di commutazione si ha che f é equivalente alla somma logica di tutti i mintermini**

$$f = \sum_{i|f_i=1} m_i$$

- **Si definisce tale espressione come forma canonica di tipo somma di prodotti (SP o SOP) la disgiunzione (somma) di tutti i mintermini della funzione**
- **Quindi si ha finalmente un espressione per f!!!!**

Esempio di forma canonica SP

$x_1 x_2 x_3$	f	**termine prodotto**	**mintermini**
000	**0**	$x_1' x_2' x_3'$	
001	**0**	$x_1' x_2' x_3$	
010	**1**	$x_1' x_2 x_3'$	m_2
011	**1**	$x_1' x_2 x_3$	m_3
100	**0**	$x_1 x_2' x_3'$	
101	**1**	$x_1 x_2' x_3$	m_5
110	**1**	$x_1 x_2 x_3'$	m_6
111	**0**	$x_1 x_2 x_3$	

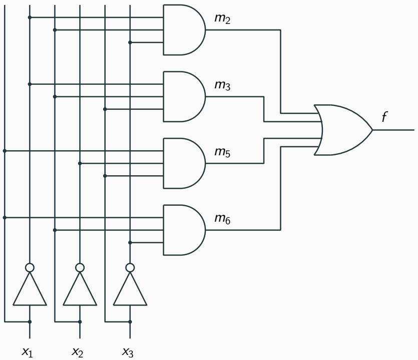
Forma canonica SP:

$$f(x_1, x_2, x_3) = m_2 + m_3 + m_5 + m_6 = x_1' x_2 x_3' + x_1' x_2 x_3 + x_1 x_2' x_3 + x_1 x_2 x_3'$$

Fissato un ordine per le variabili, questa espressione é unica (quindi a meno della proprietá commutativa di somma e prodotto)

Rete logica corrispondente (SP)

- **Funzione:**
  $f(x_1, x_2, x_3) = m_2 + m_3 + m_5 + m_6 = x_1' x_2 x_3' + x_1' x_2 x_3 + x_1 x_2' x_3 + x_1 x_2 x_3'$
- **Supponiamo che la tecnologia, oltre al NOT, metta a disposizione porte logiche AND e OR con un fan-in sufficiente**
- **Si ha una rete a due livelli di tipo SP (l'invertitore non conta)**



Maxtermini

Data una funzione $f : \mathbb{B}^n \to \mathbb{B}$, si definiscono *maxtermini*, quei termini somma in cui compaiono n variabili diverse (in forma vera o complementata) che corrispondono a righe della tabella di veritá dove la funzione vale 0

indice	$x_1 x_2 x_3$	f	**termine somma**	**maxtermini**
0	**000**	**0**	$x_1 + x_2 + x_3$	M_0
1	**001**	**0**	$x_1 + x_2 + x_3'$	M_1
2	**010**	**1**	$x_1 + x_2' + x_3$	
3	**011**	**1**	$x_1 + x_2' + x_3'$	
4	**100**	**0**	$x_1' + x_2 + x_3$	M_4
5	**101**	**1**	$x_1' + x_2 + x_3'$	
6	**110**	**1**	$x_1' + x_2' + x_3$	
7	**111**	**0**	$x_1' + x_2' + x_3'$	M_7

Seconda forma canonica (PS)

- **La funzione f puó anche essere descritta dalla seguente proposizione:** *la funzione vale 1 se il valore delle variabili di ingresso rende vero il primo maxtermine, e il secondo, ... , e il j-mo*
    - **dove j é il numero di righe dove f vale 0**
- **Trasformando tale proposizione in un espressione dell'algebra di commutazione si ha che f é equivalente al prodotto logico di tutti i maxtermini**

$$f = \prod_{\forall i \mid f(i)=0} M_i$$

- **Si definisce forma canonica di tipo prodotto di somme (PS o POS) la congiunzione (prodotto) di tutti i maxtermini della funzione**
- **Quindi si ha una seconda espressione per f! (ve lo aspettavate? .... principio di dualitá)**

Esempio di forma canonica di tipo PS

$x_1 x_2 x_3$	f	**termine prodotto**	**maxtermini**
000	**0**	$x_1 + x_2 + x_3$	M_0
001	**0**	$x_1 + x_2 + x_3'$	M_1
010	**1**	$x_1 + x_2' + x_3$	
011	**1**	$x_1 + x_2' + x_3'$	
100	**0**	$x_1' + x_2 + x_3$	M_4
101	**1**	$x_1' + x_2 + x_3'$	
110	**1**	$x_1' + x_2' + x_3$	
111	**0**	$x_1' + x_2' + x_3'$	M_7

Forma canonica PS

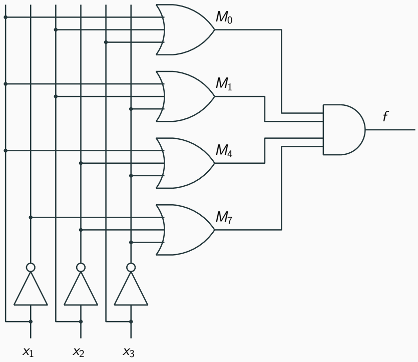
$$f(x_1, x_2, x_3) = M_0 M_1 M_4 M_7 = (x_1+x_2+x_3)(x_1+x_2+x_3')(x_1'+x_2+x_3)(x_1'+x_2'+$$

Rete logica corrispondente (PS)

- **Funzione:**
  $f(x_1, x_2, x_3) = M_0 + M_1 + M_4 + M_7 = (x_1 + x_2 + x_3)(x_1 + x_2 + x_3')(x_1' + x_2 + x_3)(x_1' + x_2' + x_3')$

t

- **Supponiamo che la tecnologia, oltre al NOT, metta a disposizione porte logiche AND e OR con un fan-in sufficiente**
- **Si ha una rete a due livelli di tipo PS (l'invertitore non conta)**



Forme canoniche SP e PS

- **Fissato un ordine per le variabili, sono uniche**
- **Attenzione: la seconda forma canonica non si ottiene complementando la prima,entrambe descrivono** f
- **Le forme canoniche SP e PS, danno tipicamente luogo a reti piuttosto costose il cui costo é proporzionale a** 2^n
- **Quale delle due conviene? Dipende dalla tecnologia e dal numero di 1 e 0**

Sommario

Il ruolo dell’ottimizzazione nella sintesi di reti logiche

- **Ingredienti:**
  - **funzione/espressione di partenza**
  - **un obbiettivo di progetto (costo, ritardo, consumo di potenza) e una metrica che descriva tale obbiettivo**
- **Il compito dell’ottimizzazione é quello di esplorare lo spazio delle possibili espressioni equivalenti cercando quella piú conveniente dal punto di vista della metrica considerata**
- **Questo puó essere fatto mediante:**
  - **una tecnica esatta che trova la soluzione migliore (es. rete dal costo minimo)**
  - **una tecnica euristica che porta a una soluzione approssimata (es. un minimo locale del costo) con un costo computazionale decisamente inferiore all’approccio esatto**

Ottimizzazione del costo

- **Le forme canoniche consentono di descrivere una funzione mediante l'algebra di commutazione**
- **In molti casi é possibile semplificare una forma canonica ottenendo un espressione equivalente piú semplice e quindi una rete meno costosa**

$$
\begin{aligned}
f(x_1, x_2, x_3) &= x_1' x_2 x_3' + x_1' x_2 x_3 + x_1 x_2' x_3 + x_1 x_2 x_3 \\
&= x_1' x_2 (x_3' + x_3) + x_1 x_3 (x_2' + x_2) \\
&= x_1' x_2 + x_1 x_3
\end{aligned}
$$

- **Occorre specificare meglio cosa vuole dire "meno costosa"**

Metriche per la stima del costo di una rete

- **Il costo di una rete é proporzionale all'area occupata dalla sua implementazione fisica (gate e interconnessioni)**
- **Chiaramente non si puó arrivare fino alla realizzazione fisica per valutare il costo di un espressione**
- **Bisogna trovare una metrica che fornisca una stima approssimata dell'area**
- **Esistono diverse metriche:**
    1. **numero di gate (non si considera il fatto che i gate con piú ingressi hanno maggiori dimensioni)**
    2. **numero di letterali nell'espressione**
    3. **numero di porte logiche pesate sui loro ingressi**
    4. **numero di termini prodotto (somma) nelle espressioni SP (PS)**

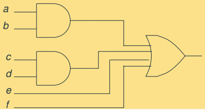
Numero di letterali

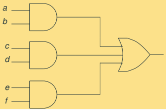
Si conta il numero di letterali (l) nell'espressione

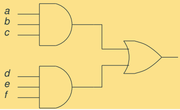
$$f = ab + c + d'e \qquad l = 5$$
$$f = abc + a'bce + d'e'f' \qquad l = 10$$

Questo metodo é semplice, ma puó essere reso piú accurato. Si considerino 3 funzioni con $l = 6$ e le reti corrispondenti

$f_1 = ab + cd + e + f \qquad f_2 = ab + cd + ef \qquad f_3 = abc + def$







Il numero totale di ingressi di gate (g) che é proporzionale all'area occupata dai gate é $g_1 = 8$, $g_2 = 9$, $g_3 = 8$

Esempi

Somma dei gate pesata sugli ingressi

- **Questa metrica puó essere facilmente utilizzata se é disponibile la rete, mentre é difficile da applicare direttamente all'espressione**
- **A questo riguardo si possono mettere tutte le parentesi (compreso quelle non necessarie) e contare il numero di operandi di ciascun operatore somma e prodotto, aggiungendo 1 tutte le volte che si incontra un invertitore**
- **Esempio** $f = abc + d' + e(f+g) = (abc) + d' + (e(f+g))$**, si ha un OR a 3 ingressi, un AND a 3 ingressi, un OR a 2 ingressi, un AND a 2 ingressi e un NOT. Quindi** $g = 11$ **(mentre** $l = 7$**)**

Il ruolo del fan-out

- **Chiaramente la valutazione del costo di un espressione corrispondente a una rete deve essere attuata tenendo in conto dell'eventuale utilizzo del fan-out evitando cosí di contare piú volte letterali e operandi**
  - **la corrispondenza uno a uno fra espressioni e rete si perde per l'utilizzo di fan-out** > 1
- **In effetti tutti i metodi considerati non tengono conto del costo delle interconnessioni**
- **Dopo aver definito una metrica per valutare il costo di una rete, possiamo iniziare a vedere come ridurre il costo delle forme canoniche**

Esempi

Sommario

Forme canoniche

Ottimizzazione

Forme normali

Insiemi funzionalmente completi

Forme normali e reti a 2 livelli

- **Considereremo per primo un problema particolare: la sintesi di reti che realizzano espressioni del tipo somma di prodotti (SP) o prodotti di somme (PS)**
- **Tali espressioni vengono definite come forme normali (le forme canoniche ne sono un caso particolare)**
- **Le reti corrispondenti (trascurando i NOT) vengono definite a 2 livelli**
- **I motivi di questa restrizione sono:**
  - **la possibilitá di trovare una soluzione esatta del problema con tempi di calcolo ragionevoli**
  - **alcuni strumenti sviluppati in questo caso servono per risolvere il problema generale**
  - **l'interesse tecnologico di metodologie di fabbricazione in grado di realizzare reti a 2 livelli in maniera molto efficiente**

Sintesi di reti a 2 livelli

- **Sono stati sviluppati diversi metodi per la sintesi a 2 livelli, sia di tipo esatto che euristico**
  - **metodi esatti: forniscono una rete di costo minimo**
  - **metodi euristici: riducono il costo della rete di partenza, ma possono produrre una rete con un costo corrispondente a un minimo locale**
- **In questo ambito vedremo un metodo di carattere euristico di tipo grafico detto delle mappe di Karnaugh che consente la sintesi e l'ottimizzazione di funzioni con fino a 6 ingressi**
- **Nell'ambito del software in laboratorio** `logic-friday` **sono disponibili 2 approcci, uno ti tipo esatto ($20 - 25$ ingressi) e l'altro euristico (> 20)**

Implicanti e implicati

- **Sia** n **il numero di variabili**
- **Nella forma normale di tipo SP di una funzione** f**, un termine prodotto** P **di** k **letterali corrispondenti a** $k \leq n$ **variabili diverse viene detto** *implicante*
- **Infatti** $P = 1 \rightarrow f = 1$ **(se** $f = P + Q$**, allora** $P \rightarrow f = P' + (P + Q) = 1$ **quindi l'espressione é sempre vera)**
- **In una forma normale di tipo PS di una funzione** f**, un termine somma** S **di** k **letterali corrispondenti a** $k \leq n$ **variabili diverse viene detto** *implicato*
- **Infatti** $S = 1 \leftarrow f = 1$ **(se** $f = SQ$**, allora** $f \rightarrow S = (SQ)' \rightarrow S = S' + Q' + S = 1$ **)**

Diagrammi di Venn

Espansione

- **Sia P un termine prodotto di $k \leq n$ letterali, vale la proprietá:**

$$x'P + xP = (x' + x)P = 1 \cdot P = P$$

- **L'espressione di partenza ha $2(k+1)$ letterali, e quella finale ne ha k (in modo analogo queste hanno rispettivamente 2 e 1 termine prodotto)**
- **La regola puó essere applicata iterattivamente**

$$\begin{aligned} f &= a'b'c'd' + a'b'c'd + a'bc'd' + a'bc'd \\ &= a'b'c'(d' + d) + a'bc'(d' + d) \\ &= a'b'c' + a'bc' \\ &= a'c'(b + b') \\ &= a'c' \end{aligned}$$

L’espansione non é sufficiente

- **Si consideri un ulteriore caso**

$$f = abc'd + abcd + a'bcd$$

- **La proprietá di espansione puó essere applicata in 2 modi diversi, si ha:**

$$f = abd(c + c') + a'bcd = abd + a'bcd$$

$$f = bcd(a + a') + abc'd = bcd + abc'd$$

- **Dopo uno di questi passi, la proprietá di espansione non puó piú essere applicata**

Idempotenza

- **Si potrebbe passare per un espressione multilivello per semplificare ulteriormente le espressioni (utilizzando la proprietá di semplificazione)**

$$f = abd + a'bcd = bd(a + a'c) = bd(a + c) = abd + bcd$$

$$f = bcd + abc'd = bd(c + ac') = bd(a + c) = abd + bcd$$

- **Come alternativa si puó utilizzare la proprietá di idempotenza (**$P + P = P$**) per poi applicare l'espansione**

$$f = abc'd + abcd + a'bcd = abc'd + abcd + abcd + a'bcd =$$

$$= abd(c' + c) + bcd(a + a') = abd + bcd$$

Espansione in espressioni PS

- **Le proprietá duali possono essere applicate a espressioni di tipo PS**

$$(x+S)(x'+S)=S$$

- **Vediamo ad esempio il caso dell'espansione:**

$$f=(a+b+c+d)(a+b+c'+d)(a+b'+c+d)(a+b'+c'+d)$$

$$=((a+b+d)+c)\left((a+b+d)+c'\right)\left((a+b'+d)+c\right)\left((a+b'+d)+c'\right)$$

$$=(a+b+d)(a+b'+d)=((a+d)+b)\left((a+d)+b'\right)=a+d$$

Implicanti e implicati primi

Definizione

Un implicante o implicato che non puó essere ulteriormente espanso si definisce come primo

Proprietá interessante, ma che al momento non riusciamo a utilizzare perché non abbiamo un metodo per costruire tali implicanti primi

La distanza Hamming viene introdotta per aiutarci da questo punto di vista

Distanza Hamming (forme SP)

- **Selezione di coppie di termini per l'espansione: termini prodotto con lo stesso numero di letterali che differiscono per un solo letterale corrispondente alla stessa variabile in forma vera e complementata**
- **Nel caso dei mintermini l'operazione puó essere fatta sia analizzando l'espressione che analizzando la tabella di veritá**
- **In particolare, possono essere espanse le configurazioni corrispondenti a uni della funzione che si trovano a distanza Hamming unitaria**
- **Si considerino ad esempio i mintermini:**

$xywz$	**1111**
$xywz'$	**1110**

- **Questi possono essere espansi come** $xyw(z + z') = xyw$

Rappresentazione dei termini prodotto come configurazioni

- **Il ragionamento sulle configurazioni binarie puó essere esteso a termini prodotto con un qualsiasi numero di letterali**
- **Questo puó essere fatto utilizzando configurazioni di $\{0, 1, -\}^n$ che utilizzano lo stesso ordinamento delle variabili usato nella tabella di veritá**
  - **in particolare si associa a ogni letterale presente nel termine prodotto il simbolo 1 se il letterale é in forma vera e 0 se é in forma negata**
  - **si associa il simbolo $-$ a quelle variabili che non compaiono nel prodotto**
- **Ad esempio $f(x, y, w, z)$: $xyw \Leftrightarrow 111-$, $xy' \Leftrightarrow 10--$**

Generalizzazione della distanza Hamming

- **La distanza Hamming puó essere generalizzata a quelle configurazioni di $\{0, 1, -\}^n$ che hanno il simbolo $-$ nelle stesse posizioni**
- **Ad esempio,** $01-0$ **e** $01-1$ **hanno distanza 1, mentre** $11--$ **e** $00--$ **hanno distanza 2**
- **Termini prodotto corrispondenti a configurazioni a distanza 1 possono essere espansi sostituendo il valore per cui differiscono con il simbolo** $-$
- **Esempio** $01-0$ **e** $01-1$ **possono essere espansi come** $01--$
- **Infatti, i termini prodotto corrispondenti sono** $x'yz + x'yz' = x'y(z+z') = xy'$

Sommario

Forme canoniche

Ottimizzazione

Forme normali

Insiemi funzionalmente completi

Insiemi funzionalmente completi

- **Un insieme di operatori si dice funzionalmente completo se é in grado di descrivere qualsiasi funzione dell'algebra di commutazione**
- **Gli operatori di disgiunzione, congiunzione e negazione costituiscono un insieme funzionalmente completo**
- **In pratica si é visto come** $\{\vee, \wedge, \neg\}$ **(**$\{+, \cdot, '\}$**) siano sufficenti a descrivere qualsiasi funzione, sono anche necessari?**
- **Esistono insiemi piú piccoli che siano funzionalmente completi**

Operatore $\uparrow$ (NAND)

- **L'operatore $\uparrow$ viene definito come** $a \uparrow b = (a \cdot b)'$
- **Corrisponde a un gate NAND**
- **Risulta da solo funzionalmente completo**
- **La dimostrazione é banale, si tratta solo di verificare che tramite tale operatore si possono descrivere tutti gli operatori dell'algebra di commutazione**
  - $a' = a \uparrow a = a \uparrow 1$
  - $a \cdot b = (a \cdot b)'' = (a \uparrow b) \uparrow (a \uparrow b) = (a \uparrow b) \uparrow 1$
  - $a + b = (a' \cdot b')' = (a \uparrow a) \uparrow (b \uparrow b) = (a \uparrow 1) \uparrow (b \uparrow 1)$

Operatore NAND $\uparrow$ e forme normali SP

- **Le forme normali SP possono essere trasformate in maniera efficente in forme a 2 livelli utilizzanti l'operatore $\uparrow$**
- **La trasformazione puó essere eseguita utilizzando De Morgan**

$$y = ab + a'cd + be = (ab + a'cd + be)'' = ((ab)' \cdot (a'cd)' \cdot (be)')'$$

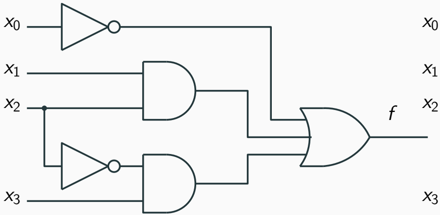
- **Come si puó osservare, l'unico operatore che compare é il NAND ($\uparrow$) da cui:**

$$y = (a \uparrow b) \uparrow (a' \uparrow c \uparrow d) \uparrow (b \uparrow e)$$

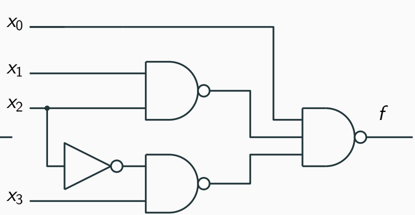
- **Si noti che l'operatore binario $\uparrow$ non é associativo:** $x \uparrow y \uparrow w \neq (x \uparrow y) \uparrow w$
- **Quando é utilizzato come operatore *n*-ario si ha:** $(x_0 \cdot x_1 .... \cdot x_{n-1})' = (x_0 \uparrow x_1 .... \uparrow x_{n-1})$

Esempi di trasformazione da reti SP a 2 livelli a reti a NAND a 2 livelli

$x_0' + x_1x_2' + x_2x_3$



$(x_0 \uparrow (x_1 \uparrow x_2') \uparrow (x_2 \uparrow x_3))$



Operatore $\downarrow$ (NOR)

- **L'operatore $\downarrow$ viene definito come** $a \downarrow b = (a+b)'$
- **Corrisponde a un gate NOR**
- **Risulta da solo funzionalmente completo**
- **Tutto quanto visto nel caso di $\uparrow$ puó essere esteso a questo caso utilizzando il principio di dualitá**
- **Si noti che in questo caso reti PS possono essere trasformate in reti a NOR a due livelli**

Conclusioni

- **Si é visto come si possa rappresentare una funzione mediante un espressione**
- **É stata sviluppata una metrica per valutare il costo di un espressione**
- **Sono stati sviluppati alcuni concetti utili per la semplificazione di un espressione**
- **Non si ha ancora un approccio sistematico**