1 Lezione del 08-10-24

1.1 Descrizione funzionale

La caratteristica più importante di una rete combinatoria è la funzione F, cioé la descrizione funzionale. Esistono più modi per esprimere questa funzione:

- A parole;
- Usando notazioni testuali (e..g. il Verilog);
- Attraverso **tabelle di verità**. In una tabella di verità contiene due insiemi di colonne: gli ingressi e le uscite. Ogni riga mostra una configurazione di stati di ingresso e il corrispondente stato d'uscita. Ad esempio:

x_2	x_1	x_0	z_1	z_0
0	0	0	0	0
0	0	1	$-$	1
0	1	0	1	0
...				

  Si dice che la variabile di uscita **riconosce** particolari stati quando si attiva in presenza di essi. Inoltre, i trattini indicano stati **non specificati**, in inglese DC, *don't care*. Questi non equivalgono alla fascia di indeterminazione, ma a uno dei due stati accettati, anche se non è importante quale. I *don't care* vanno conservati, e non fissati a variabili come 0 o 1, in quanto è importante mantenere il funzionamento interno delle reti il più semplice possibile.

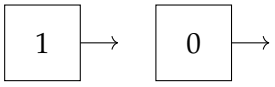
1.1.1 Descrizione e sintesi

Una **descrizione** di una rete deve essere formale, in modo che si possa capire esattamente cosa fa quella rete. La **sintesi** di una rete è il progetto stesso di realizzazione della rete, cioè quali componenti combinare in quale modo, ecc... Prima si fa la descrizione, e poi la sintesi.

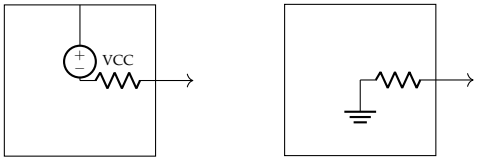
Notiamo una proprietà fondamentale: ogni rete combinatoria di N ingressi e M uscite può essere realizzata interconnettendo M reti combinatorie ad N ingressi ed una uscita. Questo ci permette di trattare tutte le reti con reti con una sola uscita.

1.2 Reti a 0 ingressi

Le reti a 0 ingresso di uscita si chiamano **generatori di costante**, e rappresentano un caso degenere. Si indicano come:



La loro uscita chiaramente vale 1 o 0 costante. Fisicamente, i generatori di costante si realizzano collegando resistori in serie al VCC (genera 1) o a massa (genera 0), ergo:



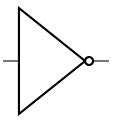
1.3 Reti a 1 ingresso

1.3.1 Invertitore

L'invertitore è una rete descritta dalla tabella di verità:

x	z
0	1
1	0

e indicata come:



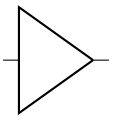
Essenzialmente nega il suo ingresso.

1.3.2 Elemento neutro

L'elemento neutro, detto anche *buffer*, è una rete descritta dalla tabella di verità:

x	z
0	0
1	1

e indicata come:



Lascia il suo ingresso invariato. Può avere un utilità come rete di rallentamento, in quanto, inevitabilmente, si perde tempo per attraversarla (pensa alla NOP). Questo è utile per le temporizzazioni delle reti.

Inoltre, dal punto di vista elettrico, l'elemento neutro ha anche un utilità per la **rigenerazione** dei segnali. Infatti, essendo collegato a massa e al VCC, può prendere segnali scadenti (vicini alla fascia di indeterminazione) e trasformarli in segnali di buona qualità (vicini al fondoscala). Questa proprietà, veramente, è comune a tutte le reti logiche, ma l'elemento neutro è l'unico che non ha altri effetti collaterali.

1.3.3 Reti costanti

Si possono interpretare i generatori di costante come reti ad un ingresso degeneri. Effettivamente, restano tali a se stesse, in quanto gli ingressi sono ignorati. Le loro tabelle di verità sono:

Generatore di 1:

x	z
0	1
1	1

Generatore di 0:

x	z
0	0
1	0

1.4 Reti a 2 ingressi

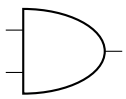
La prima domanda da porsi quando si parla di reti a 2 (come N) ingressi, è quante reti possiamo creare in tutto. Su N ingressi, la tabella di verità avrà 2^N righe. Le configurazioni possibili di 0 e 1 su 2^N righe sono 2^{2^N}. Ergo, nel caso $N = 2$, abbiamo $2^{2^2} = 16$ possibili combinazioni, che sono:

x_1	x_0	z^0	z^1	z^2	z^3	z^4	z^5	z^6	z^7	z^8	z^9	z^{10}	z^{11}	z^{12}	z^{13}	z^{14}	z^{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Ad alcune di queste corrispondono nomi speciali. Vediamole nel dettaglio:

1.4.1 Porta AND

La porta AND, indicata in z^1 , corrisponde al $\wedge$ logico, ergo $z = 1 \Leftrightarrow x_0 = x_1 = 1$. Si indica come:

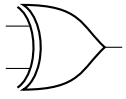


e ha tabella di verità:

x_1	x_0	z
0	0	0
0	1	0
1	0	0
1	1	1

1.4.2 Porta XOR

La porta XOR, indicata in z^6 , corrisponde all'*aut* logico, cioè esclusivo, ergo $z = 1 \Leftrightarrow x_0 \neq x_1$. Si indica come:

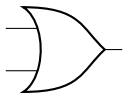


e ha tabella di verità:

x_1	x_0	z
0	0	0
0	1	1
1	0	1
1	1	0

1.4.3 Porta OR

La porta OR, indicata in z^7 , corrisponde al $\vee$ logico, ergo $z = 0 \Leftrightarrow x_0 = x_1 = 0$. Si indica come:

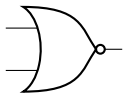


e ha tabella di verità:

x_1	x_0	z
0	0	0
0	1	1
1	0	1
1	1	1

1.4.4 Porta NOR

La porta NOR, indicata in z^8 , corrisponde alla negazione dell'$\vee$ logico, ergo $z = 1 \Leftrightarrow x_0 = x_1 = 0$. Si indica come:

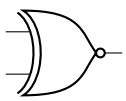


e ha tabella di verità:

x_1	x_0	z
0	0	1
0	1	0
1	0	0
1	1	0

1.4.5 Porta XNOR

La porta XNOR, indicata in z^9 , corrisponde alla negazione dell'*aut* logico, ergo $z = 1 \Leftrightarrow x_0 = x_1$. Si indica come:

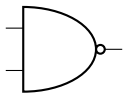


e ha tabella di verità:

x_1	x_0	z
0	0	1
0	1	0
1	0	0
1	1	1

1.4.6 Porta NAND

La porta NAND, indicata in z^14 , corrisponde alla negazione dell'$\wedge$ logico, ergo $z = 0 \Leftrightarrow x_0 = x_1 = 1$. Si indica come:



e ha tabella di verità:

x_1	x_0	z
0	0	1
0	1	1
1	0	1
1	1	0

Si dovrebbe essere notato che un pallino finale indica negazione. A volte si usa solo questa notazione, invece di tutta la porta NOT.

1.4.7 Casi degeneri

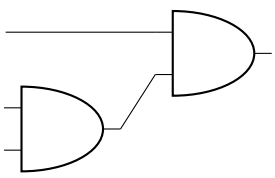
Alcuni casi speciali della tabella delle possibili reti a due porte sono degeneri: abbiamo due generatori di costante (z^0 e z^{15}), due elementi neutri, rispettivamente su x_1 e x_0 (z^3 e z_5), e due inversori sugli stessi ingressi (z_{10} e z_{12}).

1.5 AND e OR a più ingressi

Posso pensare di estendere AND e OR ad N ingressi:

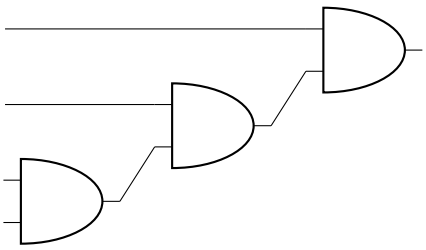
- **AND a N ingressi:** l'uscita vale 1 se tutti gli N ingressi valgono 1;
- **OR a N ingressi:** l'uscita vale 1 se almeno un'ingresso vale 1;

Questo può essere realizzato concatenando più porte logiche dello stesso tipo, come segue:

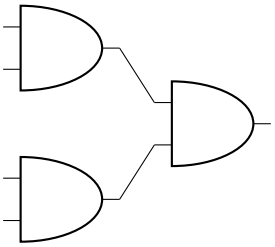


La dimostrazione è semplice dalla tabella di verità, o dalle proprietà degli operatori logici.

Una nota va fatta sulle combinazioni di più di 3 ingressi, infatti una rete del genere è sconveniente:



in quanto il segnale deve attraversare al massimo 3 livelli di logica, mentre disponendo le porte come:



il segnale dovrà attraversare al massimo 2 livelli di logica.

Conviene quindi disporre gli N ingressi e le relative porte come un'albero binario bilanciato, in modo da minimizzare gli attraversamenti di livelli di logica. Si noti che questo discorso vale per AND e OR: non per NAND, NOR, XOR o XNOR.

Possiamo osservare velocemente cosa accade se si collegano queste porte fra di loro:

- **NAND:** un singolo NAND può formare un NOT quando i suoi ingressi sono uniti insieme. Se si mettono 2 NAND in serie (a *cascata*) in questo modo, si ottiene di nuovo un AND;
- **NOR:** un singolo NOR può formare un NAND nello stesso modo del NAND. Se si mettono 2 NOR a cascata, si ottiene di nuovo un NOR;
- **XOR:** con ≥ 2 XOR, si crea effettivamente un controllore di parità, ergo una rete che si attiva quando un numero dispari dei suoi ingressi sono accesi;
- **XNOR:** con ≥ 2 XNOR, si ha l'opposto che con gli XOR: si crea una rete che si attiva quando un numero pari dei suoi ingressi sono accesi.

Queste porte si indicano solitamente come con gli input su unica orizzontale, che risulta più compatto.

1.6 Algebra di Boole

L'algebra di Boole adopera gli operatori logici conosciuti, applicati ad elementi del campo binario $GF(2) = \{0, 1\}$

Vediamo questi operatori:

- **Complemento logico:** si indica come $\bar{x}$, oppure $!x$ o $/x$. Si definisce come:

$$\bar{0} = 1, \quad \bar{1} = 0$$

- **Somma logica:** si indica con $x + y$, e ha tabella di verità:

x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

  cioè equivale all'OR.

- **Prodotto logico:** si indica con $x \cdot y$, e ha tabella di verità:

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

  cioè equivale all'AND.

Su questi operatori valgono le proprietà:

1. **Involutiva del complemento:** $\bar{\bar{x}} = x$;
2. **Commutativa della somma e del prodotto:** $x + y = y + x, \quad x \cdot y = y \cdot x$;
3. **Associativa della somma:** $x + y + z = (x + y) + z = x + (y + z)$;
4. **Associativa del prodotto:** $x \cdot y \cdot z = (x \cdot y) \cdot z = x \cdot (y \cdot z)$;
5. **Distributiva della somma rispetto al prodotto:** $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$;
6. **Distributiva del prodotto rispetto alla somma:** $x+(y \cdot z) = (x+y) \cdot (x+z)$. Bisogna fare attenzione in quanto questa non vale in $\mathbb{R}$;
7. **Complementazione:** $x \cdot \bar{x} = 0, \quad x + \bar{x} = 1$;
8. **Unione e intersezione:** $x + 0 = x, \quad x + 1 = 1$, cioè 0 è l'elemento neutro e 1 l'elemento assorbente della somma (non lo è in $\mathbb{R}$);
   $x \cdot 0 = 0, \quad x \cdot 1 = x$, cioè 1 è l'elemento neutro e 0 l'elemento assorbente del prodotto;
9. **Idempotenza:** $x + x = x, \quad x \cdot x = x$, altra che non vale in $\mathbb{R}$;
10. **Leggi di De Morgan:** $\overline{x \cdot x} = \bar{x} + \bar{x}$ e $\overline{x + x} = \bar{x} \cdot \bar{x}$.

1.6.1 Teoremi di De Morgan

Le leggi di De Morgan comuni della logica si estendono ad N variabili come:

1. $\overline{x_0 \cdot x_1 \cdot ... \cdot x_n} = \bar{x}_0 + \bar{x}_1 + ... + \bar{x}_n$
2. $\overline{x_0 + x_1 + ... + x_n} = \bar{x}_0 \cdot \bar{x}_1 \cdot ... \cdot \bar{x}_n$

Dimostrazione per induzione
Richiamiamo le basi dell'induzione:

- Si dimostra che una proprietà vale per un certo numero n_0 (passo base);
- Si dimostra che se vale per un certo $n \geq n_0$, allora vale anche per $n + 1$.

Partiamo con le dimostrazioni classiche ottenute con le tabelle di verità:

x	y	$x \cdot y$	$\overline{x \cdot y}$	$\bar{x}$	$\bar{y}$	$\bar{x} + \bar{y}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	1	0

che ci portano a $n_0 = 2$. Posso quindi porre l'ipotesi:

$$\overline{x_0 \cdot ... \cdot x_{n-1}} = \bar{x}_0 + ... + \bar{x}_{n-1}$$

e la tesi:

$$\overline{x_0 \cdot ... \cdot x_{n-1} \cdot x_n} = \bar{x}_0 + ... + \bar{x}_{n-1} + \cdot x_n$$

A questo punto faccio il passo induttivo, sfruttando l'associatività del prodotto (o della somma), e quindi riscrivendo la tesi come:

$$\overline{\alpha \cdot x_n}, \quad \alpha = x_0 + ...x_{n-1}$$

dove notiamo la variabile introdotta α, se complementata, rispetta:

$$\bar{\alpha} = \overline{x_0 \cdot ... \cdot x_{n-1}} = \bar{x}_0 + ... + \bar{x}_{n-1}$$

dall'ipotesi.

Possiamo quindi svolgere il passaggio:

$$\overline{\alpha \cdot x_n} = \bar{\alpha} + \bar{x}_n = \bar{x}_0 + ... + \bar{x}_{n-1} + \bar{x}_n$$

che conferma la tesi.

1.6.2 Algebra di Boole e reti combinatorie

Esiste una corrispondenza fra l'algebra di Boole e le reti combinatorie. In particolare, si ha che:

- **Data una rete combinatoria**, (comunque complessa), è sempre possibile trovare un'espressione booleana che mette in relazione ogni sua uscita con gli ingressi (in verità un'espressione per ogni uscita);

- **Data un'espressione booleana**, p sempre possibile sintetizzare una rete combinatoria (ad un'uscita) in cui la relazione tra ingresso ed uscita data è dall'espressione.

Si noti che, effettivamente, espressioni logiche equivalenti $\Leftrightarrow$ reti logiche che svolgono lo stesso compito, ma non per questo l'equivalenza è totale: ci conviene creare reti che usano meno componenti possibili, in quanto queste le rende più affidabili, più economiche e meno dispendiose di energia. Le proprietà dell'algebra di Boole possono quindi essere usate per ridurre il numero di porte logiche, attraverso un processo che chiameremo **minimizzazione**.