

1 Lezione del 15-10-24

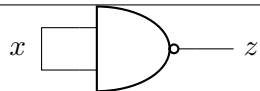
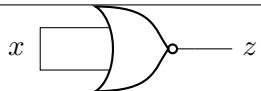
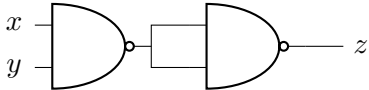
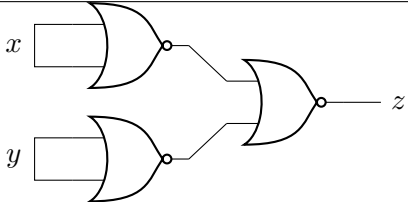
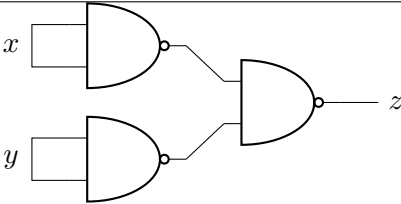
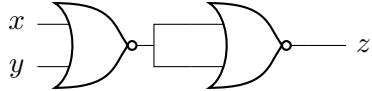
1.1 Porte logiche universali

Si dice che NAND e NOR sono **porte logiche universali**. Si possono realizzare AND, OR e NOT usando solo porte NAND o solo porte NOR.

Algebricamente, questo significa fare:

Porta	Realizzazione NAND	Realizzazione NOR
NOT	$x = x \cdot x \Rightarrow \bar{x} = \overline{x \cdot x}$	$x = x + x \Rightarrow \bar{x} = \overline{x + x}$
AND	$x \cdot y = \overline{(\overline{x \cdot y})}$	$x \cdot y = \overline{\bar{x} + \bar{y}}$ (de Morgan)
OR	$x + y = \overline{\bar{x} \cdot \bar{y}}$ (de Morgan)	$x + y = \overline{(\overline{x + y})}$

cioè collegare porte logiche fisiche nelle seguenti configurazioni:

Porta	Realizzazione NAND	Realizzazione NOR
NOT		
AND		
OR		

Potremmo sembrare che, se si poteva realizzare qualsiasi rete combinatoria con AND, OR e NOT su 2 livelli di logica, usando solo NAND o NOR dovremmo accontentarci di 4 livelli di logica (AND e OR richiedono di per sé una rete a 2 livelli di logica).

In verità, le porte NAND e NOR permettono di creare circuiti logici con gli stessi livelli di logica delle porte AND, OR e NOT.

1.1.1 Sintesi a porte NAND

Vediamo quindi il seguente algoritmo per la sintesi di un circuito con sole porte NAND:

Algoritmo 1 sintesi a porte NAND

Input: un circuito in forma SP

Output: una sintesi a porte NAND

Si sostituisce la porta OR con il suo equivalente a NAND

Si sostituisce ciascun AND con il suo equivalente a NAND

Si eliminano le coppie di NOT interne a cascata

Ignoriamo i NOT sull'ingresso, in quanto abbiamo visto sono effettivamente gratuiti. Abbiamo quindi che, rimuovendo le coppie NOT interni (creati da coppie di NAND con gli stessi input) ritorniamo in una forma a 2 livelli di logica.

Dal punto di vista algebrico si ha:

$$z = P_1 + P_2 + \dots + P_k = \overline{\overline{P_1 + P_2 + \dots + P_k}} = \overline{\overline{P_1} \cdot \overline{P_2} \cdot \dots \cdot \overline{P_k}}$$

dove il complemento superiore è l'ultima porta NAND (quella che sostituisce l'OR), e i singoli P_i complementati sono singole porte NAND (P_i è un prodotto, quindi $\overline{P_i}$ è una porta NAND).

1.1.2 Sintesi a porte NOR

Vediamo poi l'algoritmo per la sintesi di un circuito con sole porte NOR:

Algoritmo 2 sintesi a porte NOR

Input: un circuito in forma PS

Output: una sintesi a porte NOR

Si sostituisce la porta AND con il suo equivalente a NOR

Si sostituisce ciascun OR con il suo equivalente a NOR

Si eliminano le coppie di NOT interne a cascata

Anche qui ignoriamo i NOT sull'ingresso, per gli stessi motivi di prima, e rimuovendo le coppie NOT interni (creati da coppie di NAND con gli stessi input) ritorniamo nuovamente in una forma a 2 livelli di logica.

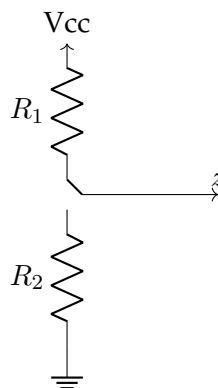
Dal punto di vista algebrico si ha:

$$z = S_1 \cdot S_2 \cdot \dots \cdot S_k = \overline{\overline{S_1 \cdot S_2 \cdot \dots \cdot S_k}} = \overline{\overline{S_1} + \overline{S_2} + \dots + \overline{S_k}}$$

dove il complemento superiore è l'ultima porta NOR (quella che sostituisce l'AND), e i singoli S_i complementati sono singole porte NOR (S_i è una somma, quindi $\overline{S_i}$ è una porta NOR).

1.2 Porte tri-state

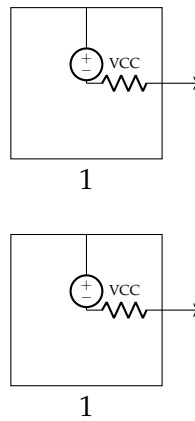
Fa comodo poter connettere insieme le uscite delle reti usando bus condivisi, cioè linee di ingresso-uscita. Abbiamo che l'uscita di una rete, dal punto di vista di una rete, corrisponde a un interruttore fra il Vcc (1 logico) o la massa (0 logico), cioè:



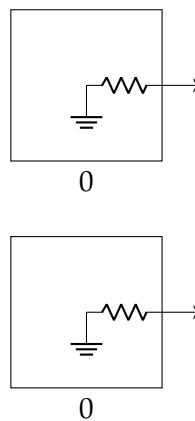
dove R_1 e R_0 sono incognite.

Quando vado a collegare più uscite sulla stessa linea possono crearsi più situazioni:

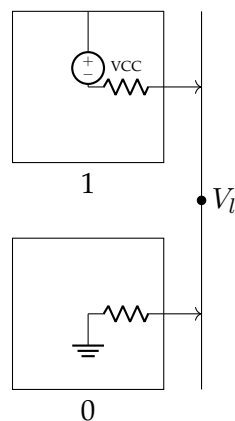
- **1 logici:** se ho due 1 logici, cioè due generatori di potenziale a V_{cc} , connessi sulla stessa linea, ho che la tensione sulla linea è sempre V_{cc} , quindi tutto ok:



- **0 logici:** allo stesso tempo, se ho due 0 logici, quindi due collegamenti a massa, sulla linea si avrà tensione nulla:



- **0 e 1 logici:** se collego un 1 logico e uno 0 logico alla stessa linea, ottengo effettivamente un partitore di tensione:

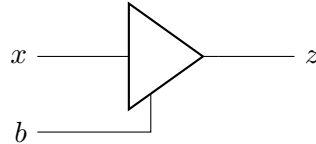


da cui ricavo:

$$V_l = \frac{R_0}{R_1 + R_0}$$

Notiamo soprattutto che se R_0 e R_1 sono molto piccoli, otteniamo correnti I molto grandi, che significa componenti bruciati.

Per risolvere il problema dato da 0 e 1 logici connessi sulla stessa linea, usiamo specifici apparecchi detti **porte tri-state**, che sono capaci di disconnettere fisicamente un'uscita da una linea condivisa. Si rappresentano come:



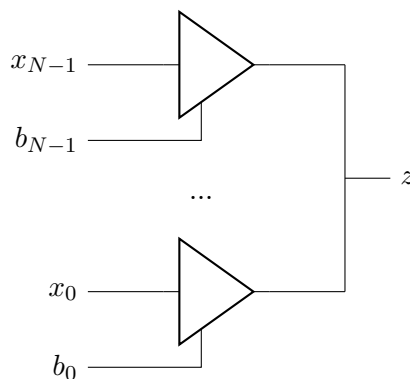
dove x è l'ingresso, z l'uscita, e b l'enabler. A $b = 1$ la porta si comporta come un'elemento neutro, mentre a $b = 0$ offre un'alta impedenza, effettivamente scollegando l'uscita. La tabella di verità corrispondente sarà:

b	x	z
1	0	0
1	1	1
0	-	Hi-Z

Notiamo che il valore Hi-Z (alta impedenza) non è un valore logico: ciò che esce da una porta in stato Hi-Z viene interpretato come un filo staccato dal resto della rete. Ogni porta logica gestisce poi questa situazione secondo le sue specifiche di realizzazione, restando comunque attaccata sia a V_{cc} che a massa, e quindi non in uno stato HiZ.

1.2.1 Multiplexer decodificati

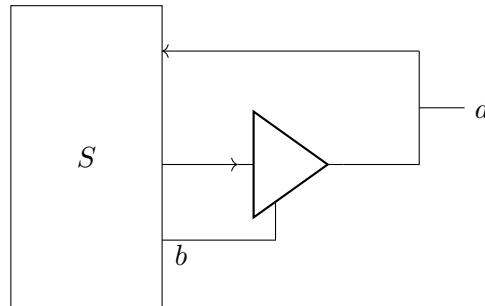
Un componente realizzato attraverso le porte tri-state è il multiplexer decodificato:



Questo componente offre alta indipendenza a tutte le variabili x_i in ingresso tranne una, quella all'indice j , selezionata attraverso una variabile di comando b_j .

1.2.2 Linea di ingresso/uscita

Si usano le porte tri-state per permettere a componenti di comunicare su linee di ingresso/uscita, ad esempio con la memoria. In questo caso, si biforca la linea, ammettendo la linea in entrata così com'è, e mettendo una porta tri-state nella linea in uscita.



Così, quando il componente S vuole comunicare con l'esterno, imposta l'enabler b a 1 e mette sulla linea d ciò che vuole comunicare. Altrimenti tiene b a 0 e ascolta ciò che arriva su d .

Se il componente S comunica con un altro componente T , questi dovranno impostare alternativamente i loro enabler b_S e b_T a 1 e 0, scambiandosi messaggi sulla linea d .

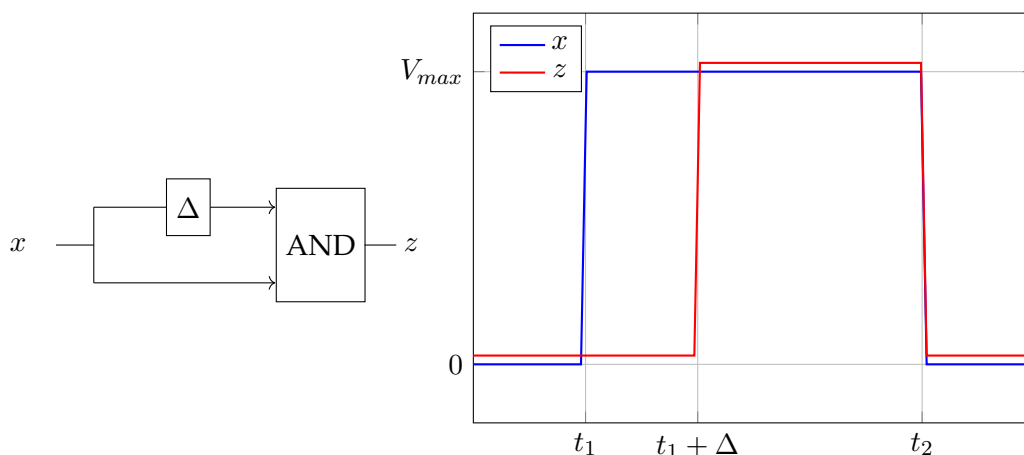
1.3 Circuiti di ritardo e formatori di impulso

A volte bisogna trattare di **segnali**. In questo caso si usano gli elementi neutri Δ (realizzati spesso con numeri pari di invertitori), che sappiamo porre un **ritardo simmetrico** agli ingressi, dove simmetrico significa identico sulle transizioni $0 \rightarrow 1$ e $1 \rightarrow 0$.

Potrebbe essere utile avere circuiti con ritardi **asimmetrici**, cioè variabili sulle transizioni $0 \rightarrow 1$ e $1 \rightarrow 0$. Indichiamo questi componenti come Δ^+ .

1.3.1 Circuito di ritardo sul fronte di discesa

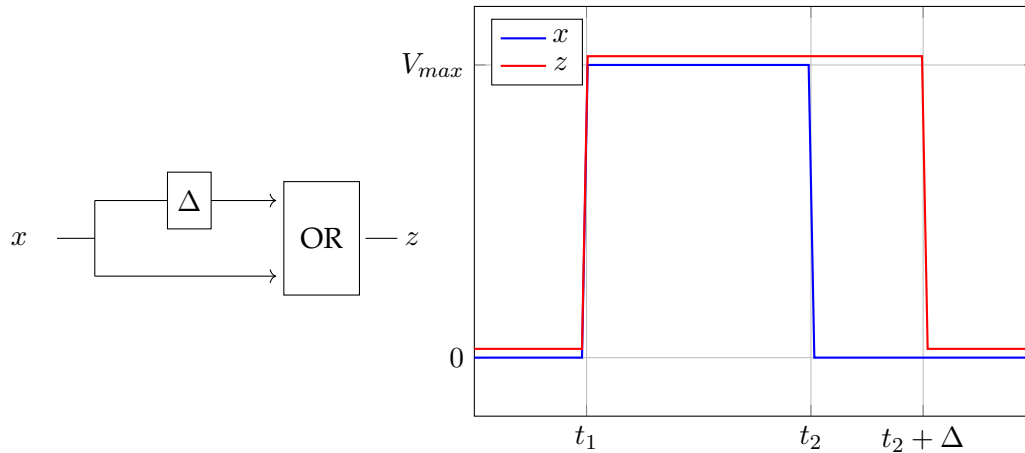
Collegando un neutro Δ assieme al segnale stesso ad una porta AND, si ottiene un'andamento del tipo:



Nello specifico, transizionando da $1 \rightarrow 0$, si ha che il primo ingresso che va a 0 porta a 0 l'uscita. C'è un ritardo piccolo da parte della porta AND. Quando invece si transiziona da $0 \rightarrow 1$, si ha che il secondo ingresso che va a 1 (quello che passa da Δ) porta a 1 l'uscita. C'è un ritardo grande da parte del Δ e della porta AND.

1.3.2 Circuito di ritardo sul fronte di salita

Allo stesso modo, collegando un neutro Δ assieme al segnale stesso ad una porta OR, si ottiene un'andamento del tipo:

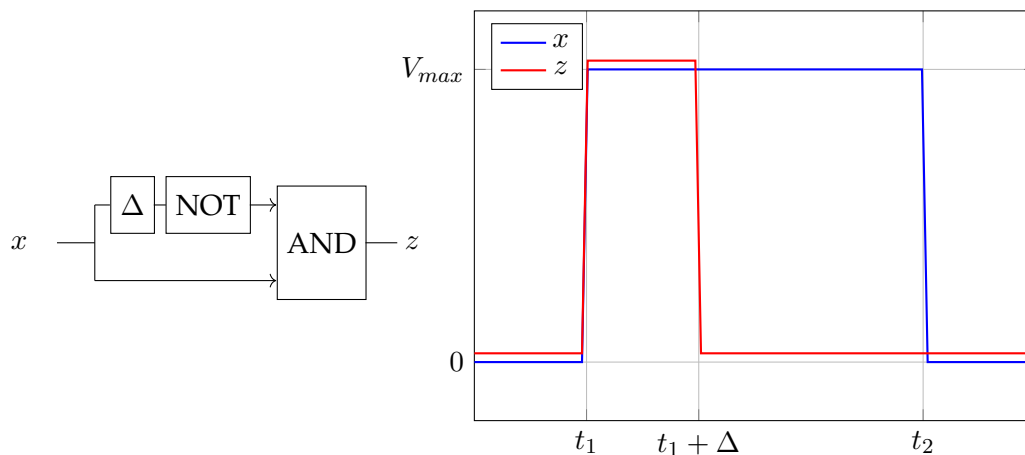


Nello specifico, transizionando da $0 \rightarrow 1$, si ha che il primo ingresso che va a 1 porta a 1 l'uscita. C'è un ritardo piccolo da parte della porta OR. Quando invece si transiziona da $1 \rightarrow 0$, si ha che il secondo ingresso che va a 0 (quello che passa da Δ) porta a 0 l'uscita. C'è un ritardo grande da parte del Δ e della porta OR.

1.3.3 Formatore di impulso sul fronte di salita

I formatori di impulso sono reti combinatorie che generano in uscita un **impulso** di durata nota. Si indicano con P^+ .

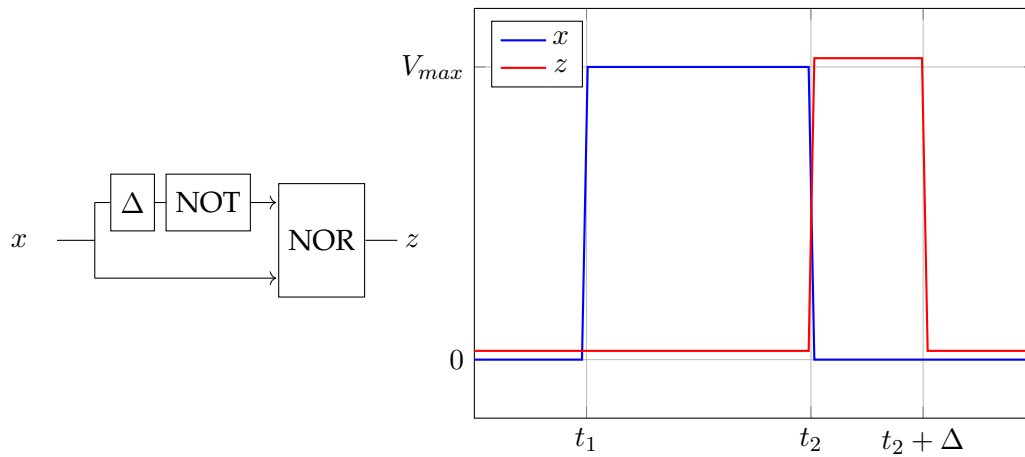
Si crea un formatore di impulso sul fronte di salita collegando la negazione di un Δ e il segnale stesso ad una porta AND, cioè:



Nello specifico, transizionando da $0 \rightarrow 1$, si ha che il segnale va a 1, attivando la AND (l'ingresso dalla NOT era già attivo). Dopo il ritardo Δ , NOT torna a 0, e quindi l'uscita della AND va a 0. Si ha quindi un'impulso di durata del ritardo Δ . Transizionando da $1 \rightarrow 0$, invece, si ha che il segnale "ancora" istantaneamente l'uscita della AND a zero, ergo non si hanno altri artefatti.

1.3.4 Formatore di impulso sul fronte di discesa

Si crea un formatore di impulso sul fronte di discesa collegando la negazione di un Δ e il segnale stesso ad una porta NOR, cioè:



Nello specifico, transizionando da $0 \rightarrow 1$, si ha che il segnale va a 1, ergo la NOR resta a 0 (l'ingresso dalla NOT era già attivo, e due ingressi attivi sono sempre 0 della NOR). Dopo il ritardo Δ , l'uscita della NOT torna a 0, così che quando si stacca il segnale, per una durata Δ entrambe le linee in entrata alla NOR vanno a 0, e quindi questa va a 1. Dopo il ritardo Δ , NOT torna a 0, e quindi l'uscita della AND va a 0. Si ha quindi, ancora una volta, un'impulso di durata del ritardo Δ .