Azzolini Riccardo 2020-05-12

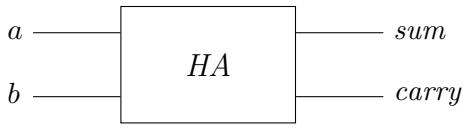
Esempio di applicazione della logica proposizionale classica – Half-adder

1 Introduzione

Per concludere la parte relativa alla logica proposizionale classica, si presenta un esempio in cui i concetti visti finora vengono utilizzati per verificare la progettazione di un semplice circuito logico.

2 Half-adder: specifica

Si vuole progettare un circuito logico che implementi un *half adder*. Esso ha due input, a e b, e due output, *sum* (la somma) e *carry* (il riporto):



Il suo comportamento è specificato da due funzioni booleane, chiamate anch'esse *sum* e *carry* poiché indicano i valori degli omonimi output al variare di a e b. Tali funzioni sono descritte dalla seguente tavola di verità:

a	b	*sum*	*carry*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Per i teoremi di completezza funzionale, si sa che è possibile scrivere delle formule logiche che descrivano il comportamento di queste due funzioni booleane.

Un modo per ottenere tali formule sarebbe costruire le DNF / CNF come indicato dalle dimostrazioni dei teoremi di completezza funzionale; in alternativa, si può ottenere una

formula più semplice ed espressiva osservando la tavola di verità e scegliendo, in base a essa, i connettivi più opportuni:

$$HA_{SPEC}(a, b, sum, carry) = (carry \leftrightarrow a \wedge b) \wedge (sum \leftrightarrow \neg(a \leftrightarrow b))$$

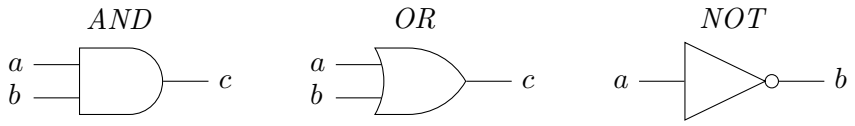
Infatti:

- il connettivo $\leftrightarrow$ corrisponde all'equivalenza tra funzioni booleane, cioè, date una valutazione v e due formule X e Y, $v \models X \leftrightarrow Y$ se e solo se $v(X) = v(Y)$;
- *carry* è equivalente a $a \wedge b$;
- *sum* vale 1 se e solo se a e b hanno valori diversi, cioè se e solo se è vera la formula $\neg(a \leftrightarrow b)$.

Osservazione: Tipicamente, una funzione booleana viene rappresentata da una formula nella quale gli argomenti della funzione corrispondono a variabili proposizionali, mentre il risultato è dato direttamente dal valore di verità assunto dalla formula. La formula appena mostrata, invece, contiene variabili proposizionali sia per gli input che per gli output, e descrive le funzioni booleane *sum* e *carry* nel senso che è verificata se e solo se la combinazione di valori di input e output è "corretta". Così facendo, si possono descrivere più funzioni booleane con una sola formula.

3 Componenti

Le componenti che verranno utilizzate per realizzare il circuito (il quale, se progettato correttamente, dovrà avere lo stesso comportamento delle funzioni booleane riportate sopra) sono le seguenti porte logiche,

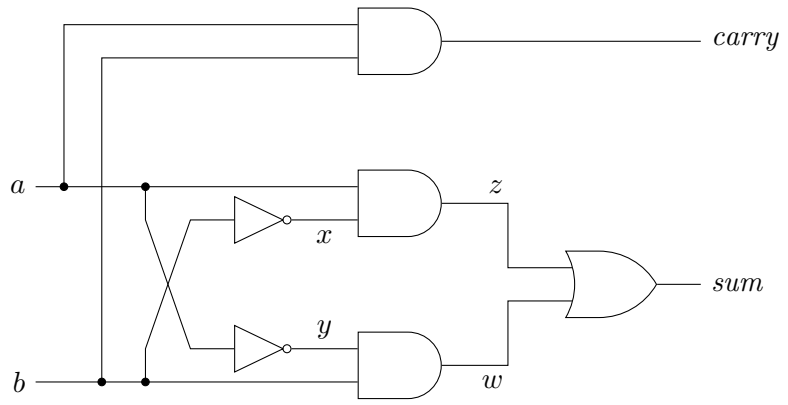


che sono descritte dalle formule

$$AND(a, b, c) \equiv (c \leftrightarrow a \wedge b)$$
$$OR(a, b, c) \equiv (c \leftrightarrow a \vee b)$$
$$INV(a, b) \equiv (b \leftrightarrow \neg a)$$

4 Implementazione

L'half-adder viene implementato mediante il seguente circuito:



Avendo caratterizzato le porte logiche usate in questo circuito con delle formule che mettono in relazione input e output di ciascuna porta, il comportamento dell'intero circuito può essere descritto istanziando tali formule con i "canali" di input e output opportuni:[1]

$$HA_{IMPL}(a, b, sum, carry) \equiv AND(a, b, carry) \wedge INV(b, x) \wedge AND(a, x, z) \\ \wedge INV(a, y) \wedge AND(b, y, w) \wedge OR(z, w, sum)$$

Sostituendo le specifiche delle componenti con le formule che le rappresentano, si ottiene:

$$HA_{IMPL}(a, b, sum, carry) \equiv (carry \leftrightarrow a \wedge b) \wedge (x \leftrightarrow \neg b) \wedge (z \leftrightarrow a \wedge x) \\ \wedge (y \leftrightarrow \neg a) \wedge (w \leftrightarrow b \wedge y) \wedge (sum \leftrightarrow z \vee w)$$

5 Formalizzazione della verifica di correttezza

Adesso, si vuole dimostrare la *correttezza* del circuito progettato (cioè che esso rispetta effettivamente la specifica dell'half-adder), utilizzando gli strumenti della logica proposizionale classica.

Formalmente, la correttezza dell'implementazione è verificata se si dimostra l'equivalenza logica tra le formule che descrivono l'implementazione e la specifica:

$$HA_{IMPL}(a, b, sum, carry) \equiv HA_{SPEC}(a, b, sum, carry)$$

[1] Una descrizione del genere potrebbe essere generata automaticamente da appositi strumenti software.

Siccome il problema può essere ridotto a quello della verifica di una tautologia, è possibile utilizzare un calcolo (indicato in seguito con $\vdash$) per svolgere la dimostrazione:

$$\begin{aligned} Impl \equiv Spec &\iff \models Impl \leftrightarrow Spec \\ &\iff \models (Impl \rightarrow Spec) \wedge (Spec \rightarrow Impl) \\ &\iff \vdash (Impl \rightarrow Spec) \wedge (Spec \rightarrow Impl) \end{aligned}$$

Osservazione: In realtà, ci si potrebbe accontentare di mostrare anche solo che $Impl \models Spec$, ovvero che, ogni volta che l'implementazione dà un risultato, questo corrisponde alla specifica.