Esperienza 13: Semaforo

Gruppo BN
Lisa Bedini, Federico Belliardo, Marco Costa

1 maggio 2017

1 Scopo dell'esperienza

Lo scopo dell'esperienza realizzare un semaforo come macchina a stati finiti tale che

- nello stato ENABLED esegua un ciclo in cui si abbiano accesi (per la durata di un colpo di clock e nel seguente ordine) Led verde, Led verde e giallo, Led rosso.
- nello stato NOT ENABLED faccia lampeggiare il Led giallo (sincronamente col clock).

Il semaforo va realizzato sia tramite circuiti integrati, sia programmando Arduino.

2 Materiale a disposizione

- SN7474 Dual D-FlipFlop
- SN74LS94 Quad NAND gate
- SN74LS74 Dual D-Latch
- SN74LS86 Quad XOR gate
- DIP switch
- 4 LED

3 Stato Enabled

Per realizzare il solo stato enabled abbiamo optato per una macchina di Moore non essendoci alcun tipo di input. In figura 1 abbiamo disegnato le transizioni e la codifica dei vari stati. Abbiamo deciso di usare solo 2 FF in quanto due bit erano sufficienti per codificare i tre stati richiesti. Indicheremo SEMPRE (anche nei punti successivi della relazione) Q_1 il bit pi significativo, mentre Q_0 sarmpre il bit meno significativo. Abbiamo codificato gli stati in modo che i Led verde e giallo siano pilotati da due bit distinti, rispettivamente Q_1 e Q_0 nel nostro caso. Cos sar facile in seguito poter far lampeggiare il solo Led giallo. Si osservi inoltre che lo stato 01 sociato al solo Led giallo acceso: questo puadere solo nella eventualite i FF si accendano in questo stato. Salvo questa eventualito stato 01 non compare pi nei cicli successivi. Le tabelle di transizione sono riportate in tabella **??**.

$Q_{1,n}$	$Q_{0,n}$	$Q_{1,n+1}$	$Q_{0,n+1}$	LV	LG	LR
0	0	1	0	0	0	1
0	1	0	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	1	1	0

Tabella 1: Tabella di veritlla funzione di transizione fra stato n e il successivo $n+1$ dei due D-FF

Con un rapido studio tramite mappe di Karnaugh ci si convince facilmente che ponendo i due don't care[1] pari a 0 si ottiene che la funzione richiesta $Q_{1,n+1} = \bar{Q}_{0,n}$ $Q_{0,n+1} = \bar{Q}_{0,n}1,n(0)$Cos ha che lo stato non richiesto 01 ("solo giallo acceso"), transisce nello stato 00 ("solo rosso acceso").

[1]cito assegnare a questi don't care tutte le combinazione tranne 01: in questo caso infatti la macchina resterebbe perpetuamente in questo stato

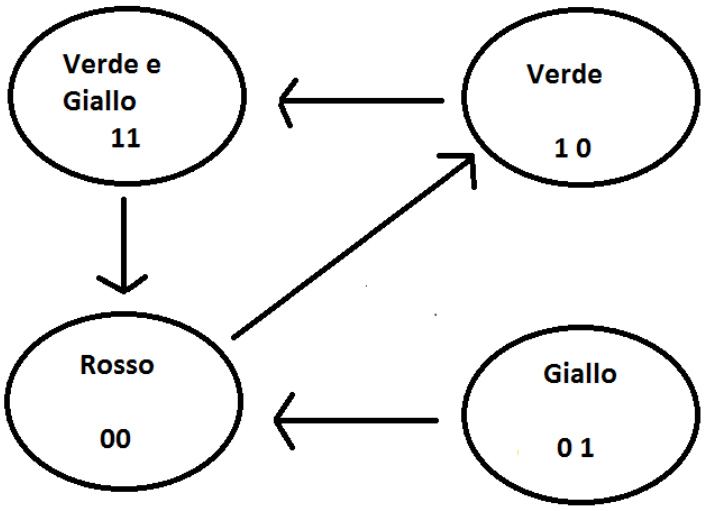

Figura 1: Diagramma dello stato Enabled. Le tranizioni avvengono aad ogni colpo di clock

E	$Q_{1,n}$	$Q_{0,n}$	$Q_{1,n+1}$	$Q_{0,n+1}$	LV	LG	LR
0	0	0	0	1	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	1	1	0	0
0	1	1	0	0	1	1	0
1	0	0	1	0	0	0	1
1	0	1	0	0	0	1	0
1	1	0	1	1	1	0	0
1	1	1	0	0	1	1	0

Tabella 2: Matrice di transizione e valori di veritr le uscite nel caso di semaforo completo.

Una volta codificati gli stati, abbiamo assegnato alle uscite L_V (Led Verde), L_G (Led giallo), L_R (Led rosso) i seguenti valori:

$$L_V = Q_1 \qquad L_G = Q_0 \qquad L_R = \bar{Q}_0 \cdot \bar{Q}_1 \tag{1}$$

Questo era in effetti il modo pi semplice per realizzare i collegamenti fra i FF e le uscite: per come erano stati codificati gli stati, i led verde e giallo sono pilotabili direttamente dal rispettivo bit, mentre per il Led rosso abbiamo scelto l'unica funzione che valga 1 solo sullo stato 00. In effetti avremmo pure potuto prendere come funzione per pilotare il Led rosso $\bar{Q}_1$, a patto pervere 01 stato in cui sia il giallo che il rosso sono accesi. Abbiamo preferito usare una porta in pi qui e avere la possibilit controllare il Led giallo direttamente.

4 Semaforo completo

Abbiamo optato di usare una macchina di Mealy per realizzare il semaforo completo. Si ntenuta la stessa codifica degli stati in termini di bit utilizzata precedentemente. In questo modo la matrice di transizione nello state Enabled entica a quella precedente. Abbiamo chiamato E il valore logico dell'enable. Abbiamo deciso di scegliere lo stato Enabled attivo alto (ossia quando $E = 1$):In tabella 2 abbiamo la matrice di transizione e i valori delle uscite implementata (i don't care sono stati assegnati) Nel caso Not Enabled, abbiamo deciso di mantenere lo stato 01 ad essere con solo il Led giallo acceso. Abbiamo deciso di farlo transire verso 10 : in modalitabled lo stato 00 aveva solo il Led rosso acceso, pertanto con una aggiunta di un AND con il bit E si usciti a mantenere il circuito gintato in precedenza. Le transizioni degli altri stati (10 e 11) sono state scelte in modo che le relative mappe di Karnaugh risultassero le pi semplici possibili compatibilmente col fatto che gli stati 10 e 11 non devono permanere mai. Abbiamo costenuto come funzioni logiche:

$$Q_{0,n+1} = (Q_{1,n} \cdot \bar{Q}_{0,n}) + (\bar{E} \cdot \bar{Q}_{1,n}) \qquad Q_{1,n+1} = E \cdot \bar{Q}_{0,n} \tag{2}$$

Con lo stesso modo si sono scelte le funzioni di output dei Led relativamente ai soliti stati indesiderati 11 e 10. Stavolta i valori logici degli output sono dei veri don't care. Abbiamo ottenuto le seguenti equazioni:

$$LV = Q_1 \qquad LG = Q_0 \qquad LR = E \cdot (\bar{Q}_1 \cdot \bar{Q}_0) \tag{3}$$

Si osservi che non abbiamo alterato le connessioni delle uscite realizzate nel punto precedente, e ci si mitato ad aggiungere un AND con E in un solo punto. Si osservi che in questo caso abbiamo implementato un meccanismo di abilitazione asincrona: non appena si cambia lo stato di E, indipendentemente dal fronte del clock, si ha che il semaforo cambia stato di abilitazione. Il Led Rosso llegato all'enable non tramite altri D-Latch, quindi se ceso mentre E passa da 1 a 0, avviene immediatamente il cambio dei valori logici delle uscite previste nel caso di $E = 0$, ossia si passa all'uscita $LG = 1$ senza aspettare il clock. Per le altre uscite non ci sono problemi in quanto sono collegate ad E solo tramite i vecchi D-Latch. Per realizzare un meccanismo di abilitazione sincrona, ossia per fare in modo che il semaforo registri solo cambiamenti di E che avvengono sul fronte alto del clock, si llegato E direttamente ad un terzo D-Latch, come in figura**??**.

5 Semaforo con Arduino

Adesso vogliamo implementare lo stesso semaforo programmando Arduino. Abbiamo optato per una macchina di tipo Moore, pi semplice concettualmente da realizzare. Abbiamo collegato le uscite di Arduino 9, 10, 11 tramite il buffer rispettivamente ai Led Verde, Giallo, Rosso, inserendo resistenza da 330Ω su ciascuno per limitare la corrente. Queste uscite sono state dichiarate come OUTPUT nel programma utilizzato. L'enable ato collegato all'uscita 8, e nel programma chiarato come INPUT-PULLUP (ossia alto se interruttore aperto e basso se chiuso). Dato che con Arduino si hanno in genere un numero di bit a sufficienza, nello scriere il programma abbiamo assegnato ad ogni LED un bit diverso. In questo modo si ha uno stato per ogni LED acceso, pi lo stato spento e lo stato in cui Verde e Giallo sono accesi. Abbiamo considerato solo i vari stati che ci servono: quelli indesiderati non si presentano mai dato che l'inizializzazione (nello stato in cui tutto ento) viene fatta da noi nel programma e non casualmente come nel caso dei FF. Il funzionamento del programma mima il comportamento della macchina a stati finiti precedentemente costruita: legge lo stato in cui si trova attualmente, legge il valore di Enable (attivo alto) e a quel punto decide in che stato finire. Il vantaggio rispetto a prima e ssibile settare il tempo di permanenza di uno stato in modo indipendente dagli altri. Dopo viene retto il valore di Enable (abilitazione sincrona) e vengono cambiati i valori di output secondo lo stato in cui si trova la macchina (che essendo di tipo Moore, ha ad ogni stato associato un ben definito valore di output)

6 Conclusioni