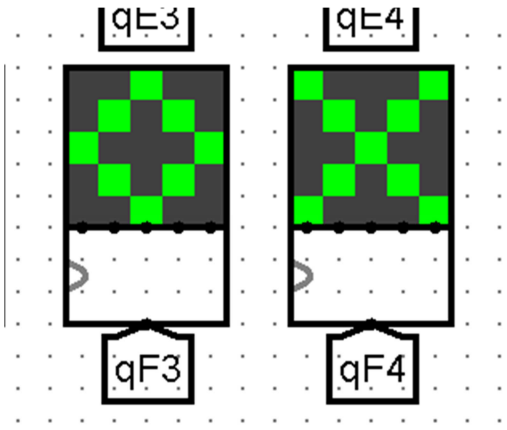
Progetto Architettura degli Elaboratori I

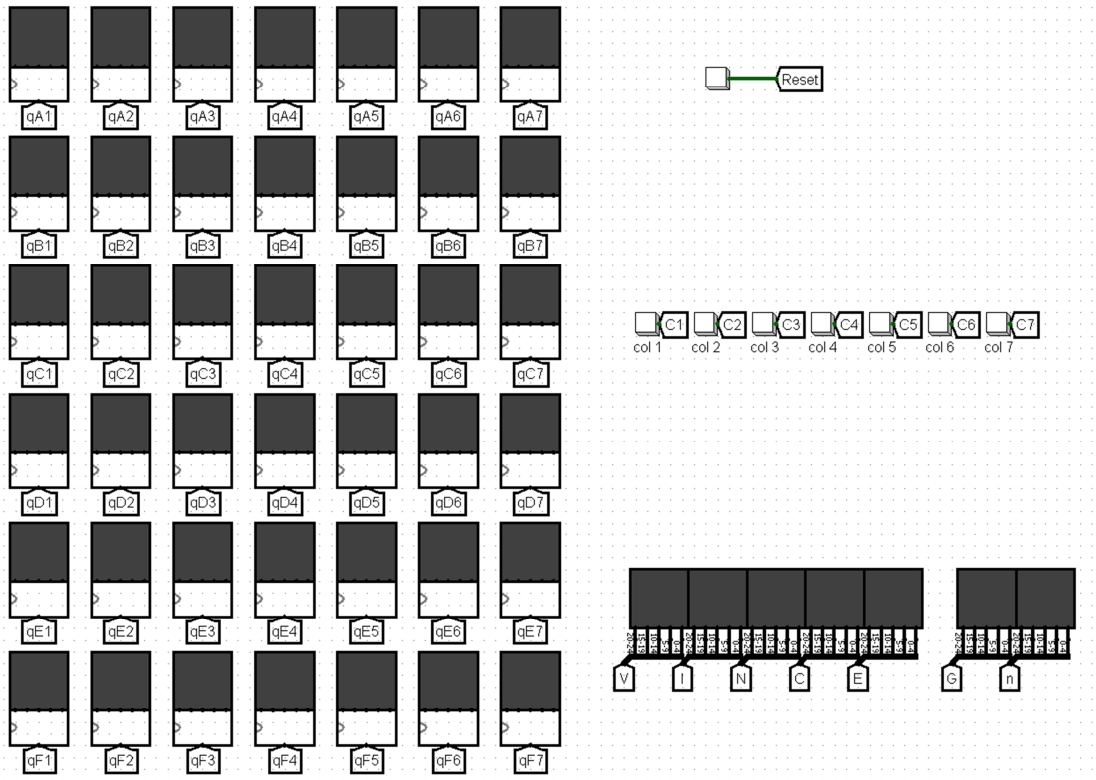
FORZA 4

Il progetto prevede una simulazione di più partite del gioco "Forza 4" mediante la scelta da parte dell'utente della colonna in cui desidera inserire la propria "tessera".

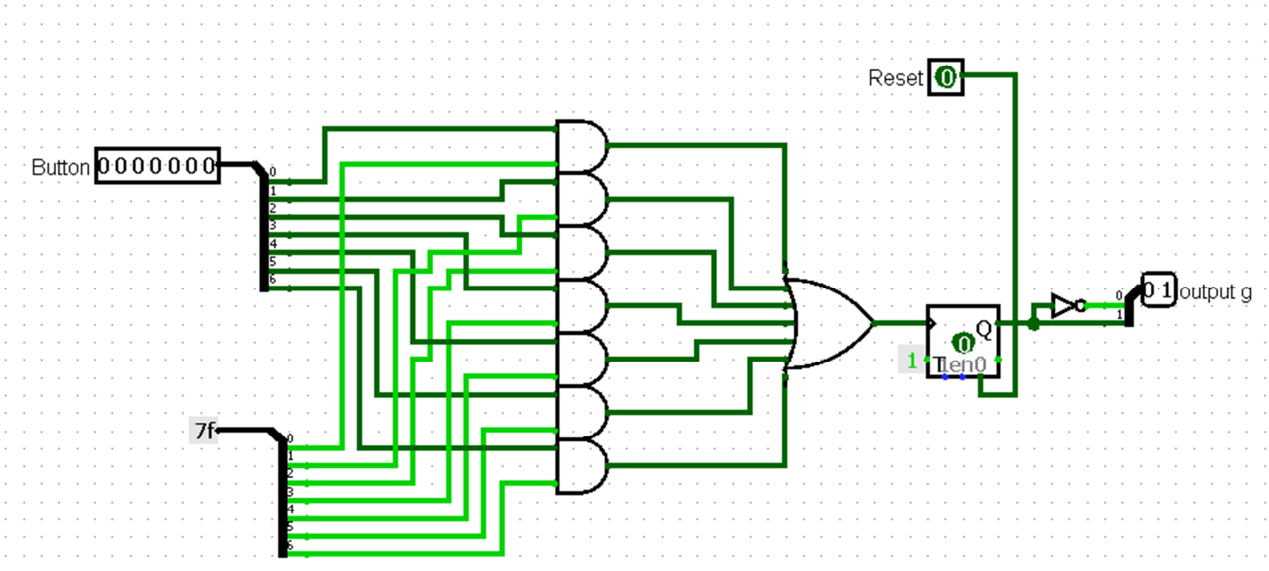


Il gioco gestisce una partita tra due giocatori, ognuno identificato da una "tessera" differente da concatenare orizzontalmente, verticalmente o diagonalmente per formare una combinazione di 4 tessere consecutive. L' aspetto grafico è gestito tramite una serie di Led Matrix in formato 5x5 e organizzati su una tabella 6x7, in aggiunga vi sono una serie di ulteriori Led Matrix per la visualizzazione di un messaggio indicante l'ipotetico vincitore di fine partita.
L'input è gestito tramite 8 bottoni intaragibili, 7 rappresentanti le colonne della tabella e uno che assume lo scopo di permettere l'annullamento della partita e l'inizio di una nuova (bottone Reset).

Dettagli:
- Input: 7 bottoni rappresentanti le 7 colonne del gioco, assumono il valore 1 al momento della pressione. La tessera sarà inserita automanticamente nella riga più bassa possibile della colonna per simulare il gioco reale;
- Led Matrix: matrice 6x7, ogni led è legato ad un registro che ne stabilisce lo stato;
- Registri: 7 sotto-circuiti Registro, uno per colonna, ognuno contenente 6 registri corrrispondenti alle righe. All'inserimento di un valore, verrà automaticamente incrementato quello orrispondente alla riga più bassa possibile;
- Contatore: 7 contatori legati alla pressione dei bottoni, una pressione aumenta il loro valore di uno. Sono necessari per l'inserimento del valore del giocatore nel registro corernto, in quanto svolgono il compito di selectors nei decoder presente in Registro;



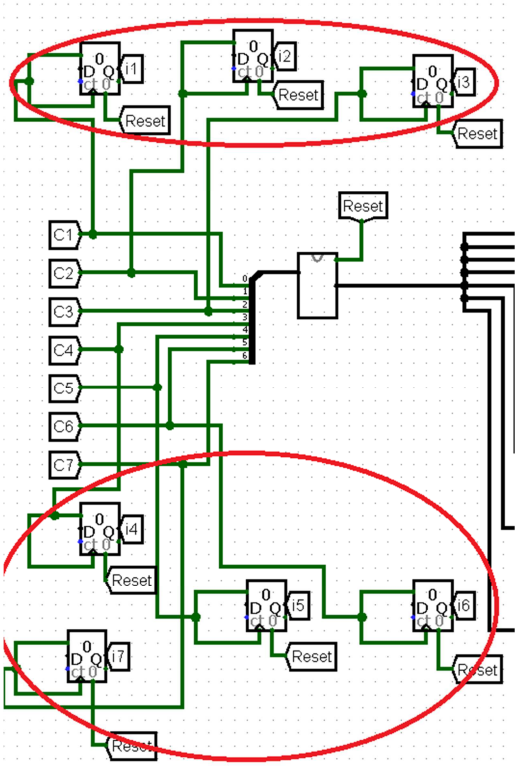
Sotto-circuiti e dettagli



- Alterna

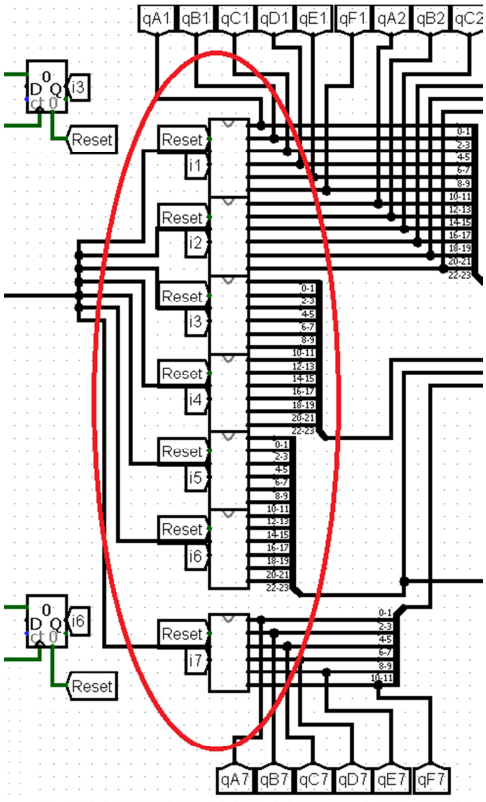
Qualisiasi cambiamento di stato di un bottone causa un cambiamento nel flipflop, ciò determinerà una variazione nell'output, in particolare, essendo l'output la rappresentazione del giocatore che sta svolgendo la sua mossa, un cambiamento di stato determina il cambio del turno.
Reset ha dunque lo scopo di ripristinare il tutto allo stato iniziale per permettere l'avvio di una nuova partita.

- Contatori

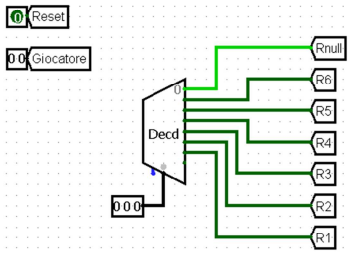


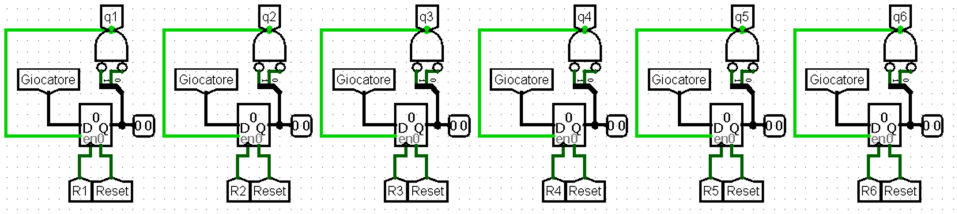
Svolgono la funzione di selettori per i registri, nel dettaglio, ad ogni click del bottone, il loro valore incrementerà, spostando il target di scrittura nel registro di una riga in su, evitando tentativi riscrittura su registri già inizializzati. Il ritorno dello stato a 0 non mette a rischio la solidità della struttura in quanto i register non sono sovrascrivibili.

- Registro

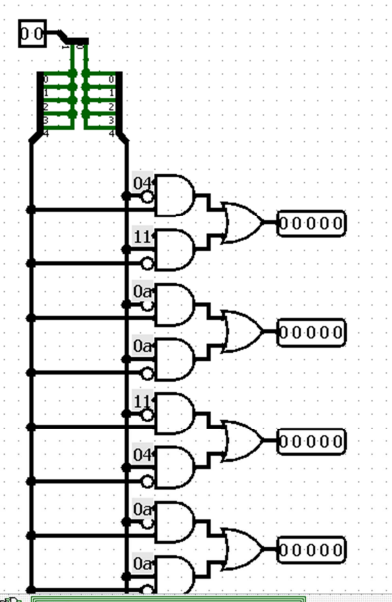


Formato da un Decoder e 6 register, riceve Reset, in caso si debba eliminare i dati registrati, l'output di Alterna che diventerà lo stato del registro selezionato ed infine riceve il corrispondente contatore che fungerà da insieme di bit di selezione per il Decoder.
Ogni register da poi come output il proprio stato, che oltre che essere un input necessario per il sottocircuito successivo, verrà usato tramite Player (vedi sotto) per la rappresentazione attraverso Led Matrix dedll'aspetto grafico del programma.
Poichè le caselle da gestire sono 42, il limite di 32 bit imposto dalla macchina mi ha costretto a ricorrere a più di questi sottocircuiti e non ad uno singolo.
A destra è visibile il suo utilizzo all'interno del main, mentre sotto, la sua composizione



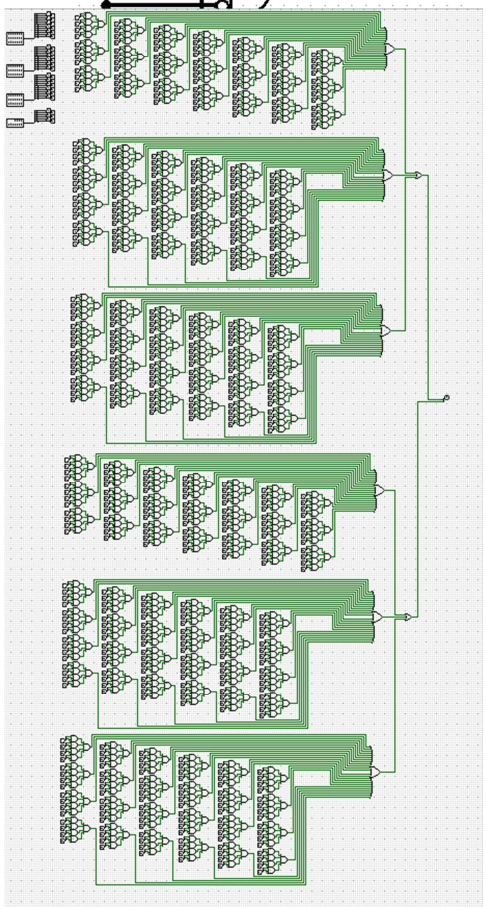


- Player

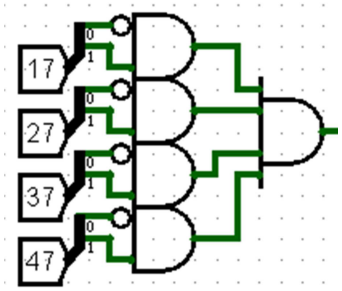


Un seplice circuito che riceve come input un codice giocatore e in base ad esso rilascia 5 output da 5 bit che inseriti in un Led Matrix danno ad esso la forma della "tessera" corrisondente al giocatore.
Ciò benchè inutile ai fini del circuito per il calcolo della vittoria è essenziale per lo svolgimento di una partita. Le costanti applicate sono state opportunatamente impostate in modo da dare nel caso del giocatore 1 (cod. In bit 10) un rombo, in caso di giocatore 2 (cod. In bit 01) una croce.

- Vittoria



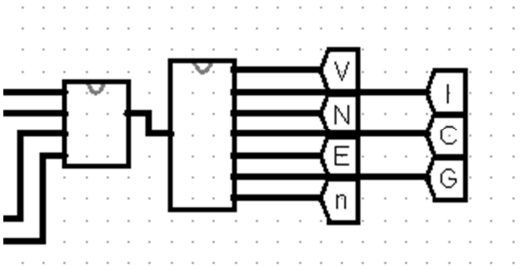
Tale circuito riceve in ingresso tutti i 42 registri formati da 2 bit l'uno, e li ripartisce tramite tunnel, così da permetterne lo studio. Esso è formato da tutte e 69 le possibili combinazioni di vittoria di "forza 4" e in caso dell'avverarsi di una di queste, rilascia il codice di colui che l'ha ottenuto. Più precisamente, ilsottocircuito si divide in 6 gruppi che culminano tutti in un or che verifica, avverarsi di una delle combinazioni, i primi 3 gruppi verificano se la combinazione è stata eseguita del giocatore con codice 10,gli altri 3, il giocatore con codice 01. Questi 3 gruppi per comodità nel svilupparli, sono stati suddivisi in 1 gruppo: combinazioni orizzonali, 2 gruppo, combinazioni verticali, e infine 3 gruppo possibili combinazioni diagonali per la vittoria.
Ho scelto questo tipo di circuito in quanto benchè sia a primo inpatto "brutto" e poco intuitivo, l'ho reputato il più semplice, per aiutarmi a mantenere una chiarezza mentale nel quali fossero i suoi obbiettivi e nel cosa volessi ottenere. Altri possibili metodi come la divisione in altri sottocircuiti per delegare la gestione dei controlli ad esempio li ho trovati meno intuitivi e pericolosi da realizzare in quanto l'eccessiva suddivisione di un operazione potrebbe portare ad errori in fasi di realizzazione.
(A destra, il circuito sviluppato per verificare se una

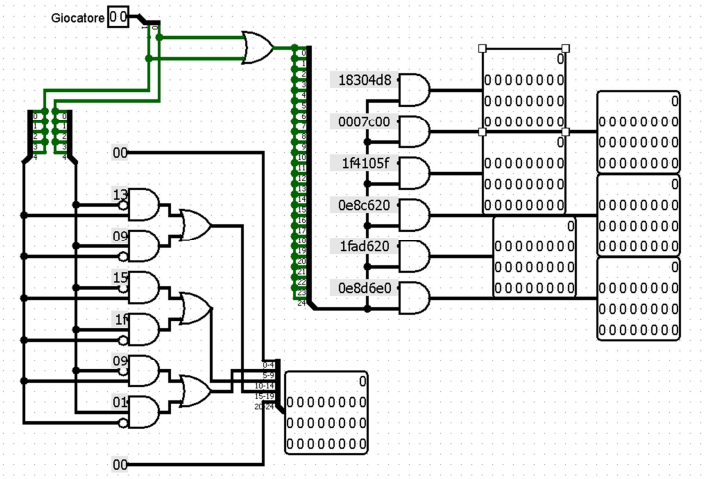


delle combinazioni fosse avvenuta)

- Messaggio

Messaggio riceve come input il valore in output di Vittoria, formato da 2 bit e lo utilizza per determinare che valore dare all'ultimo output, mentre i primi sei infatti sono indipendenti dall'input, l'ultimo assumerà un valore che rappresenta graficamente "1" in caso l'input sia 10, e "2" se l'input è 01.





Conclusioni

Per realizzare questo progetto ho deciso di partire decidendo come gestire i registri e superare il limite dei 32 bit massimi rispetto ai 42 di cui avessi necessità ho ipotizzato e testato la soluzione dei registri comprensivi di decoder da assegnare quali eclusivi alle varie colonne e deciso l'uso di un counter per la gestione del decoder e la simulazione dell'obbligo di assegnare un valore esclusivamente al registro più "basso" possibile. Implementato questo, ho svolto qualche ricerca e optato per la soluzione delle combinazioni di registri per la determinazione del vincitore per i motivi descritti nel rispettivo paragrafo. Infine, ho composto il circuito per la visualizzazione del vincitore per dare maggior peso e caratterizzazione all'aspetto grafico.

Eventuali miglioramenti possono essere una migliore gestione del decoder in modo da rendere non necessario un canale per la fase in cui il counter assume valore 0 ed una diversa implementazione del circuito per la determinazione del vincitore.