SPACE INVADERS

SPACE INVADERS

Il progetto consiste nel simulare il gioco "Space Invaders" riproducendo un giocatore e degli avversari da colpire e sconfiggere per poter conquistare la vittoria.
L'area di gioco consiste in una matrice led da 32colonne x 18righe in cui verranno rappresentati i nemici nella metà superiore, e il giocatore nell'ultima riga.
Si voleva riprodurre il giocatore sotto forma di tre led accesi e consecutivi che possono scorrere verso destra e verso sinistra a seconda dei comandi dell'utente.
L'utente potrà guidare il giocatore tramite la tastiera che permette:
• lo scorrimento verso sinistra
• l'attacco
• lo scorrimento verso destra
(Precedentemente si progettava lo spostamento attraverso i buttons ma per comodità del giocatore si utilizzeranno i comandi da tastiera).
Verranno fatti dei controlli in modo tale da vincolare l'utente a non oltrepassare il lato sinistro o il lato destro arrivando al lato opposto, quindi per arrivare da un estremo all'altro dell'area di gioco, egli dovrà attraversare quest'ultima.
Per gestire la matrice led, sono stati utilizzati tanti splitter quante le colonne, per gestire separatamente la serie di bit che rappresenta ogni colonna e per poter effettuare i controlli necessari per le collisioni.
Vengono utilizzati due splitter connessi, per ogni colonna in modo tale da visualizzare su uno di essi l'attacco del giocatore mentre l'altro verrà utilizzato per generare l'output effettivo, cioè ciò che verrà effettivamente visualizzato dopo aver effettuato i vari controlli:

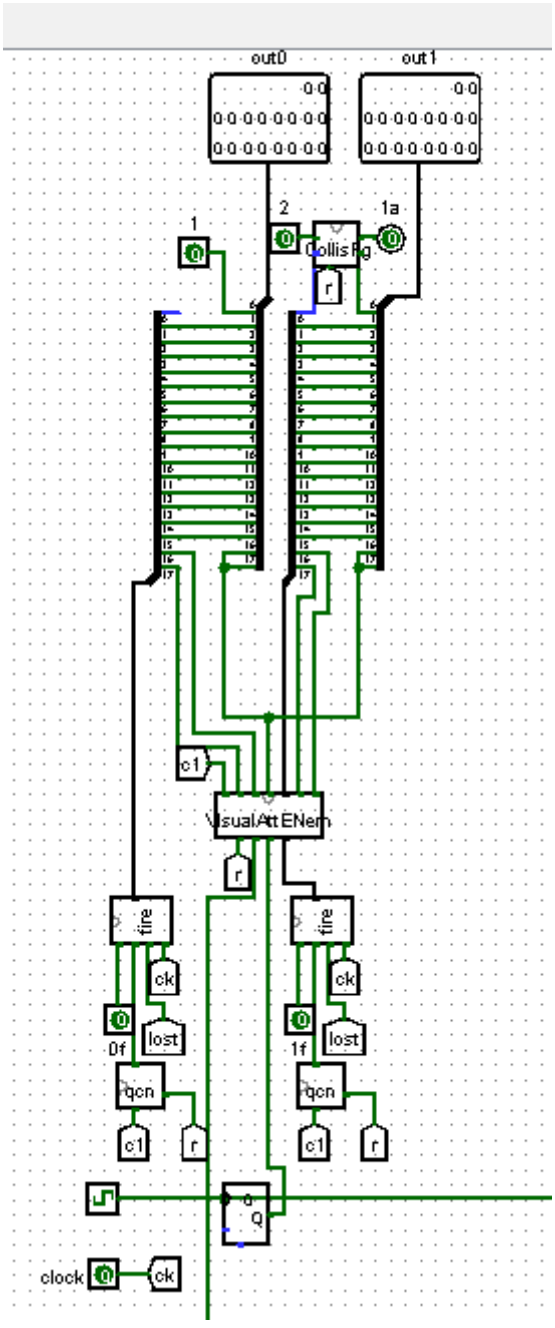
- nel caso in cui il bit scorrevole, si trova nella stessa posizione di uno dei bit accesi, che rappresentano un nemico, allora bisognerà disattivare questo nemico, ovvero rendere i bit presenti nella sua posizione uguali a 0.
- nel caso in cui si presentino bit di attacco del giocatore e del nemico sulla stessa colonna, allora bisognerà far in modo che non si contrastino, ma che scorrino liberamente.
- Nel caso in cui uno bit di attacco del nemico, si trovi nella stessa posizione del giocatore, vuol dire che l'ha colpito e che quindi è stato sconfitto (i controlli di questo genere verranno fatti solo sugli splitter corrispondenti alle colonne in cui sono presenti nemici).
- E' stato utilizzato un generatore casuale di bit per generare casualmente il bit di attacco del nemico.

Il giocatore inizialmente si troverà sull'estremo sinistro dell'area di gioco, e la sua posizione verrà memorizzata di volta in volta in un registro che verrà aggiornato a seconda delle mosse dell'utente e resettato nel caso in cui si spenga e si riaccenda il gioco attraverso il tasto “On-Off”:
• lo scorrimento verso sinistra indica che la posizione del primo led del giocatore sarà posizionato sulla colonna successiva. Viceversa, lo scorrimento verso destra consisterà nel sottrarre una posizione alla posizione attuale ovviamente controllando che il giocatore non vada oltre ai limiti dell'area di gioco.
• l'attacco non modificherà la posizione del giocatore, ma come conseguenza verrà illuminato il led al di sopra del led centrale del giocatore (questo serve a creare l'effetto dell'attacco). Ovviamente per decidere la posizione alla quale illuminare il led, bisognerà memorizzare in un registro la posizione del led centrale e mantenerlo aggiornato per poterne usufruire appena il tasto d'attacco viene premuto.
Per ottenere l'effetto di scorrimento del led verso il lato superiore della matrice, quando verrà premuto il corrispondente tasto, verrà generato un bit, che verrà memorizzato in un registro a scorrimento da 17 bit i quali rappresentano i led di una colonna escludendo la riga in cui è posizionato il giocatore. Il registro genererà un output corrispondente ad un valore binario che rappresenterà una colonna della matrice led in cui deve essere simulato l'attacco. Si utilizza il registro a scorrimento perchè in questo modo il bit equivalente a "1" (led acceso) scorrerà verso la fine del registro in cui l'ultimo flip-flop corrisponde al bit più significativo (così l'ultima posizione del led acceso sarà sull'ultima riga superiore).
Nel momento in cui il giocatore colpisce un nemico oppure viene sconfitto durante la partita, il bit sparisce e anche i nemici vengono “disattivati”.
La generazione del bit scorrevole dell'attacco nemico verrà generata usufruendo nello stesso modo del registro scorrevole. In questo caso sono stati fatti dei controlli sullo stato del nemico (attivo o meno) in modo tale da evitare la generazione dei bit casuali nel caso in cui egli non sia attivo.

Come è stato spiegato precedentemente, non appena un bit di attacco del giocatore si troverà nella stessa posizione del nemico, vuol dire che è avvenuta una collisione. In questo caso verrà generato un bit che non cambierà finché non verrà resettato il gioco e che permetterà la disattivazione del nemico durante questa partita. Sono stati utilizzati i pin “clear” dei flipflop per mantenere a 0 tutto il registro in cui scorreva il bit di attacco del nemico, perché una volta disattivati, non dovranno più attaccare.
I bit di collisione relativi a ogni nemico verranno memorizzati per il calcolo del punteggio.
Il calcolo del punteggio viene effettuato “sfruttando” i ritardi di porta:

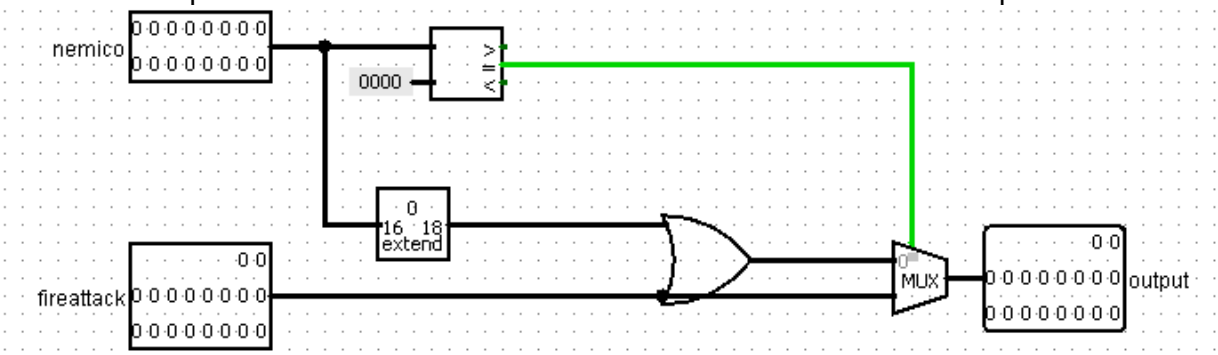
- Se si otterrà almeno una collisione, vuol dire che è morto un nemico e quindi il punteggio aumenterà di 10 punti.
- Non appena si rileva le collisioni totali però non può rimanere sullo stato alto, perché se dovesse avvenire un'altra collisione non lo si rileverebbe più, e il punteggio continuerebbe ad aumentare inutilmente. Si decide quindi di memorizzare in un flipflop SR (S=bit di collisione, R=0), il bit di collisione, così in questo modo la memorizzazione del bit=1 sarà permanente (finché il circuito non verrà resettato). Successivamente si memorizza in un altro flipflop D il contenuto del ff SR che verrà utilizzato come Clear in un successivo flipflop D. La memorizzazione del contenuto dell'SR in un altro ff permette di avere un impulso positivo, (il segnale va a1 ma poi velocemente va a 0).

Simulazione gioco: mantenendo attiva la simulazione, premere On-Off, per far partire o terminare la partita.
Schemi a Blocchi dei vari Componenti: in allegato.
Di seguito verranno riportati i circuiti più importanti del progetto:
-Splitters:



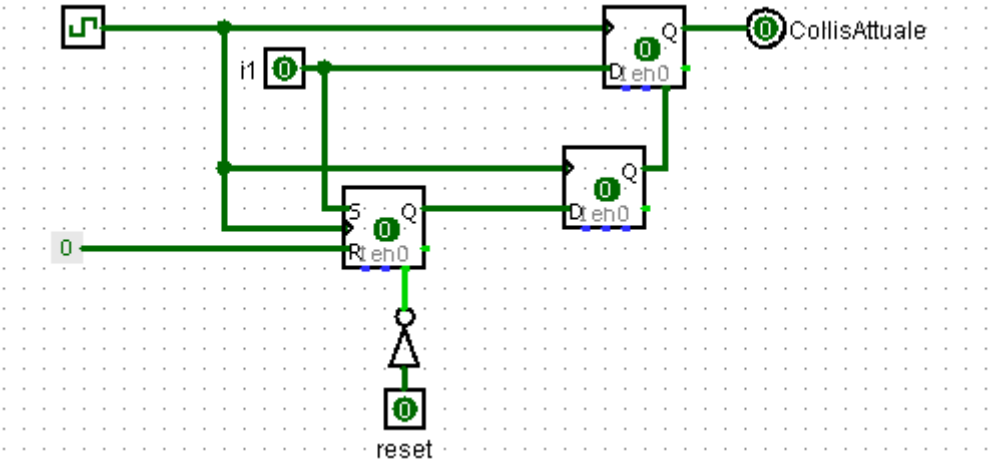
Purtroppo non è possibile visualizzare tutto il circuito contenente tutti gli splitter, perché è molto grande. La foto mostra due coppie di splitter, due per ogni colonna. Su entrambe le colonne avremo un nemico, ma quest'ultimo può attaccare solo sul suo lato destro, quindi sulla seconda colonna. E' per questo motivo che per visualizzare l'attacco sulla prima colonna non vengono fatti controlli per evitare che l'attacco nemico contrasti quello del personaggio. Tutti i controlli precedentemente spiegati sulle collisioni, verranno fatte però sulla seconda colonna tenendo conto solo di alcuni bit della prima.

- Se uno dei bit accesi posizionati per rappresentare il nemico sono accesi, ma nello stesso momento è stato generato il bit di attacco del giocatore su una delle due colonne, allora il nemico verrà disattivato.
- In contemporanea a tutto ciò si visualizza la serie di attacchi utilizzando questo circuito:

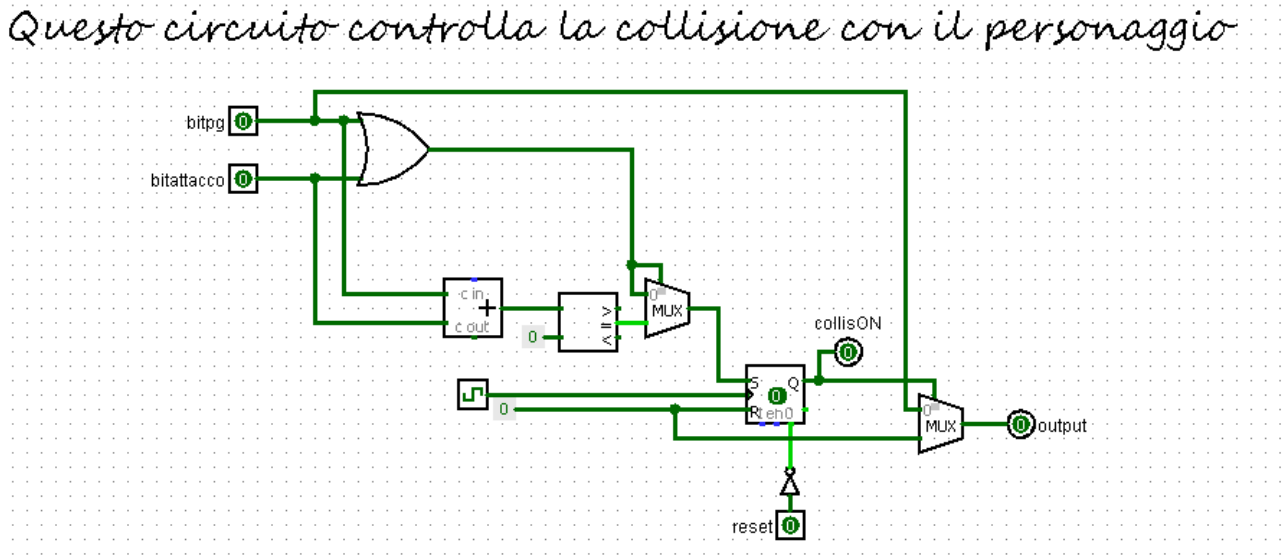


In questo modo i due attacchi non si contrastano.

- Il circuito “qcn” (“Quando colpisco un nemico”) viene utilizzato per settare a 0 il bit di attacco non appena avviene una collisione con il nemico. E' il seguente:



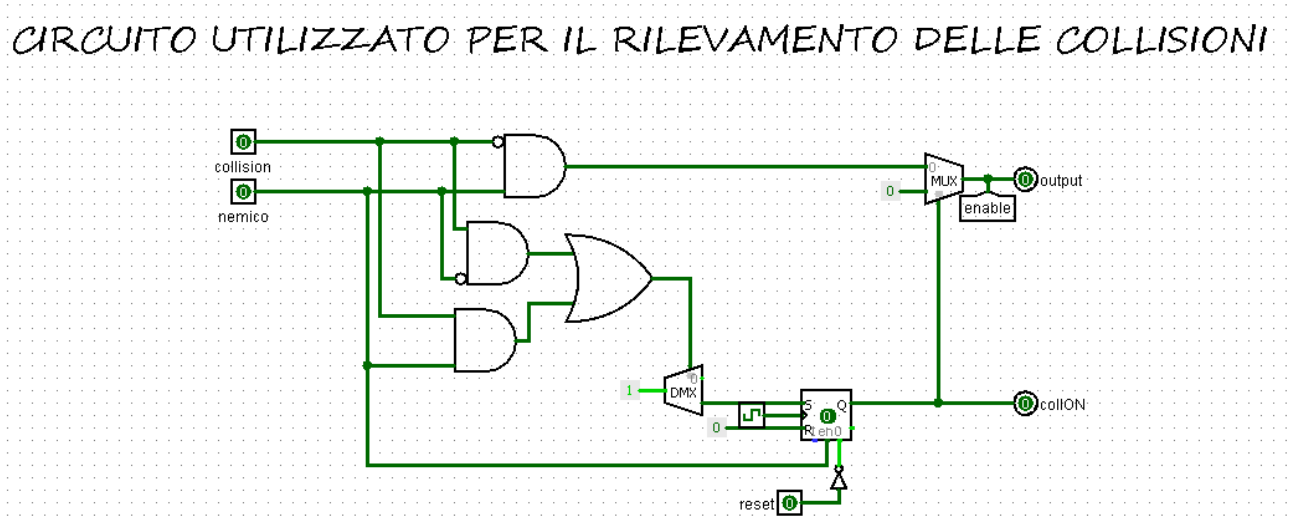
Utilizzando questa tecnica, creo un impulso che utilizzo per resettare il registro scorrevole dell'attacco del personaggio.
Per il punteggio sono stati utilizzati tanti circuiti fatti in questo modo quanto il numero dei nemici (Il circuito è denominato “ControlloreCollisioni”), e non appena si ottiene almeno un impulso (è stata quindi utilizzata la porta or), il punteggio viene incrementato.
-Circuito “CollisioneConPg”:
Permette di ottenere un bit che permette di controllare quando il giocatore viene sconfitto, e quindi spegne i led accesi del giocatore:



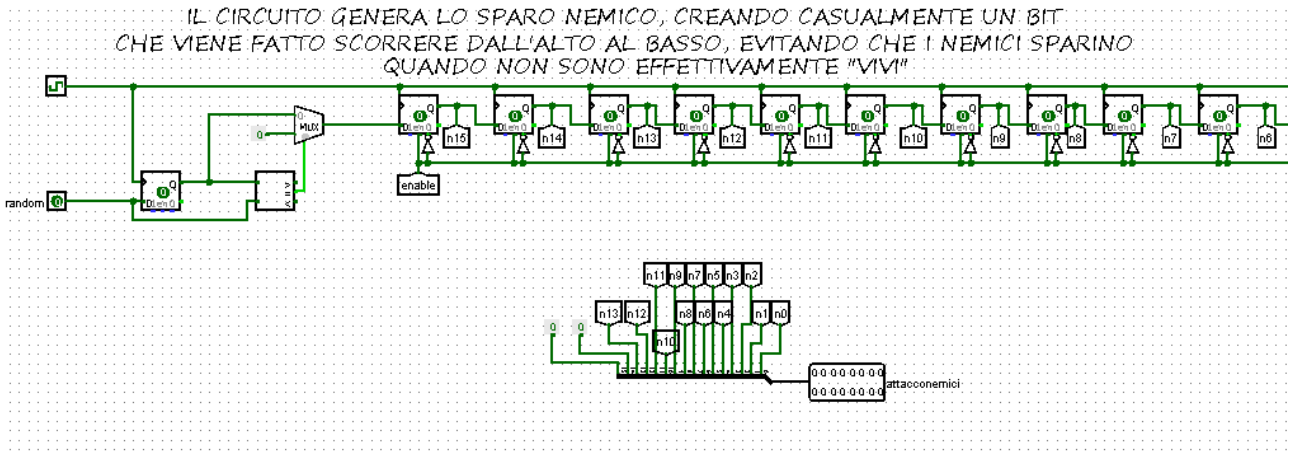
Questo controllo viene effettuato solo sulla colonna in cui il nemico potrà attaccare. Vengono confrontati con 0 i bit di posizione meno significativa degli splitter corrispondenti, dopo essere stati sommati, se il confronto da un risultato positivo allora è avvenuta una collisione e si memorizza permanentemente (finché non viene effettuato il reset) il bit di collisione che viene utilizzato come selettore del multiplixer: se è pari a 1 allora devo spegnere i led del personaggio, altrimenti visualizzo quelli attuali.

-Circuito “CollisioneConNemico”:

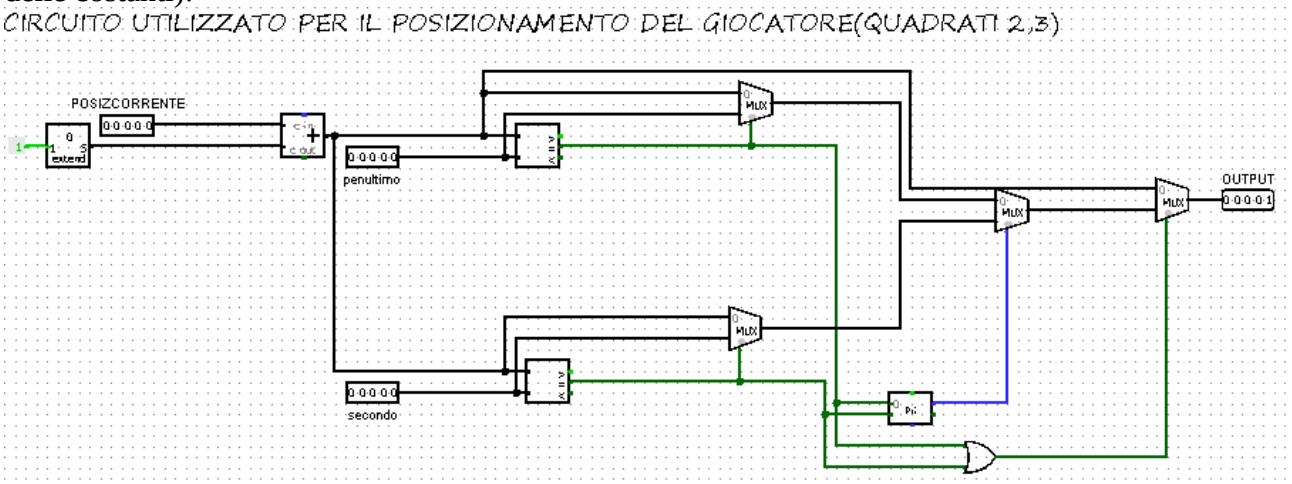
Si confrontano nuovamente due bit, ma questa volta sono quelli posizionati dove è presente il nemico con il bit di attacco del giocatore, non appena avviene la collisione si disattiva il nemico. Questo bit di collisione viene utilizzato come “Clear” nel registro scorrevole dell'attacco del nemico ucciso:



Questo è il registro (parte di esso):



Il confronto effettuato all'inizio permette di non generare due bit consecutivi di attacco.
Un registro molto simile viene utilizzato per la generazione dell'attacco del giocatore (circuito “fire”).
-Circuto “Quadrati2e3DelPg” (2° e 3° led del giocatore):
Il circuito preleva la posizione del primo led, ricava la posizione successiva sommandogli 1 (ne verranno utilizzati due in coda), e controlla che non siano l'ultima posizione che il secondo led può assumere(ovvero la penultima colonna) oppure la prima che egli può assumere(ovvero la seconda colonna). Riguardo al terzo led il funzionamento è analogo, ma l'ultima posizione che può assumere coincide con l'ultima colonna, mentre la prima coincide conla terza colonna (vengono utilizzate delle costanti):

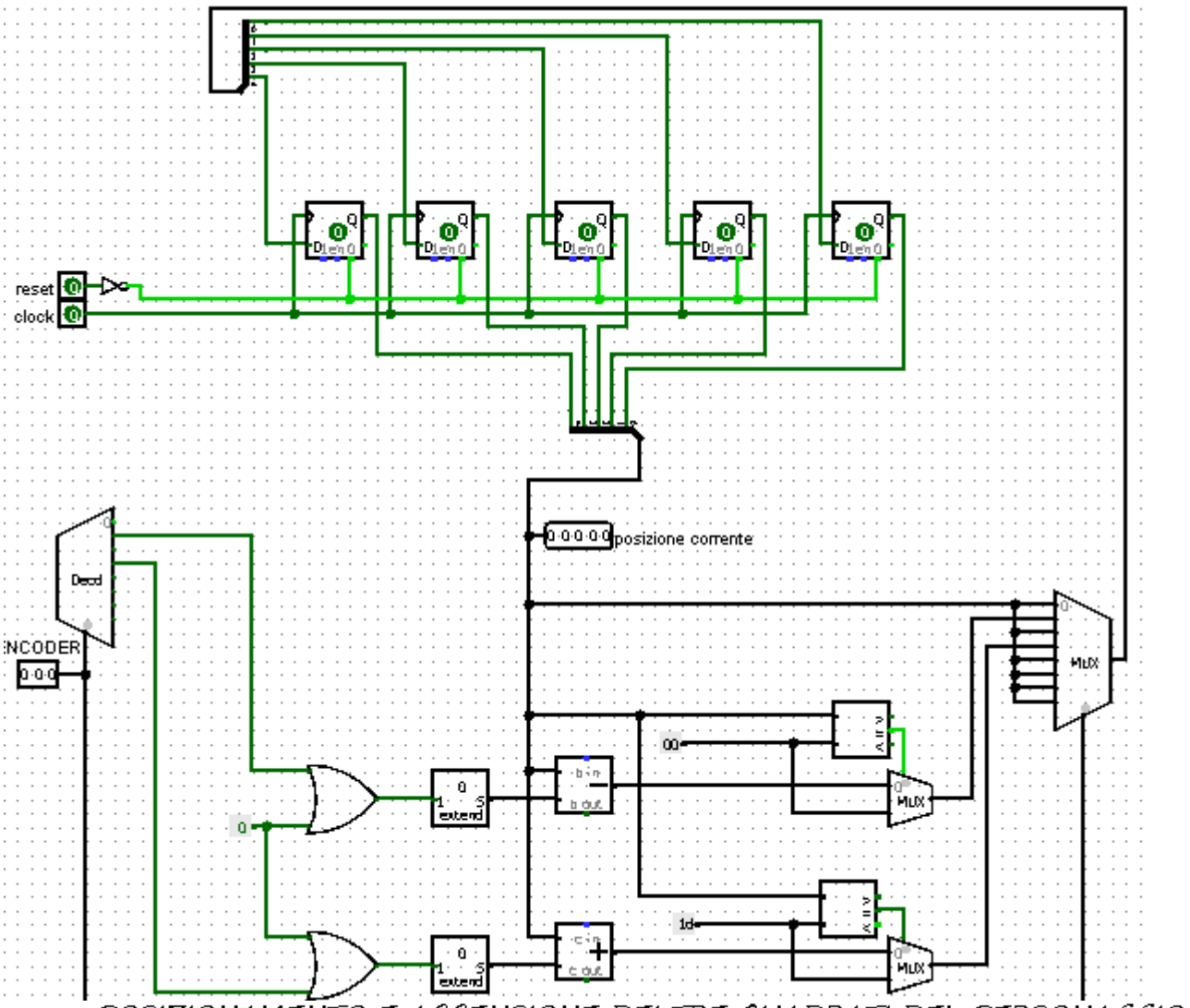


In questo modo, se la posizione ha raggiunto il limite destro o sinistro, allora non verrà utilizzata come posizione quella calcolata, ma quella corrispondente al limite, perché il giocatore non può andare oltre essa.

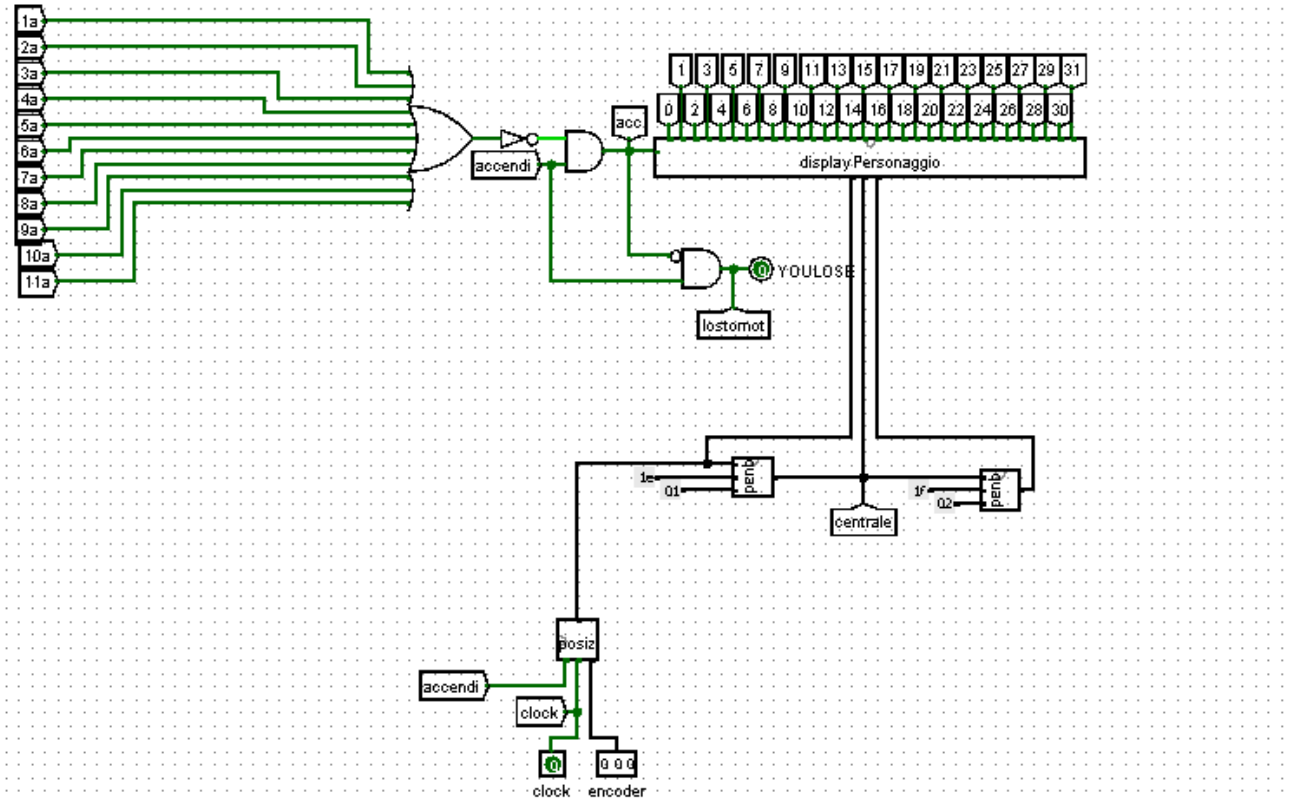
Il circuito sottostante (“PosizionamentoPg”) illumina il primo led del giocatore, aggiornandolo a seconda dell'input rilevato da tastiera:

- Se l'utente ha premuto “a” (left) la posizione memorizzata precedentemente verrà decrementata di 1
- Se l'utente ha premuto “d” (right) la posizione verrà incrementata di 1
- In tutti gli altri casi, si mantiene la posizione precedentemente

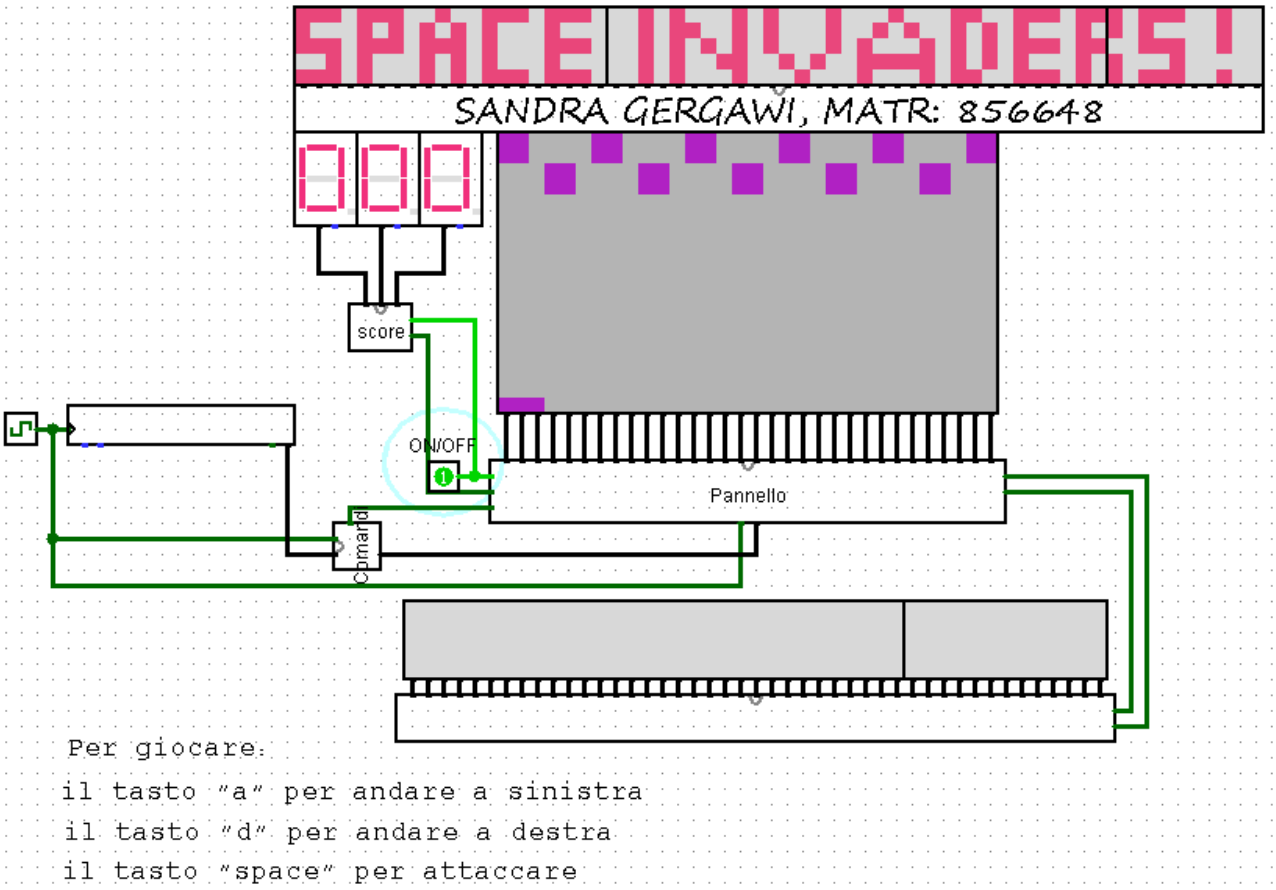
La posizione calcolata verrà utilizzata come riferimento per il posizionamento del secondo e del terzo led. Per l'accensione dei led nelle colonne corrispondenti , sono stati utilizzati 3 demultiplexer (1 per ogni led, ma il risultato visualizzato sulla matrice led sarà l'uscita di una or che come input ha tutte le uscite aventi la stessa posizione, in questo modo, se almeno una di queste uscita è pari a 1 vuol dire che il led deve accendersi. Dato che i tre led sono discostati di uno, i tre ingressi di ogni or non saranno mai accesi contemporaneamente, ma viene comunque fatta questa operazione per evitare che il display riceva in ingresso bit diversi creando errori.



POSIZIONAMENTO E ACCENSIONE DEI TRE QUADRATI DEL PERSONAGGIO



Sono stati utilizzati molti tunnel per evitare che il foglio di lavoro risultasse confusionario, ulteriori informazioni riguardanti gli ingressi sono stati fatti nel circuito stesso.
I circuiti “quad” sono quelli del posizionamento del secondo e del terzo led, e infatti prendono in ingresso l'uscita della posizione del led iniziale. “Posiz” infatti è il circuito “PosizionamentoPg”.
“DisplayPersonaggio” contiene i 3 demultiplexer commentati precedentemente. La sconfitta del giocatore viene misurata in base allo stato dei tre led.
Altri circuiti sono stati utilizzati per rendere i fogli di lavoro meno confusionari possibile.



Questa è la schermata principale.
Miglioramenti possibili:

- l'aggiunta di più nemici
- l'aumento delle dimensioni del display

Progetto Architettura Degli Elaboratori 1
Sandra Gergawi
Matricola: 856648