Relazione del Progetto di Laboratorio Architettura degli Elaboratori I

Giuseppe Del Campo*
Matricola: 935114, Turno: A
giuseppe.delcampo@studenti.unimi.it

1 Introduzione

Il progetto realizzato consiste nell'implementazione di un gioco che permetta all'utente di gestire un **orto** tramite alcuni strumenti a sua disposizione. Una volta accesa la simulazione, l'utente dovrà far crescere le piantine e infine potarle per raccogliere punti; per permettere la loro crescita, dovrà mantenere l'umidità nel **range** specificato. Lo scopo del gioco sarà totalizzare il maggior numero di punti possibile, senza arrivare allo 0% di umidità, che comporterebbe il game over.

1.1 Inizio del gioco

Appena iniziato il gioco, l'utente si troverà davanti a diversi **display**: quelli fondamentali mostrano lo *sviluppo delle piante*, la *quantità di acqua disponibile* e il *valore dell'umidità*. L'umidità sarà automaticamente impostata al 50% e le piante si troveranno nello stato iniziale. Inoltre, per facilitare l'utente, il serbatoio dell'acqua sarà completamente pieno.

1.2 Difficoltà del gioco

Per rendere tutto leggermente più impegnativo, è stata implementata una **difficoltà dinamica**. Inizialmente, per agevolare l'apprendimento e l'utilizzo del gioco, essa non influenzerà particolarmente le scelte dell'utente. Col passare del tempo, però, l'umidità scenderà sempre più velocemente, fino ad arrivare all'ultimo stadio di difficoltà: da quel momento in poi, gestire l'orto non sarà molto facile.

1.3 Istruzioni per giocare

1. Avviare il gioco premendo l'apposito pulsante di accensione.
2. Irrigare il terreno attraverso il pulsante di irrigazione, e tenere l'umidità nel range 50 - 70%,
3. Aspettare la crescita delle piantine, e potarle per ottenere punti.
4. Mantenere il serbatoio dell'acqua pieno, per evitare di rimanerne senza.

*Progetto approvato il 22/08/2019, consegnato il 06/09/2019

2 Fattore umidità

2.1 Visualizzazione su schermo

Dato che l'umidità è il fattore fondamentale del gioco, aveva bisogno di essere mostrata su schermo in modo chiaro e preciso. Per questo sono stati utilizzati due *7-Segment Display*, talché da poter mostrare quest'ultima dal valore 0 a 99.

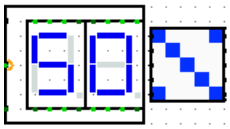

Figura 1: Visualizzazione dell'umidità su schermo

Per la gestione dei due display, è stato creato un componente chiamato *Display Umidità*; questo ricava le due cifre da dover visualizzare a partire dal valore dell'umidità in ingresso. All'interno di esso è presente anche un ulteriore componente chiamato ***7 Segment Driver***, che gestisce un singolo 7-Segment Display per far visualizzare la relativa cifra (da 0 a 9).

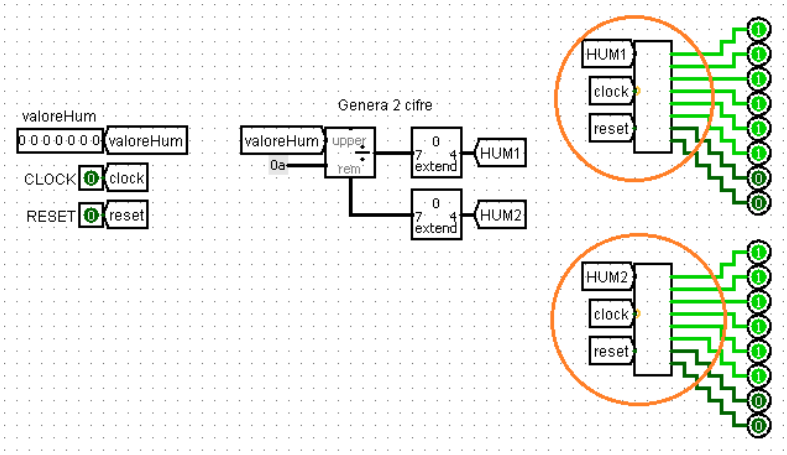

Figura 2: Componente di gestione del Display Umidità

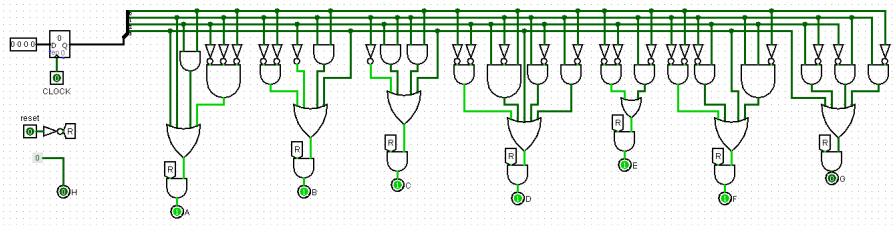

Figura 3: Componente 7 Segment Driver

2.2 Gestione

Per gestire l'umidità è stato creato un apposito componente chiamato *Gestione Umidità*. In esso sono presenti due circuiti fondamentali (gestione del valore dell'umidità e gestione della difficoltà del gioco) e anche un controllo sull'acqua, ovvero che se non ce n'è abbastanza nel serbatoio, non sarà possibile irrigare; è stato deciso di ottimizzare leggermente mettendo tutto dentro un singolo componente.

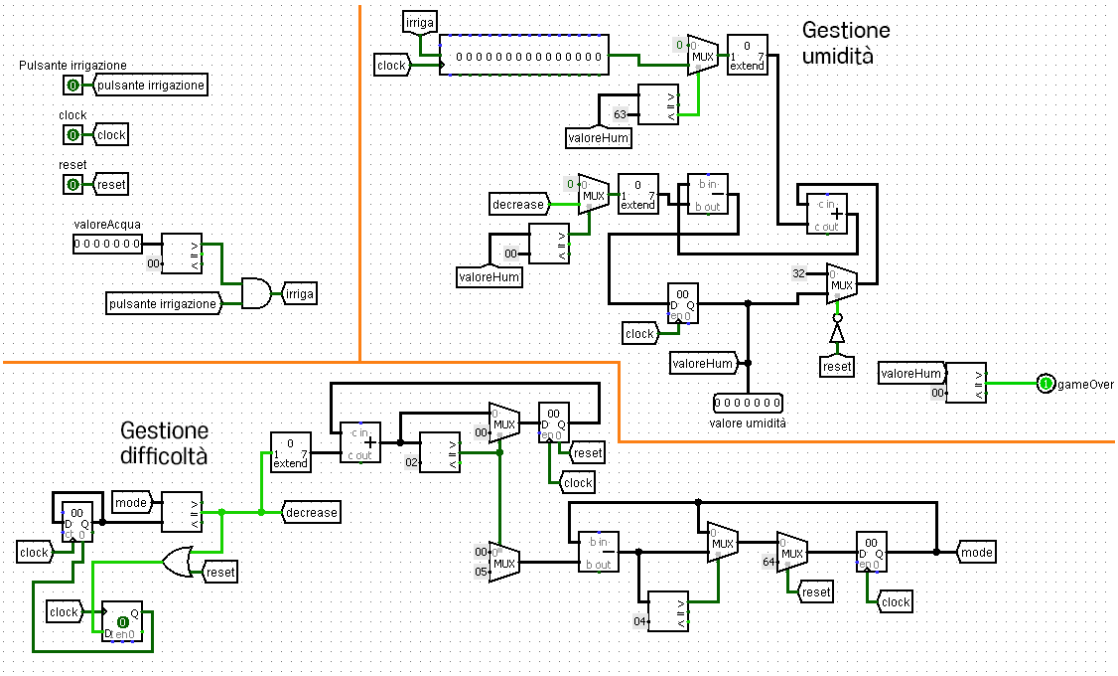

Figura 4: Componente di Gestione dell'umidità

2.2.1 Valore umidità

Il valore dell'umidità è impostato su 7 bit, per poter arrivare a 99. Viene memorizzato in un registro, e cambiato a seconda degli eventi. Quando viene premuto il pulsante di irrigazione, viene incrementato di 1 dopo essere passato in uno Shift Register a 15 stadi (per simulare l'attesa della caduta dell'acqua dal rubinetto), oppure viene decrementato di 1 quando arriva il segnale di decrease. Sono presenti anche due controlli realizzati con dei comparatori e dei multiplexer, che non permettono al valore di scendere sotto lo 0, o sforare la soglia prevista di 99. Se arriva il segnale di reset il registro viene impostato a 50 (valore iniziale dell'umidità). Se l'umidità arriva a 0, viene attivato il game over.

2.2.2 Difficoltà

E' stato deciso di inserire la gestione della difficoltà nello stesso circuito di quella dell'umidità, in quanto è la difficoltà che decide quando l'umidità va decrementata. Il circuito del decrease è un counter che incrementa fino ad arrivare a mode (un valore che serve a diminuire l'umidità), quando lo raggiunge il counter viene resettato, e l'umidità diminuisce di 1. Il valore mode viene gestito nell'altra parte del circuito: viene contato il numero di volte che arriva il segnale di decrease, e quando raggiunge 2 il valore mode viene diminuito di 5 (solo se mode è maggiore di 4, ovvero soglia prestabilita che determina la massima difficoltà). In questa parte di circuito sono presenti registri o latch per memorizzare i dati, comparatori, adder e subtractor per fare i calcoli, e multiplexer.

3 Gestione degli stati

L'orto ha 4 stati complessivi; ogni *stato* comporta la relativa condizione delle piante. Lo stato 0 (iniziale) presuppone che le piante siano state appena piantate o potate, mentre gli altri 3 prevedono delle piantine con stadi di crescita più avanzati. Per gestire gli stati complessivi, è stato creato un componente chiamato *Gestione Stati*. A parte questa funzione, ne è presente anche un'altra, ovvero la gestione del punteggio: dato che esso viene calcolato in base allo stato attuale, è stato posizionato all'interno di questo componente.

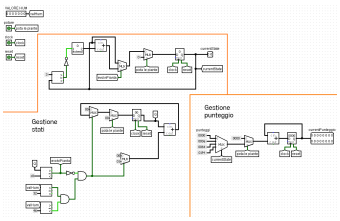

Figura 5: Componente di Gestione degli Stati

3.1 Stati

Lo stato attuale delle piantine può variare da 0 a 3, quindi è stato memorizzato su 2 bit. E' presente un controllo tramite comparatore che evita allo stato di andare oltre il 3 (se lo stato è 3, rimane fermo). Tramite un multiplexer, quando arriva il segnale di evoluzione delle piantine, il nuovo stato viene sommato ad 1 grazie a un adder, ed esso sarà il nuovo stato attuale. L'evoluzione delle piantine avviene in questo modo: se l'umidità si trova nel range corretto (50-70%) il registro che conta i colpi di clock viene incrementato di 1 ogni volta. Quando questo registro arriva a 200, si attiva il segnale di evoluzione delle piantine, e viene azzerato il registro contatore. La potatura delle piante azzera sia quest'ultimo, che il registro degli stati.

3.2 Punteggio

Quando vengono potate le piante, viene assegnato un *punteggio* a seconda dello stato attuale delle piantine:

Stato	***Punti ottenuti dopo la potatura***
0	0
1	10
2	100
3	500

Il punteggio, inizializzato a 0, viene memorizzato all'interno di un registro a 16 bit, e incrementato a seconda se vengono potate le piante o meno. Questo è reso possibile attraverso l'utilizzo di un multiplexer a 4 ingressi (4 punteggi per i 4 diversi stati); una volta selezionato il punteggio, viene sommato a quello totale attraverso l'utilizzo di un adder.

4 Fattore acqua

Per introdurre maggiore difficoltà nel gioco, è stato creato il *fattore acqua*: con esso non è possibile irrigare il terreno in maniera illimitata, ma sarà necessario tenere il serbatoio abbastanza pieno. Per la gestione di essa, è stato creato un componente chiamato *Gestione Acqua*.

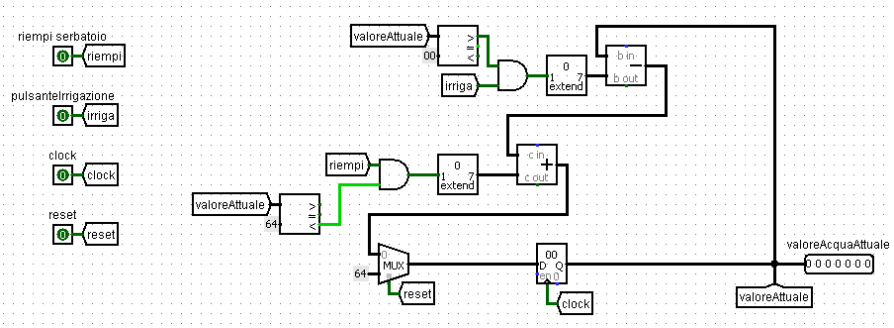

Figura 6: Componente di Gestione dell'acqua

Il valore dell'acqua viene memorizzato in un registro a 7 bit, e varia da 0 a 100 per simulare la percentuale di acqua disponibile. Viene decrementato o incrementato di 1 tramite adder o subtractor, a seconda se si irriga o si riempie il serbatoio: sono presenti inoltre dei controlli per non far sforare il valore sotto lo 0 o sopra il 100, realizzati tramite dei comparatori e porte and. Tramite un multiplexer il segnale di reset porta il valore a 100, che è quello iniziale.

5 Display grafici

Il componente più importante del progetto è sicuramente il display ***LED Matrix***, che ha permesso di visualizzare tutti gli elementi grafici del gioco; all'interno del progetto ce ne sono molti, ognuno con una diversa funzione.

5.1 Display piante

Questo display è grande 16x27, ed è quello principale dato che al suo interno vengono visualizzate le piantine e quanto sono cresciute.

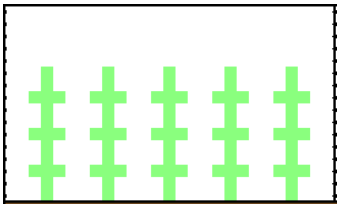

Figura 7: Visualizzazione su schermo delle piantine

Per la gestione di esso è stato creato un componente chiamato *Display piante*. Esso si occupa della gestione delle 16 righe del display; a seconda dello stato, ogni riga avrà un certo valore che permetterà la corretta visualizzazione delle piantine. Per fare ciò, sono stati utilizzati 16 multiplexer (con lo stato come bit di selezione) e 16 registri collegati a 16 uscite: ad esempio, se lo stato è 2 (come in figura 7) ogni riga del display viene impostata in modo tale da mostrare le piantine correttamente.

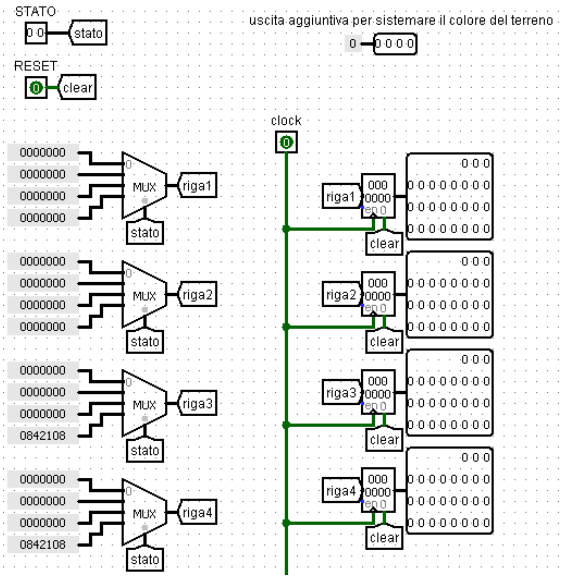

Figura 8: Parte del componente di gestione del Display Piante

5.2 Display rubinetto

Questo display grande 16x5 serve a simulare l'uscita dell'acqua dal rubinetto; per il rubinetto invece, è stato semplicemente creato un componente che mandasse le uscite corrette per mostrare il disegno giusto.

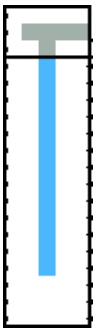

Figura 9: Visualizzazione su schermo del rubinetto

La gestione del rubinetto è stata implementata all'interno del componente *Display rubinetto*. Per simulare la caduta dell'acqua sono stati usati 16 registri, collegati in modo tale da formare uno shift register complessivo. All'ingresso della sedicesima riga (ovvero prima riga dall'alto del display) è presente un controllo sulla presenza di acqua nel serbatoio, posto in and con il pulsante di irrigazione, e mandato in un multiplexer. Se entrambe le condizioni sono soddisfatte, esce la sequenza di bit per creare una sorta di "goccia" sulla riga del display, che prosegue su ogni riga fino ad arrivare in fondo; tenendo premuto il pulsante di irrigazione, si crea un'effettiva uscita continua di acqua.

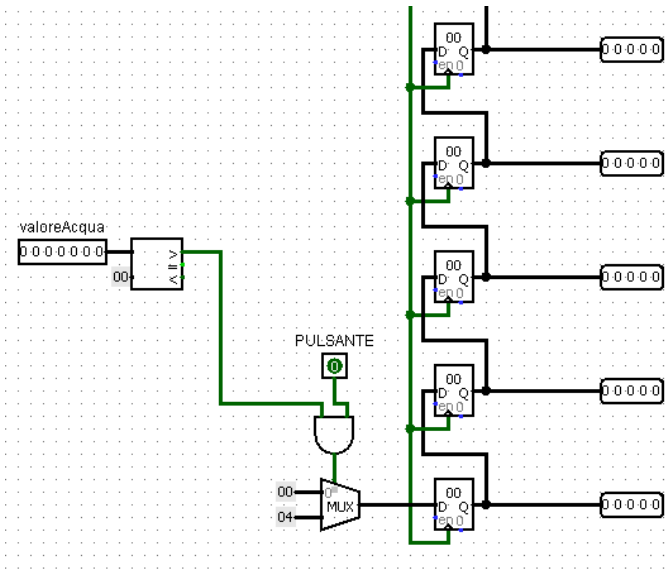

Figura 10: Parte del componente di gestione del Display Rubinetto

5.3 Display serbatoio

Questo display è grande 10x12, e serve a mostrare la quantità di **acqua** rimasta per irrigare; è formato da 10 righe per simulare la percentuale di acqua disponibile (dal 0 al 100%).

Figura 11: Visualizzazione su schermo del serbatoio

Per la gestione di esso è stato creato un componente chiamato *Display serbatoio*. Il funzionamento è simile ai display precedenti, ma stavolta sono presenti 10 comparatori e 10 multiplexer che servono ad attivare le righe corrette del display e simulare quindi la presenza dell'acqua. Ogni comparatore controlla se il valore dell'acqua è maggiore a una certa soglia; se lo è, attiva la riga corrispondente. Se ad esempio è maggiore a 0, 9, 19 e 29, verranno attivate le righe 1, 2, 3 e 4. Anche qui sono stati usati 10 registri con le relative 10 uscite.

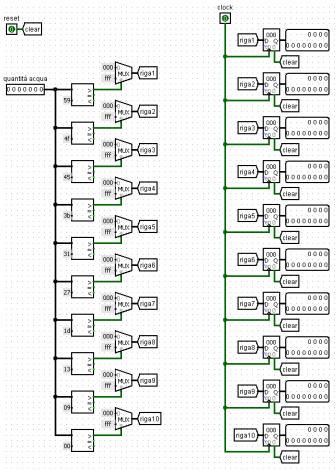

Figura 12: Componente di gestione del Display Serbatoio

5.4 TTY Display

Per mostrare il punteggio è stato utilizzato il componente ***TTY*** di grandezza 1x16, che è un display che lavora su caratteri ascii. E' stato necessario quindi creare un componente di gestione che servisse a mostrare il punteggio attuale.

Figura 13: Visualizzazione su schermo del Punteggio

5.4.1 TTY Display Driver

Dato il punteggio di 16 bit in ingresso, il punteggio massimo ottenibile è di 65.536; è stato necessario dunque prendere ognuna delle 5 cifre del punteggio da mostrare sul display. Questo è stato possibile grazie all'utilizzo di 4 divider, che hanno diviso la cifra per 10 per 4 volte, ottenendo ognuna delle cifre (l'ultima cifra è l'ultimo resto calcolato). Dato che il display lavora su caratteri ascii, è stato sommato 48 ad ognuna delle cifre, ottenendo i caratteri corretti da mostrare su display. Grazie a dei multiplexer, il segnale di reset decide se far passare i codici appena ottenuti o la costante hex 30 (che sarebbe il carattere 0 in ascii). Uno shift register a 5 stadi viene precaricato con i valori, fungendo effettivamente da serializzatore, in modo tale da far uscire i caratteri uno per ogni colpo di clock. L'uscita sarà quindi un singolo carattere ascii a 7 bit; con 5 colpi di clock si andrà a formare il punteggio per intero.
Quando lo shift register non manda valori del punteggio in uscita, dà come output 0 (spazio vuoto): questo evita problemi nell'output.

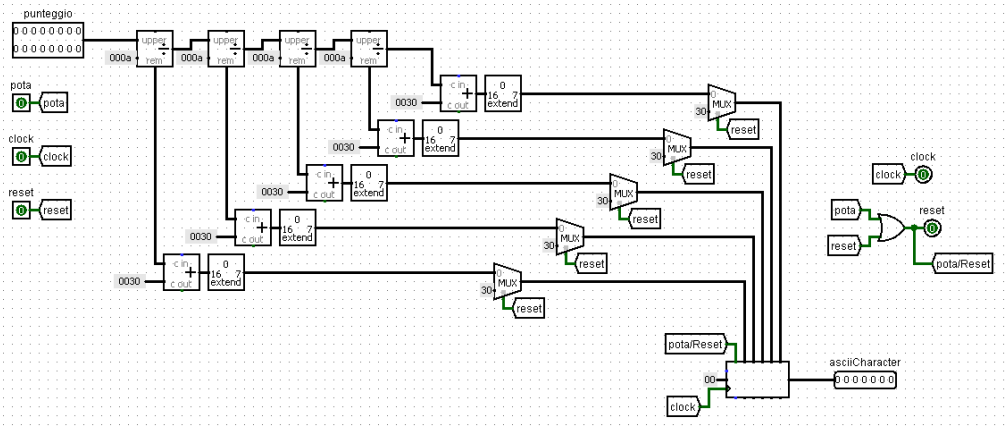

Figura 14: Componente TTY Display Driver

6 Note di implementazione e commenti

Il circuito è stato progettato per funzionare ad una frequenza di 64Hz. Ogni componente realizzato è stato strutturato in modo tale da essere resettato dal segnale globale di reset; per semplicità di implementazione, il segnale di reset è stato usato come modalità OFF / spento, e anche come game over; inoltre sono stati utilizzati Bit Extender a seconda delle necessità nella maggior parte dei componenti creati.
Alcuni display LED Matrix, come si può ben notare, sono statici, come ad esempio quello per simulare il terreno, quello del rubinetto e quello del segno di percentuale. Per visualizzarli correttamente sono state mandate le uscite corrispondenti in modo da formare i disegni voluti.

6.1 Possibili miglioramenti ed estensioni

Ci sono alcuni aspetti che potrebbero essere inclusi per rendere il gioco più interessante, come ad esempio maggiore difficoltà introdotta dal deterioramento delle piante se l'umidità non viene tenuta nel range prestabilito; oppure la suddivisione del display primario delle piantine in più sottodisplay, ognuno impostato per gestire un diverso tipo di piantina, a seconda della scelta del giocatore (Es: mettendo una pianta grassa, cresce più lentamente ma si ottengono più punti potandola). Inizialmente era presente anche la gestione della temperatura, ma essendo un orto, si è pensato che non era possibile gestirla: un'altra espansione sarebbe quella di poter scegliere tra orto e serra (dato che la serra è chiusa, sarebbe possibile gestire anche la temperatura in modo preciso, per far crescere le piante più velocemente). Per quanto riguarda invece gli aspetti tecnici del gioco, si potrebbe migliorare la funzione di game over, implementando magari un display che comunica quando si perde.