XXX XXX matricola n. XXXXXXXXX A.A. 2016/2017

Progetto Architettura degli elaboratori 1

Controllo serra automatizzata

Indice

* Specifiche di progetto [1]
  + Descrizione generale [1]
  + Interfaccia utente [1]
  + Condizioni iniziali [1]
  + Condizioni finali [2]
  + Ciclo di utilizzo [2]
* Circuiti [3]
  + Sensori [3]
  + Scritte cont [4]
  + Scritte [5]
  + Lettere [6]
  + Parole [6]
  + Media [7-8]
  + Mem update [9]
  + Sens cont [10]
  + Memoria [11]
  + Controller riga [12]
  + Switch scritte [13]
  + Num to display [14]
  + Hub switch numeri [15]
  + Mux valori [16]
  + Switch numeri [17]
* Circuito principale [18]
* Interazione sottocircuiti [19-20]
* Note finali [21]
* Miglioramenti [21]
* Allegati (tabelle codifiche) [22]

SPECIFICHE DI PROGETTO

DESCRIZIONE GENERALE

Si vuole progettare un sistema di circuiti in grado di monitorare tre sensori (simulati): temperatura, umidità e luce. L’utente sostanzialmente deve effettuare una sola azione: impostare le soglie secondo le quali verranno controllati i sensori. Come ulteriore interazione, viene messo a disposizione un sistema di display che, tramite un apposito pulsante, visualizzano diversi dati in diverse disposizioni. Il circuito calcola automaticamente valori di picco (minimi e massimi) e medie mobili; tali dati vengono memorizzati per una successiva visualizzazione o reset.

INTERFACCIA UTENTE

L’interfaccia utente è composta da:

* INPUT:
  + Input numerici: 3 input per l’inserimento delle soglie (in binario);
  + Pulsanti di Reset: un pulsante per il reset generale del sistema ed uno per il reset di valori di picco e medie;
  + Tasto “Enable”: per abilitare il salvataggio dei dati;
  + Tasto “Modifica” e pulsante “Save”: per modificare le soglie;
  + Pulsante “>”: per modificare i dati visualizzati e la loro disposizione.
* OUTPUT:
  + 6 linee di display: le prime tre per visualizzare i dati memorizzati e le ultime tre per mostrare i valori dei sensori in tempo reale;
  + 3 led: per segnalare valori fuori soglia (simulano l’attivazione di azionatori).

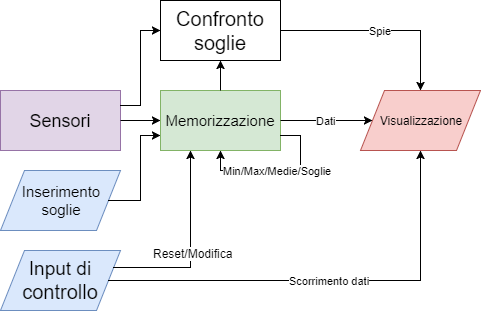
CONDIZIONI INIZIALI

Per il corretto funzionamento, inizialmente i sensori vengono preimpostati a valori predefiniti. I registri di minimo vengono impostati a 100 (fondo scala), mentre quelli di massimo a 0. In output vengono visualizzati tutti i minimi. Le soglie sono a 0 di default.

CONDIZIONI FINALI

Non c’è un vero e proprio stato finale, infatti il circuito, ad ogni ciclo di clock, genera nuovi valori che vengono valutati dal resto del circuito. Perciò ad ogni ciclo di clock i valori di picco e le medie variano, insieme a ciò che viene visualizzato.

CICLO DI UTILIZZO



Appena avviata la simulazione, il circuito inizia a generare numeri pseudo-casuali. Tuttavia, per far funzionare correttamente il tutto, è necessario premere il tasto “Enable”. Una volta fatto ciò, i valori di picco e le medie (calcolate in un apposito sotto circuito) verranno inviate alla memoria e salvati nei suoi registri interni, che vengono scritti ad ogni ciclo di clock (a meno che l’”Enable” sia disattivato). Contemporaneamente l’utente può modificare le soglie rispettando la seguente procedura:

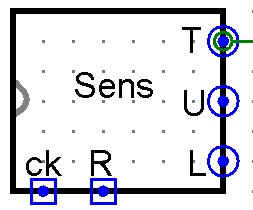
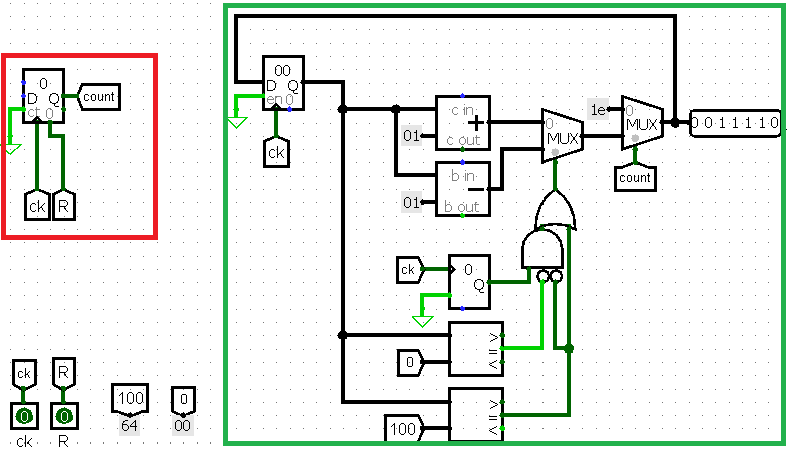
1. Bisogna inizializzare la procedura premendo il tasto “Modifica”. In questo modo verrà automaticamente visualizzata la pagina delle soglie;
2. Le soglie vengono inserite tramite gli appositi input numerici;
3. Per salvare le impostazioni, basta premere il pulsante “Save”. Contestualmente al salvataggio, sui display è possibile visualizzare i valori immessi;
4. Alla fine della modifica, si deve disattivare “Modifica”.

In questo modo i dati vengono scritti nei registri della memoria.

Da questo momento il circuito si può definire “a regime”, infatti non necessita di ulteriori input per il proprio funzionamento.   
Durante le varie fasi (tranne quella di modifica delle soglie), l’utente può visualizzare i diversi dati tramite il pulsante “>”.

CIRCUITI

Sensori



Composizione:

* Un input per il clock
* Un input per il Reset (generale)
* Due costanti: 0x64 e 0x00 rispettivamente, valore massimo e minimo

Principio di funzionamento:

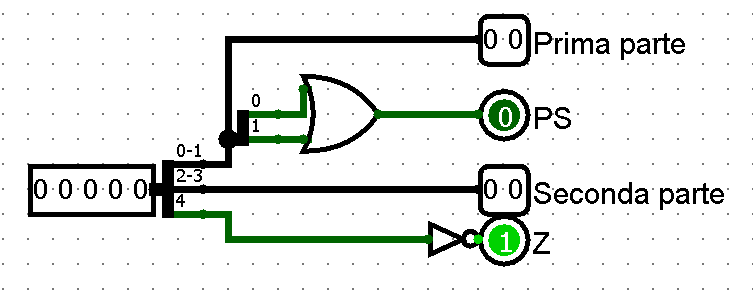
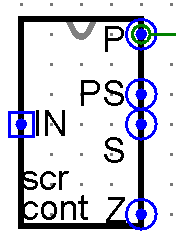
Ad ogni ciclo di clock, viene generato un nuovo numero, se questo esce dal range prefissato, viene calcolato il numero più vicino a quello da scartare ma dentro al range.

Progettazione:

Per generare i valori si può partire da un numero prefissato ed incrementare/ decrementare di 1 il valore precedente. Considerando n il valore attuale, n+1 e n-1 vengono calcolati simultaneamente, poi, tramite un generatore di numeri casuali, ne viene selezionato uno che viene mandato in output e retroazionato per calcolare il valore successivo.

Se il valore raggiunge il minimo del range, viene automaticamente incrementato; se invece raggiunge il massimo, viene forzatamente decrementato.

Scritte cont



Composizione:

* Un input numerico da 5 bit
* Due output numerici da 2 bit
* Due output booleani

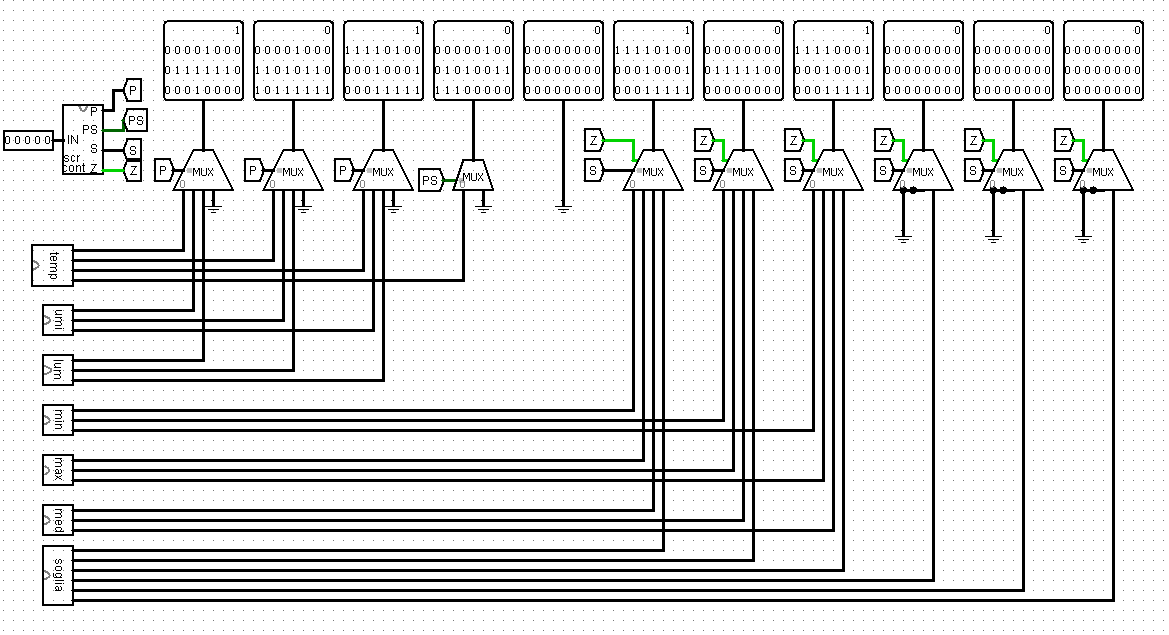
Principio di funzionamento:

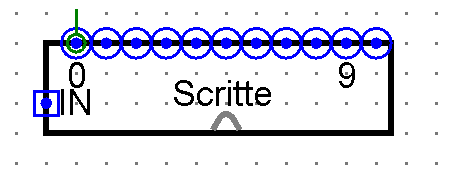
Sui display vengono sempre generate 2 parole, per la prima ci sono tre alternative: “Temp”, “Umi” e “Lum”. Siccome “Temp” occupa uno spazio in più rispetto alle altre, bisogna determinare quando il quarto display deve visualizzare “p” o uno spazio vuoto. Inoltre, come codice per visualizzare le parole, viene utilizzata una stringa di 5 bit. Tali bit devono essere divisi in due sequenze da 2 bit (per selezionare le parole) ed un segnale per la cancellazione della seconda parola.

Progettazione:

Per determinare lo stato del quarto display occorre ragionare sulla codifica dell’input; la parola più lunga è identificata da “00”, perciò se la prima sequenza non rispetta questa caratteristica, bisogna spegnere il display. La negazione sul segnale di azzeramento è solo una questione di utilizzabilità, infatti è più immediato pensare che l’azzeramento avvenga quando il relativo bit viene posto a 1.

Scritte





Composizione:

* Un input numerico
* Le sette parole da visualizzare
* Undici output numerici da 25 bit
* Il sotto-circuito “Scritte cont”

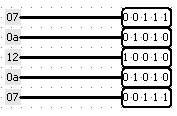
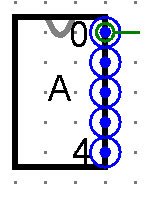
Principio di funzionamento:

Il circuito si occupa di inviare ai display la corretta combinazione di scritte. In input riceve un codice numerico che è la codifica delle varie combinazioni di parole.

Progettazione:

Il numero in ingresso, è codificato appositamente pensando alle scritte da visualizzare, perciò è abbastanza semplice sfruttarlo. Ottenute le diverse stringhe dal circuito “Scritte cont” è del tutto naturale utilizzare dei multiplexer che a seconda del numero in input, selezionano (per ogni display) quale lettera mostrare. Per ottimizzare i componenti, l’ultimo pin dei primi 3 multiplexer viene usato per la cancellazione. Per cancellare anche la seconda parola, basta sfruttare il segnale “Z” che, quando posto a uno, disabilita gli ultimi sei multiplexer.

A-D-E-G-I-L-M-N-O-P-S-T-U-X



Composizione:

* 5 costanti numeriche
* 5 output numerici

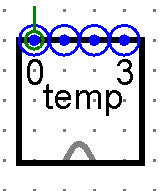
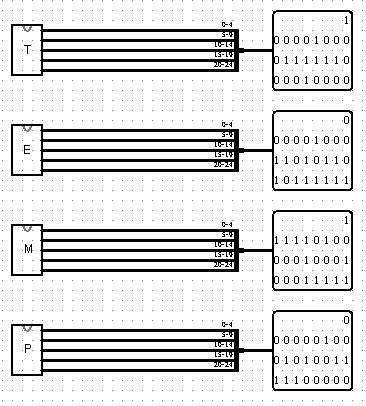
Principio di funzionamento:

Siccome i display (per la visualizzazione delle lettere) non sono altro che matrici led, serve, per ogni colonna, una sequenza binaria per accendere o meno i led.

Progettazione:

La progettazione di questi componenti è pressoché nulla, infatti servono solo da “contenitori” per le costanti che rappresentano le sequenze binarie per il controllo dei display. Si ottiene pertanto un circuito per ogni lettera.

Temp-Umi-Lum-Min-Max-Med-Soglia



Composizione:

* Un circuito per ogni lettera
* 4 output numerici a 25 bit
* 4 splitter

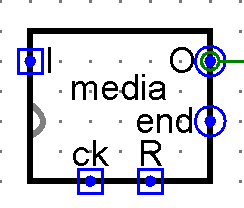
Principio di funzionamento:

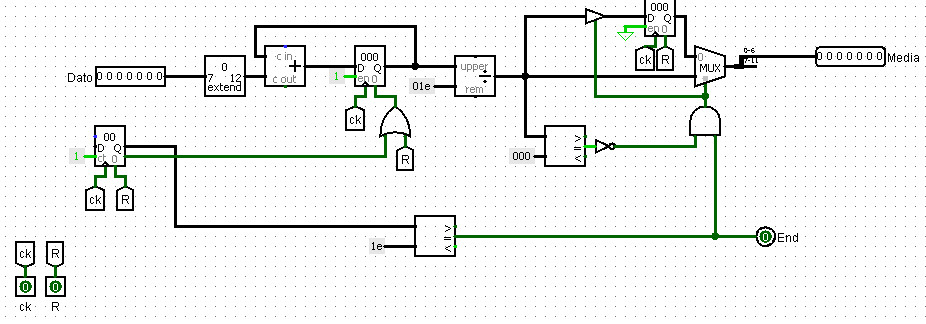
Come per i circuiti raffiguranti le lettere, serve un circuito per ogni parola.

Progettazione:

Visto che ogni lettera è rappresentata da cinque stringhe binarie, ogni parola sarà composta dall’insieme di tutte le stringhe di ogni lettera. Per ovviare al limite di 32 bit per output, si può creare un circuito con m output (dove m è pari al numero di lettere), dove ogni output è lungo 25 bit.

Media





Composizione:

* Un input numerico a 7 bit;
* Due input per clock e Reset (generale);
* Due registri a 12 bit;
* Un output numerico a 7 bit.

Principio di funzionamento:

Per il calcolo delle medie serve sapere tre cose: il valore attuale, i valori precedenti e quando si hanno abbastanza dati per calcolare la media. Riflettendo risulta inutile memorizzare i dati precedenti, infatti basta che ad ogni ciclo di clock si sommi il valore attuale alla somma precedente; così facendo, occorre un solo registro (per salvare la somma) anziché uno per ogni valore. Serve un modo per capire quando calcolare la media; visto che, ad ogni ciclo di clock viene aggiornata la somma, si può sfruttare lo stesso clock, per contare quante somme sono state fatte. Così il contatore, arrivato ad un certo numero, genera un segnale che esegue la divisione e lo manda in output.

Progettazione:

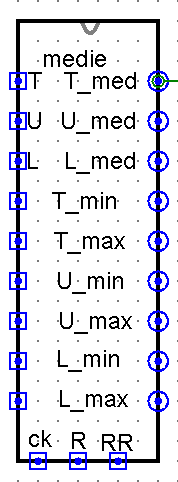
Prima di realizzare il circuito bisogna ragionare sui bit necessari per effettuare tutte le operazioni. Visto che i valori in input variano da 0 a 100, servono 7 bit per rappresentarlo. Dato che l’output è una media, il suo valore non può essere maggiore di 100, perciò anche l’output sarà da 7 bit. Ovviamente 7 bit non bastano per memorizzare la somma e farne la divisione, l’unico limite è Logisim. Si è scelto di eseguire i calcoli di media su 12 bit, il che permette fare la media al massimo di 40 valori (2^12-1/100).

Detto ciò, si può dividere il circuito in due porzioni:

1. La prima che calcola la somma;
2. La seconda che esegue la media e manda in output il risultato.

Per realizzare la prima parte è sufficiente prendere l’ultimo valore e sommarlo al totale precedente. Quindi serviranno un registro che salvi il totale ed un sommatore che sommi il valore in input con quello salvato nel registro.

La seconda parte è più complessa, infatti oltre ad eseguire la divisione, deve prevedere un sistema che, durante il calcolo della media, mandi in output la media precedente. Infatti se non venisse realizzato tale sistema, tra una media e l’altra, nella memoria verrebbe salvato 0. Siccome serve “ricordare” qualcosa, la soluzione più semplice è l’utilizzo di un registro che viene aggiornato ad ogni fine calcolo.

Mem Update

Date le dimensioni del circuito, per la sua visualizzazione, bisogna aprire il relativo circuito Logisim.

Composizione:

* Tre circuiti per il calcolo delle medie
* 9 Input numerici da 7 bit
* 9 Output numerici da 7 bit
* 9 Multiplexer a due ingressi
* Un contatore modulo 1
* Un ingresso per il clock
* Due ingressi per i due Reset

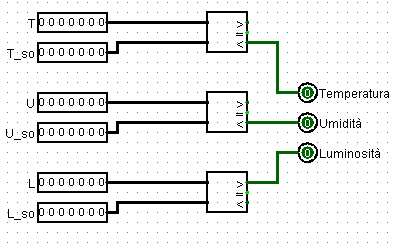
Principio di funzionamento:

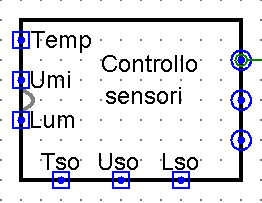
Dato che si vogliono memorizzare e mostrare i dati di picco e le medie, è necessario un circuito che le calcoli. In linea di massima, per il calcolo dei minimi è sufficiente confrontare il minimo precedente con il valore attuale ed inviare in output (e quindi alla memoria) il minore dei due, per il massimo si segue lo stesso ragionamento.

Progettazione:

Per le medie basta utilizzare il sotto-circuito dedicato per i tre sensori. Per i massimi si può seguire l’idea intuitiva di calcolo. Mentre per il minimo occorre realizzare un circuito più complesso, ovvero:

* Si confronta il valore attuale con il minimo precedente e si seleziona il minore dei due;
* Prima dell’output si deve frapporre un ulteriore circuito. Infatti inizialmente tutti i registri sono a 0, perciò dopo il primo ciclo di clock risulterebbe che i minimi siano tutti 0. Una soluzione è porre inizialmente tutti i minimi a 100.

Sens cont



Composizione:

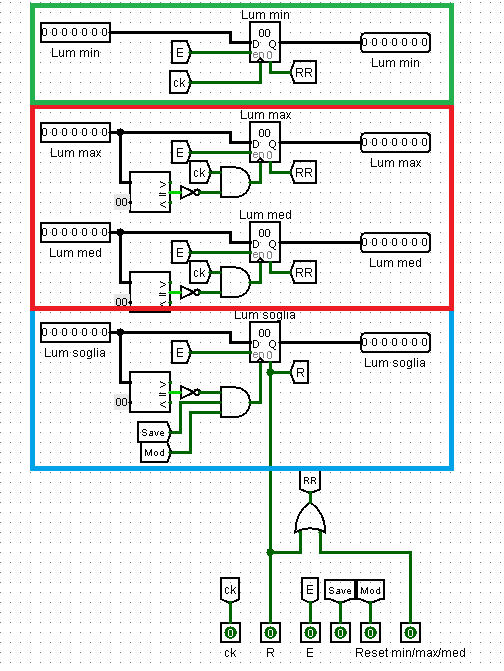
* 6 Input numerici da 7 bit
* 3 Output booleani
* Tre comparatori

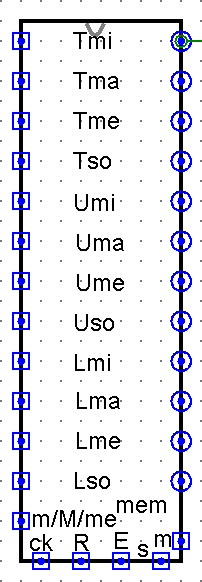
Principio di funzionamento:

Generati i valori dei sensori, e salvate le relative soglie, occorre confrontarle. Per farlo basta utilizzare i comparatori standard di Logisim. I segnali generati da essi sono i tre output cercati.

Progettazione:

Nel caso del valore di luminosità, si è scelto di invertire il senso di comparazione per ottenere un funzionamento più verosimile; se la luce rilevata supera la soglia impostata, viene segnalato un problema.

Memoria



Composizione:

* Tutti i comandi tranne “>”
* I due input per Save e Modifica
* 12 Input numerici da 7 bit
* 12 Output numerici da 7 bit

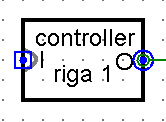
Principio di funzionamento:

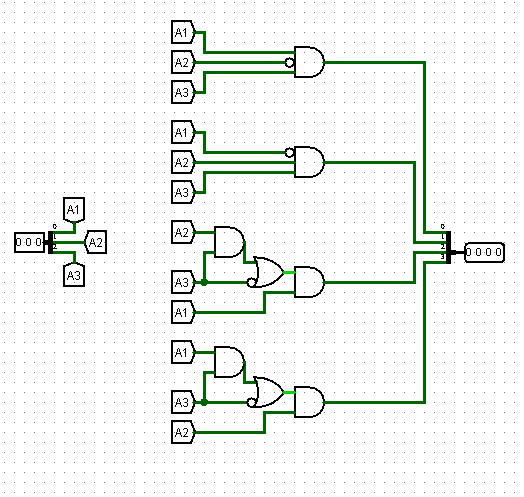
Per salvare i dati provenienti dal circuito “Mem Update” e le soglie, serve una memoria che però, per evitare errori, non si può aggiornare ad ogni ciclo di clock. I minimi (in verde) devono essere aggiornati ad ogni ciclo di clock, i massimi (in rosso) per essere scritti devono risultare diversi da 0 e le soglie (in azzurro) possono essere aggiornate solo se viene rispettata la procedura.

Progettazione:

1. I massimi devono essere aggiornati solo quando, in corrispondenza del clock, il dato in ingresso è diverso da 0. Perciò si può utilizzare un comparatore che con il clock esegue l’aggiornamento;
2. I segnali da monitorare per le soglie sono tre: il confronto con zero, il pulsante save e il tasto modifica. Se il dato è valido, il registro può essere scritto solo se entrambi gli input (“Modifica” e “Save”) sono a 1.

Le soglie hanno come unico reset quello generale, mentre il resto viene resettato anche dal reset parziale.

Controller riga 1-2-3



Composizione:

* Un input numerico da 3 bit
* Un output numerico da 4 bit

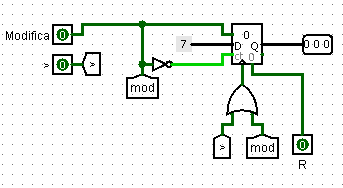
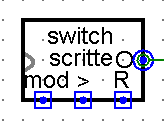
Principio di funzionamento:

Siccome si vuole utilizzare un solo pulsante per il controllo di tre righe diverse, è necessaria una codifica che, tramite un numero, identifichi in modo univoco (in modo diverso per ogni riga) la scritta da visualizzare.

Progettazione:

Siccome le combinazioni di scritte da visualizzare sono 7, servono 3 bit per numerarle. Come visto durante la progettazione del circuito “Scritte cont”, il concetto di “quale scritta” è rappresentato a sua volta da un numero di 4 bit (più uno per l’azzeramento). Perciò ciò che si cerca è una serie di funzioni booleane (4 per ogni riga) che, prendendo in input il numero di 3 bit, “costruiscano” le 3 stringhe a 4 bit. Per leggibilità del circuito, ogni quadrupla è stata racchiusa in un sotto-circuito.

Switch scritte



Composizione:

* Un input per “Modifica”
* Un input per “>”
* Un input per il reset generale
* Un output numerico da 3 bit

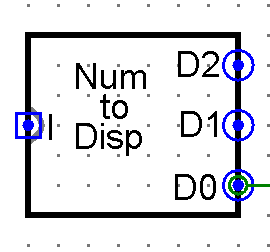
Principio di funzionamento:

Tutte le righe sono controllate tramite un solo numero a 3 bit, tale numero viene variato dal pulsante “>”. Perciò serve un modo di ricordarsi cosa si è selezionato al click precedente e un circuito che vari questo numero.

Progettazione:

La stringa a 3 bit *numera* le varie scritte, l’ordine non è importante al fine di questo circuito. Perciò si può affermare che ogni scritta corrisponde ad un numero. Si può sfruttare un contatore che, quando viene premuto il pulsante “>”, viene incrementato fino a far “scorrere” tutte le scritte. L’input “Modifica” serve per la procedura di modifica delle soglie, quando questa viene avviata, il contatore viene portato forzatamente a “3”.

Num to display

Viste le dimensioni del circuito, per la visualizzazione bisogna aprire il circuito Logisim.

Composizione:

* Un input numerico da 7 bit
* 3 output numerici da 4 bit

Principio di funzionamento:

Purtroppo in Logisim non esiste un componente che, da un ingresso binario, restituisca la rappresentazione decimale. Perciò serve un circuito che converta i numeri binari su 7 bit, nel corrispettivo decimale.

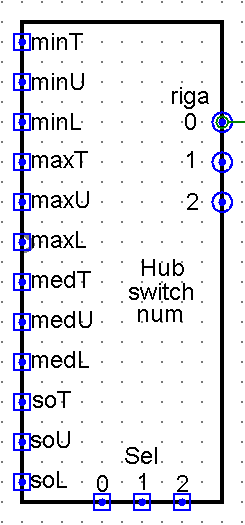
Progettazione:

Come da specifiche, i numeri generati variano tra 0 e 100, perciò anche tutti vari dati della memoria varieranno in questo range. Sfruttando questo fatto, si può pensare a creare un circuito combinatorio che prende questo numero e lo converte in più passaggi: Decoder->divisione in cifre->codifica cifre.

Purtroppo non si può realizzare direttamente questo sistema come descritto; infatti anche per i Decoder vige la restrizione che impone Logisim, ovvero: ogni componente standard, ha al massimo 32 input/output. Questo modifica leggermente il circuito risultante ma allunga di molto il lavoro di collegamento e riordino; infatti si può ovviare ragionando su una caratteristica dei numeri binari, ovvero: se si prende la stringa di 7 bit, e la si divide in due stringhe, una contenente i primi 3 bit e la seconda gli ultimi 4; si osserva che i primi 3 bit identificano tutti i numeri che possono essere visti come: m+8\*k (dove m è il numero di 3 bit e k è un numero naturale). La seconda stringa rappresenta ‘8\*k’.

Quindi una possibile struttura è un decoder ad 8 vie che abilita o meno altri 8 decoder a 16 vie. Raggruppati a seconda delle diverse cifre, i numeri vengono convertiti tramite 3 priority encoder.

Hub switch numeri

 Viste le dimensioni del circuito, per la visualizzazione bisogna aprire il circuito Logisim.

Composizione:

* 12 input numerici da 7 bit
* 3 input numerici da 4 bit
* 3 output numerici da 7 bit

Principio di funzionamento:

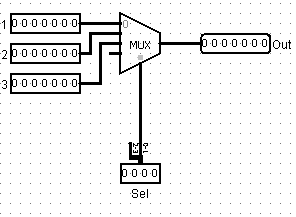
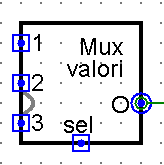
Per ogni riga ci sono diversi numeri visualizzabili, quindi serve un circuito che in ingresso abbia tutti i numeri visualizzabili sulle diverse righe ed in uscita il numero giusto per ogni riga.

Progettazione:

Per semplificare la progettazione, si può pensare che qualsiasi valore sia visualizzabile in qualsiasi riga. Come visto in precedenza, il circuito “Switch scritte” insieme ai controller di riga generano un codice che identifica quali scritte visualizzare su ogni riga. Per la selezione del numero da visualizzare si può sfruttare tale stringa.

Utilizzando ulteriori due sotto-circuiti, per ogni riga si può prima “capire” quale grandezza si deve considerare e poi quale dei suoi aspetti visualizzare. Questo lo si può fare dividendo nuovamente il codice di 4 bit.

Mux valori



Composizione:

* 3 input numerici da 7 bit
* Un input numerico da 4 bit
* Un output numerico da 7 bit

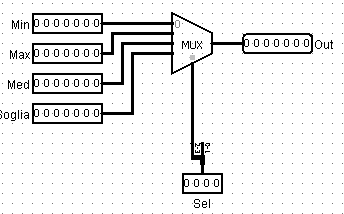
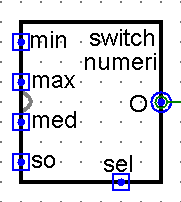
Principio di funzionamento:

Serve per capire quale grandezza analizzare nel circuito “Hub switch numeri”.

Progettazione:

In ingresso si ha un generico dato delle tre grandezze (la tipologia di dato è in comune), inoltre come input si ha anche il codice di 4 bit che identifica le scritte. Basta quindi considerare i primi due bit di tale stringa per identificare la grandezza (i primi due bit selezionano la prima parola).

Switch numeri



Composizione:

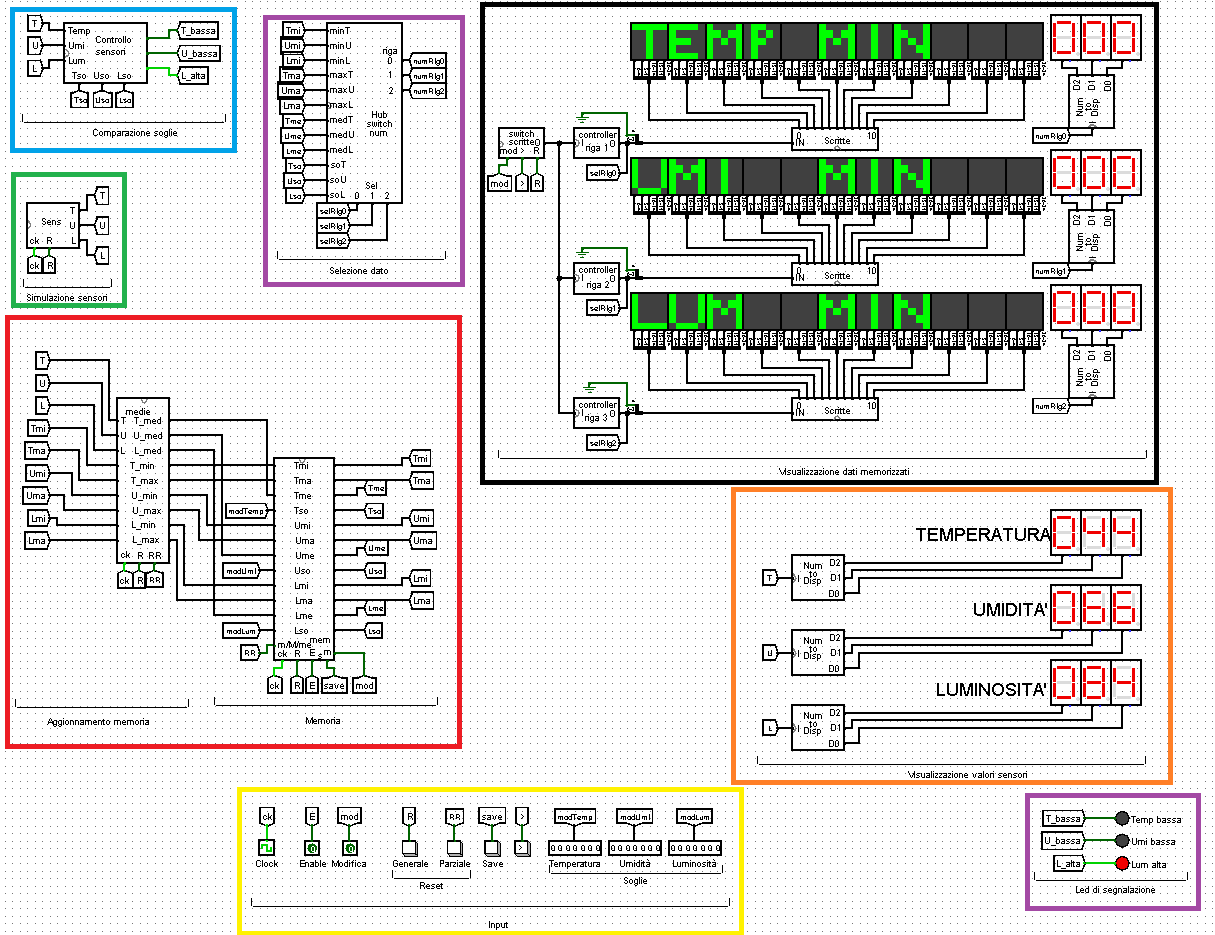
* 4 input numerici da 7 bit
* Un input numerico da 4 bit
* Un output numerico da 7 bit

Principio di funzionamento:

“Switch numeri” riceve in input tutti i dati relativi ad una grandezza. La funzione che svolge è quella di selezione di uno degli input tramite il codice a 4 bit in ingresso.

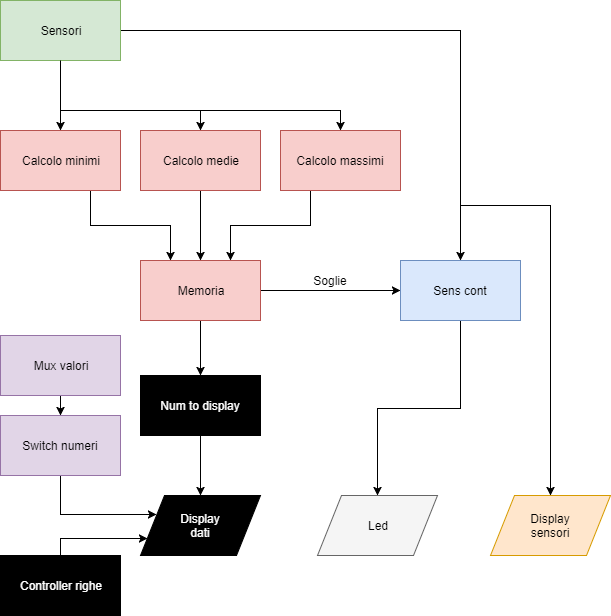
Progettazione:

Anche in questo caso visto che si tratta di selezionare un dato piuttosto che un altro, la scelta più semplice è utilizzare un multiplexer, resta da capire come pilotarlo. Nuovamente si può sfruttare la struttura del codice a 4 bit; siccome la seconda metà identifica la seconda parola di ogni riga, allora identificherà anche il dato da visualizzare

CIRCUITO PRINCIPALE

* Blocco giallo: contiene tutti gli input necessari per il funzionamento del sistema, inoltre contiene anche il clock;
* Blocco rosso: in input riceve tutti i valori, più tutti i segnali I/O tranne “>”. In output presenta i valori di picco, le medie e le soglie. Tutto viene memorizzato all’interno del secondo circuito, le soglie vengono modificate direttamente in quest’ultimo, mentre gli altri dati nel primo circuito;
* Blocco verde: in input riceve il clock ed il reset generale. All’interno genera continuamente nuovi numeri che manda in output;
* Blocco azzurro: in input riceve i sensori e le soglie, a seguito di una comparazione, mostra i risultati in output;
* Blocco nero: come input ha solo “Modifica”, “>” e il reset generale. Tramite una serie di sotto-circuiti, ad ogni pressione di “>” mostra scritte diverse (con i relativi valori). Per la visualizzazione dei numeri, costruisce una stringa binaria che invia al blocco viola;
* Blocco viola: in input riceve tutti i dati memorizzati più le stringhe calcolate nel blocco nero. Grazie ad una batteria di multiplexer, fornisce in output il dato da visualizzare su ogni riga;
* Blocco arancione: visualizza sui display i valori generati che riceve in ingresso;
* Blocco grigio: replica i risultati del blocco azzurro ma sotto forma di led.

INTERAZIONE SOTTOCIRCUITI



Il circuito “Sensori” può essere considerato il primo blocco del sistema; come tale non riceve alcun segnale da altri circuiti, ma solo il clock ed il reset. Essendo il primo (anche in ordine di progettazione), esso impone su quanti bit vengono rappresentati i numeri in tutto il circuito. I valori generati vengono utilizzati da: “Sens cont”, memoria e nella parte di display sensori. Nelle prime fasi di progettazione, generava numeri casuali all’interno del range 0/100; ciò nonostante in seguito si è migliorato il circuito sfruttando la retroazione dei dati per riutilizzarli. Visto che creava ridondanza le retroazioni esterne sono state sostituite da registri interni.

Il blocco che segue nel funzionamento della macchina è la memorizzazione. I comparatori e i sottocircuiti “Media” all’interno di “Mem Update” elaborano i valori dei sensori e li utilizzano, insieme a quelli già presenti in memoria, per calcolare i nuovi dati da salvare. Inizialmente si era pensato di racchiudere entrambe le fasi in un unico circuito, tuttavia la retroazione dei segnali causava problemi di ridondanza. Nel calcolo di minimo si è scelto di impostare inizialmente il suo valore a 100 anziché escludere lo 0, per evitare problemi di aggiornamento se un sensore dovesse raggiungere lo 0.

Una volta immagazzinati i dati, il passaggio successivo è il confronto tra i valori dei sensori e quelli delle soglie impostate dall’utente. Quindi la memoria manda i valori delle soglie al sottocircuito “Sens cont” che esegue la comparazione con i numeri che provengono da “Sensori” e manda i risultati direttamente ai led di segnalazione.

L’ultimo apparato è quello di visualizzazione. Questa porzione di circuito sfrutta tutti i segnali provenienti dagli altri blocchi e li visualizza. In primis, “Switch scritte” e i controller di riga, generano tre segnali; questi vengono sfruttati in due ambiti. I tre circuiti “Scritte” utilizzano tali segnali per determinare le scritte e controllare i display; ciò ha causato alcuni problemi dati dal limite di bit per output che impone Logisim, alla fine si è scelto di creare un output per ogni display. Il codice dai controller viene utilizzato anche per la selezione dei dati che viene effettuata internamente a “Hub switch numeri”, che tramite “Mux valori” e “Switch numeri” seleziona quale numero (che riceve in input dalla memoria) mandare in uscita.

Prima della visualizzazione però, la rappresentazione binaria di ogni numero deve essere elaborata da “Num to display” che genera 3 stringhe binarie per il controllo dei display esadecimali.

Il blocco di display sensori, riceve in ingresso i valori generati da “Sensori”, li converte tramite “Num to display” e li visualizza su display esadecimali.

I problemi di compatibilità di dato sono stati principalmente due:

1. Per il calcolo della media non bastano 7 bit. Questo è stato risolto semplicemente aumentando la dimensione di tutti i componenti.
2. Logisim non offre alcun componente che permetta di visualizzare un numero in binario; perciò si è dovuto creare un apposito circuito che prendesse in input il numero e lo “convertisse” nella rappresentazione binaria cifra per cifra.

NOTE FINALI

Indubbiamente il circuito che ha richiesto più tempo per la sua realizzazione è stato “Num to display”; infatti contiene tantissimi collegamenti e oltretutto prima di notare il fatto che si potessero “raggruppare” i numeri si è dovuta compilare una tabella piuttosto estesa.

Tuttavia i circuiti più problematici sono stati “Sensori” e “Media”;

* “Sensori” ha richiesto diverso tempo per la progettazione, prima di tutto per far in modo di avere la retroazione (inizialmente si ha avuto alcuni problemi di loop) e poi per evitare che il numero generato andasse fuori range;
* “Media”; di primo acchito è risultato difficile immaginare un circuito che calcolasse continuamente la media e fornisse in uscita sempre un dato valido. Dopo diversi tentativi e miglioramenti, si è giunti ad una versione il cui unico limite è la capacità di memorizzazione dei registri interni.

Il “percorso” che seguono i segnali all’’interno del circuito, rispecchia ciò che è stato immaginato all’inizio della progettazione. Naturalmente sono state effettuate “deviazioni” inevitabili (come lo scorporamento della memoria). In linea di massima però si è riuscito a creare un sistema semplice, funzionale ma poco flessibile.

MIGLIORAMENTI

Ci sono diverse estensioni e miglioramenti possibili:

1. Un sistema che verifichi che le soglie immesse siano all’interno del range;
2. L’opzione di impostare un range a piacere (bisognerebbe aggiungere ulteriori 6 registri nella memoria);
3. Rendere flessibile il circuito “Num to display”;
4. Sostituire tutti i display con un unico display TTY (si aggrava il problema della visualizzazione dei numeri);
5. Permettere l’inserimento delle soglie in decimale (servirebbe un circuito che converta da decimale a binario);
6. Dare la possibilità d’impostare un’isteresi piuttosto che una soglia (ulteriore ampliamento della memoria);
7. Sostituire la batteria di registri che compongono “Memoria” con un componente RAM (complicherebbe molto il controllo di scrittura).

ALLEGATI

Tabella codifica scritte



Tabella codifica per controller riga

|  |  |  |  |
| --- | --- | --- | --- |
| Codice | Codice riga 1 | Codice riga 2 | Codice riga 3 |
| 000 | 00000 | 00001 | 00010 |
| 001 | 00100 | 00101 | 00110 |
| 010 | 01000 | 01001 | 01010 |
| 011 | 01100 | 01101 | 01110 |
| 100 | 00000 | 00100 | 01000 |
| 101 | 00001 | 00101 | 01001 |
| 110 | 00010 | 00110 | 01010 |