Progetto Finale di Reti Logiche

a.a. 2020/2021

Alessandra Moro
codice persona: 10620673
alessandra.moro@mail.polimi.it

Alberto Mosconi
codice persona: 10653349
albertomaria.mosconi@mail.polimi.it

Indice

1 Introduzione

1.1 La specifica del problema

Lo scopo del progetto è la realizzazione di un componente hardware per svolgere l'equalizzazione dell'istogramma di una immagine in scala di grigi.

Questa tecnica permette di manipolare i livelli di grigi dei pixel di una data immagine in modo da incrementarne il contrasto. La figura 1 illustra il prima e il dopo di questo processo.

Figure 1: Confronto prima (sinistra) e dopo (destra) l'equalizzazione [wikipedia]

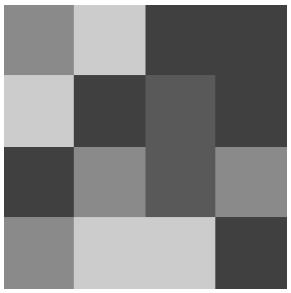
Il componente sviluppato si interfaccia con una memoria RAM, con indirizzi di 16 bit e celle da 8 bit, da cui legge prima le due dimensioni dell'immagine, e poi sequenzialmente tutti i valori dei pixel.

Sempre sulla RAM, dopo aver concluso l'elaborazione, scrive i nuovi valori dei pixel.

INDIRIZZO	VALORE	NOTE
0	4	`numero di colonne`
1	4	`numero di righe`
2	138	`primo byte immagine`
3	204	
4	64	
5	64	
6	204	
7	89	
...	...	

1.2 Un esempio di equalizzazione

Si prende come esempio l'immagine sottostante, di dimensioni 4x4 pixel, e si vuole svolgere il processo di equalizzazione.



138	204	64	64
204	64	89	64
64	138	89	138
138	204	204	64

Figure 2: A sinistra l'immagine e a destra i valori dei pixel corrispondenti.

Per ricalcolare il valore di ogni pixel bisogna prima trovare il massimo e minimo valore che possono assumere nell'immagine originale, e calcolare la differenza tra i due.

$$\texttt{DELTA\_VALUE} = \texttt{MAX\_PIXEL\_VALUE} - \texttt{MIN\_PIXEL\_VALUE}$$

Nel caso dell'esempio si ha che il minimo livello di grigio assunto da un pixel è pari a 64, e dunque `MIN_PIXEL_VALUE=64`, e il massimo è 204, `MAX_PIXEL_VALUE=204`.
Ricavo quindi che `DELTA_VALUE=140`.
A questo punto si può ricavare il valore `SHIFT_LEVEL`, che serve per gli step successivi.

$$\texttt{SHIFT\_LEVEL} = (8 - \texttt{floor}(\log_2(\texttt{DELT\_VALUE} + 1)))$$

Nel caso dell'esempio:

$$\texttt{SHIFT\_LEVEL} = (8 - \texttt{floor}(\log_2(141))) = (8 - \texttt{floor}(7.139)) = (8 - 7) = 1$$

Ora, per ognuno dei pixel che compongono l'immagine, si trova il nuovo valore temporaneo:

$$\texttt{TEMP\_PIXEL} = (\texttt{CURRENT\_PIXEL\_VALUE} - \texttt{MIN\_PIXEL\_VALUE}) << \texttt{SHIFT\_LEVEL}$$

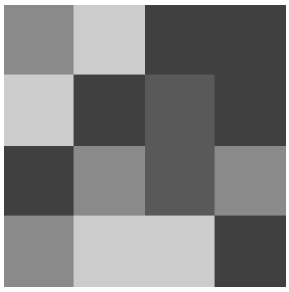
E si salva il nuovo valore limitandolo ad un massimo di 255:

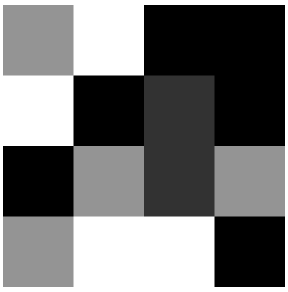
$$\texttt{NEW\_PIXEL\_VALUE} = \texttt{min}(255, \texttt{TEMP\_PIXEL})$$

La seguente tabella mostra questi calcoli applicati a tutta l'immagine.

CURRENT_PIXEL_VALUE	TEMP_PIXEL	NEW_PIXEL_VALUE
138	148	148
204	280	255
64	0	0
64	0	0
204	280	255
64	0	0
89	50	50
64	0	0
64	0	0
138	148	148
89	50	50
138	148	148
138	148	148
204	280	255
204	280	255
64	0	0

Sono stati ottenuti così tutti i nuovi valori dei pixel, e l'immagine è stata equalizzata.





148	255	0	0
255	0	50	0
0	148	50	148
148	255	255	0

Figure 3: In ordine, l'immagine originale, l'immagine equalizzata e i valori dei suoi pixel corrispondenti.

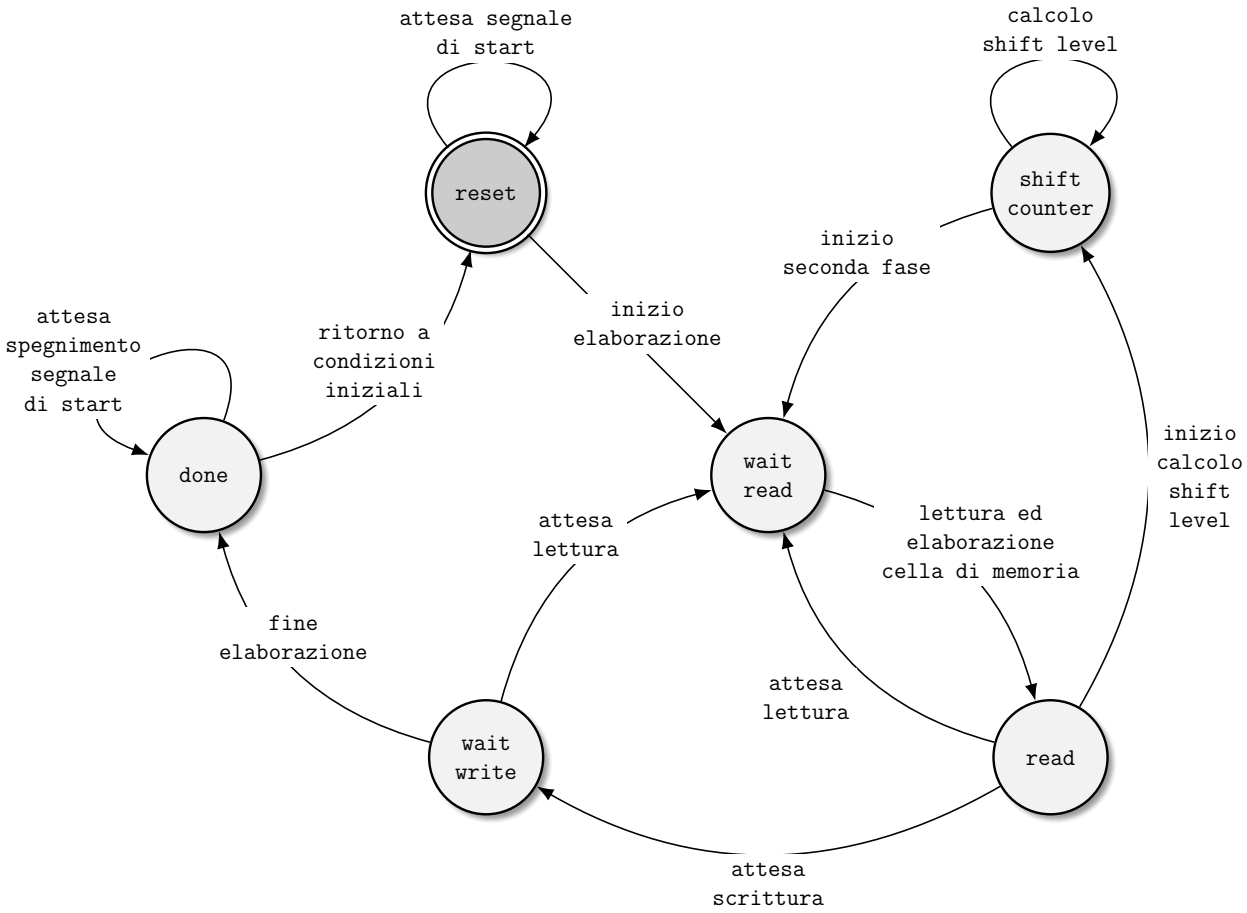
2 Architettura

Il processo di equalizzazione si divide in due fasi:

- una prima fase in cui vengono cercati e riconosciuti il valore massimo ed il valore minimo tra i pixel costituenti l'immagine ed infine calcolato lo shift level,
- una seconda fase in cui, ripartendo dall'indirizzo contenente il primo byte, vengono calcolati i valori dei nuovi pixel e scritti nelle celle di memoria della RAM.

Per attuare queste due fasi, il modulo implementato si comporta come una Macchina a Stati Finiti (FSM).

2.1 La macchina a stati finiti



2.1.1 Prima fase

Partendo dallo stato di `RESET`, in cui tutti i segnali corrispondono al loro valore di default, si attende che il segnale di start venga portato a 1.

Quando ciò si verifica, sequenzialmente, iniziano la lettura ed il salvataggio delle dimensioni di righe e colonne ed il confronto tra i valori dei pixel per determinarne un massimo ed un minimo. Questo processo avviene con l'alternarsi di due stati.

- Il primo, `WAIT READ`, ha lo scopo di attendere che la RAM legga il valore richiesto e lo restituisca in output.
- Il secondo, `READ`, invece, riceve in input il valore letto dalla cella di memoria RAM, e si occupa di salvare rispettivamente il numero di colonne, il numero di righe, ed i valori di massimo e di minimo.

Dopo aver stabilito massimo e minimo, avviene il passaggio allo stato `SHIFT COUNTER`, dove inizia il calcolo del valore di shift (shift level).

Riprendendo la formula per calcolare lo shift level, definita nella specifica dell'algoritmo,

$$\texttt{SHIFT\_LEVEL} = (8 - \texttt{floor}(\log_2(\texttt{DELTA\_VALUE} + 1)))$$

si può notare che il sottraendo $\texttt{floor}(\log_2(\texttt{DELTA\_VALUE} + 1))$ equivale al minimo numero di bit necessari per rappresentare $(\texttt{DELTA\_VALUE} + 1)$.

Pertanto, servendosi di una variabile `pos_count` inizializzata al numero massimo di bit (8), si analizza il vettore $(\texttt{DELTA\_VALUE} + 1)$ partendo dal bit più significativo, ovvero quello in posizione `pos_count`.

Finché si incontra un bit uguale a 0, `pos_count` viene decrementato di 1 e si rimane sullo stato `SHIFT COUNTER`.

Nell'istante in cui si incontra il bit più significativo pari ad 1, `pos_count` non viene ulteriormente decrementato poiché il suo valore corrisponde a quello cercato.

Svolti questi passaggi, si calcola lo shift level e si inizia la seconda fase tornando allo stato di `WAIT READ`.

2.1.2 Seconda fase

Ripartendo dall'indirizzo contenente il primo byte, si inizia un processo analogo alla prima fase, leggendo iterativamente tutti i byte dell'immagine.

Ogni valore letto viene utilizzato nello stato `READ` per il calcolo del pixel equalizzato, come indicato nella specifica dell'algoritmo.

Ottenuto il nuovo valore, passando attraverso lo stato `WAIT WRITE`, si attende la scrittura dello stesso nella prima cella di memoria libera, il cui indirizzo si può ricavare facilmente conoscendo le dimensioni dell'immagine.

Supponendo che il byte letto si trovi ad un indirizzo X, l'indirizzo X_1 di scrittura risulta:

$$X_1 = X + (n\_righe \times n\_colonne)$$

In seguito all'avvenuta scrittura, si torna allo stato WAIT READ per la lettura del successivo byte.

Esauriti tutti i byte dell'immagine, si è concluso il processo di equalizzazione e dallo stato di WAIT WRITE si passa direttamente allo stato DONE portando ad 1 il segnale o_done.

La permanenza in questo stato termina non appena il segnale di start viene spento; a questo punto si torna nello stato RESET in attesa della prossima immagine.

2.2 Implementazione in VHDL

La FSM è implementata con un singolo processo, sensibile agli eventi di clock e di reset. Se il segnale di reset viene portato ad 1 in qualsiasi momento, la macchina è riportata nello stato RESET, riassegnando ad ogni segnale il proprio valore default.

Sono elencati di seguito i segnali e le variabili utilizzati.

SEGNALE/VARIABILE	DETTAGLI	FUNZIONE
n_col	vector, 8bit	Numero di colonne dell'immagine
n_rig	vector, 8bit	Numero di righe dell'immagine
current_address	vector, 16bit	Indirizzo cella in lettura
current_state	state_type	Stato corrente della FSM
max_pixel_value	vector, 8bit	Valore massimo assunto dai pixel dell'immagine
min_pixel_value	vector, 8bit	Valore minimo assunto dai pixel dell'immagine
shift_level	vector, 4bit	Numero di posizioni di shift, usato nel ricalcolo dei valori dei pixel
pos_count	integer	Contatore usato per ricavare il numero minimo di bit necessari per rappresentare il vettore delta_plus_one
second_phase	single bit	Assume valore 0 durante la prima fase e valore 1 durante la seconda fase
delta_plus_one	vector, 9bit	Variabile contenente la differenza tra valore massimo e minimo dei pixel incrementata di 1
temp_pixel_value	vector, 9bit	Variabile utilizzata per passaggi intermedi durante il ricalcolo del valore dei pixel

2.2.1 Osservazioni

1. I valori dei pixel dell'immagine sono rappresentati su 8 bit, con un massimo, quindi, di 255. Di conseguenza anche la `DELTA VALUE` definita nell'algoritmo ha una dimensione di 8 bit. Tuttavia, essendo necessario incrementarla di 1, ne consegue che `delta_plus_one` debba comprendere 9 bit.

2. Durante il calcolo del nuovo valore dei pixel, è possibile che questo superi 255. Risulta quindi necessario che la variabile `temp_pixel_value` sia di 9 bit per salvare il valore temporaneo prima di imporre il limite massimo a 255.

3. Per evitare comportamenti imprevedibili durante la sintesi del circuito, all'inizio dell'esecuzione del processo è stato scelto di assegnare esplicitamente ad ogni segnale il proprio valore precedente. Questo verrà poi adeguatamente sovrascritto a seconda dello stato corrente. In questo modo viene preventivamente impedita la generazione automatica da parte di Vivado di *inferred latches*.

3 Risultati Sperimentali

3.1 Sintesi

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui varius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor sed augue. Nulla nec lacus.

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

4 Simulazioni

4.1 TestBench 1

Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

4.2 TestBench 2

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

Etiam ac leo a risus tristique nonummy. Donec dignissim tincidunt nulla. Vestibulum rhoncus molestie odio. Sed lobortis, justo et pretium lobortis, mauris turpis condimentum augue, nec ultricies nibh arcu pretium enim. Nunc purus neque, placerat id, imperdiet sed, pellentesque nec, nisl. Vestibulum imperdiet neque non sem accumsan laoreet. In hac habitasse platea dictumst. Etiam condimentum facilisis libero. Suspendisse in elit quis nisl aliquam dapibus. Pellentesque auctor sapien. Sed egestas sapien nec lectus. Pellentesque vel dui vel neque bibendum viverra. Aliquam porttitor nisl nec pede. Proin mattis libero vel turpis. Donec rutrum mauris et libero. Proin euismod porta felis. Nam lobortis, metus quis elementum commodo, nunc lectus elementum mauris, eget vulputate ligula tellus eu neque. Vivamus eu dolor.

4.3 TestBench 3

Nulla in ipsum. Praesent eros nulla, congue vitae, euismod ut, commodo a, wisi. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nonummy magna non leo. Sed felis erat, ullamcorper in, dictum non, ultricies ut, lectus. Proin vel arcu a odio lobortis euismod. Vestibulum ante ipsum primis in faucibus orci luctus

et ultrices posuere cubilia Curae; Proin ut est. Aliquam odio. Pellentesque massa turpis, cursus eu, euismod nec, tempor congue, nulla. Duis viverra gravida mauris. Cras tincidunt. Curabitur eros ligula, varius ut, pulvinar in, cursus faucibus, augue.

Nulla mattis luctus nulla. Duis commodo velit at leo. Aliquam vulputate magna et leo. Nam vestibulum ullamcorper leo. Vestibulum condimentum rutrum mauris. Donec id mauris. Morbi molestie justo et pede. Vivamus eget turpis sed nisl cursus tempor. Curabitur mollis sapien condimentum nunc. In wisi nisl, malesuada at, dignissim sit amet, lobortis in, odio. Aenean consequat arcu a ante. Pellentesque porta elit sit amet orci. Etiam at turpis nec elit ultricies imperdiet. Nulla facilisi. In hac habitasse platea dictumst. Suspendisse viverra aliquam risus. Nullam pede justo, molestie nonummy, scelerisque eu, facilisis vel, arcu.

5 Conclusioni

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

Nulla non mauris vitae wisi posuere convallis. Sed eu nulla nec eros scelerisque pharetra. Nullam varius. Etiam dignissim elementum metus. Vestibulum faucibus, metus sit amet mattis rhoncus, sapien dui laoreet odio, nec ultricies nibh augue a enim. Fusce in ligula. Quisque at magna et nulla commodo consequat. Proin accumsan imperdiet sem. Nunc porta. Donec feugiat mi at justo. Phasellus facilisis ipsum quis ante. In ac elit eget ipsum pharetra faucibus. Maecenas viverra nulla in massa.