Documentazione del progetto di reti logiche 2017/2018.

Leonardo Barilani - Numero Matricola: 848050 - Codice Persona: 10528015
William Bonvini - Numero Matricola: 847640 - Codice Persona: 10526185

Specifica

La specifica del progetto consiste nel sintetizzare un componente hardware che, data in ingresso un'immagine in scala di grigi restituisca un'area di un rettangolo circoscritto nella stessa.
L'immagine viene rappresentata come una matrice di numeri interi (8 bit) , ogni valore identifica una sfumatura della scala di grigi.
Vengono forniti in input la lunghezza e la larghezza della matrice, un valore di soglia e la matrice stessa.
L'area circoscritta corrisponde alla minima area all'interno della matrice in cui sono contenuti tutti i valori uguali o superiori al valore di soglia.

Scelte Progettuali

L'algoritmo scelto per la realizzazione della specifica consiste nel calcolare l'area totale dell'immagine per poi individuare all'interno della stessa 4 sotto-aree *(top_area, bot_area, lft_area, rgt_area)* da sottrare a quella totale.
Più specificatamente, ognuna delle sotto-aree viene calcolata a partire dai valori più esterni della matrice fornita in ingresso e contiene solo valori della stessa che non superano il valore di soglia.
Per rendere più chiaro l'algoritmo ci avvaliamo di un esempio:
Di seguito mostriamo una rappresentazione semplificata della matrice, i cui unici valori possibili sono 0 o 1.
Lo 0 corrisponde a un valore minore della soglia, l'1 corrisponde a un valore maggiore o uguale.

0	0	0	0	0	0
0	0	1	0	0	0
1	0	1	1	0	0
0	1	1	0	0	0
1	1	1	1	0	0
0	0	0	0	0	0

top_area
rgt_area
bot_area

Sottraendo le aree colorate dall'area totale si ottiene l'area non evidenziata, ovvero l'area richiesta dalla specifica.
In VHDL sono stati realizzati 5 componenti, 4 di essi per calcolare le aree esterne ($comp\_top, comp\_bot, comp\_lft, comp\_rgt$) e uno per calcolare l'area totale dell'immagine ($multiplier$).

Ottimizzazioni

L'algoritmo prevede l'esecuzione in parallelo dei componenti *multiplier e comp_top*, in quanto indipendenti tra loro.
Infatti *comp_top* riceve in ingresso la lunghezza e l'altezza dell'immagine, e quindi non necessita dell'area totale.
Inoltre, i componenti comp_*lft* e comp_*rgt*, poiché eseguiti dopo comp_*top* e *comp_bot*, sono stati ottimizzati per effettuare la scansione a partire dalla prima riga non appartenente ad *area_top* fino all'ultima riga non appartenente ad *area_bot.*
Riavvalendoci della matrice semplificata abbiamo:

0	0	0	0	0	0
0	0	1	0	0	0
1	0	1	1	0	0
0	1	1	0	0	0
1	1	1	1	0	0
0	0	0	0	0	0

area_top
area_rgt
area_bot
riga di inizio computazione di comp_*lft* e comp_*rgt*
riga di fine computazione di comp_*lft* e comp_*rgt*

Descrizione dello schema funzionale

Da ogni stato si passa allo stato S_BEGIN nel caso in cui venga alzato il segnale *i_rst*.
Non è stato incluso nello schema dell' FSA per maggior chiarezza dello stesso.

Far riferimento alla macchina a stati finiti alla pagina precedente.

Inizializzazione

Il programma inizia la computazione nello stato S_BEGIN, al fronte di discesa del clock quando il segnale di reset è a 1.
In questo stato vengono settati a zero i segnali di *o_done, o_en, o_we, start_top* e azzerati tutti i bit degli indirizzi presenti nelle variabili *width, height,* e *threshold.*
Quando il segnale *i_start* passa da basso ad alto viene alzato il segnale di *o_en* e ci si sposta nello stato S_WIDTH**,** che si salva nella variabile *width* la larghezza dell'immagine, per poi passare allo stato S_HEIGHT, dove viene salvata nella variabile *height* l'altezza.
Di conseguenza viene alzato il segnale di *multiply* che scaturisce l'inizio del processo *Multiplier.*
Dallo stato di S_HEIGHT si passa allo stato S_THRESHOLD**.**
Nel caso in cui la larghezza o la lunghezza dell'immagine siano nulle, vengono impostate a 0 i segnali *tmp_area* e *o_data* e si passa allo stato END_0, che verrà esplicitato nella fase di salvataggio, altrimenti viene salvata nella variabile *threshold* il valore della soglia e viene alzato il segnale *start_top,* che ha il compito di avviare la fase di calcolo e scaturisce l'avvio del componente *comp_top*.
A questo punto si passa allo stato S_WAITING_TOP.

Calcolo

Durante questa fase vengono individuate le quattro aree *area_top, area_bot, area_lft, area_rgt.*
Di seguito vengono descritte le operazioni eseguite in ogni stato.

S_WAITING_TOP:
Nello stato S_WAITING_TOP si aspetta la fine del calcolo della top_area.
alla salita del segnale di *end_top*, se l'area calcolata dal componente top è uguale all'area totale dell'immagine viene inviata in output la dimensione dell'immagine, pari a zero (*o_data=00000000),* e si passa allo stato S_END_0.
Se invece l'area calcolata (*top_area*) è inferiore a quella totale, viene salvata in *tmp_area* la differenza tra l'area totale e *top_area,* viene scaturito l'inizio del componente *comp_bot* e si passa allo stato di S_WAITING_BOT**.**

S_WAITING_BOT:
Una volta terminato il calcolo della *bot_area,* ovvero, alla salita del segnale *end_bot,* viene salvata in *tmp_area* la differenza tra la *tmp_area* stessa e la *bot_area,* viene alzato il segnale di *start_left,* così scaturendo l'inizio della computazione di *comp_lft* e si passa allo stato S_WAITING_LFT.

S_WAITING_LFT:
Alla salita del segnale *end_lft* viene aggiornata l'area temporanea e viene alzato il segnale di *start_rgt,* che scaturisce l'inizio del componente *comp_rgt* e il passaggio allo stato S_WAITING_RGT.

S_WAITING_RGT:
Alla salita del segnale *end_rgt* viene aggiornata l'area temporanea e si passa allo stato S_WAITING_RES, che sancisce l'inizio della fase di salvataggio.

Salvataggio

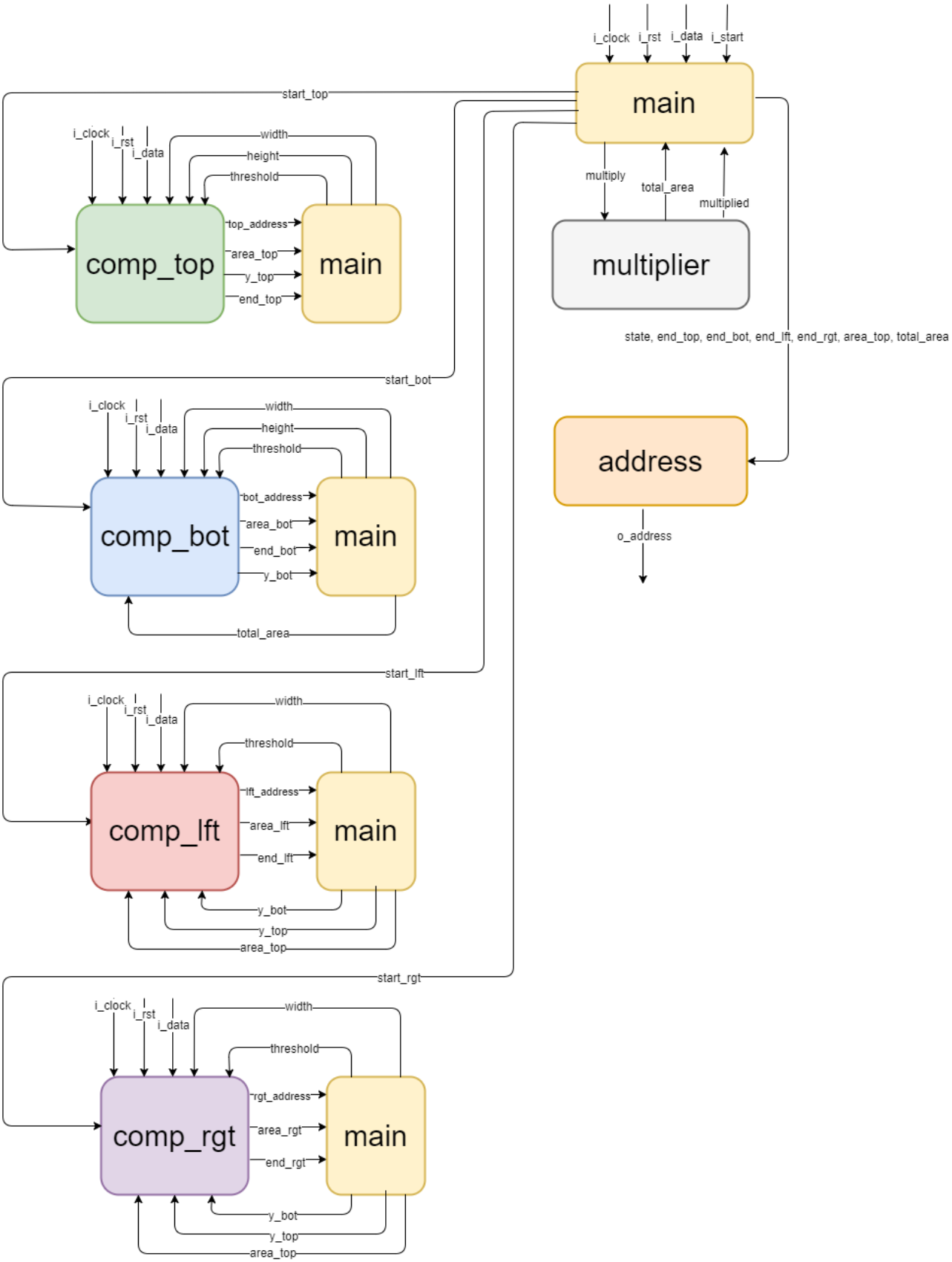
S_WAITING_RES:
Nello stato di S_WAITING_RES vengono inviati in output, tramite *o_data,* gli 8 bit meno significativi dell'area temporanea (*tmp_area),* che ora corrisponde all'area che circoscrive l'area di interesse.
Dopodiché si passa allo stato S_END_0.

S_END_O:
In questo stato vengono inviati in output, tramite *o_data,* gli 8 bit più significativi dell'area di interesse per poi passare allo stato S_END_1.

S_END_1:
Vengono abbassati i segnali *o_en* e *o_we,* e viene alzato il segnale *o_done.*
Si passa allo stato S_END_2.

S_END_2:
Questo stato è stato introdotto in quanto è necessario che l'algoritmo torni disponibile per una nuova computazione.
Durante la sua esecuzione il segnale *o_done* viene riportato a zero.

Schema a blocchi



Far riferimento all'immagine alla pagina precedente per una maggiore comprensione del seguente testo.
Il processo *main* è unico, è ripetuto più volte all'interno dello schema solo per semplificarne la comprensione.

Lo schema a blocchi è formato da due processi (*main, multiplier)*, un multiplexer (*address)*, e quattro componenti (*comp_top,comp_bot,comp_lft,compr_rgt)*.

Main
Il *main* è un processo interno al componente principale *project_reti_logiche* che gestisce quale componente avviare e quali segnali dargli in ingresso.

Address
è uno switch che, dato in ingresso uno stato, e dei segnali interni al componente *project_reti_logiche*, restituisce l'indirizzo in cui leggere o scrivere un'informazione.

Multiplier
processo interno al componente *project_reti_logiche* adibito al calcolo dell'area totale dell'immagine, moltiplicando lunghezza e altezza della stessa.

Flusso
Alla salita del segnale esterno di *i_start*, il *main* alza i segnali di *start_top* e di *multiply*, così attivando rispettivamente *compt_top* e *multiplier*.
comp_top non ha accesso diretto alla RAM, quindi richiede al *main* di accedere alla RAM al posto suo utilizzando il segnale di output *top_address*
Riceverà il valore della cella analizzata dal segnale di input *i_data.*
Il componente comunica al *main* la fine della computazione alzando il segnale *end_top* e restituendo *y_top*, ovvero la prima riga che andrà analizzata dai componenti *comp_lft* e *comp_bot*, e *area_top*, ovvero l'area calcolata dal componente.

Gli altri componenti si comportano in modo analogo tranne per qualche differenza.
comp_bot riceve in ingresso *total_area* perché parte dall'ultima riga della matrice, e necessita di saltare al suo indirizzo per iniziare la computazione.
Inoltre restituisce al *main* il segnale *y_bot*, che indica l'ultima riga che andrà analizzata dai componenti *comp_lft* e *comp_bot.*
comp_lft e *comp_rgt* ricevono ovviamente dal *main* i segnali di *y_bot* e *y_top* per il motivo già esplicitato.
Rispetto a *comp_top* e *comp_bot* non ricevono in ingresso l'altezza della matrice perché è un'informazione superflua, ottenibile dalla sottrazione tra *y_bot* e *y_top*, ma ricevono *area_top* in quanto necessaria per calcolare l'offset dell'indirizzo da quale inizia la loro computazione.

Struttura interna dei componenti
Analizzando dall'interno i componenti addetti al calcolo delle sotto-aree, abbiamo deciso di caratterizzare ognuno di essi coi seguenti segnali:

- *state*
  in cui viene salvato lo stato della computazione (S_BEGIN; S_COUNT; S_END)
- *zero*
  indirizzo assoluto che individua il primo indirizzo della memoria che andrà letto dal componente (una cella della matrice).
- *x*
  il numero della colonna che si sta esaminando (utilizzato per capire quando una riga è terminata)
- *y*
  il numero della riga che si sta esaminando (utilizzato per capire quando una colonna è terminata)
- *tmp_address*
  l'indirizzo assoluto usato per accedere alla RAM.
- *o_area*
  segnale in cui viene salvata l'area calcolata dal componente.

Ogni componente, dopo aver ricevuto un segnale *i_rst* si sposta nello stato S_BEGIN e inizializza a zero ogni bit dei segnali *x, y, tmp_area, o_area, o_end.*
Nei componenti *comp_top* e *comp_bot* è presente anche un'uscita o_y in cui viene salvata l'ultima riga appartenente all'area del componente.

Componente top
Quando viene alzato il segnale *i_start*, viene settato il *tmp_address* uguale all'indirizzo salvato in *zero* (ovvero il primo indirizzo in cui vengono memorizzati i valori dell'immagine) per poi passare allo stato **S_COUNT**.
In questo stato viene incrementata la variabile *tmp_address* fintanto che non si trova un valore maggiore o uguale alla soglia.
Ogni volta che si raggiunge il termine di una riga viene incrementata l'altezza dell'area (y), e non appena viene trovato un valore maggiore o uguale alla soglia, o quando l'altezza dell'area del componente corrisponde all'altezza dell'immagine, si pone *o_end=1*, *o_y=y* e si passa allo stato **S_END**.

La descrizione degli altri componenti è analoga.
Le uniche differenze consistono nell'inizializzazione del segnale zero e nella direzione di scorrimento della matrice.
Più specificatamente:
Il componente *comp_bot* inizializza il segnale *zero* all'indirizzo dato dalla somma tra il primo indirizzo in cui vengono memorizzati i valori della matrice e l'area totale dell'immagine, meno la larghezza della stessa e meno uno.
Questo componente scorre la matrice dunque partendo dall'ultima riga e decrementando l'indirizzo d'accesso finché non trova un valore maggiore o uguale alla soglia.
il componente *comp_lft* inizializza l'indirizzo *zero* all'indirizzo dato dalla somma tra il primo indirizzo in cui vengono memorizzati i valori della matrice e l'area calcolata da *comp_top* e scorre la matrice in verticale, fintanto che non arriva a *y_bot*, per poi cambiare colonna.
il componente *comp_rgt* inizializza l'indirizzo *zero* all'indirizzo dato dalla somma tra il primo indirizzo in cui vengono memorizzati i valori della matrice, l'area calcolata da *comp_top*, la larghezza dell'immagine meno uno.
Scorre la matrice in verticale, a partire dall'ultima colonna, e decrementa la colonna d'accesso quando arriva a *y_bot*. Itera finché non trova un valore maggiore o uguale alla soglia.

Test

I test sono stati eseguiti per verificare il corretto funzionamento dell'algoritmo in situazioni limite.
I test da noi pensati sono:

- Matrice nulla
  - Viene fornita in ingresso una matrice di altezza e lunghezza pari a zero.
- Matrice vuota
  - Viene fornita in ingresso una matrice la cui area d'interesse è nulla, ovvero tutti i valori al suo interno sono inferiori al valore di soglia.
- Matrice piena
  - Viene fornita in ingresso una matrice la cui area d'interesse corrisponde all'area totale.
- Matrice grande
  - Viene fornita in ingresso una matrice di lunghezza e altezza pari a 8
- Matrice piccola
  - Viene fornita in ingresso una matrice di lunghezza e altezza pari a 5.
- Riga
  - Viene fornita in ingresso una matrice la cui area d'interesse ha altezza unitaria.
- Colonna
  - Viene fornita in ingresso una matrice la cui area d'interesse ha lunghezza unitaria.