APPUNTI SIS:

COMANDI editor di testo:

* .model🡪serve per dare un nome al file
* .inputs🡪serve per inizializzare i nomi degli input
* .outputs🡪serve per inizializzare i nomi degli output
* .names🡪mi permette di scrivere dei nodi logici (logic gates), unità che al suo interno ha delle logiche. In queste porte logiche posso assegnare una sola porta di output.
* .end🡪chiude il componente
* .exdc🡪è il comando che precede il .names di un don’t care set, il quale verrà poi descritto con un on set.

COMANDI bash:

* (rl)read\_blif🡪serve per leggere il file blif che vogliamo eseguire
* (sim)simulate…🡪serve per simulare le nostre porte di input (in ordine!)
* write\_eqn🡪scrive l’equazione booleana in somma di prodotti
* print\_stats🡪posso vedere le statistiche del componente dove pi->porte di input, po->porte di output, node->ci dice quanti nodi ci sono nel componente, latches->quanti registri ci sono nel componente, lits->quanti letterali abbiamo.
* full\_simplify🡪esegue l’algoritmo di Q&Mk.
* write\_blif🡪se non gli si passa alcun parametro lui stamperà a video il file blif.
* print\_level🡪serve per vedere di quanti livelli è il nostro circuito
* sweep🡪eliminazione dei nodi con un’unica linea di ingresso e di nodi con valore costante
* eliminate n🡪elimina un nodo sostituendo la sua espressione in tutti gli altri nodi, l’eliminazione sarà fatta in modo che il numero di letterali non sia maggiore di n. (se si vuole eliminare un nodo utilizzato una sola volta si utilizzerà -1)
* fx(estrazione)🡪estrae sottoespressioni comuni a più nodi creandone uno nuovo (il nuovo nodo avrà valore numerico
* per lanciare degli script bisogna scrivere source
* script.rugged🡪script che chiama sweep, eliminate, fx e simplify con una serie di parametri in modo da avere un giusto bilanciamento tra area e ritardo
* set autoexec comando🡪stampa automaticamente il risultato del comando specificato dopo l’esecuzione di un qualunque altro comando.

MAPPING TECNOLOGICO:

l’ottimizzazione va fatta sempre prima del mapping!!

* read\_library🡪carica la libreria tecnologica nel formato genlib(noi utilizzeremo mcnc.genlib e synch.genlib)
* print\_library🡪visualizza le informazioni della libreria caricata
* map🡪esegue l’operazione di mapping, -m 0 permette di ottenere un circuito minimizzato per area, -n 1 un circuito minimizzato per ritardo. L’opzione -s permette di visualizzare le informazioni relative ad area e ritardo dopo il mapping. **Total gate area** fornisce il valore dell’area come numero di celle standard della libreria tecnologica mentre **maximum arrival time** indica il ritardo.
* write\_blif -n🡪 mostra la rappresentazione del circuito associate alle porte della libreria
* print\_delay🡪stampa informazionii relative al ritardo del circuito
* reduce\_depth🡪riduce la lunghezza dei cammini critici(questa è un ottimizzazione, va fatta prima del mapping)
* print\_map\_stats 🡪stampa le statistiche della mappa

MODELLAZIONE FSM:

* dopo l’inserimento degli inputs e outputs bisognerà aprire la descrizione con il comando start\_kiss, alla fine bisognerà chiudere con end\_kiss.
* all’interno della kiss dovremo indicare inizialmente 5 elementi:.i🡪inputs, .o🡪outputs, .s🡪stati, .p🡪numero di transizioni, .r🡪stato di reset. L’ordine di descrizione è ingressi/stato corrente/stato prossimo/uscita.
* Chiusa la kiss avremo 2 possibilità, la prima è un assegnazione manuale di una codifica per gli stati, la seconda è un assegnazione automatica tramite il comando **state\_assign jedi.** Fatto ciò lanceremo il comando **stg\_to\_network.**

I passaggi sono:

1. Modellazione della macchina a stati
2. Ottimizzazione degli stati🡪state\_minimize stamina
3. Codifica degli stati🡪state\_assign jedi
4. stg\_to\_network
5. ottimizzazione🡪source script.rugged
6. mappare il circuito per area o ritardo in base alle esigenze

CONTROLLORE DATAPATH:

.subckt nomeComp paramFromale=paramAttuale …

.search nomeFileComp.blif

Per gestire un latch la keyword è:

.latch <input> <output> <type> <controlo> <init-val>

type🡪modo in cui il registro lavora, decideremo su quale fronte del clock il latch lavora: falling edge(fe), rising edge(re), active high(ah) ,active low(al) o asynchronous(as)

control🡪è il segnale di clock per il registro, NIL quando utilizza il clock interno del circuito in cui è inserito

init-val🡪è lo stato iniziale del latch 0,1,2,3. 2 indica don’t care, 3 indica unknow.