

FPGA Tutorial on DE1 Board

TUTORIAL MODELSIM

Using Quartus II V13.0

Marsi 2014 - University of TRIESTE

Tutorial Modelsim

Simulazioni attraverso Modelsim di circuiti logici

Descrizione: Si utilizzi il tool Modelsim per realizzare diverse simulazioni di circuiti logici descritti in Verilog HDL a diversi livelli di astrazione.

Scopo: familiarizzare col tool di sviluppo Modelsim e col linguaggio di descrizione hardware Verilog HDL.

Apprendimento previsto:

- Descrizione di un semplice circuito logico utilizzando Verilog HDL
- Simulazione del circuito in maniera interattiva (step by step)
- Realizzazione di un file di stimoli e descrizione del medesimo attraverso il linguaggio HDL
- Utilizzo del file di stimoli per una simulazione batch
- Comprensione della differenza tra descrizioni comportamentali, RTL, e strutturali.
- Realizzazione di simulazioni sia a livello RTL che a livello di Gate richiamando Modelsim dall'interno del programma Quartus

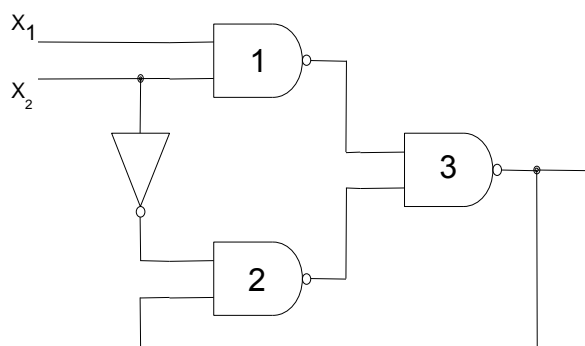
Introduzione a ModelSim

Modelsim [1] è un tool di simulazione sviluppato da Mentor-Graphics [2] che è stato adottato come “state of the art” all'interno di diversi software EDA (Electronic Design Automation) dedicati al progetto di circuiti elettronici. Per cui è piuttosto comune che i produttori di un certo sistema di sviluppo (nel nostro caso Altera) forniscano le interfacce verso software sviluppati da terze parti ad esempio ModelSim per la Simulazione logica, Synopsys per la sintesi HSpice per la simulazione analogica e per l'analisi dei segnali, ...

In questo tutorial analizzeremo le basi per una simulazione di un circuito logico descritto in VerilogHDL utilizzando ModelSim e come questo possa interfacciarsi all'ambiente Quartus II [3-4-5].

Introduzione al problema

Si voglia simulare il circuito qui di seguito riportato.



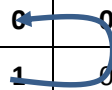
Questo è un circuito sequenziale asincrono che può risultare affetto da un'alea statica [6]. Questo comporta un'interessante criticità: in pratica il funzionamento è fortemente legato dai ritardi di propagazione dei segnali attraverso i vari percorsi e si potrà notare che alterando questi, si viene a modificare il funzionamento del circuito stesso.

Il funzionamento del circuito è descritto da questa tavola di flusso:

st \ x ₁ x ₂	00	01	11	10
0	0	0	1	0
1	1	0	1	1


Si supponga che all'accensione entrambi i segnali d'ingresso x₁ e x₂ siano a 0. L'uscita può assumere in tal caso indifferentemente sia il valore 0 che il valore 1 ed autosostenersi attraverso il loop di reazione (in altre parole al momento di accensione del circuito non c'è modo di stabilire se l'uscita sia a 0 oppure a 1). Se però si alza il segnale x₂ l'uscita assume il valore 0 e questo viene mantenuto anche se successivamente x₂ si abbassa riportandosi alle condizioni iniziali.

st \ x ₁ x ₂	00	01	11	10
0	0	0	1	0
1	1	0	1	1



Viceversa se dopo l'accensione con entrambi gli ingressi a 0 si va ad alzare il segnale x₁ l'uscita assume il valore 1 e lo mantiene anche quando x₁ dovesse tornare ad abbassarsi.

st \ x ₁ x ₂	00	01	11	10
0	0	0	1	0
1	1	0	1	1



Se entrambi i segnali sono a livello alto l'uscita assume il valore 1, valore che dovrebbe essere mantenuto anche quando x₂ si abbassa. Si può però notare che a seconda dei tempi di propagazione potrebbe anche avvenire la transizione sbagliata e l'uscita potrebbe portarsi allo stato basso. In pratica se $\Delta i + \Delta 2 - \Delta 1 > \Delta 3$ l'uscita passerà al valore 0, altrimenti la transizione sarà corretta e l'uscita rimarrà allo stato alto.

st\ x_1x_2	00	01	11	10
0	0	0	1	0
1	1	0	1	1

st\ x_1x_2	00	01	11	10
0	0	0	1	0
1	1	0	1	1

E' pertanto fondamentale poter simulare il circuito analizzando nel dettaglio i tempi di propagazione attraverso i vari elementi del circuito.

Procedimento

Attraverso un comune text editor si realizzi un file scritto in Verilog HDL che descriva (a livello comportamentale) il circuito sopra riportato con l'introduzione (manuale) di eventuali tempi di ritardo nei vari elementi e lo si salvi in un opportuno direttorio.

Il file in questione potrebbe essere il seguente:

```

`timescale 1ps / 1ps

module alea (x1,x2,y);
input x1,x2;
output y;

wire i1,i2,i3;

    assign #20 i1=~(x1 & x2);
    assign #20 i2=~(i3 & y);
    assign #30 i3=~x2;

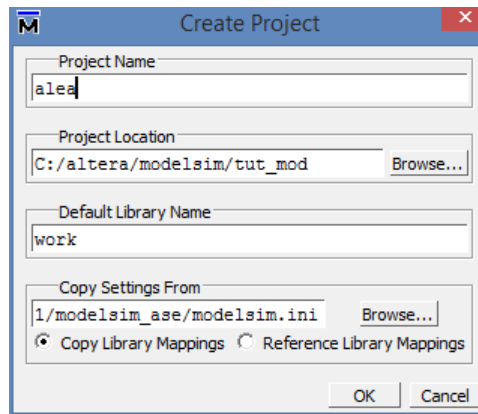
    assign #20 y=~(i1 & i2);
endmodule

```

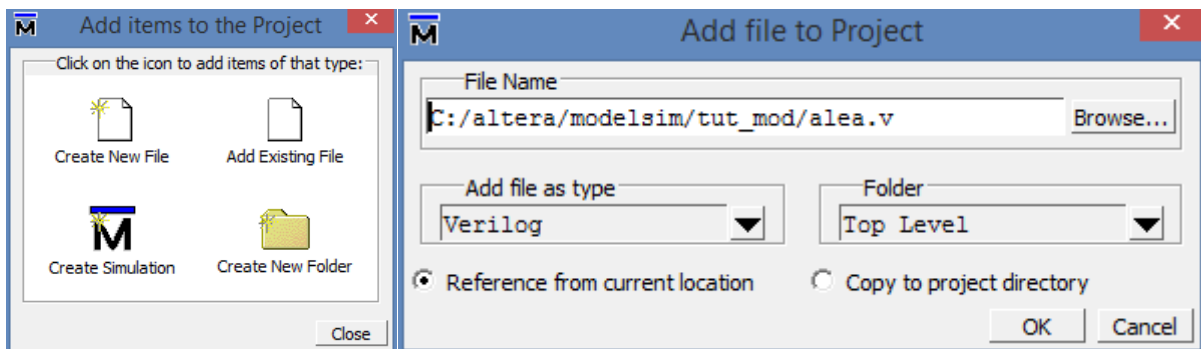
1. Compilazione del sorgente

Si apra il tool Modelsim o tramite icona sul desktop o dall'elenco dei programmi installati

Si crei un nuovo progetto specificandone nome e ubicazione del direttorio nel quale farlo risiedere. Si mantenga pure il nome di default "work" della libreria alla quale appoggiarsi per farvi risiedere i files compilati. Si copi il file di configurazione di default o dal direttorio di installazione o dall'ultimo progetto svolto.



Si aggiunge al progetto il file sviluppato “add existing file”



- > OK
- > Close

Si compila il file in tal modo esso entra a far parte di una delle varie librerie disponibili. Per default è bene utilizzare allo scopo la libreria “work” (che conterrà tutti i moduli del progetto sul quale si sta lavorando).

Compile > Compile All

Eventuali errori di sintassi verranno evidenziati in questa fase. Se non vi sono errori nella libreria “Work” apparirà il modulo “alea”.

Eventualmente per far comparire la scheda relativa verificare che la scheda relativa alla gestione delle librerie sia attiva

View > Library (alt-vu)

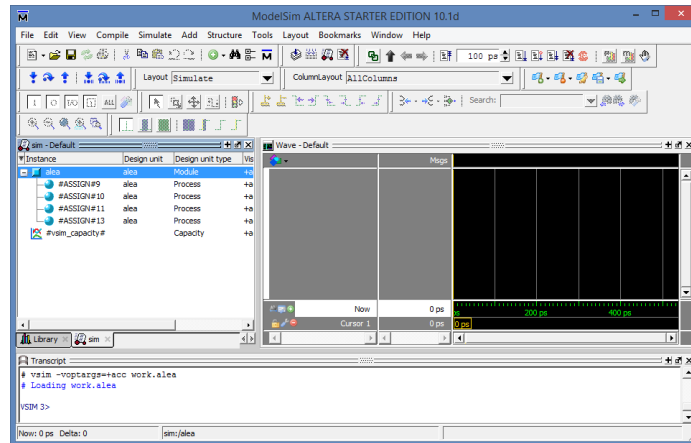
2. Simulazione step by step

All’interno della finestra Library cliccare col tasto destro sopra il modulo alea (all’interno della libreria work) e selezionare **Simulate**.

Compare una nuova scheda orientata alla gestione dei segnali in fase di simulazione.

Accertarsi che sia visibile la schermata per la visualizzazione grafica dei segnali (Wave)

View > Wave (alt-vv)

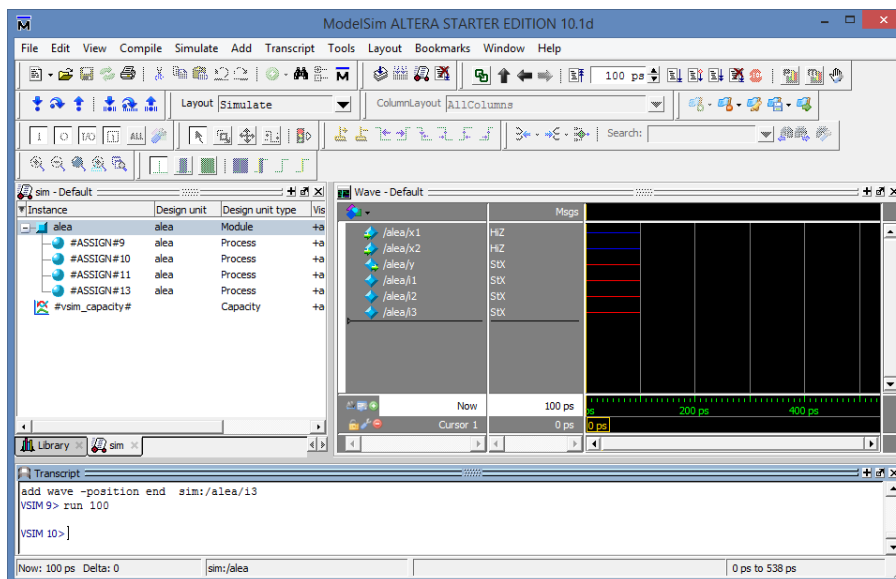


Trascinare l'istanza denominata "alea" dalla finestra sim alla finestra wave. Si noti che appaiono tutti i segnali inerenti tale blocco, compresi i segnali interni.

Nella finestra di Console digitare

```
> run 100
```

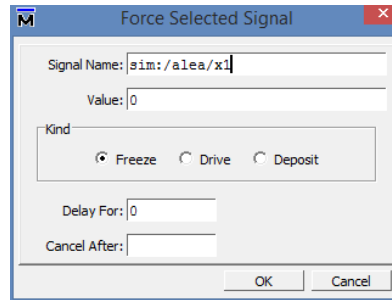
Il sistema esegue i primi 100 passi di simulazione (equivalenti nel nostro caso a 100 ps)



Si noti che inizialmente i segnali di ingresso non sono definiti (highZ) e pertanto i segnali d'uscita nonché i segnali interni risultano interterminati (StX).

Si forzino ora i segnali di ingresso al livello logico 0.

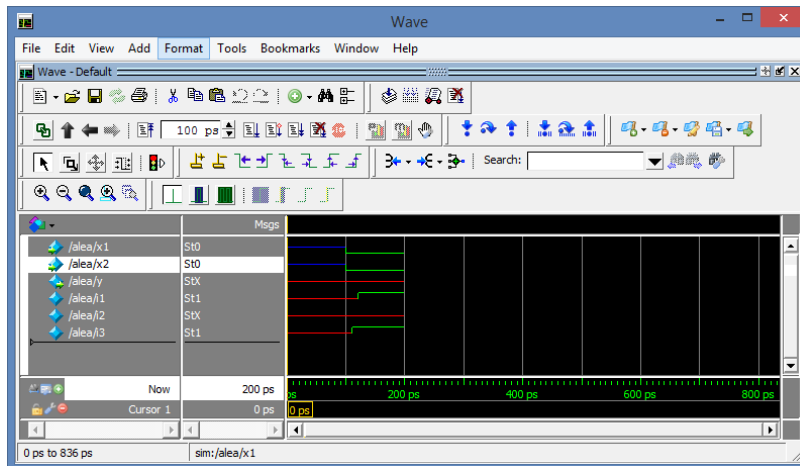
- Right click sul segnale /alea/x1 nella finestra Wave
- Force ...
- Nella finestra che appare si scriva all'interno del campo Value il valore 0
- OK



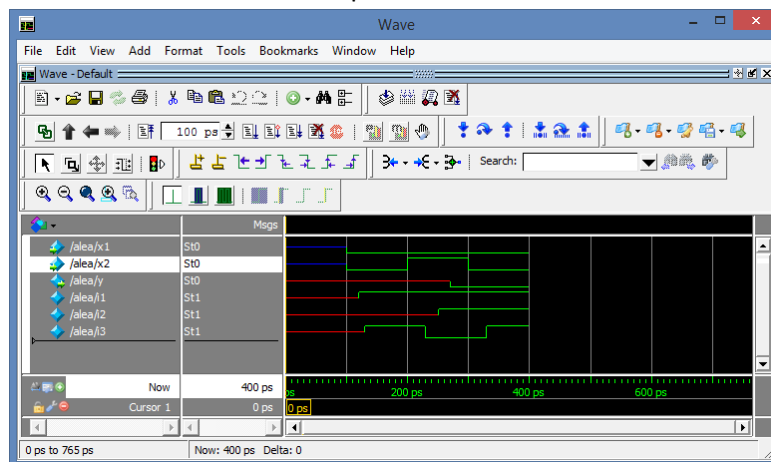
Si replichi la procedura anche per il segnale x2
 All'interno della finestra di Console si digiti

```
> run 100
```

Ora diversi segnali assumono il corretto valore logico dopo un certo intervallo temporale compatibile con i tempi di ritardo introdotti in fase di descrizione del circuito, ma altri (y ed i2) permangono allo stato "StX" (indefinito). Questo è dovuto alla natura stessa del circuito, infatti entrambi i valori logici (0 e 1) sarebbero compatibili col funzionamento e non vi è modo come detto nell'introduzione, di discriminare in quale dei due stati si trovi il sistema.

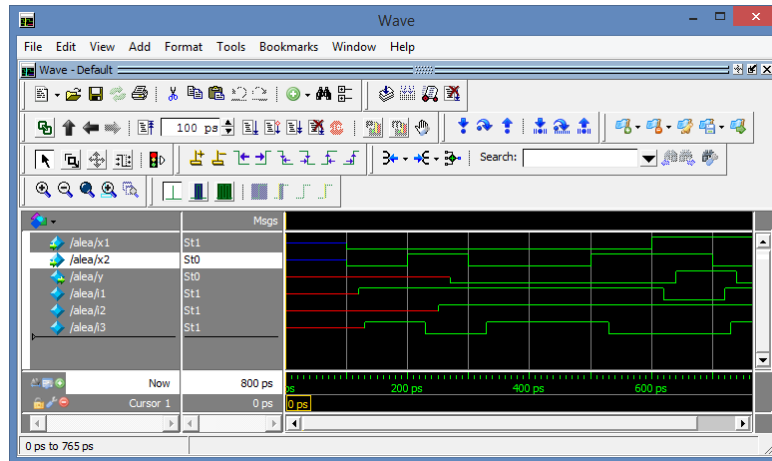


Ripetendo la procedura illustrata sopra si provi ad alzare e ad abbassare il segnale x2, facendo seguire ad ogni modifica un periodo di funzionamento di 100ps.



Si noti che con questa procedura il sistema ha portato l'uscita in uno stato ben definito (lo stato 0) e che in tale stato esso permane anche quando i segnali in ingresso sono ritornati entrambi allo stato 0.
 Si provi ora ad assegnare seguendo la procedura sopra suggerita la sequenza di stimoli (relativa ai segnali x1 ed x2) 00 – 01 – 11 – 10, facendo seguire a ogni stimolo un sufficiente periodo di simulazione (ad es. 100ps) tale da garantire che tutti i ritardi si siano estinti.

st\ x_1x_2	00	01	11	10
0	0	0	1	0
1	1	0	1	1



Si può notare che correttamente l'uscita si porta al valore alto in corrispondenza al valore 11 per gli ingressi, ma al passaggio a 10 invece di mantenersi al valore alto come previsto nella tavola di flusso, per la presenza di un'alea si riporta al valore 0.

st\ x_1x_2	00	01	11	10
0	0	0	1	0
1	1	0	1	1

Si può infatti notare che se il ritardo dell'invertitore viene ridotto dagli attuali 30ps a 10ps l'alea non si presenta ed il funzionamento sarà diverso. Per effettuare questa simulazione adatteremo un metodo diverso è più intuitivo.

3. Simulazione in batch attraverso definizione delle f.d.o.

- All'interno della finestra Library cliccare col tasto destro sull'istanza "alea" e scegliere Edit.
- Nella finestra di testo che compare modificare il ritardo dell'invertitore da #30 a #10

```

. . .
assign #20 i1=~(x1 & x2);
assign #20 i2=~(i3 & y);
assign #10 i3=~x2;

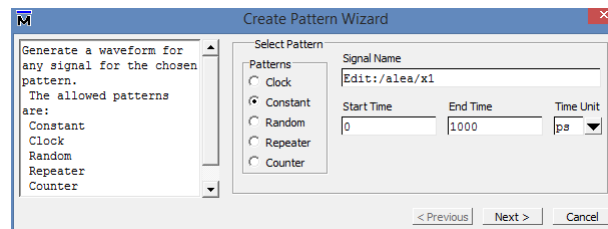
assign #20 y=~(i1 & i2);
. . .

```

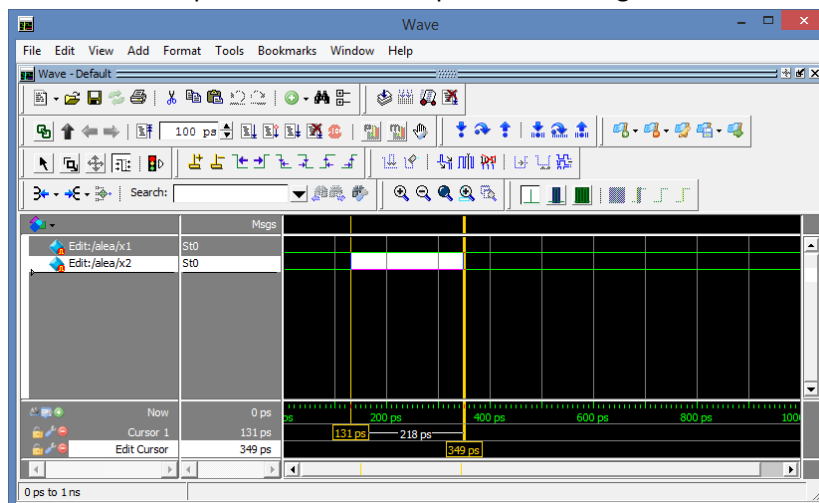
- Salvare
- Sempre attraverso il click destro scegliere "Recompile"
- e successivamente "Simulate"
- Sempre nella finestra Library (presente eventualmente sotto alla finestra "sim") con un click destro sull'istanza "alea" scegliere "Create Wave".

Nella finestra Wave appaiono ora 3 segnali (x_1 , x_2 e y)

- Cancellare il segnale y (right click > Edit > Delete)
- Editare i segnali x1 e x2 perché assumano tra 0ps e 1000ps il valore costante 0:
 - o Right click > Edit > Create/Modify Waveform
 - o Nella finestra che appare scegliere
 - Costant,
 - Start Time=0
 - End Time=1000
 - Time Unit ps
 - Next



- o Nella successiva finestra
 - fissare il valore a 0
 - Finish
- Ripetere la procedura anche per il segnale x2
- Modificare la modalità di funzionamento del mouse
 - o Wave > Mouse Mode > Edit Mouse
- Utilizzare il mouse per evidenziare una porzione del segnale x2

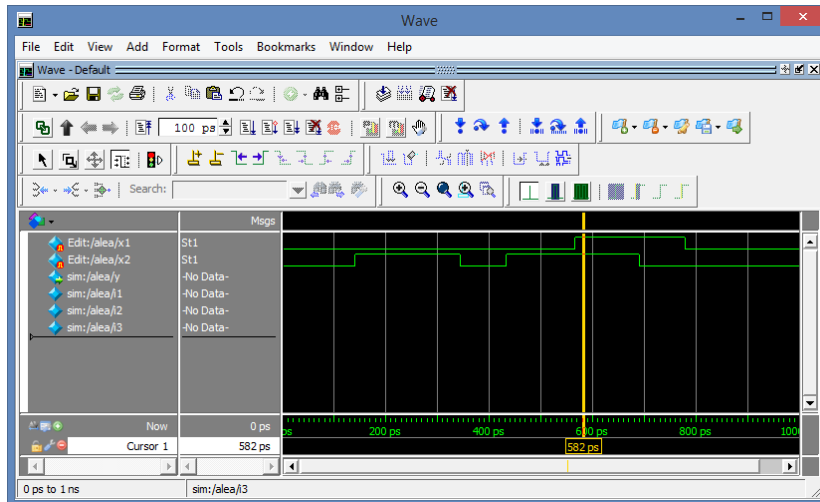


- Modificare il valore di questa porzione
 - Wave > Wave Editor > Invert

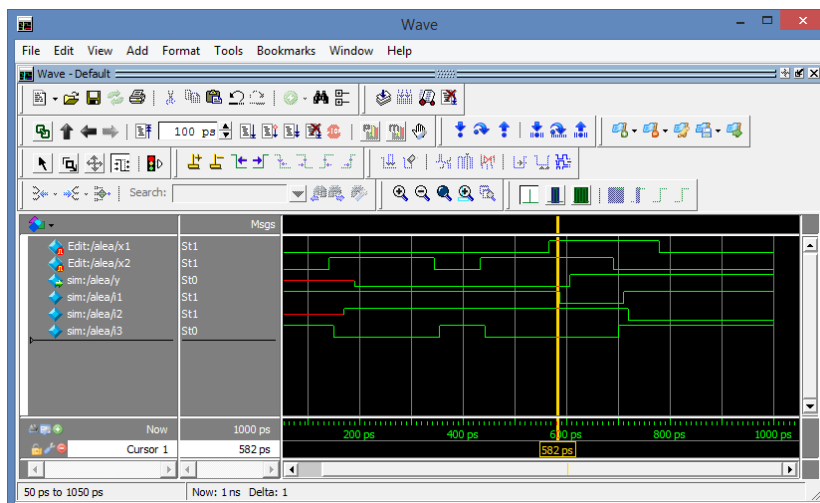
Eventualmente si può sfruttare la "Edit Toolbar" per avere a disposizione pulsanti atti editare più rapidamente i segnali

Window > Toolbars > Wave Edit

- Utilizzare questa tecnica per definire completamente le f.d.o con cui stimolare il circuito
- Trascinare dalla finestra "Sim" alla finestra "Wave" tutti i segnali che si intendono monitorare



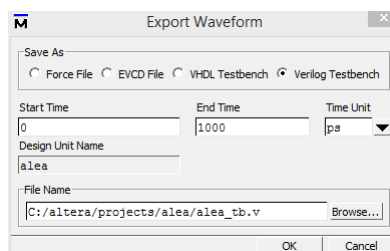
- Nella finestra console digitare
 - o Run -all



Si può notare che con un diverso ritardo dell'invertitore il fenomeno dell'alea non si verifica più ed il circuito funziona in modo corretto.

Se si vogliono simulare più soluzioni circuitali attraverso i medesimi stimoli è possibile salvare sia gli stimoli che l'unità da testare all'interno di un file Verilog HDL.

- Assicurarsi che la finestra "Wave" risulti evidenziata (eventualmente cliccandoci sopra)
- File > Export > Waveform
 - o Scegliere il formato Verilog Testbench
 - o La durata della simulazione
 - o Il nome del file ed il direttorio in cui salvarlo



- o OK

Si consiglia di aprire il file appena generato con un text editor e di analizzarne la sintassi che ne descrive il comportamento e di come gli stimoli vengano forniti alla UUT (Unit Under Test). Questo in esame è un file che impiega il linguaggio Verilog HDL per realizzare una descrizione “comportamentale” del sistema completo (UUT + generatore di stimoli) e che evidentemente non deve venir sintetizzato ne’ tantomeno sarebbe possibile sintetizzare.

4. Simulazione di un testbench completo descritto in Verilog HDL

Il file generato al passo precedente contiene sia la definizione degli stimoli, sia come questi vengono portati alla unità da testare (UUT). Pertanto esso, insieme alla descrizione Verilog HDL dell’unità da testare possono essere impiegati per rigenerare la simulazione o eventualmente possono essere modificati per generare diversi test.

Si apra Modelsim o se quest’ultimo è già aperto si termini la simulazione in corso:

- Simulate > End Simulation

Si importi nel corrente progetto il file in oggetto: facendo riferimento alla finestra Project (eventualmente aprendola se essa dovesse essere chiusa)

```
View > Project (Alt - vx)
```

```
Project > Add To Project > Existing File
```

Al momento il file (nella sua versione testuale) fa parte del progetto, ma per poterlo utilizzare esso deve venir compilato. Si evidenzi pertanto il file cliccandovi sopra

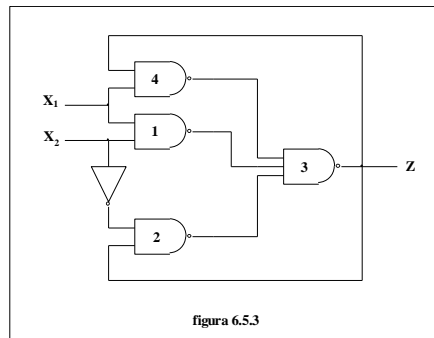
```
Compile > Compile Selected (Alt - ce)
```

Ora all’interno della libreria work vi sono due moduli: il file che realizza il testbench ed il file che descrive il circuito sotto test.

- Si clicchi col tasto destro sul modulo relativo al test bench e si scelga *Simulate*
- Nella Finestra “Wave” si trascinino i segnali che si vogliono analizzare (ad esempio si può trascinare l’intera istanza DUT contenuta all’interno del testbench per visualizzare tutti i segnali che ad essa fanno riferimento)
- Nella finestra di Console di digitare
 - o Run -all

Ora si può eventualmente modificare il sorgente Verilog HDL che descrive il circuito da analizzare modificandone ritardi e/o collegamenti e dopo una ricompilazione si può rilanciare la simulazione con i medesimi stimoli e verificare i risultati.

Ad esempio si potrebbe modificare il circuito introducendo un percorso ridondante per annullare gli effetti dell’alea:



Tale circuito potrebbe venir descritto in Verilog HDL col seguente sorgente:

```

`timescale 1ps / 1ps

module alea (x1,x2,y);
input x1,x2;
output y;

wire i1,i2,i3,i4;

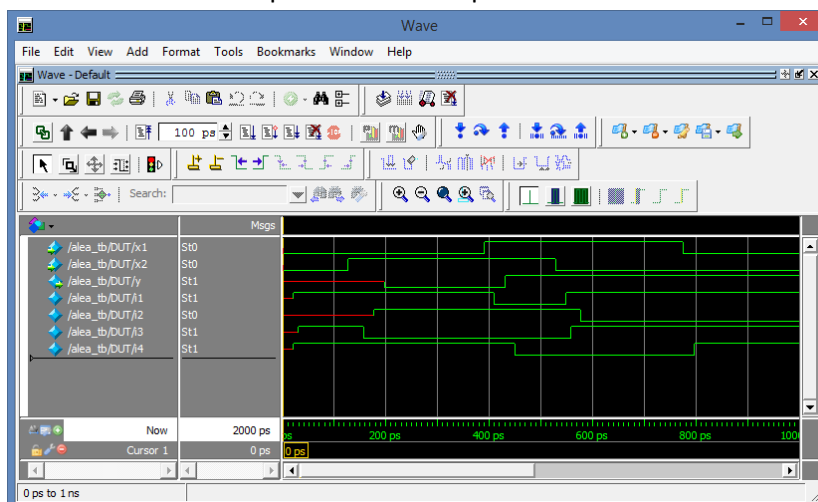
assign #20 i1=~(x1 & x2);
assign #20 i2=~(i3 & y);
assign #30 i3=~x2;
assign #20 i4=~(x1 & y);

assign #20 y=~(i1 & i2 & i4);
endmodule

```

Si può pertanto

- Fermare la simulazione attuale
- Reditare il modulo “alea.v” originale
- Ricompilare il modulo alea così modificato
- Rilanciare la simulazione
- Definire i segnali da visualizzare
- Far procedere la simulazione per un certo tempo.



I risultati dimostrano che nonostante il tempo di ritardo dell’invertitore possa essere critico, l’introduzione del nuovo percorso garantisce l’assenza di alee ed un corretto funzionamento della reazione.

5. Simulazione all'interno di Quartus

Introduzione:

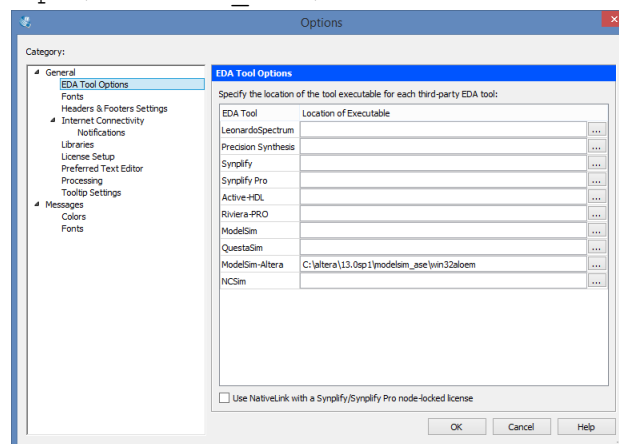
Modelsim può essere richiamato anche direttamente all'interno dell'ambiente di sviluppo Quartus, ma per garantire l'interazione tra i due ambienti bisogna realizzare dei settaggi a priori:

Seguendo il percorso

Tools > Option > General > EDA Tool Option

Definire il direttorio ove risiede l'eseguibile di modelsim. Di default si trova in:

C:\altera\13.0sp1\modelsim_ase\win32aloem



Inoltre seguendo il percorso

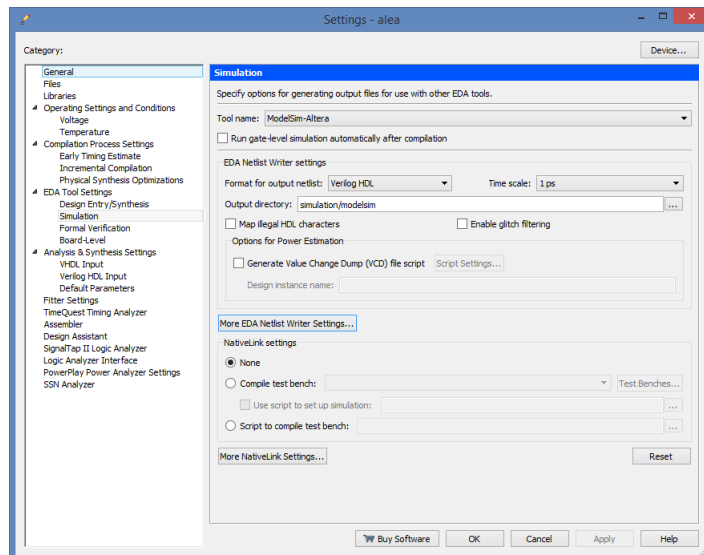
Assignment > Settings > EDA Tool Settings

Definire come

- tool di simulazione: ModelSim - Altera
- formato: Verilog HDL
- Run Gate Level simulation automatically – Disattivato

Inoltre seguendo il percorso

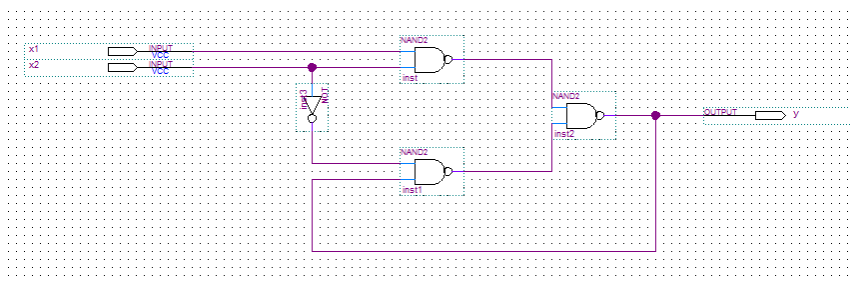
- Assignment > Settings > EDA Tool Settings > Simulation



- Nel **Tool name**, selezionare **ModelSim-Altera**.
- Assicurarsi che **Run gate-level simulation automatically after compilation** sia disattivato.
- Nella sezione **EDA Netlist Writer settings**, per il campo **Format for output netlist** selezionare **Verilog HDL**.
- Assicurarsi che i campi **Map illegal HDL characters**, **Enable glitch filtering**, e **Generate Value Change Dump (VCD) file script** siano disattivati
- Nella selezione **NativeLink settings**, selezionare **None**.

Simulazione a livello RTL

Supponiamo che all'interno di Quartus si voglia effettuare la simulazione di un progetto realizzato sotto forma di schematico quale ad esempio quello riportato qui di seguito.

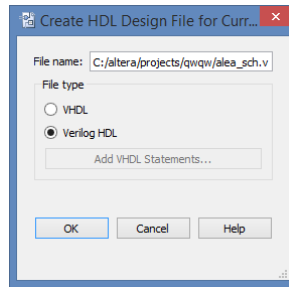


Poiché Modelsim è un simulatore per linguaggi di descrizione hardware (quali Verilog HDL oppure VHDL) il primo passo deve essere quello di convertire il nostro schematico in un equivalente file descritto in Verilog HDL.

Verificando di avere a vista la scheda con lo schematico

File > Create/Update > Create HDL design File From Current File

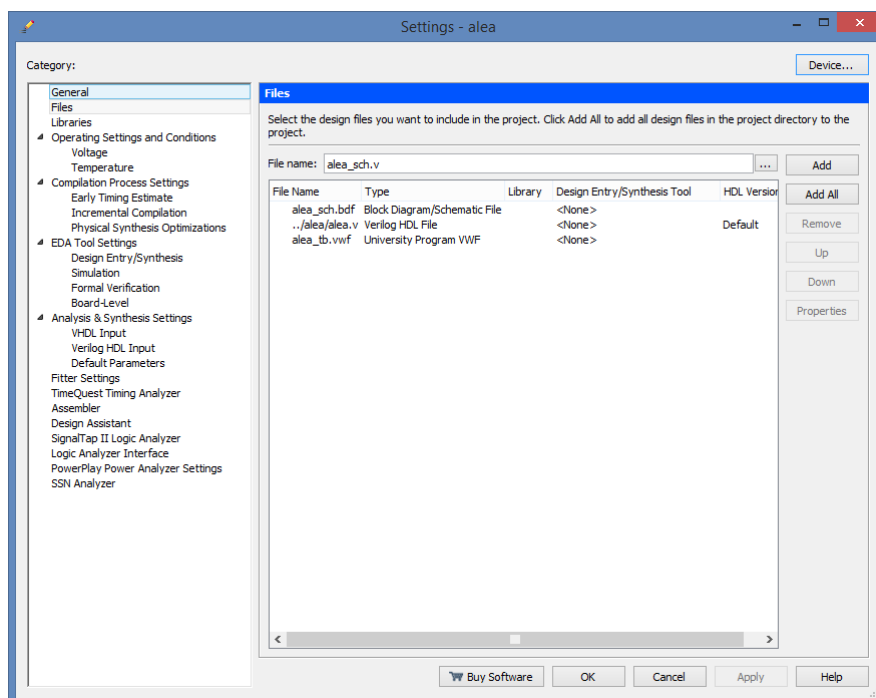
Scegliere linguaggio (Verilog HDL) ed il file di destinazione e cliccare OK



Viene generato il file corrispondente che può essere importato nel progetto al posto dello schematico.

Project > Add/Remove Files in project

Ed utilizzare l'interfaccia per rimuovere dal progetto lo schematico (Remove) ed includere il file Verilog HDL appena generato (Add).



Nella sezione Project Navigator, all'interno della scheda Files cliccare col tasto destro sul file appena importato e scegliere "Set as Top Level Entity" (Ctrl-Shift-J).

Processing > Start > Start Analysis and Synthesis (ctrl-k)

Quindi

Tools > Run Simulation Tool > RTL simulation

Si apre il simulatore Modelsim e all'interno di esso si può procedere come descritto al paragrafo 3:

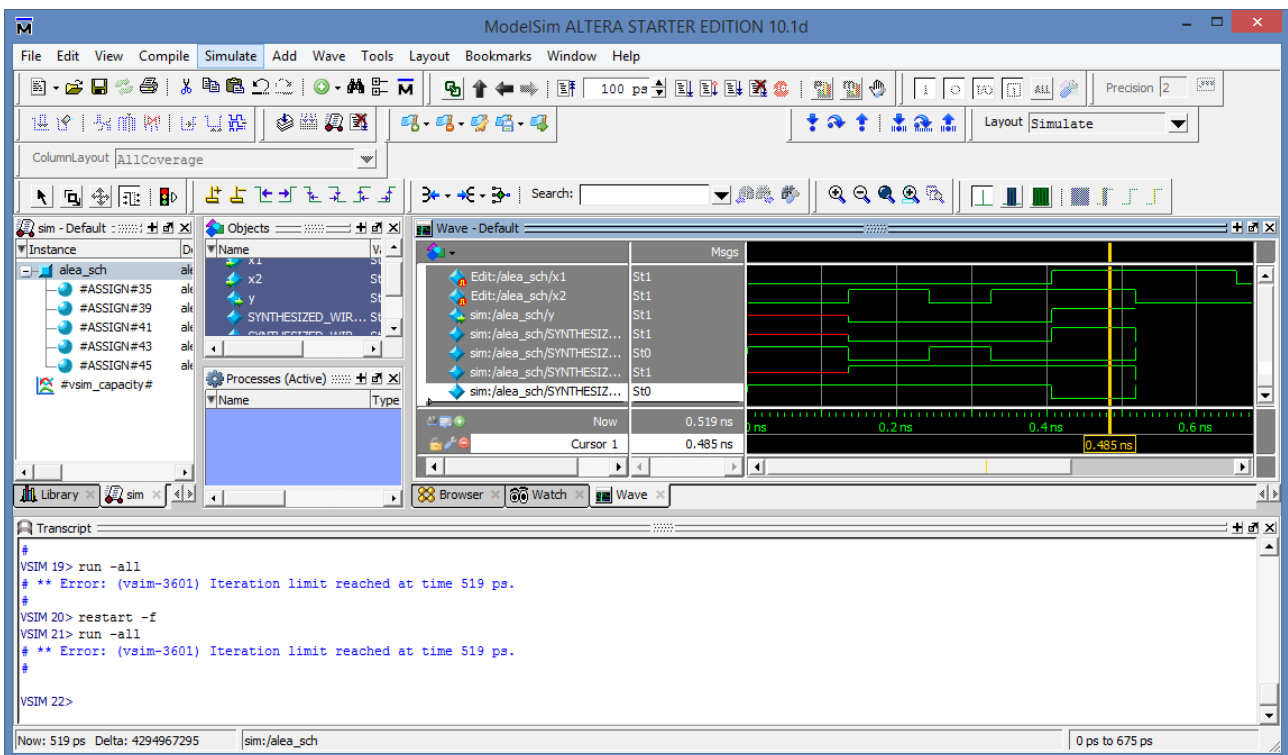
- Si faccia partire la simulazione dell'istanza da analizzare (click destro > Simulate)
- Si generino le forme d'onda di test (click destro > Create Wave)
- Si eliminino i segnali non utilizzati
- Attraverso il tool grafico si disegnano le f.d.o. opportune
- Si trascinino nella finestra wave i segnali da visualizzare

- Nella finestra di Console si lanci un “run”

Alcune note:

I segnali interni assumono nomi di difficile interpretazione in quanto il sistema di conversione da Schematico a Verilog ha assegnato loro nomi di default.

La simulazione si interrompe con un errore proprio in presenza della transizione da 11 a 10. Infatti come visto in precedenza il comportamento del circuito dipende essenzialmente dai ritardi del circuito, ma in questa prima fase di progetto nella quale i ritardi non sono ancora definiti (ovvero sono tutti idealizzati a 0), il sistema non ha modo di stabilire se vi sarà o meno un’alea.



Si propone perciò di salvare le f.d.o. ed effettuare una seconda simulazione a più basso livello di astrazione, ovvero a livello di gate.

File > Export > WaveForm

E si scelga il nome e la posizione dove salvare detto file (in formato Verilog HDL).

Si chiuda momentaneamente il tool di simulazione.

Sfruttando il file di stimoli appena salvato si possono eseguire simulazioni automatiche senza dover ridefinire di volta in volta l’andamento degli ingressi. Per fare questo bisogna, all’interno di Quartus modificare i settaggi:

Assignments > Settings

E all’interno della scheda

Eda Tool Settings > Simulation

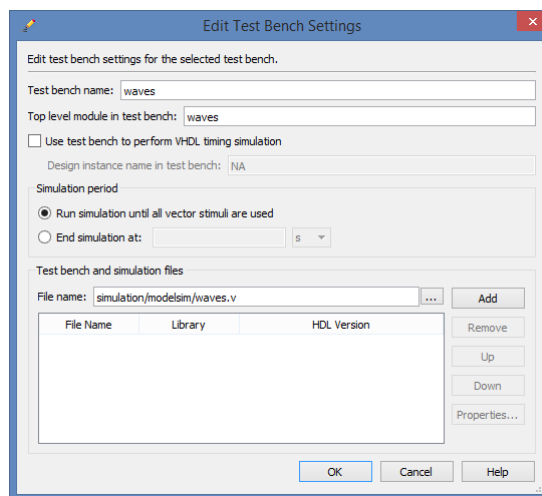
Attivare **Compile Test Bench** e cliccare sul tasto **Test Benches**

Nella Finestra Test Benches cliccare su New

Nella nuova finestra definire

- Un nome mnemonico per il test bench che si va a definire
- **IMPORTANTE:** Il nome della Top Entity così come appare all'interno del file di stimoli (solitamente coincide col nome del file di stimoli stesso)
- Si aggiunga all'elenco dei file da utilizzare il file di stimoli salvato al passo precedente

Nota: solitamente i tre nomi di default coincidono



- > OK
- > OK
- > OK

Ora il medesimo file di stimoli può essere adottato per effettuare rapidamente ed in modo automatico simulazioni ai diversi livelli di astrazione.

Simulazione a livello di GATE

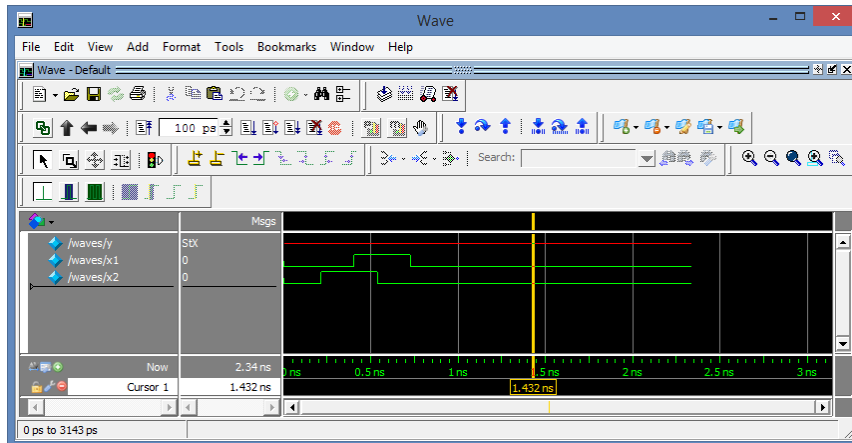
Si compili fino a livello di fitting il circuito:

Processing > Start Compilation (ctr-L)

Si rilanci la simulazione a livello di gate

Tools > Run Simulation Tool > Gate Level Simulation

Si scelga un modello comportamentale (Fast /Slow) e si completi la simulazione.



Dai risultati si nota che il comportamento non è esattamente quello che ci si poteva aspettare e l'uscita rimane sempre indefinita. Questo comportamento deriva dal fatto che ora la simulazione tiene conto di tutti i ritardi di propagazione all'interno dell'FPGA e probabilmente la durata degli impulsi sugli ingressi non è sufficiente a far commutare il circuito.

Si provi pertanto a ridefinire un diverso file di stimoli dove i segnali siano stabili per qualche decina di ns.

Per fare questo si può o ricominciare dalla simulazione RTL oppure si può ri-lanciare Modelsim ma senza abbinare ad esso il file di stimoli: Si chiuda ModelSim, si ritorni pertanto alla scheda dei settaggi e si disattivi l'utilizzo del Test Bench.

Assignments > Settings

Eda Tool Settings > Simulation e nella sezione "Native Link Settings" scegliere "None"

Si rilanci quindi la simulazione a livello di gate.

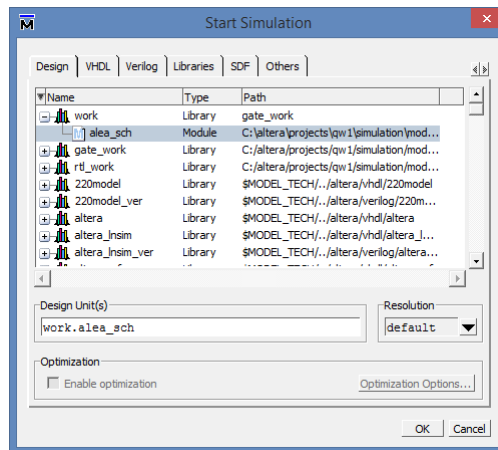
Tools > Run Simulation Tool > Gate Level Simulation

La procedura di realizzazione delle f.d.o ed il salvataggio delle medesime è già stata analizzata precedentemente, però nel caso in esame vi è una ulteriore procedura da seguire durante la fase di simulazione.

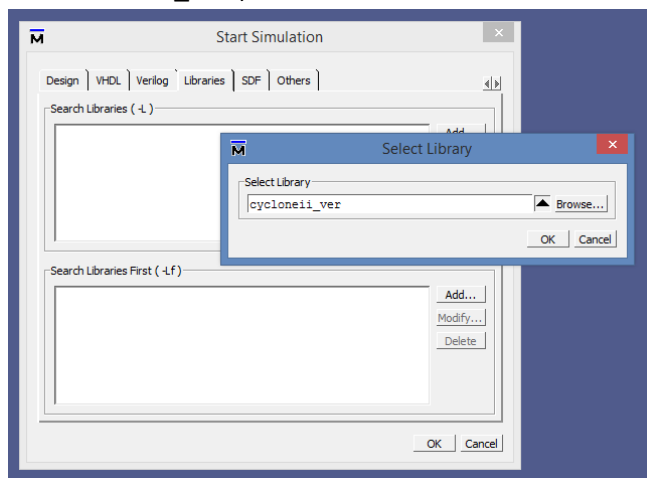
Infatti per poter effettuare la simulazione del circuito a livello di gate il simulatore deve aver avere informazione sui gate impiegati, pertanto si devono creare i collegamenti (links) con alcune librerie dove vengono descritti dettagliatamente i parametri ed i ritardi associati all'FPGA specifica che viene utilizzata. Per fare ciò la simulazione deve essere lanciata la simulazione dalla toolbar:

Simulate > Start Simulation

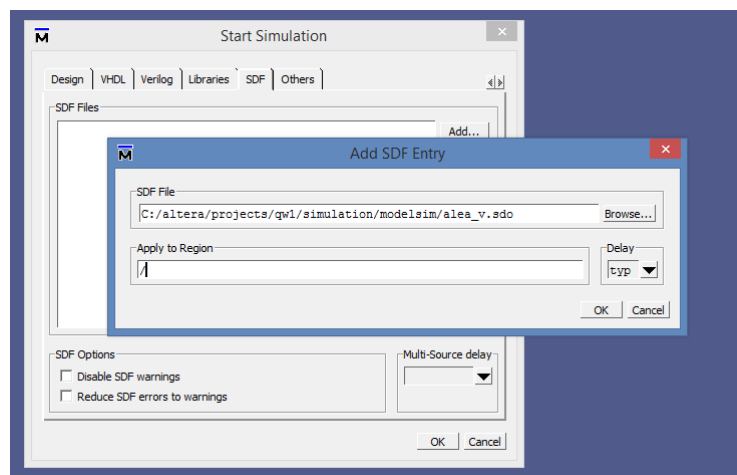
- Nella scheda relativa al "Design" si definisca l'unità da simulare (che tipicamente è all'interno della libreria work)



- Nella scheda relativa alle Libraries si aggiunga la libreria relativa al dispositivo impiegato (se si usa una Cyclone II si opti per la "cicloneii_ver")



- Nella scheda relativa alle tempistiche (SDF) si vada ad integrare col file .sdo che è stato generato in fase di compilazione

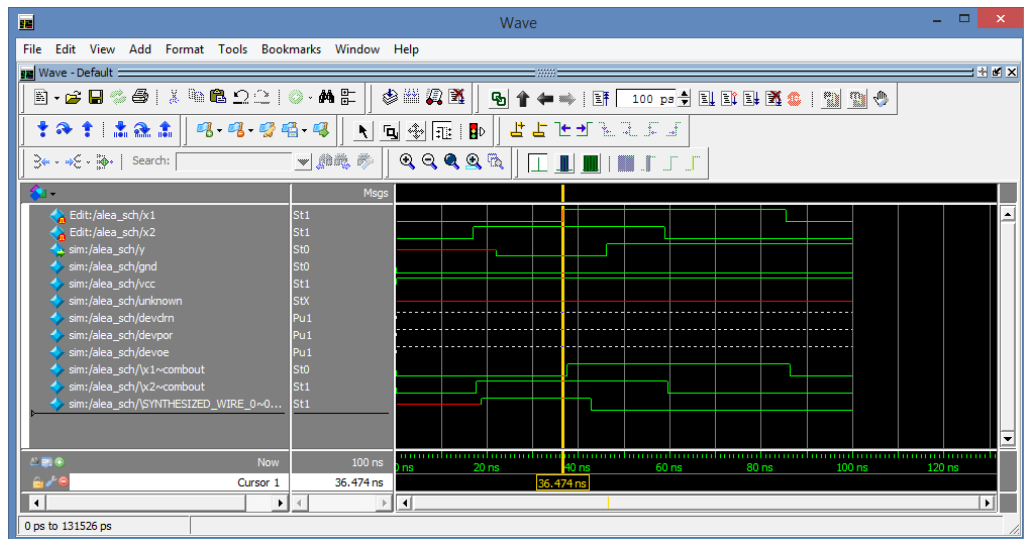


Ora si può completare la simulazione.

> OK

A questo punto seguendo la procedura già illustrata precedentemente si possono

- definire le f.d.o. degli stimoli prestando attenzione che questi presentino una durata dell'ordine dei "ns" anzichè dei "ps" come fatto finora (per cui si preveda una durata della simulazione di circa 100 ns)
- aggiungere nella finestra wave i segnali che si desidera da visualizzare (tenendo conto che questi saranno segnali interni all'FPGA nati dal processo di sintesi e che probabilmente risultano diversi dai segnali presenti nello schematico dal quale siamo partiti.
- Far procedere la simulazione per un tempo sufficiente.



La risposta del circuito adesso è coerente e si evidenziano i tempi di ritardo relativi alla propagazione dei segnali nell'FPGA. Il file di stimoli può adesso eventualmente essere salvato e riutilizzato attraverso la procedura sopra descritta per effettuare altre simulazioni.

NOTA: poiché le "timescale" dei files Verilog generati automaticamente sono espresse in ps è bene rifarsi sempre a questa unità per evitare confusione tra i vari files. Ad esempio, quando nella fase di salvataggio del file di stimoli, quando si deve fornire l'informazione di durata è bene che questa sia espressa come 100.000 ps piuttosto che come 100 ns. Altrimenti il file di stimoli generato si fermerà dopo 100 unità di tempo, che dal suo punto di vista equivalgono a 100 ps.

Bibliografia

[1] Modelsim Home page

<http://www.mentor.com/products/fv/modelsim/>

[2] Mentor-Graphics Home page

<http://www.mentor.com/>

[3] ENEE 359a: Digital VLSI Circuits by B. Jacob – “**Project 1: ModelSim Tutorial and Verilog Basics**”

<http://www.ece.umd.edu/class/enee359a.S2008/p1.pdf>

[4] Altera Corporation: “**Introduction to Simulation of Verilog Designs Using ModelSim Graphical Waveform Editor**”

ftp://ftp.altera.com/up/pub/Altera_Material/9.1/Tutorials/Verilog/ModelSim_GUI_Introduction.pdf

[5] Altera Corporation: “**Using ModelSim to Simulate Logic Circuits in Verilog Designs**”

ftp://ftp.altera.com/up/pub/Altera_Material/13.0/Tutorials/Verilog/Using_ModelSim.pdf

[6] Antonio D’Amore : “**Dispense del corso di Reti Logiche**”

<http://www2.units.it/marsi/reti/dispense/capitolo%2006.pdf>

[7] Altera Corporation: “**Introduction to Simulation of Verilog Designs Using ModelSim Graphical Waveform Editor**”

ftp://ftp.altera.com/up/pub/Altera_Material/13.0/Tutorials/Verilog/ModelSim_GUI_Introduction.pdf

[8] Mentor Graphics: “**ModelSim Reference Manual - Software Version 10.1a**”

http://moodle.units.it/moodle/file.php/670/Manuals/Modelsim/ModelSim_Reference_Manual_v10.1c.pdf

[9] Mentor Graphics: “**ModelSim User Manual - Software Version 10.1a**”

http://moodle.units.it/moodle/file.php/670/Manuals/Modelsim/ModelSim_Users_Manual_v10.1c.pdf

[10] Mentor Graphics “**ModelSim Tutorial -Software Version 10.1a**”

<http://www.fatih.edu.tr/~onur/download/EEE546/modelsimTutorial.PDF>

[11] Larbi Boughaleb “**ModelSim Tutorial**”

http://www.ece.northwestern.edu/~ismail/courses/c92/ModelSim_Tutorial.pdf

[12] “**ModelSim Tutorial**”

<http://ee.hawaii.edu/~sasaki/EE361/Fall06/Lab/Lab4.1/ModelSim.pdf>

