

# SIMULAREA FUNCȚIONALĂ A CIRCUITELOR CU SIMULATORUL MODELSIM

## 1. Scopul lucrării

În această lucrare se prezintă principiul simulatoarelor și al simulării funcționale a circuitelor, iar apoi se descriu programele disponibile în cadrul sistemului CAD Xilinx WebPACK pentru simularea funcțională a circuitelor. Este prezentat programul HDL Bench, care permite crearea unor bancuri de test necesare pentru simulare, și simulatorul ModelSim, care permite simularea circuitelor descrise prin scheme logice sau limbaje de descriere hardware.

## 2. Considerații teoretice

### 2.1. Simularea asistată de calculator

#### 2.1.1. Principiul simulării asistate de calculator

Simularea unui circuit permite verificarea funcționării circuitului pe baza descrierii acestuia prin oricare din metodele uzuale (limbaje de descriere hardware, scheme, diagrame de stare), înainte de implementarea efectivă a circuitului descris. Simularea asistată de calculator este metoda cea mai utilizată pentru simularea circuitelor, utilizarea acestei metode devenind posibilă și pentru sistemele digitale complexe odată cu creșterea semnificativă a performanței sistemelor de calcul moderne. Această simulare se realizează cu ajutorul unui program simulator, care construiește un model al circuitului pe baza descrierii acestuia de către proiectant, model care este echivalent cu o copie virtuală a circuitului proiectat. Simulatorul execută apoi modelul circuitului și analizează răspunsul acestuia la o serie de combinații ale intrărilor aplicate circuitului într-un anumit interval de timp, o asemenea combinație fiind numită *vector de test* sau *stimul*.

Pe lângă vectorii de test, proiectantul poate specifica și ieșirile așteptate ale circuitului pentru fiecare vector. În acest caz, simulatorul va compara ieșirile generate prin execuția modelului cu cele specificate de proiectant, care sunt cunoscute ca fiind corecte, afișând mesaje de eroare în cazul existenței unor diferențe între acestea. Dacă se detectează erori de funcționare, descrierea circuitului poate fi corectată mult mai simplu decât în cazul în care circuitul a fost implementat.

Pentru simularea sistematică a funcționării unui circuit, de multe ori se creează un *banc de test* ("*testbench*") constând din circuitul care trebuie testat și module adiționale pentru generarea și aplicarea vectorilor de test la intrările circuitului testat. Vectorii de test pot fi reprezentați fie de toate combinațiile posibile ale intrărilor circuitului (dacă este posibil), fie de combinațiile reprezentative pentru toate situațiile de funcționare. Prin utilizarea bancurilor de test se simulează funcționarea în condiții reale a circuitului, când acestuia i se aplică intrări din mediul exterior, de la un operator uman, sau de la un alt circuit sau sistem digital.

Simularea asistată de calculator are următoarele avantaje principale:

- Simularea permite experimentarea cu diferite variante de proiectare ale circuitului și condiții de funcționare ale acestuia, fără a fi necesară implementarea acestor variante. Proiectantul va putea implementa apoi varianta cea mai eficientă din punctul de vedere al performanțelor și al costului.

- Prin utilizarea simulării asistate de calculator este posibilă testarea sistematică a circuitelor, prin aplicarea unui număr mare de vectori de test la intrările circuitului, vectori care pot fi generați prin program sau pot fi specificați sub formă tabelară.
- Pentru circuite sau sisteme digitale complexe, simularea asigură reducerea costurilor și reducerea numărului erorilor de proiectare comparativ cu cazul în care se utilizează un prototip hardware pentru testare. Realizarea unor circuite complexe cum sunt microprocesoarele moderne nu ar fi fost posibilă fără utilizarea intensivă a simulării asistate de calculator.

### 2.1.2. Tipuri de simulare

Simularea poate fi executată în diferite etape ale procesului de proiectare și poate utiliza diferite nivele de abstractizare ale circuitului descris printr-o anumită metodă.

*Simularea funcțională* constă în simularea unei descrieri la nivel înalt a circuitului, această descriere specificând funcționarea circuitului, și nu structura acestuia. Pentru descrierea circuitului se pot utiliza diferite limbaje de modelare, cum sunt limbajele de programare. În acest caz, simularea constă în compilarea și execuția modelului respectiv. În cele mai multe cazuri, simularea funcțională se bazează pe un model al circuitului sub forma unei descrieri într-un limbaj de descriere hardware (HDL).

*Simularea logică* reprezintă analiza funcționării circuitului pe baza valorii unor variabile logice. Această simulare este numită și simulare *la nivelul transferurilor între registre* (RTL – *Register Transfer Level*), deoarece variabilele simulate sunt cele păstrate în registre. Simularea evaluează în fiecare ciclu de ceas funcțiile logice ale căror valori sunt înscrise în registre.

*Simularea la nivel de circuite* se referă la analiza funcționării unor modele ale circuitelor reprezentate prin interconectarea unor dispozitive electronice cum sunt tranzistoare, rezistențe și condensatoare. Această simulare constă în calcularea nivelelor de tensiune în funcție de timp din toate nodurile circuitului sau o parte a nodurilor. Aceasta presupune formularea și rezolvarea unui mare număr de ecuații diferențiale, necesitățile de memorie și de resurse de calcul fiind ridicate.

*Simularea temporală* constă în simularea funcționării circuitului după determinarea întârzierii reale a semnalelor pe diferitele căi de date. Această simulare se utilizează în special pentru circuitele programabile, la care întârzierile semnalelor nu vor fi cunoscute decât în urma implementării circuitului, atunci când se poate determina numărul de conexiuni programabile utilizate pentru rutarea diferitelor semnale. Informațiile despre întârzierile semnalelor sunt furnizate simulatorului prin operația numită *adnotare inversă*. Simularea temporală permite analiza funcționării în condiții reale a circuitului și determinarea frecvenței maxime de funcționare a acestuia.

### 2.1.3. Funcționarea unui simulator

Un program de simulare necesită două tipuri de intrări: un fișier cu descrierea circuitului care trebuie simulat și un set de stimuli care definesc toate semnalele de intrare pe durata timpului de simulare. Circuitul poate fi descris printr-o listă de conexiuni sau printr-un limbaj de descriere hardware. În cazul în care circuitul este specificat prin scheme sau diagrame de stare, trebuie creată mai întâi o descriere echivalentă a circuitului sub forma unei liste de conexiuni. Această descriere poate fi într-un format specific listelor de conexiuni (de exemplu, EDIF – *Electronic Design Interchange Format*), sau poate fi o descriere structurală a circuitului într-un limbaj de descriere hardware (de exemplu, VHDL sau Verilog).

Proiectantul trebuie să specifice setul de stimuli care va fi aplicat la intrările circuitului pe durata simulării acestuia. Opțional, proiectantul poate specifica și semnalele de ieșire care trebuie generate de circuit pentru fiecare stimul. Pentru definirea stimulilor, sistemele CAD sau simulatoarele pun la dispoziție diferite interfețe, cele mai utilizate fiind interfețele grafice, interfețele bazate pe fișiere text și cele la nivelul liniei de comandă. Interfețele grafice permit specificarea valorii semnalelor aplicate la intrările circuitului sub forma unor diagrame de timp. Aceste interfețe sunt utile atunci când trebuie definit un număr redus de intrări. Atunci când există un număr mai mare de semnale de intrare, sau trebuie definit un număr mare de tranziții ale acestor semnale, este mai utilă specificarea

într-un fișier text a unor comenzi care definesc valorile semnalelor de intrare și momentele de timp în care semnalele își modifică valoarea. Pot exista și comenzi care specifică execuția simulării pentru un anumit timp. Introducerea comenzilor de simulare prin linia de comandă este utilă atunci când trebuie modificată valoarea unor semnale care au fost definite anterior prin alte metode.

Simulatoarele utilizate în proiectarea asistată de calculator sunt simulatoare bazate pe evenimente. Aceste simulatoare împart intervalul de timp pentru care se realizează simularea în intervale cu durate foarte reduse, un asemenea interval fiind numit *pasul simulării*. Acest pas este exprimat printr-un multiplu întreg al unei unități de timp care se numește *limita de rezoluție*. Simulatorul nu poate măsura intervale de timp mai reduse decât această limită. Limita de rezoluție poate fi, de exemplu, de 1 picosecundă (ps), multiplul acesteia fiind specificat de proiectant. După trecerea unui interval de timp egal cu pasul simulării, simulatorul determină toate semnalele ale căror valori au fost modificate pe durata ultimului pas al simulării, o asemenea modificare fiind numită *eveniment*. Pentru fiecare semnal de intrare care s-a modificat, simulatorul reevaluează modelul circuitului și determină noile valori ale semnalelor care sunt afectate de această modificare. Această reevaluare determină modificarea altor semnale și generarea altor evenimente.

Un semnal nu poate fi actualizat în același timp cu un alt semnal din care este generat, deoarece semnalele nu își pot modifica valorile instantaneu. De aceea, modificarea valorii unui semnal este planificată de simulator pentru un moment de timp ulterior față de timpul curent de simulare ( $t_c$ ), după o întârziere numită *întârziere delta* ( $\delta$ ). Deci, modificarea valorii semnalului va fi executată la momentul de timp  $t_{c+\delta}$ .

La modificarea valorii unui semnal, se execută un *ciclu delta* în care se modifică valorile tuturor semnalelor care depind de primul semnal. Dacă semnalele modificate afectează alte semnale, vor fi planificate alte evenimente pentru timpul curent de simulare, la momentele de timp  $t_{c+2\delta}$ ,  $t_{c+3\delta}$ , ... La aceste momente vor fi executate alte cicluri delta pentru actualizarea valorii tuturor semnalelor. Toate aceste cicluri delta se execută pentru același moment de simulare ( $t_c$ ), astfel încât timpul de simulare nu avansează decât atunci când nu mai sunt alte evenimente planificate pentru timpul curent de simulare și toate semnalele au fost actualizate. Deci, întârzierea delta este de fapt o întârziere zero, doar în mod conceptual fiind considerată ca o întârziere infinezimală, fiind introdusă pentru a indica succesiunea operațiilor executate de simulator pentru actualizarea valorii semnalelor.

## 2.2. Generarea bancurilor de test cu programul HDL Bencher

Sistemul Xilinx WebPACK permite simularea funcțională a circuitelor proiectate cu ajutorul simulatorului ModelSim, care poate fi lansat în execuție din fereastra *Project Navigator*. Simularea se realizează pe baza listei de conexiuni generate în etapa de sinteză, listă care este generată sub forma unei descrieri structurale în limbajul VHDL sau Verilog. Sistemul permite și simularea circuitelor proiectate după etapele de translatare și mapare, sau simularea temporală după etapele de plasare și rutare. În această lucrare este descrisă doar simularea funcțională.

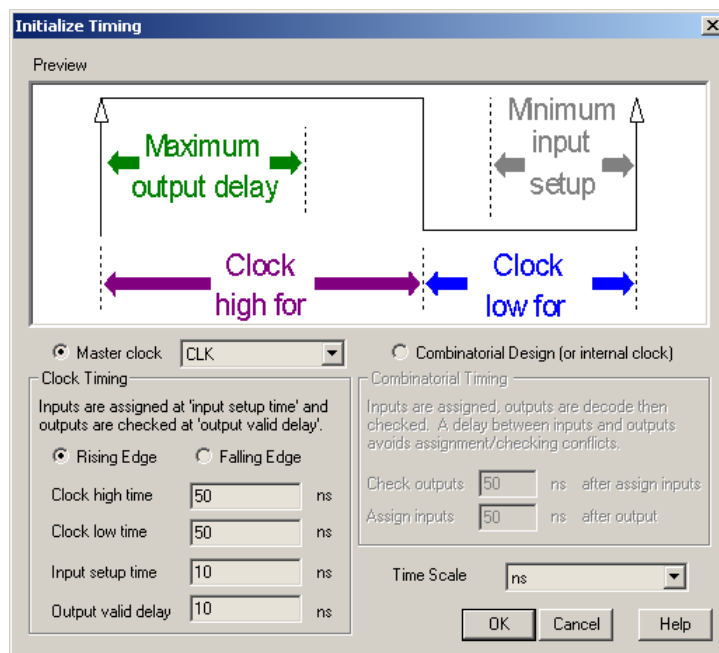
Simularea funcțională se poate realiza fie utilizând un banc de test specificat sub forma unei descrieri HDL, fie utilizând un banc de test generat în mod automat cu ajutorul programului HDL Bencher. Acest program, care este integrat în sistemul Xilinx WebPACK, permite introducerea sub formă grafică a stimulilor și a semnalelor de ieșire așteptate, generând în mod automat un banc de test care poate fi utilizat cu simulatorul ModelSim. Deși acest simulator permite specificarea stimulilor și prin comenzi introduse din linia de comandă, se recomandă ca înainte de lansarea simulatorului să se creeze un fișier de stimuli, din care programul HDL Bencher va genera un banc de test.

### 2.2.1. Crearea unui fișier de stimuli

Pentru crearea unui fișier de stimuli cu ajutorul programului HDL Bencher, se procedează în modul următor:

1. În fereastra *Sources in Project* din ecranul *Project Navigator* se selectează fișierul sursă conținând descrierea circuitului care trebuie simulat.

2. Din meniul *Project Navigator* se selectează *Project* → *New Source* pentru crearea unui nou fișier.
3. În fereastra de dialog *New* se selectează *Test Bench Waveform* ca tip al noului fișier.
4. Se introduce numele fișierului care va fi creat; acest nume trebuie să fie diferit de numele fișierului sursă care va fi simulat.
5. Se selectează *Next*. În fereastra de dialog *Select* se va selecta fișierul sursă cu care trebuie asociat fișierul de stimuli. În mod implicit, în această fereastră va apare selectat fișierul care a fost selectat în fereastra *Sources in Project* (pasul 1).
6. Se selectează *Next*, iar apoi se selectează *Finish*. Va fi lansat în execuție programul HDL Benchner și va apare fereastra *Initialize Timing*, care permite setarea unor parametri de timp care vor fi utilizați în timpul simulării (Figura 8.1). În mod obișnuit, se pot păstra setările implicite.



**Figura 8.1.** Fereastra *Initialize Timing* pentru setarea parametrilor de timp care vor fi utilizați la simulare.

Parametrii care pot fi setați depind de tipul circuitului care va fi simulat: secvențial sau combinațional. Acest tip este determinat de programul HDL Benchner, iar setările implicite vor fi selectate în mod automat; aceste setări pot fi modificate de utilizator. Unitatea de timp utilizată poate fi setată în caseta *Time Scale*. Pentru circuitele secvențiale se poate seta frontul activ al semnalului de ceas: frontul crescător (*Rising Edge*) sau frontul descrescător (*Falling Edge*). Acest front este indicat printr-o săgeată în fereastra *Preview*. De asemenea, se poate seta timpul cât semnalul de ceas va avea valoarea logică 1 (*Clock high time*) și timpul cât acest semnal va avea valoarea logică 0 (*Clock low time*). Aceste două intervale definesc perioada semnalului de ceas. Valoarea implicită a acestei perioade este de 100 ns, ceea ce corespunde unei frecvențe de 10 MHz.

Parametrul *Input setup time* definește momentul de timp înainte de frontul activ al semnalului de ceas când programul HDL Benchner setează valoarea semnalelor de intrare. Parametrul *Output valid delay* definește întârzierea de la frontul activ al semnalului de ceas după care programul HDL Benchner verifică valorile semnalelor de ieșire. Acești parametri reprezintă constrângeri de temporizare, după implementarea circuitului fiind posibilă verificarea respectării acestor constrângeri.

7. După setarea parametrilor de timp, se execută un clic pe butonul *OK*.

În urma acestor operații, se vor afișa două ferestre principale ale programului HDL Bencher (Figura 8.2). Fereastra de sus este cea a formelor de undă (*Waveform*), utilizată pentru vizualizarea sau editarea formelor de undă ale stimulilor aplicați la intrările circuitului și ale semnalelor de ieșire așteptate. În partea stângă a acestei ferestre se afișează toate semnalele cărora li s-au atașat porturi de intrare, de ieșire sau bidirecționale. Direcția portului este indicată grafic la dreapta numelui semnalului. În fereastra de jos se afișează conținutul fișierului HDL cu care a fost asociat fișierul de stimuli. De exemplu, în această fereastră se afișează lista de conexiuni care a fost generată dintr-o schemă, sub forma unei descrieri VHDL.

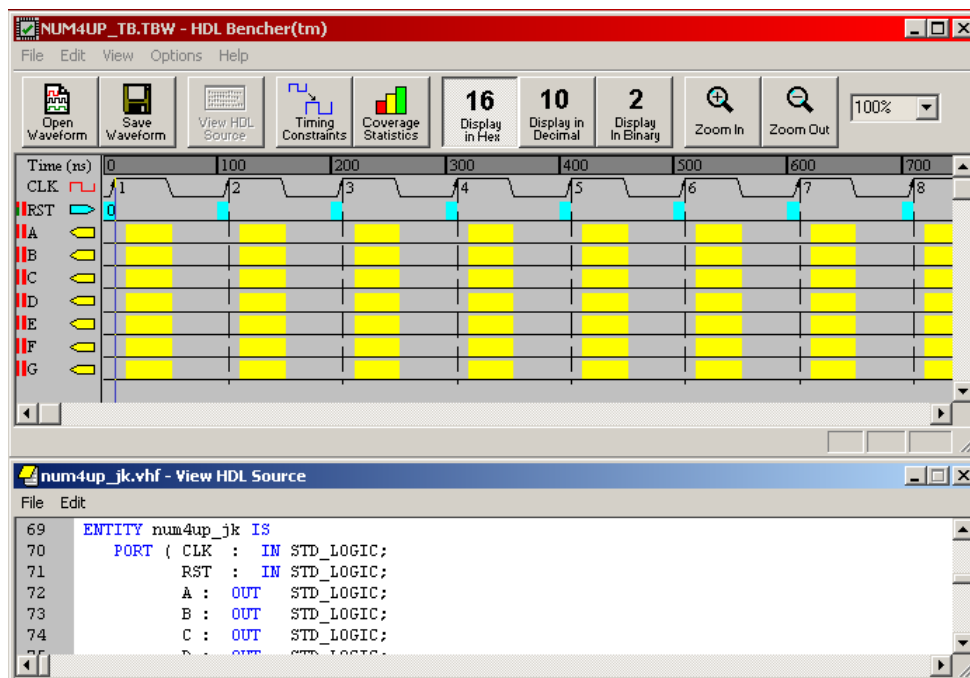


Figura 8.2. Ferestrele principale ale programului HDL Bencher.

Programul HDL Bencher setează valoarea semnalelor de intrare și verifică valorile semnalelor de ieșire la momentele de timp setate în fereastra *Initialize Timing*. Această fereastră este afișată la crearea unui nou fișier de stimuli, iar setările parametrilor de timp pot fi modificate ulterior prin comanda *Options* → *Timing Constraints* sau butonul *Timing Constraints*. Intervalul de timp în care poate fi setată valoarea unui semnal de intrare este indicat prin culoarea albastru deschis, iar intervalul de timp în care se verifică valoarea unui semnal de ieșire este indicat prin culoarea galbenă.

Dacă se poziționează cursorul deasupra numelui unui semnal, se vor afișa într-o casetă atributele semnalului respectiv. Dacă se poziționează cursorul deasupra formei de undă a unui semnal, se va afișa valoarea semnalului la momentul de timp respectiv. Prin poziționarea cursorului pe linia care indică timpul (implicit, prin diviziuni la fiecare 100 ns), se afișează valoarea timpului pentru poziția respectivă.

## 2.2.2. Inițializarea semnalelor prin comutarea valorii biților

Pentru definirea stimulilor (vectorilor de test), trebuie să se inițializeze semnalele de intrare. Metoda cea mai simplă pentru inițializarea semnalelor care sunt biți este comutarea valorii acestora în momentele de timp dorite. Pentru aceasta, se execută un clic pe forma de undă a semnalului în poziția corespunzătoare momentului de timp la care trebuie să se modifice valoarea aceluia semnal. Valoarea semnalului se va modifica din 0 în 1 sau din 1 în 0, iar această comutare se va produce la începutul intervalului de timp precedent în care este permisă modificarea valorii semnalului. Acest interval este indicat printr-o celulă de culoare albastră. Printr-o succesiune de comutări se pot defini în mod simplu formele de undă ale semnalelor de intrare.

În mod opțional, se pot inițializa și semnalele de ieșire cu valorile așteptate ale acestora. Pentru fiecare vector de intrare, se pot defini valorile corecte ale semnalelor de ieșire. Aceste valori se definesc în mod similar cu valorile semnalelor de intrare, comutând valorile existente. Comutarea se va produce la începutul intervalelor de timp indicate prin celule de culoare galbenă. În timpul simulării, simulatorul va compara valorile introduse de utilizator cu valorile generate la simulare. Simulatorul va afișa mesaje de eroare pentru fiecare moment de timp la care există o diferență între cele două valori. Dacă nu se asignează valori semnalelor de ieșire, simulatorul nu va afișa mesaje de eroare.

### 2.2.3. Inițializarea semnalelor prin metoda tabelară

Programul HDL Bencher permite introducerea rapidă a valorii semnalelor printr-o metodă tabelară. Pentru aceasta se execută un clic dublu pe forma de undă a semnalului la momentul de timp la care trebuie să se modifice valoarea aceluia semnal. Se va deschide o casetă de editare, în care se introduce valoarea semnalului la momentul respectiv (Figura 8.3). După introducerea valorii, se apasă tasta **Enter**, iar caseta se va deplasa la dreapta, la următorul moment de timp la care se poate modifica valoarea semnalului. Se continuă cu introducerea altor valori, după care se apasă tasta **Esc**. Se pot utiliza tastele cu săgeți pentru deplasarea la un moment de timp următor sau precedent, sau la un alt semnal.

Time (ns)	0	100	200
H0	0	Pattern 1	
H1	0		

Figura 8.3. Inițializarea semnalelor prin metoda tabelară.

În cazul semnalelor care sunt vectori, valorile acestora se pot introduce în hexazecimal, zecimal sau binar. Baza care va fi utilizată este cea utilizată și pentru afișare, aceasta fiind indicată printr-unul din butoanele “16 Display in Hex”, “10 Display in Decimal” sau “2 Display in Binary”. La introducerea valorilor, se verifică dacă acestea se încadrează în domeniul admis, în funcție de baza selectată. Se pot introduce valori cuprinse între ghilimele duble, acestea fiind interpretate întotdeauna ca valori binare, indiferent de baza selectată.

În caseta de editare deschisă la utilizarea metodei tabelare, se poate introduce o listă de valori separate prin virgule. De exemplu, se pot introduce toate valorile corespunzătoare unui semnal, pentru întregul interval de simulare. Dacă se introduc două virgule consecutive, se păstrează valoarea curentă a semnalului. După introducerea unei liste, se apasă tasta **Enter** pentru memorarea valorilor, actualizarea semnalului și trecerea la semnalul următor. La terminarea editării, se apasă tasta **Esc**.

### 2.2.4. Inițializarea semnalelor prin fereastra Pattern Wizard

Inițializarea semnalelor cu o formă complexă este simplificată prin utilizarea ferestrei *Pattern Wizard*. Pentru utilizarea acestei metode, se execută un clic dublu pe forma de undă a unui semnal, prin care se va deschide aceeași casetă de editare ca și la inițializarea prin metoda tabelară. Se execută un clic pe butonul *Pattern* (Figura 8.3), prin care se va deschide fereastra *Pattern Wizard* (Figura 8.4).

Opțiunile disponibile depind de tipul semnalului, aceste opțiuni fiind afișate în caseta *Choose Pattern*. Pentru semnalele care sunt biți, această fereastră permite generarea unei succesiuni de impulsuri de o anumită lățime, asignarea aleatoare a semnalelor sau comutarea valorii lor după un număr de cicluri specificat. Pentru semnalele care sunt vectori, este posibilă în plus incrementarea sau decrementarea valorii lor cu un anumit număr, sau deplasarea la stânga sau la dreapta a valorii lor după un număr de cicluri specificat. Forma semnalului care va fi generat este afișată în partea de jos a ferestrei.

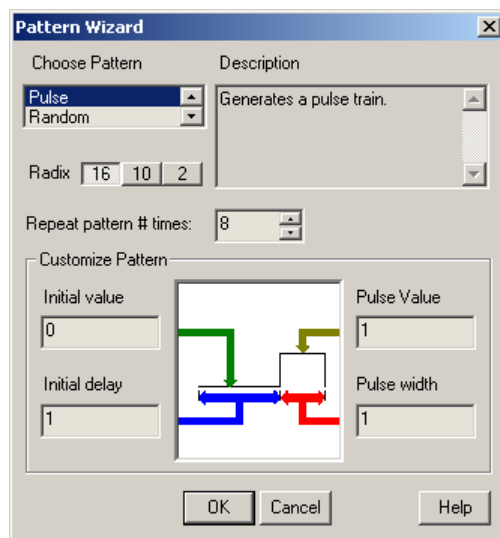


Figura 8.4. Fereastra *Pattern Wizard*.

### 2.2.5. Salvarea fișierului de stimuli

După editarea semnalelor de intrare și, opțional, a celor de ieșire, se salvează fișierul cu forma de undă a semnalelor prin comanda *File* → *Save Waveform* sau butonul *Save Waveform*. Poate apare o fereastră de dialog pentru specificarea numărului unităților de timp care trebuie să treacă de la ultima modificare a unui semnal de intrare până la terminarea simulării. Se poate selecta valoarea implicită (1). Fișierul salvat va avea extensia **.tbw** și va fi adăugat automat la proiectul curent deschis în fereastra *Project Navigator*. Acest fișier poate fi deschis ulterior prin execuția unui dublu clic pe numele acestuia în fereastra *Sources in Project*.

După salvarea fișierului de stimuli, programul HDL Bencher generează în mod automat un fișier HDL care conține bancul de test corespunzător fișierului salvat. Acest fișier poate fi vizualizat în ecranul *Project Navigator* prin selectarea fișierului cu extensia **.tbw** în fereastra *Sources in Project* și execuția unui clic dublu pe numele procesului *View Behavioral Testbench* din fereastra *Processes for Current Source*.

## 2.3. Simularea funcțională cu simulatorul ModelSim

Simulatorul *ModelSim* (*Model Technology*) permite simularea funcțională a unui sistem digital înainte de implementarea acestuia pentru a verifica dacă funcționarea este corectă. Simularea funcțională poate fi executată pentru sisteme digitale specificate prin scheme sau prin limbaje de descriere HDL. Sistemul ISE WebPACK permite lansarea în execuție a simulatorului din fereastra *Project Navigator*. Simulatorul poate fi lansat în execuție și separat, în afara sistemului ISE WebPACK.

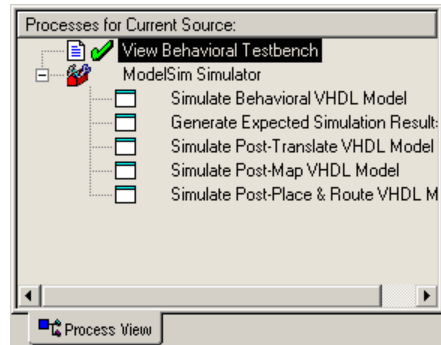
Pe lângă simularea funcțională, simulatorul *ModelSim* permite și alte tipuri de simulare, care pot fi executate în diferitele etape de implementare ale sistemului digital proiectat. Acestea nu sunt descrise în lucrarea prezentă.

### 2.3.1. Lansarea în execuție a simulatorului

După crearea unui fișier de stimuli cu programul HDL Bencher, poate fi lansat în execuție simulatorul *ModelSim* pentru simularea funcțională a proiectului. Sistemul ISE WebPACK realizează toate operațiile preliminare necesare înaintea execuției simulării: setarea proprietăților sesiunii de simulare, crearea directorului de lucru, compilarea fișierelor sursă, crearea unui fișier de comenzi pentru simulator (fișier cu extensia **.fdo**) și inițializarea simulării.

Pentru lansarea în execuție a simulatorului *ModelSim*, se execută următoarele operații:

1. În fereastra *Sources in Project*, se selectează fișierul de stimuli creat cu programul HDL bench (fișierul cu extensia **.tbw**).
2. În fereastra *Processes for Current Source*, se execută un clic pe semnul + din stânga liniei *ModelSim Simulator*. Prin aceasta, se vor afișa procesele disponibile pentru fișierul de stimuli (Figura 8.5).
3. Se execută un clic dublu pe linia *Simulate Behavioral VHDL Model*.

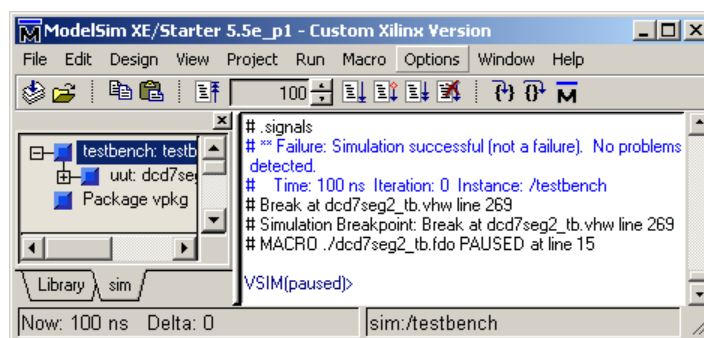


**Figura 8.5.** Vizualizarea operațiilor care pot fi executate cu simulatorul ModelSim.

Se va lansa în execuție simulatorul ModelSim, care va încărca bibliotecile necesare pentru simularea diferitelor componente, va compila modelele funcționale ale componentelor, va executa comenzile aflate în fișierul de comenzi creat de sistemul ISE WebPACK, după care va afișa mai multe ferestre și va opri procesul de simulare. În mod implicit, simulatorul va afișa fereastra principală a acestuia, fereastra cu structura proiectului simulat, fereastra cu semnalele utilizate pentru simulare și fereastra cu formele de undă ale semnalelor de intrare și a celor de ieșire obținute prin simulare. Aceste ferestre sunt prezentate în secțiunile următoare. Ferestrele afișate, ca și alte opțiuni de simulare, se pot modifica din fereastra *Processes for Current Source*, executând un clic cu butonul din dreapta pe linia *Simulate Behavioral VHDL Model* și selectând *Properties*.

### 2.3.2. Fereastra principală a simulatorului

Fereastra principală a simulatorului ModelSim, în cazul în care simulatorul a fost lansat în execuție din sistemul ISE WebPACK, are forma din Figura 8.6.



**Figura 8.6.** Fereastra principală a simulatorului ModelSim.

Partea din stânga ferestrei principale este rezervată pentru un spațiu de lucru. Acest spațiu permite accesul simplu la proiecte, unități de proiectare compilate și date utilizate la simulare. Spațiul de lucru poate fi ascuns sau afișat prin comanda *View* → *Hide/Show Workspace*.

În partea din dreapta ferestrei principale se afișează comenzile care au fost executate de simulator și mesajele generate în urma execuției. Simulatorul afișează un prompter care permite introducerea unor comenzi în linia de comandă. Conținutul acestei zone a ecranului este salvat în mod



automat într-un fișier cu numele implicit **transcript**. Este posibilă salvarea acestui conținut într-un fișier cu un alt nume prin comanda *File* → *Save Transcript As*.

### 2.3.3. Fereastra Structure

Fereastra *Structure* permite vizualizarea sub formă ierarhică a structurii proiectului. Informațiile afișate în această fereastră sunt afișate și în zona spațiului de lucru din fereastra principală. Se afișează toate unitățile de proiectare utilizate de simulator, cum sunt: bancul de test, unitatea testată (etichetată cu **uut** – “*unit under test*”, pachetele utilizate în cadrul proiectului și cele utilizate pentru simulare. Această fereastră este utilă în cazul circuitelor complexe, atunci când se caută o anumită componentă aflată la un nivel ierarhic inferior.

Prima linie din fereastra *Structure* indică unitatea de nivel superior din cadrul proiectului. În cazul bancurilor de test create cu programul HDL Bencher, numele acestei unități este **testbench**. În continuare se afișează unitatea testată și pachetele utilizate. Structura unității testate poate fi vizualizată prin expandarea ierarhiei acesteia, executând un clic pe semnul + din stânga numelui **uut** (Figura 8.7). Structura celorlalte componente poate fi expandată în mod similar.

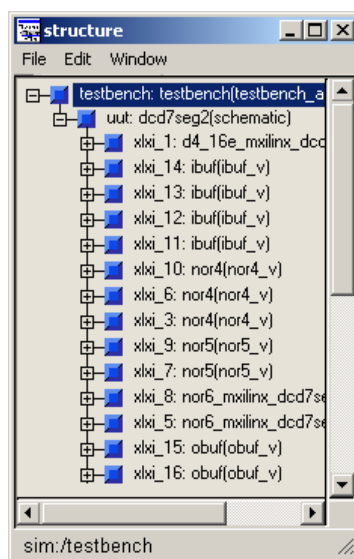


Figura 8.7. Vizualizarea structurii unității testate în fereastra *Structure*.

În fereastra *Structure* se poate selecta o regiune, care va deveni regiunea curentă. Conținutul ferestrei semnalelor și cel al ferestrei fișierului sursă este actualizat pentru a vizualiza informațiile pentru acea regiune.

### 2.3.4. Fereastra Signals

Această fereastră afișează numele și valoarea curentă a semnalelor din regiunea selectată în fereastra *Structure*. Fereastra *Signals* este împărțită în două zone. Zona din stânga indică numele semnalelor din regiunea curentă (selectată în fereastra *Structure*). Zona din dreapta conține valorile semnalelor la sfârșitul simulării (Figura 8.8).

Prin execuția unui clic dublu pe numele unui semnal se va selecta linia din fișierul sursă în care este definit semnalul respectiv. Dacă fereastra fișierului sursă nu a fost deschisă în prealabil (cu comanda *View* → *Source*), se va deschide în mod automat această fereastră. Se poate utiliza comanda *Edit* → *Sort* pentru sortarea semnalelor în ordine ascendentă (*Ascending*), descendentă (*Descending*), sau în ordinea declarării lor (*Declaration Order*).

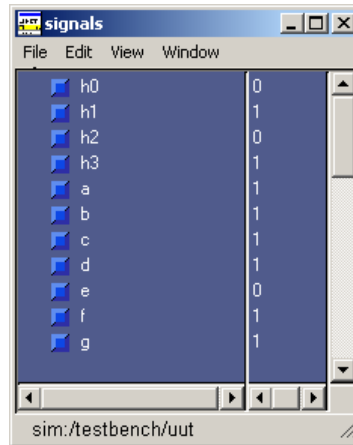


Figura 8.8. Vizualizarea semnalelor în fereastra *Signals*.

### Adăugarea semnalelor în fereastra *Wave*

Fereastra *Wave* vizualizează sub formă grafică semnalele în timpul simulării (această fereastră este descrisă în secțiunea 2.3.5). Sistemul ISE WebPACK specifică simulatorului *ModelSim* adăugarea în fereastra *Wave* a tuturor semnalelor care sunt porturi de I/E. Pentru vizualizarea altor semnale, este necesară adăugarea lor în fereastra *Wave* de către utilizator. Aceasta se poate realiza din fereastra *Signals*.

Pentru adăugarea unor semnale în fereastra *Wave*, se procedează astfel:

1. În fereastra *Structure*, se expandează ierarhia unității testate până când se ajunge la componenta în care sunt definite semnalele dorite.
2. Se selectează această componentă în fereastra *Structure*. Conținutul ferestrei *Signals* va fi actualizat pentru a afișa semnalele din regiunea selectată în fereastra *Structure*.
3. În fereastra *Signals*, se selectează semnalele care trebuie adăugate.
4. Se selectează comanda *View* → *Wave* → *Selected Signals*. Pentru adăugarea tuturor semnalelor din regiunea selectată în fereastra *Structure*, se poate selecta comanda *View* → *Wave* → *Signals in Region*.

După adăugarea unor semnale în fereastra *Wave*, nu se trasează în mod automat forma de undă a acestora. Pentru aceasta, este necesară fie executarea din nou a simulării, fie continuarea simulării. Modul de execuție al acestor operații este descris în secțiunea 2.3.5.

### Forțarea valorii semnalelor

Comanda *Edit* → *Force* afișează o fereastră de dialog care permite aplicarea unor stimuli semnalului selectat. Se pot selecta mai multe semnale pentru a forța valoarea acestora. Fereastra *Force Selected Signal* este ilustrată în Figura 8.9.

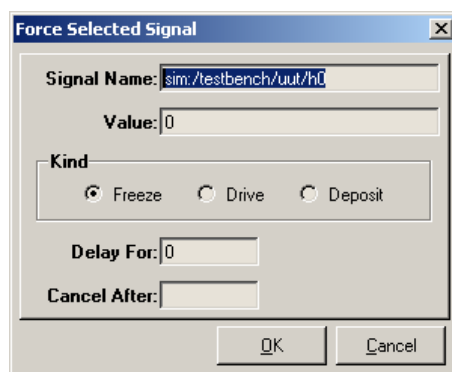


Figura 8.9. Fereastra de dialog pentru forțarea valorii semnalelor.

Opțiunile pentru forțarea valorii semnalelor sunt următoarele:

- **Value**  
Se afișează valoarea curentă a semnalului, care poate fi modificată prin introducerea unei noi valori în acest câmp. În cazul unei magistrale, se poate specifica o valoare într-o bază diferită de cea zecimală, sub forma *bază#valoare*. De exemplu, *16#0F* specifică valoarea hexazecimală *0Fh*.
- **Kind: Freeze**  
Semnalul este setat la valoarea specificată până când acesta este forțat din nou sau până când semnalul este eliberat prin comanda *Edit* → *NoForce*.
- **Kind: Drive**  
Se atașează un driver semnalului, care va forța semnalul la valoarea specificată până când acesta este forțat din nou sau până când semnalul este eliberat prin comanda *Edit* → *NoForce*.
- **Kind: Deposit**  
Semnalul este setat la valoarea specificată și va rămâne la această valoare până la următoarea tranzacție a driverului, până când semnalul este forțat din nou, sau până când acesta este eliberat prin comanda *Edit* → *NoForce*.
- **Delay For**  
Permite specificarea numărului unităților de timp pentru care trebuie aplicat stimulul. Acest timp este măsurat de la timpul de simulare curent.
- **Cancel After**  
Forțarea semnalului va fi terminată după timpul de simulare specificat prin această opțiune.

La selectarea butonului *OK*, se va lansa în execuție o comandă **force** cu parametri setați, comanda fiind afișată în fereastra principală. Dacă au fost selectate mai multe semnale, va apare următorul semnal în fereastra de dialog. Pentru fiecare semnal pot fi setați parametri diferiți.

### Setarea punctelor de întrerupere ale simulării

Se pot specifica puncte de întrerupere a execuției simulării (“*breakpoints*”), condițiile în care se va produce suspendarea simulării și acțiunile pe care trebuie să le execute simulatorul atunci când condițiile specificate sunt îndeplinite. De exemplu, se poate specifica un punct de întrerupere atunci când un semnal are o anumită valoare. La întâlnirea unui punct de întrerupere, se va afișa un mesaj în fereastra principală, indicând semnalul care a determinat întreruperea.

Pentru accesul la comenzile care permit setarea punctelor de întrerupere pentru un semnal, se selectează semnalul și se execută un clic cu butonul din dreapta. Pentru setarea unui punct de întrerupere, se selectează opțiunea *Add Breakpoint* din meniu. Pentru eliminarea unui punct de întrerupere setat anterior, se selectează opțiunea *Remove Signal Breakpoint*. Pentru eliminarea tuturor punctelor

de întrerupere din regiunea curentă, se selectează opțiunea *Remove All Signal Breakpoints*. Pentru afișarea listei punctelor de întrerupere care sunt setate, se selectează opțiunea *Show Breakpoints*.

Comanda *Edit Breakpoint* deschide fereastra de dialog *Edit When* (Figura 8.10). Prin setarea opțiunilor din această fereastră, se va genera o comandă **when**, care va fi afișată în fereastra principală.

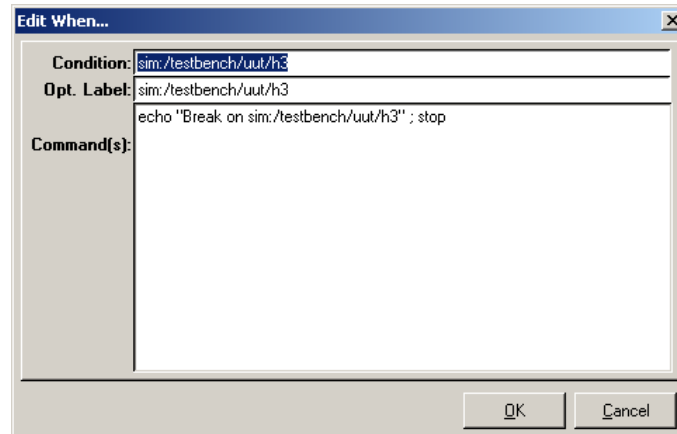


Figura 8.10. Fereastra de dialog pentru editarea punctelor de întrerupere.

Opțiunile din fereastra de dialog *Edit When* sunt următoarele:

- Condition**  
 Permite specificarea condiției sau a condițiilor în care se va produce suspendarea simulării și executarea comenzilor specificate prin opțiunea **Command(s)**. O condiție se poate specifica sub forma unei expresii conținând următorii operatori: egal ( $\equiv$  sau  $=$ ), diferit ( $\neq$  sau  $\neq$ ), ȘI logic (**&&** sau **AND**), SAU logic (**||** sau **OR**). Operanzii pot fi nume de componente, nume de semnale sau constante. De exemplu, egalitatea unui semnal scalar *a* cu o valoare se poate specifica sub forma **a=1**, iar egalitatea unui semnal vectorial *num* cu o valoare se poate specifica sub forma **num="1010"**. Operatorul de egalitate nu se poate utiliza pentru compararea valorii a două semnale, ci numai pentru compararea valorii unui semnal cu o constantă. Ca și condiție se poate specifica și modificarea valorii unui semnal, sub forma **a'EVENT**.
- Opt. Label**  
 Permite specificarea unei etichete opționale pentru comanda **when**.
- Command(s)**  
 Permite specificarea unei comenzi sau a mai multor comenzi care vor fi executate la îndeplinirea condiției sau condițiilor setate. Implicit, se va executa o comandă **echo**, care afișează un mesaj în fereastra principală, și o comandă **stop**, care oprește execuția simulării. Execuția poate fi reluată ulterior.

### 2.3.5. Fereastra Wave

În fereastra *Wave* se vizualizează rezultatele simulării atât sub formă grafică, cât și sub formă numerică. Această fereastră este împărțită în mai multe zone (Figura 8.11). Zona din stânga conține numele semnalelor și ierarhia componentei în care sunt generate. Este posibilă afișarea doar a numelui semnalelor, fără ierarhia componentelor. În zona din mijloc se afișează valorile semnalelor din fereastră. Valorile afișate se modifică în mod dinamic în funcție de deplasarea cursorului în zona care conține forma semnalelor. Valorile se pot afișa în diferite baze de numerație. În zona din dreapta se afișează formele de undă ale semnalelor. Formatul de afișare poate fi setat individual pentru fiecare semnal. În această zonă se pot afișa un număr de până la 20 de cursoare.

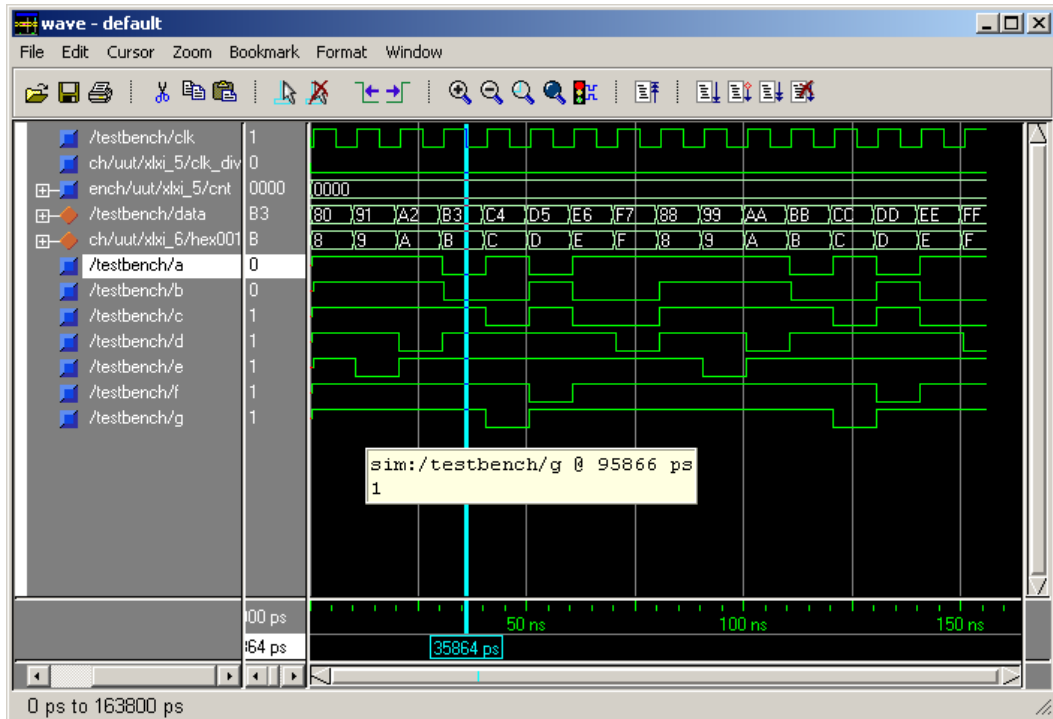


Figura 8.11. Fereastra *Wave* a simulatorului ModelSim.

Sub aceste trei zone se mai află alte trei zone, dintre care cea din stânga nu este utilizată. Zona din mijloc indică timpul curent de simulare și valoarea timpului corespunzătoare pentru fiecare cursor setat. Valoarea corespunzătoare cursorului activ se afișează pe un fond alb. Se poate selecta un anumit cursor prin selectarea valorii corespunzătoare acesteia în zona din mijloc. Zona din dreapta afișează timpul absolut de simulare pentru fiecare cursor și, în cazul existenței mai multor cursoare, timpul relativ între cursoare.

### Gruparea semnalelor în fereastra Wave

Semnalele din fereastra *Wave* pot fi grupate în magistrale. O magistrală este o colecție de semnale concatenate într-o anumită ordine pentru a crea un nou *semnal virtual*. Un asemenea semnal este util pentru afișarea mai compactă a acestuia atât sub formă grafică, cât și sub formă numerică, sau pentru setarea mai simplă a valorii acestuia comparativ cu setarea valorii semnalelor individuale. Un semnal virtual este indicat printr-o icoană sub forma unui romb de culoare portocalie.

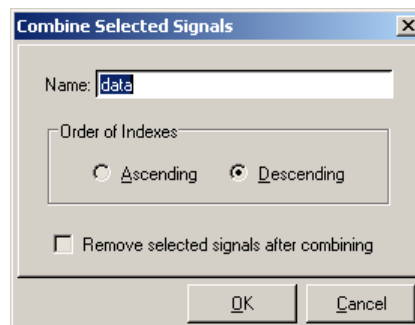


Figura 8.12. Fereastra de dialog pentru combinarea unor semnale într-o magistrală.

Pentru gruparea unor semnale, se selectează numele semnalelor în zona din stânga a ferestrei *Wave*, după care se selectează comanda *Edit* → *Combine*. Va apare fereastra de dialog *Combine Selected Signals* (Figura 8.12). În câmpul *Name* se introduce un nume pentru magistrala formată din semnalele grupate. Se selectează apoi ordinea în care vor fi indexate semnalele în cadrul magistralei. Dacă se selectează ordinea ascendentă (opțiunea *Ascending*), primul semnal selectat în fereastra *Wave*

va avea indexul 0. Dacă se selectează ordinea descendentă (opțiunea *Descending*), primul semnal selectat va avea indexul maxim. Dacă se dorește eliminarea semnalelor selectate din fereastra *Wave* după combinarea lor, se activează opțiunea *Remove selected signals after combining*.

### Formatarea afișării semnalelor în fereastra *Wave*

Există mai multe opțiuni pentru selectarea modului de afișare a semnalelor în fereastra *Wave*. Acestea se pot aplica fie semnalelor individuale, fie mai multor semnale simultan. Principalele opțiuni sunt descrise în continuare.

Pentru selectarea modului în care se afișează valoarea și forma de undă a unui semnal, se selectează numele semnalului, după care se selectează comanda *Edit* → *Signal Properties*. Această comandă este disponibilă și dacă se execută un clic cu butonul din dreapta pe numele semnalului selectat, iar apoi se selectează opțiunea *Signal Properties*. Se va deschide fereastra *Wave Signal Properties*, conținând două grupe de opțiuni, *View* și *Format*. Grupa de opțiuni *View* permite setarea bazei de numerație în care se afișează valoarea semnalului (*Radix*), a culorii cu care se afișează forma de undă a semnalului sau numele acestuia (*Wave Color*, respectiv *Name Color*). Setarea bazei de numerație se poate realiza și dacă se selectează numele semnalului, se execută un clic cu butonul din dreapta, se selectează opțiunea *Radix*, iar apoi baza dorită.

Grupa de opțiuni *Format* permite setarea formatului de afișare al semnalului. Formatul *Literal* afișează semnalul sub forma unor casete care conțin valorile semnalului respectiv (dacă există spațiu suficient). Formatul *Logic* afișează valori ale semnalului cum sunt 0 sau 1, ca și unele valori speciale, ca: U (valoare neinițializată), X (valoare nedefinită), Z (valoare de înaltă impedanță), - (valoare indiferentă). În cazul selectării formatului *Event*, se marchează fiecare tranziție a semnalului în timpul simulării. Setarea formatului de afișare se poate realiza și dacă se selectează numele semnalului, se execută un clic cu butonul din dreapta, se selectează opțiunea *Format*, iar apoi formatul dorit.

Pentru selectarea modului în care se afișează numele unui semnal în partea din stânga ferestrei *Wave*, se selectează numele semnalului, după care se selectează comanda *Edit* → *Display Properties*. Această comandă este disponibilă și dacă se execută un clic cu butonul din dreapta pe numele semnalului selectat, iar apoi se selectează opțiunea *Display Properties*. Se va deschide fereastra *Wave Window Properties*, în care modul de afișare poate fi setat în caseta *Display Signal Path*. Valoarea 0 din această casetă va determina afișarea ierarhiei complete a componentei care generează semnalul, de exemplu, */testbench/uut/clk\_div*. O valoare diferită de 0 indică numărul elementelor ierarhice care vor fi afișate. Valoarea 1 va determina afișarea doar a numelui semnalului.

### Salvarea rezultatelor simulării







Forma de undă a semnalelor, obținută în urma simulării, este salvată în mod automat de simulator într-un fișier cu formatul *WLF* (*Wave Log Format*), numele implicit al acestui fișier fiind **vsim.wlf**. Acest fișier este plasat în directorul de lucru al simulatorului, care este același cu directorul proiectului creat cu sistemul ISE WebPACK. În cazul în care acest fișier trebuie păstrat pentru vizualizarea ulterioară, trebuie copiat într-un alt director sau trebuie redenumit, deoarece la o nouă simulare conținutul fișierului va fi rescris. Fișierul cu forma de undă a semnalelor poate fi deschis ulterior pentru vizualizare și compararea cu rezultatele unei simulări ulterioare. Pentru aceasta, în fereastra *Wave* a simulatorului se execută comanda *File* → *Open Dataset* și se selectează fișierul salvat anterior.

Simulatorul permite și salvarea listei semnalelor din fereastra *Wave*, ceea ce este util atunci când au fost adăugate semnale sau stimuli suplimentari în această fereastră, iar simularea trebuie executată din nou. Lista completă a semnalelor va putea fi încărcată în mod simplu înaintea unei sesiuni de simulare, fără a fi necesară adăugarea din nou a semnalelor. Pentru salvarea listei semnalelor, în fereastra *Wave* a simulatorului se execută comanda *File* → *Save Format*. Lista este salvată într-un fișier de comenzi (numit și de macro-uri), cu extensia **.do**. Numele implicit al fișierului este **wave.do**, dar acest nume poate fi schimbat înainte de salvarea fișierului.

## Execuția simulării

Atunci când simulatorul este lansat în execuție din fereastra *Project Navigator* a sistemului ISE WebPACK, se execută în mod automat simularea pentru întreaga perioadă de timp în care există vectori de test definiți. Dacă se adaugă însă noi semnale în fereastra *Wave*, forma de undă a acestora nu va fi afișată în această fereastră, astfel încât va fi necesară executarea din nou a simulării. De asemenea, dacă se adaugă noi stimuli, poate fi necesară continuarea simulării. Pentru executarea acestor operații și a altor operații similare se pot utiliza butoanele sau comenzile descrise în Tabelul 8.1. Majoritatea acestor butoane sunt disponibile atât în fereastra *Wave*, cât și în fereastra principală a simulatorului. Opțiunile de meniu echivalente butoanelor sunt disponibile în fereastra principală.

Tabelul 8.1. Butoane și opțiuni de meniu pentru execuția simulării.

Buton	Denumire	Meniu echivalent	Funcție
	<b>Restart</b>	Run → Restart	Reîncarcă elementele proiectului și resetează timpul de simulare la zero, cu opțiunea de a păstra formatarea curentă, punctele de întrerupere și fișierul WLF.
	<b>Run Length</b>	--	Specifică timpul pentru care se execută simularea curentă.
	<b>Run</b>	Run → Run <time implicit>	Execută simularea curentă pentru timpul specificat în caseta <b>Run Length</b> .
	<b>Continue Run</b>	Run → Continue	Continuă execuția simulării curente.
	<b>Run All</b>	Run → Run -all	Execută simularea curentă la infinit, sau până când se întâlnește un punct de întrerupere, sau până când se apasă butonul <b>Break</b> .
	<b>Break</b>	--	Oprește execuția simulării curente.

## 2.4. Exemplu de simulare

Se prezintă în continuare etapele care trebuie executate pentru simularea schemei unui număr sincron de 4 biți realizat cu bistabile JK, sensul de numărare fiind direct (de la 0000 la 1111). Schema acestui numărător este prezentată în lucrarea de laborator Nr. 4, figura 4.33.

### 2.4.1. Crearea proiectului

Pentru crearea unui nou proiect și adăugarea fișierului cu schema numărătorului, se execută următoarele operații:

1. Se lansează în execuție sistemul ISE WebPACK. În ecranul *Project Navigator* se selectează *File* → *New Project*. Va apare fereastra de dialog *New Project*. În câmpul *Project Location* din această fereastră se selectează directorul în care se va crea proiectul (un subdirector din directorul `C:\Student\`). În câmpul *Project Name* se introduce numele proiectului. Se verifică setările corecte în câmpurile *Device Family* și *Device* (**Spartan2**, respectiv **xc2s50-5tq144**). În câmpul *Design Flow* se selectează fluxul de proiectare **XST VHDL**. Se selectează apoi butonul *OK* pentru a crea proiectul.
2. Se selectează comanda *Project* → *Add Copy of Source* pentru adăugarea fișierului cu schema numărătorului. Acest fișier se găsește în directorul `C:\Laborator\AC\proiecte\lab8_1`, numele fișierului fiind **num4jk.sch**.
3. În fereastra *Sources in Project* se execută un clic dublu pe numele fișierului **num4jk.sch** pentru vizualizarea schemei numărătorului. Se închide apoi fereastra editorului schematic.

### 2.4.2. Crearea unui fișier pentru bancul de test

Pentru crearea unui fișier care va conține vectorii de test, fișier care va fi utilizat apoi pentru crearea unui banc de test, se procedează astfel:

1. În fereastra *Sources in Project* se selectează fișierul sursă cu schema numărătorului.
2. Se selectează comanda *Project* → *New Source* pentru crearea unui nou fișier. În fereastra de dialog *New* se selectează *Test Bench Waveform* ca tip al noului fișier. În câmpul *File Name* se introduce numele fișierului care va fi creat, de exemplu, **num4jk\_tb** (numele trebuie să fie diferit de numele fișierului care conține schema).
3. Se selectează *Next*, iar în fereastra de dialog *Select* se păstrează asocierea implicită cu fișierul sursă care conține schema numărătorului (**num4jk**).
4. Se selectează *Next*, iar apoi *Finish*. Prin aceste operații, se va lansa în execuție programul HDL Bencher și va apare fereastra *Initialize Timing*.
5. În fereastra *Initialize Timing*, se setează parametrii de timp (acești parametri sunt descriși în paragraful 2.2.1). În mod implicit, frontul activ al semnalului de ceas este cel crescător (*Rising Edge*) și se păstrează această setare. Timpul cât semnalul de ceas are valoarea logică 1 (*Clock high time*) se setează la 5 ns, iar timpul cât semnalul de ceas are valoarea logică 0 (*Clock low time*) se setează de asemenea la 5 ns. Parametrii *Input setup time* și *Output valid delay* se setează la 1 ns. Se selectează apoi butonul *OK*. Vor apare cele două ferestre ale programului HDL Bencher: fereastra formelor de undă ale stimulilor și fereastra cu fișierul sursă VHDL care a fost creat pe baza schemei circuitului.

### 2.4.3. Inițializarea vectorilor de test

Semnalele de intrare ale numărătorului sunt *CLK* (semnalul de ceas) și *RST* (semnalul de resetare). Semnalul de ceas este inițializat în mod automat conform parametrilor care au fost setați în fereastra *Initialize Timing*, astfel încât este necesară doar inițializarea semnalului *RST*. În plus, se vor inițializa și semnalele de ieșire ale numărătorului cu valorile corecte (0000, 0001, 0010, ..., 1111), valori care vor fi comparate cu cele obținute la simularea circuitului, semnalându-se eventualele erori.

Pentru inițializarea semnalelor *RST*, *Q0*, *Q1*, *Q2* și *Q3*, în fereastra programului HDL Bencher care afișează formele de undă ale semnalelor se execută următoarele operații:

1. Pentru inițializarea semnalului *RST*, se setează acest semnal la valoarea 1 la timpul de simulare 0 ns, iar apoi se setează semnalul la valoarea 0 la timpul de simulare 20 ns. Pentru aceasta, se execută un clic în prima celulă albastră din dreptul semnalului *RST* (celulă marcată cu valoarea 0), iar apoi se execută un clic în a treia celulă albastră (cea corespunzătoare timpului de simulare de 20 ns).
2. Se inițializează mai întâi semnalele *Q0*, *Q1*, *Q2* și *Q3* astfel încât acestea să fie 0 în primul ciclu de ceas. Pentru aceasta, se execută un clic cu butonul din dreapta în prima celulă galbenă din dreptul semnalului *Q0* și se selectează opțiunea *Set Value*. În caseta *Get Value* se păstrează valoarea implicită 0 și se execută un clic pe butonul *OK*. Se procedează similar pentru setarea semnalelor *Q1*, *Q2* și *Q3* la valoarea 0 în primul ciclu de ceas.
3. Pentru inițializarea semnalului *Q0* astfel încât valoarea acestuia să se modifice la fiecare impuls de ceas (începând cu al doilea ciclu), se va utiliza fereastra *Pattern Wizard* pentru comutarea acestui semnal. Se execută mai întâi un clic dublu în a doua celulă galbenă din dreptul semnalului *Q0* (nu în prima celulă în care semnalul a fost setat la 0), iar apoi se execută un clic pe butonul *Pattern*. Prin aceasta se va deschide fereastra *Pattern Wizard*. În această fereastră se setează următoarele opțiuni:
  - *Choose Pattern: Toggle*
  - *Repeat pattern # times: 8*
  - *Initial value: 0*



- *Toggle every: 1*

Se selectează apoi butonul *OK*.

4. Se procedează similar pentru inițializarea semnalului *Q1* astfel încât valoarea acestuia să se modifice la fiecare două impulsuri de ceas începând cu al doilea ciclu. În fereastra *Pattern Wizard* se modifică următoarele opțiuni:

- *Repeat pattern # times: 4*
- *Toggle every: 2*

Se selectează butonul *OK*.

5. Se procedează similar pentru inițializarea semnalului *Q2* astfel încât valoarea acestuia să se modifice la fiecare patru impulsuri de ceas începând cu al doilea ciclu. În fereastra *Pattern Wizard* se modifică următoarele opțiuni:

- *Repeat pattern # times: 2*
- *Toggle every: 4*

Se selectează butonul *OK*.

6. Pentru inițializarea semnalului *Q3* astfel încât valoarea acestuia să fie 0 în ciclurile de ceas 2-9 și 1 în ciclurile de ceas 10-17, se poate utiliza de asemenea fereastra *Pattern Wizard*, modificând următoarele opțiuni:

- *Repeat pattern # times: 1*
- *Toggle every: 8*

Se selectează butonul *OK*. În primele 17 cicluri de ceas, ieșirile *Q3*, *Q2*, *Q1* și *Q0* ale numărătorului trebuie să fie setate la valorile corespunzătoare combinațiilor 0000, 0000, 0001, 0010, ..., 1111. Forma de undă a semnalelor trebuie să fie cea ilustrată în Figura 8.13.

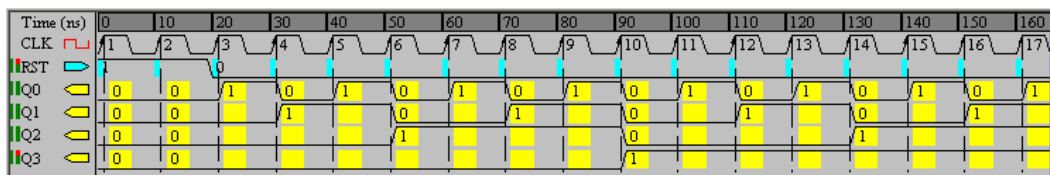


Figura 8.13. Formele de undă obținute prin inițializarea semnalelor de intrare și de ieșire.

7. Se salvează vectorii de test într-un fișier cu extensia **.tbw** utilizând comanda *File* → *Save Waveform* sau butonul *Save Waveform*.
8. Se închide programul HDL Bencher utilizând comanda *File* → *Exit*.

#### 2.4.4. Simularea funcțională

Pentru simularea funcțională a numărătorului, se procedează astfel:

1. În fereastra *Sources in Project* din ecranul *Project Navigator* se selectează fișierul cu extensia **.tbw** care a fost creat anterior cu programul HDL Bencher.
2. În fereastra *Processes for Current Source*, se execută un clic pe semnul + din stânga liniei *ModelSim Simulator* pentru afișarea proceselor disponibile.
3. Se execută un clic dublu pe linia *Simulate Behavioral VHDL Model*. Prin aceasta, se va lansa în execuție simulatorul *ModelSim*, care va executa simularea utilizând vectorii de test specificați, va compara semnalele de ieșire așteptate cu cele obținute la simulare, iar apoi va afișa rezultatele simulării în fereastra principală. Simulatorul va deschide de asemenea fereastra cu structura proiectului, fereastra semnalelor și fereastra cu forma de undă a semnalelor.

4. Se examinează fereastra principală a simulatorului. Se observă afișarea unui mesaj indicând faptul că au apărut erori la simulare (Figura 8.14).

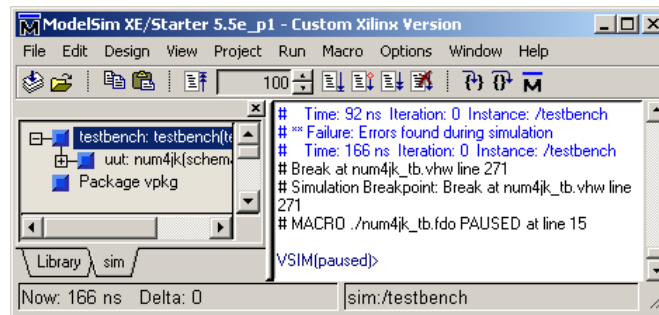



Figura 8.14. Fereastra principală a simulatorului ModelSim afișând detectarea unor erori la simulare.

5. În fereastra *Wave* se execută un clic pe butonul *Zoom Full*  pentru vizualizarea formei de undă complete a semnalelor.
6. Se grupează semnalele  $q0$ ,  $q1$ ,  $q2$  și  $q3$  într-un semnal virtual și se modifică formatul de afișare al acestui semnal. Pentru aceasta, se execută următoarele operații:
- Se selectează comanda *Edit* → *Sort* și se selectează opțiunea *Descending*. Semnalele vor fi sortate în ordine inversă celei alfabetice.
  - Se selectează semnalele  $q3$ ,  $q2$ ,  $q1$  și  $q0$ , iar apoi se selectează comanda *Edit* → *Combine*. Se va deschide fereastra *Combine Selected Signals*. În câmpul *Name*: se introduce numele semnalului virtual obținut prin combinarea celor patru semnale, de exemplu,  $q$ , se selectează opțiunea *Descending*, iar apoi se selectează butonul *OK*. Noul semnal va fi afișat în fereastra *Wave*.
  - Se selectează semnalul virtual creat, se execută un clic cu butonul din dreapta și se selectează opțiunea *Radix*, iar apoi *Hexadecimal*. Valoarea semnalului va fi afișată în hexazecimal.
7. În fereastra *Wave* se observă că secvența de numărare obținută, indicată prin succesiunea valorilor semnalului  $q$ , este 0, 1, 2, 3, 4, 5, E, F, 0, 1, 2, 3, 4, 5, E, F. Această secvență nu este cea corectă.
8. Se salvează lista semnalelor din fereastra *Wave*, executând comanda *File* → *Save Format*. Se păstrează numele implicit al fișierului (**wave.do**) și se selectează butonul *Save*.
9. Se închide sesiunea de simulare prin închiderea ferestrei principale a simulatorului sau prin execuția comenzii *File* → *Quit* din fereastra principală.

#### 2.4.5. Corectarea erorilor

Pentru corectarea erorilor din schema numărătorului, se revine în fereastra *Project Navigator*. Se execută un clic dublu pe numele fișierului **num4jk.sch** pentru lansarea în execuție a editorului schematic și se examinează schema numărătorului. Se poate observa că intrarea de sus a porții  $\Sigma I$  din dreapta este conectată la semnalul  $Q0$ , și nu la ieșirea porții  $\Sigma I$  din stânga.

Se selectează conexiunea eronată, iar apoi se șterg conexiunile selectate prin apăsarea tastei **Delete**. Se refac conexiunile care au fost șterse și se conectează în mod corect intrarea de sus a porții  $\Sigma I$  din dreapta. Se adaugă apoi conexiunea pentru semnalul  $Q0$ , se atașează numele semnalului la această conexiune și se adaugă un port de ieșire pentru acest semnal.

Se salvează schema și se închide fereastra editorului schematic.

### 2.4.6. Reexecutarea simulării funcționale

Se lansează din nou în execuție simulatorul, executând un clic dublu pe linia *Simulate Behavioral VHDL Model*. Se observă că în fereastra principală a simulatorului apare un mesaj indicând faptul că nu au fost detectate erori:

```
# ** Failure: Simulation successful (not a failure). No problems detected.
```

În fereastra *Wave*, se selectează toate semnalele, iar apoi se șterg prin apăsarea tastei **Delete** (cursorul trebuie să se afle în partea din stânga a ecranului). Se execută comanda *File* → *Load Format*, se păstrează numele **wave.do** și se selectează butonul *Open*. Astfel se va încărca lista semnalelor care a fost salvată anterior, inclusiv formatul de afișare al acestora. Se observă că secvența de numărare este acum cea corectă.

Se închide sesiunea de simulare.

## 3. Desfășurarea lucrării

**3.1.** Executați toate etapele descrise în secțiunea 2.4 pentru simularea funcțională a numărătorului de 4 biți realizat cu bistabile JK.

**3.2.** Întocmiți tabelul de adevăr al unui multiplexor 4:1. Intrările de date ale multiplexorului sunt D0, D1, D2, D3, intrările de selecție sunt S1, S0, iar ieșirea este Z. Scrieți ecuația ieșirii, desenați schema multiplexorului cu editorul schematic, creați un banc de test cu programul HDL Bencher și simulați funcționarea multiplexorului cu simulatorul ModelSim.

**3.3.** Scrieți ecuațiile ieșirilor unui decodificator 3:8. Intrările decodificatorului sunt A, B, C, iar ieșirile sunt Y0, Y1, ..., Y7. Desenați schema decodificatorului cu editorul schematic, creați un banc de test cu programul HDL Bencher și simulați funcționarea decodificatorului cu simulatorul ModelSim.

**3.4.** Utilizați un multiplexor 4:1 din biblioteca de componente a sistemului ISE WebPACK (**m4\_1e**) pentru implementarea următoarei funcții booleene:

$$F(A, B, C) = \Sigma(0, 2, 4, 5)$$

Desenați schema circuitului cu editorul schematic, conectând intrarea de validare E la 1 logic. Creați un banc de test cu programul HDL Bencher și simulați funcționarea circuitului cu simulatorul ModelSim.

**3.5.** Implementați prin porți logice codificatorul prioritara cu 4 intrări definit prin tabelul de adevăr de mai jos (Tabelul 8.2).

**Tabelul 8.2.** Tabelul de adevăr al unui codificator prioritara cu 4 intrări.

EI	Intrări				Ieșiri			
	I3	I2	I1	I0	Y1	Y0	GS	EO
0	X	X	X	X	0	0	0	0
1	0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1	0
1	0	0	1	X	0	1	1	0
1	0	1	X	X	1	0	1	0
1	1	X	X	X	1	1	1	0

Intrarea EI este utilizată pentru validarea circuitului, iar I3, I2, I1, I0 sunt intrările de date. Y1 și Y0 sunt ieșirile de date, la care se obține codul intrării activate. În cazul în care sunt activate mai multe intrări simultan, la ieșire se obține codul intrării celei mai prioritare. Intrarea I3 este cea mai prioritara, iar I0 este cea mai puțin prioritara. De exemplu, dacă intrarea I3 este 1, codul de la ieșire

este 11, indiferent de valorile celorlalte intrări. Dacă circuitul nu este validat ( $EI = 0$ ), ieșirile sunt inactice (0 logic). Ieșirea GS este activată dacă cel puțin una din intrările de date este activată, iar ieșirea EO este activată atunci când nici una din intrările de date nu este activată.

Întocmiți diagramele Karnaugh pentru ieșirile codificatorului în funcție de intrările de date I3, I2, I1, I0, scrieți ecuațiile minimizate ale ieșirilor, iar apoi modificați ecuațiile pentru a ține cont de intrarea de validare EI. Desenați schema circuitului cu editorul schematic, creați un banc de test cu programul HDL Bencher și simulați funcționarea circuitului cu simulatorul ModelSim.

**3.6.** Simulați funcționarea circuitului pentru detectarea secvenței de intrare 1010 din lucrarea de laborator Nr. 4 (problema 3.3). Desenați schema circuitului, creați un banc de test, iar apoi simulați funcționarea.

**3.7.** Simulați funcționarea numărătorului cu secvența de numărare 0, 1, 2, 4, 5, 6, 0 din lucrarea de laborator Nr. 4 (problema 3.7). Desenați schema numărătorului utilizând bistabile JK, creați un banc de test, iar apoi simulați funcționarea.