

Simulera med ModelSim



ModelSim - simuleringsprogramvara

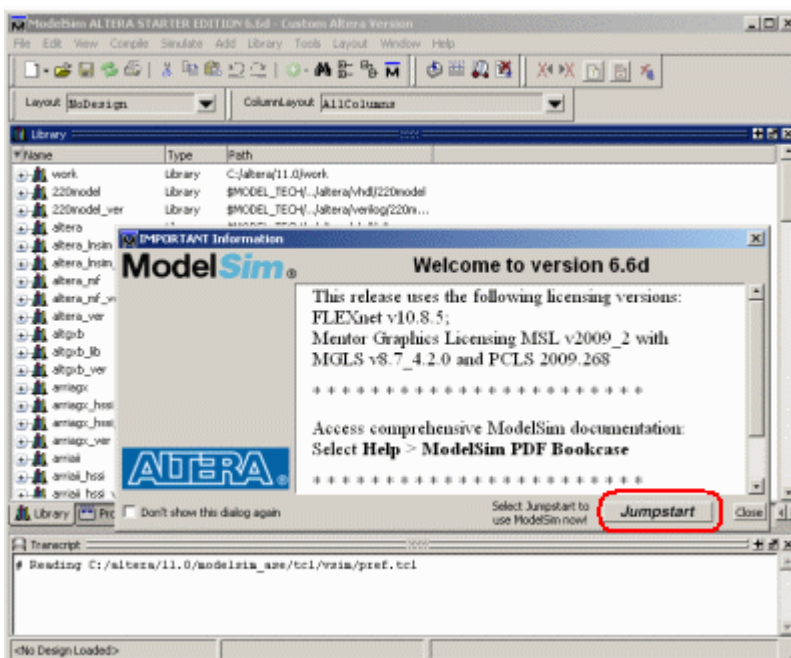
ModelSim kan användas till att simulera VHDL-kod, för att avgöra om den är "rätt" tänkt. Alteras version av **ModelSim** är också kopplad till en "databas" med fakta om Altera-kretsar, tex. MAX-kretsarna, så man kan också göra simuleringar som tar hänsyn till "tidsfördröjningar" och andra fenomen inuti den tänkta målkretsen. (Så länge som målkretsen är av Altera's fabrikat ...).



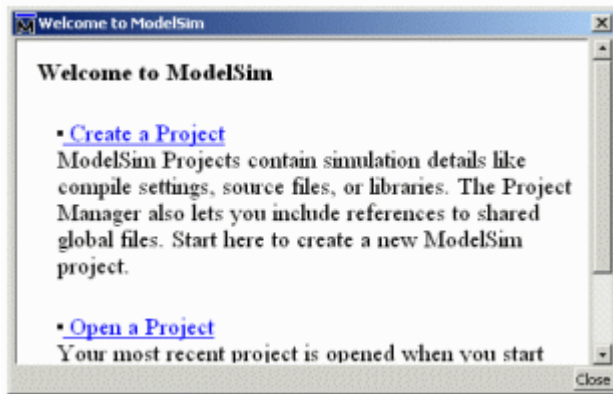
Välj rätt programversion - i skolan finns flera olika installerade under startmenyn!

```
Altera 13.0.1.232 Web edition\  
  ModelSim-Altera Starter Edition 13.0.1.232\  
    ModelSim-Altera 10.1d(Quartus II 13.0sp1)
```

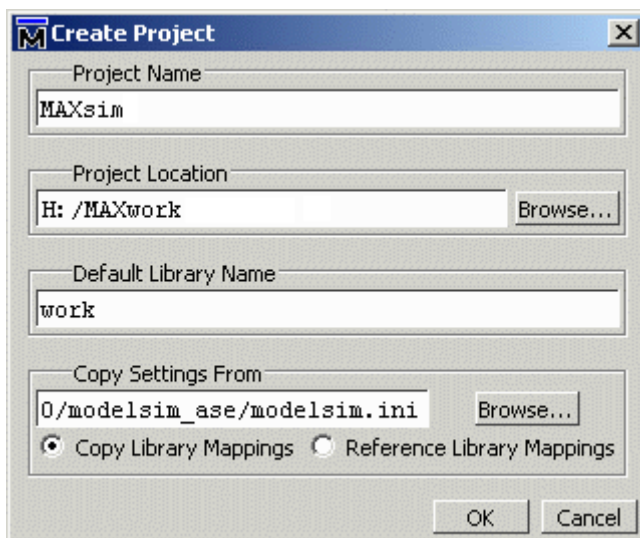
Starta ModelSim.



I fönstret "important information" klickar man på **Jumpstart** för att få hjälp med att sätta upp ett projekt.



Därefter klickar man på länken "Create a Project" i välkomstfönstret.



Skapa ett projekt.

Project Name

MAXsim kan vara ett lämpligt namn

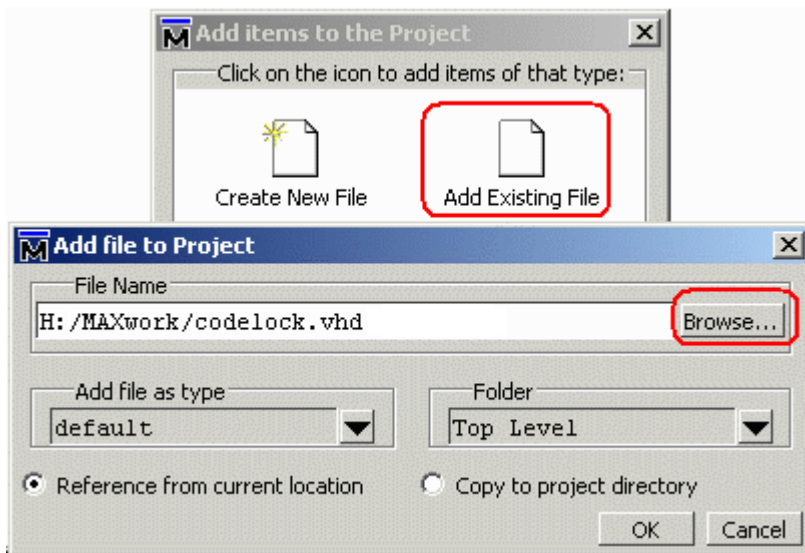
Project location

H: /MAXwork bläddra dig fram till samma arbetsmapp som Du använde för Quartus

Default Library Name

work behåll det föreslagna namnet, det är standard vid VHDL-simulering

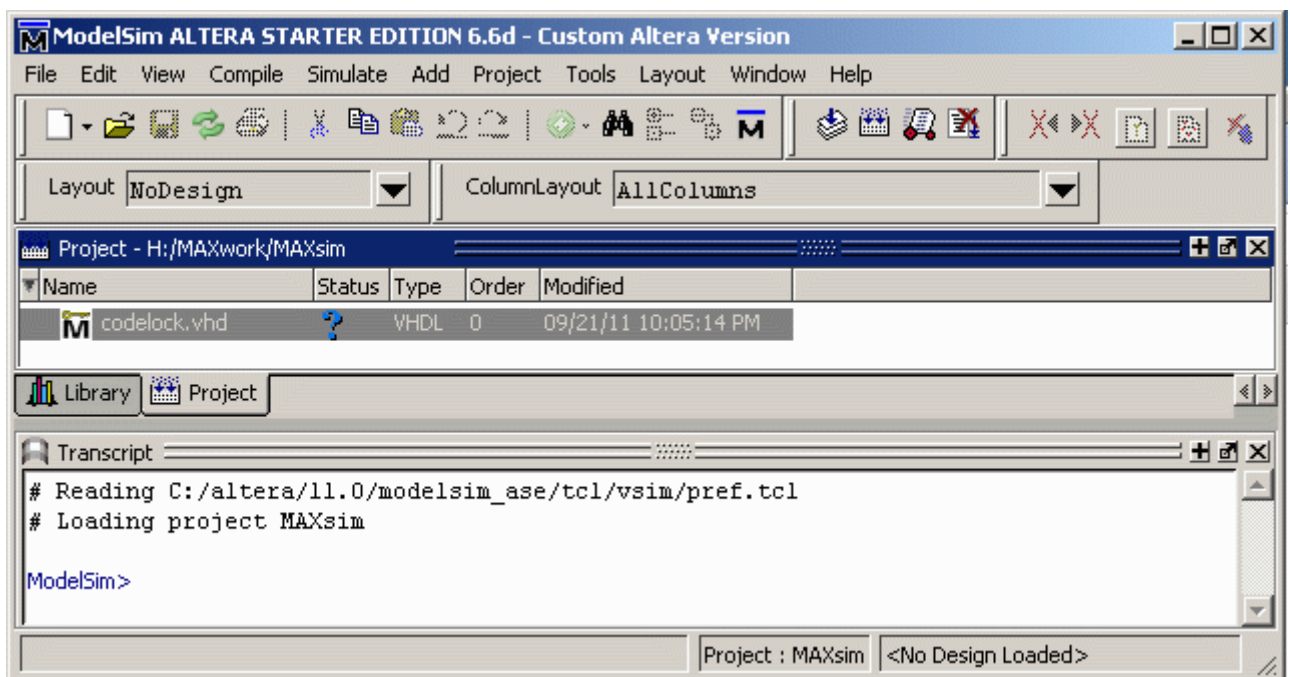
Klicka på **OK**.



Vi väljer "Add Existing File" för att lägga till en VHDL-fil till projektet.
 "Bläddra" fram till filen `codelock.vhd` som vi tidigare skapade med **Quartus**.

Klicka på **OK**. Därefter klicka på **Close**.



Kodlåskoden i ModelSim



ModelSim har en *egen* kompilator för att ta fram simuleringen ur VHDL-koden. Fast vi har kompilerat VHDL-koden i **Quartus** måste vi trots det kompilera den igen för **ModelSim**.

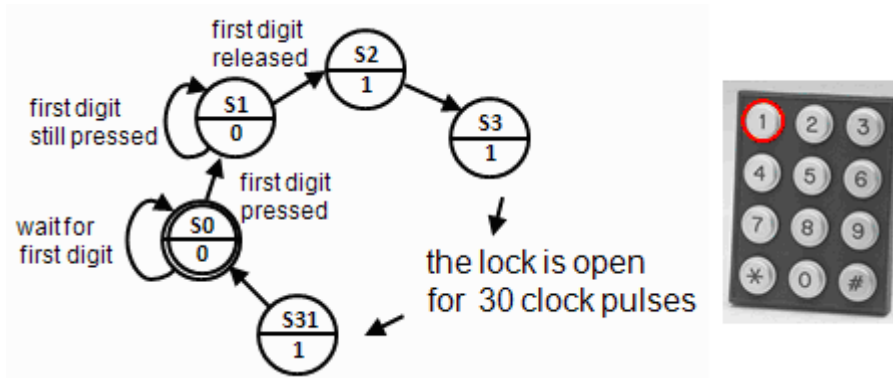
Name	Status
codelock.vhd	?

Välj **Compile** menyn, alternativet **Compile All**.

Name	Status
 codelock.vhd	

Nu är VHDL-koden också kompilerad för **Modelsim**.
Statussymbolen ändras från ett blått frågetecken till en grön bock!

Simulera kodlås-mallen!

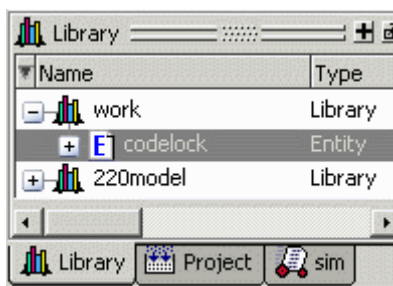


Simuleringen går till så att vi ger olika komandon i **Transcript**-fönstret, och sedan följer ett utvalt antal signaler i fönstret **Wave**.

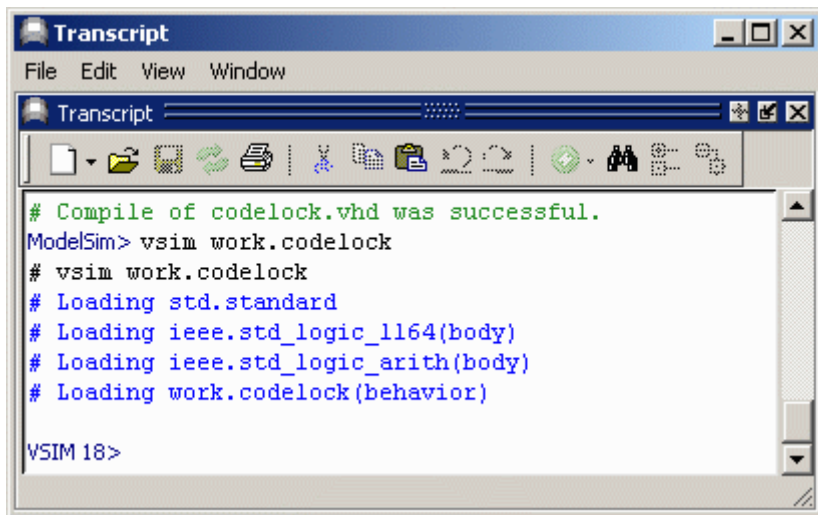
Transcript-fönstret är ett klassiskt teckenbaserat terminalfönster där man ger kommandon, men man kan även ge de flesta kommandon genom menyval, eller genom att klicka på knappar. Kommandon skrivs dock ut i ut **Transcript**-fönstret, oavsett hur dom givits.

Ladda Designen till simulatorn.

Välj fliken **Library**, och öppna mappen **work**. Dubbelklicka på "Entity" **codeLock**. En serie kommandon utförs nu som resulterar i att designen laddats in till simulatorn.



I **Transcript**-fönstret kan man följa vilka kommandon det är som utförts.

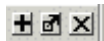
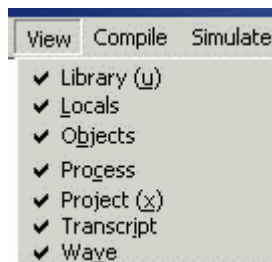


Förberedelser inför simuleringen

Vi behöver ha ett antal fönster öppna för att kunna följa simuleringen.

Ge kommandon i **Transcript**-fönstret eller klicka för i **View**-menyn.

```
VSIM> view objects
VSIM> view locals
VSIM> view source
VSIM> view wave -undock
```



Modelsim består av ett otal "fönster". Det kan vara svårt att se allt på en gång. Med knappen **Zoom/Unzoom** förstorar man fönstret (på andra fönsters bekostnad). Med knappen **Dock/Undock** kan fönstret flyttas till valfri plats, det alternativet valde vi för **Wave**-fönstret. Med knappen **Close** kan man stänga fönster som inte behövs för tillfället.

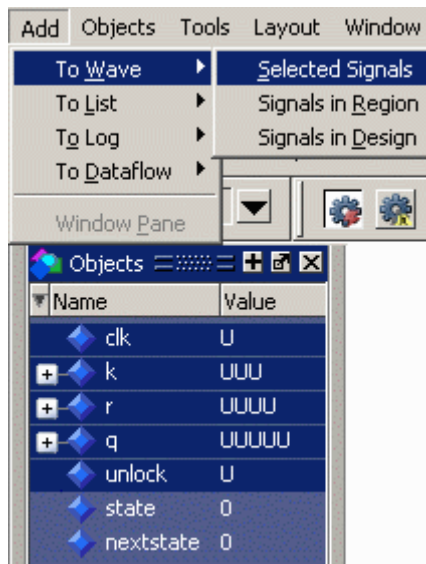
Signaler i Wave

Nästa steg är att ange för simulatören vilka signaler Du vill följa i **Wave**-fönstret. Har man många signaler är det en bra idé att välja ut de signaler man är intresserad av, men här väljer vi att följa alla:

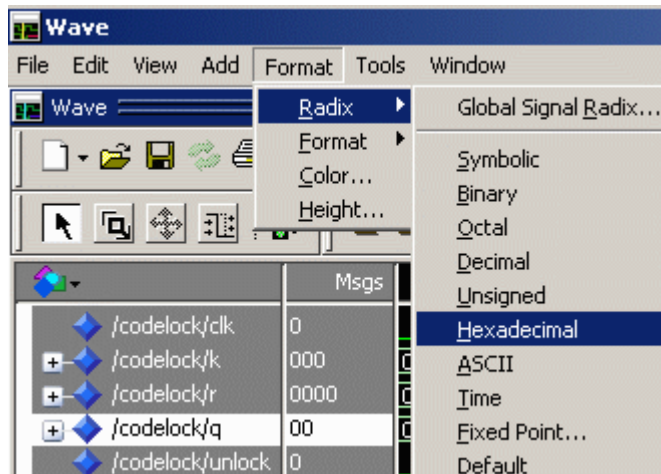
```
add wave *
```

Det finns flera alternativa sätt att lägga till signaler till **Wave**-fönstret:

- Välj signaler i **Object**-fönstret (Shift+Vänster Musknapp) och "dra och släpp" urvalet till **Wave**-fönstret.
- Högerklicka i **Object**-fönstret och välj **Add to Wave**.
- Ett **Add to Wave** dialog-fönster nås från menyraden under **Add**.

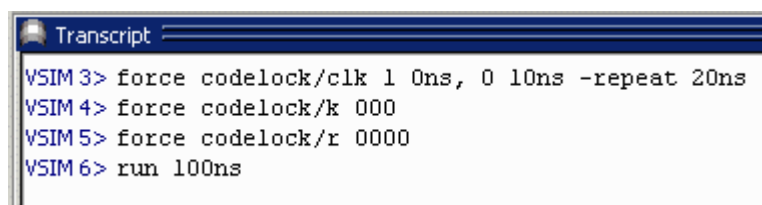


Format, Radix, Hexadecimal



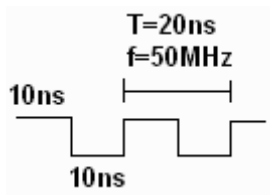
Tillståndsvariabeln `q` har 32 olika tillstånd, en sådan variabel är lättare att följa om den anges som en hexadecimal siffra, `00 . . . 1F` i stället för som ett femsiffrigt binärtal. Vi föreslår därför att Du markerar den variabeln och byter den till Hexadecimal. `UUUUU` byts då mot `XX` i **Wave**-fönstret. Övriga variabler passar bäst som binärtal.

Skapa stimuli



Den förinställda tidsupplösningen i **Wave** är nanosekunder, `ns`. En lämplig klockfrekvens för ett kodlås kan däremot vara så låg som 5 Hz, dvs. en periodtid om 0,2 sek. Enklast, för att inte behöva göra omfattande omställningar av programmet, är att "skala" om

problemet till en högre klockfrekvens med periodtiden 20 ns. Vi får i så fall då också tänka oss att det är "flinka" fingrar som trycker på knapparna.



Stimuli, dvs. signaler som klockpulser eller knapptryckningar, kan skapas med kommandot `force` i **Transcript**-fönstret.

```
force codelock/clk 1 0ns, 0 10ns -repeat 20ns
```

Genererar klockpulser för evigt.

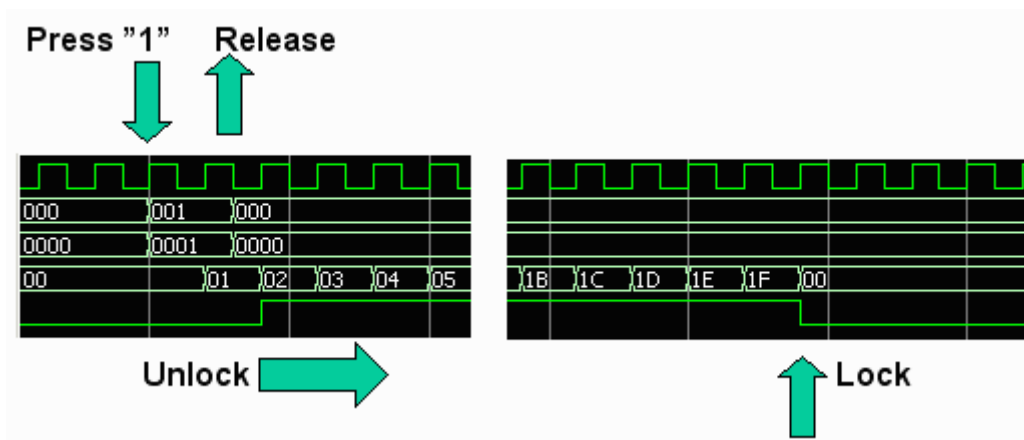
```
force codelock/k 000
force codelock/r 0000
```

Initierar variablerna `r` och `k`.

```
run 100ns
```

Kör simuleringen i 100 ns, dvs. fem hela perioder.

Simulera knapptryckningen



```
force codelock/k 001
force codelock/r 0001
run 30ns
force codelock/k 000
force codelock/r 0000
run 800ns
```

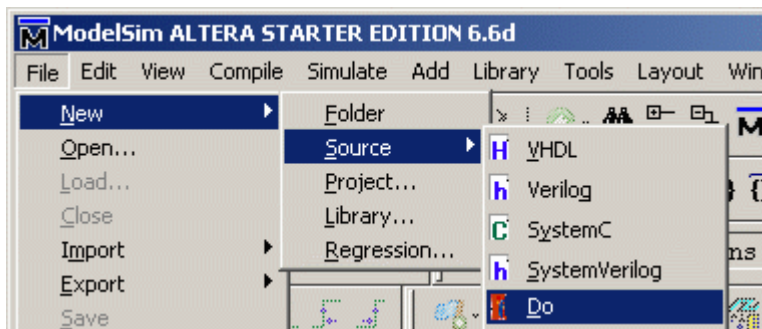
30 ns (20+10) innebär att knapptryckningen med säkerhet kommer i mellanrummet mellan klockflankerna. Hela simuleringstiden 100 ns + 30ns + 800 ns = 930 ns motsvarar 46,5 klockpulser. Detta räcker för att visa låsets hela öppningsförlopp.

Do-file

I stället för att skriva kommandon direkt i **Transcript**-fönstret, kan man köra många kommandon i följd som står i en sk. Do-file.

Alternativt kan man i Windows kopiera text (Ctrl-C) och klistra in den (Ctrl-V) i **Transcript**.

```
delete wave *
add wave codelock/clk
add wave codelock/k
add wave codelock/r
add wave codelock/q
add wave codelock/unlock
force codelock/clk 1 0ns, 0 10ns -repeat 20ns
force codelock/k 000
force codelock/r 0000
run 100ns
force codelock/k 001
force codelock/r 0001
run 30ns
force codelock/k 000
force codelock/r 0000
run 800ns
```

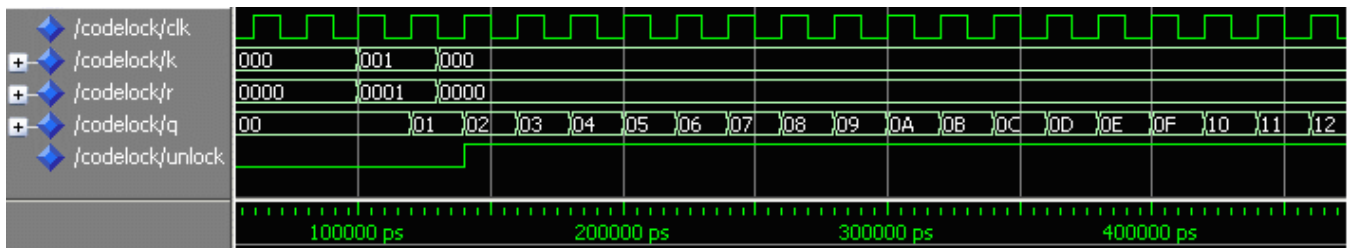


Så här skapar man en Do-file. Klistra in textkommandona ovan i filen. Spara den sedan bland de övriga filerna (i MAXwork) med filnamnställägget `.do`.

Du kör sedan en Do-file med dessa kommandon (i **Transcript**):

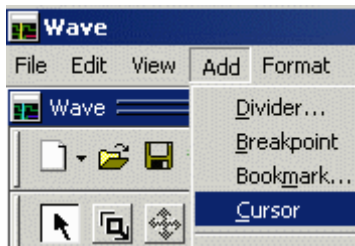
```
restart -f
do lock.do
```

Hitta i Wave-fönstret

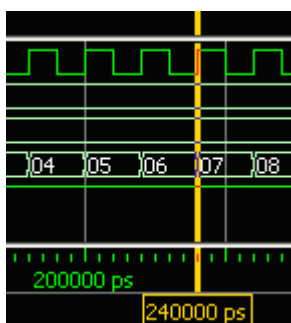
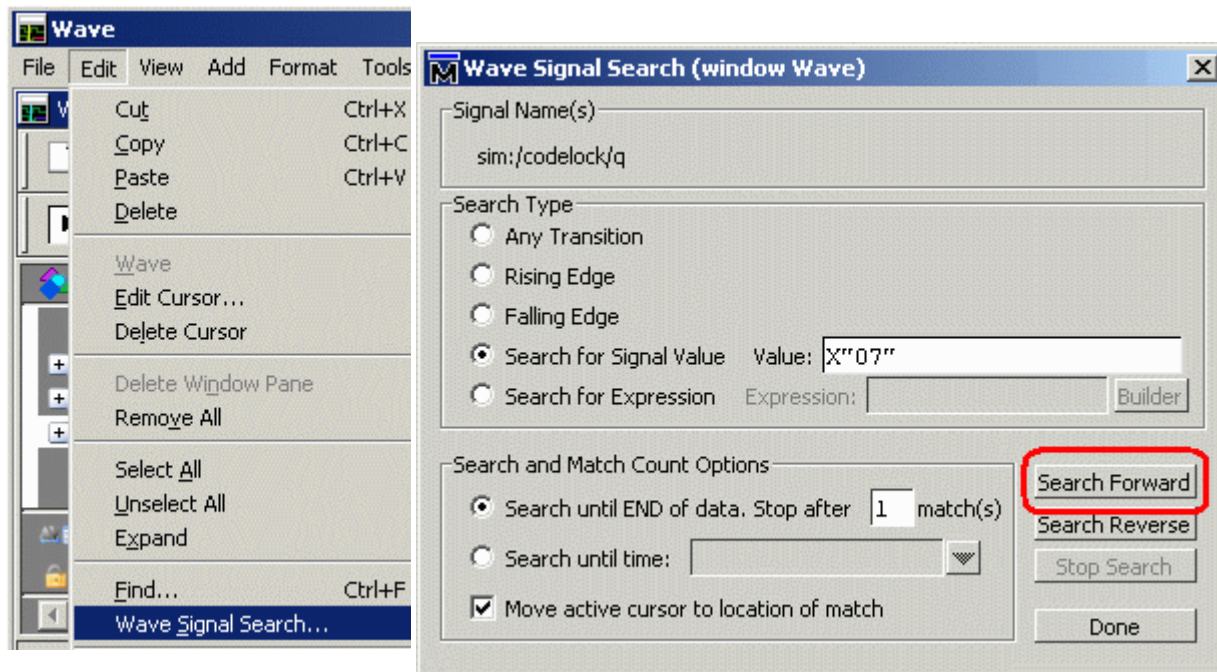


Det kan vara svårt att hitta det man söker efter i Wavefönstret. Det finns därför en hel rad med hjälpverktyg som Zoom, Expanded time, Cursors mm.

Add, Cursor.



En Cursor kan användas tillsammans med funktionen **Edit, Wave Signal Search**.



Nu pekar Cursorn ut vad som händer (tydligen inget speciellt!) när q har tillståndet 07.

Ägna nu lite tid åt att prova olika verktyg som finns för att orientera sig i Wavefönstret!

Simuleringen har visat att låset öppnar för den tilltänkta knapptryckningen, men detta är inte tillräckligt - det behövs mycket mer "testande" innan man törs lita på konstruktionen!

William Sandqvist william@kth.se