1η Εργαστηριακή Άσκηση

Εξοικείωση με την Προσομοίωση Κυκλωμάτων

Εξέταση Άσκησης: 15/3/2023
Παράδοση αναφοράς: 26/3/2023

Στην 1η εργαστηριακή άσκηση να εξοικειωθείτε με την προσομοίωση κυκλωμάτων που έχουν περιγραφεί σε HDL (Verilog) καθώς και στον έλεγχο της ορθής λειτουργίας τους (functional simulation). Για simulation θα χρησιμοποιήσετε το εργαλείο VCS της Synopsys και τα προβολή των κυματομορφών το Synopsys DVE (υπάρχει εγκατεστημένο και το Verdi που προσφέρει περισσότερες debugging ιδιότητες). User Guide για αυτά τα εργαλεία υπάρχουν στα έγγραφα του eclass. Στα user guides μπορείτε να βρείτε αναλυτικές λεπτομέρειες και διάφορα παραδείγματα.

Αντιγράψτε στο home σας τα απαιτούμενα αρχεία για την 1η εργαστηριακή άσκηση και αρχικοποιήστε τις απαραίτητες μεταβλητές περιβάλλοντος:

```
cp -r /usr/local/vlsi2_2023/lab1 .
cd lab1
source env.sh
```

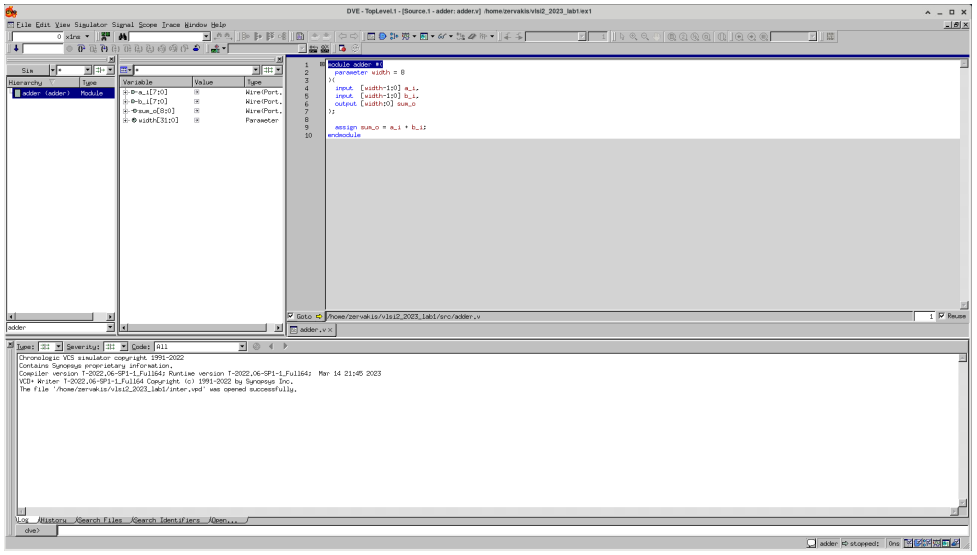
Στον φάκελο `src` μπορείτε να βρείτε τους πηγαίους κώδικες των designs για τα παραδείγματα αυτής της άσκησης. Θα χρησιμοποιήσετε την εντολή `make` για να εκτελέσετε τα παραδείγματα αυτά. Μελετείστε το `Makefile` για να δείτε τις εντολές που εκτελούνται σε κάθε παράδειγμα.

Παράδειγμα 1

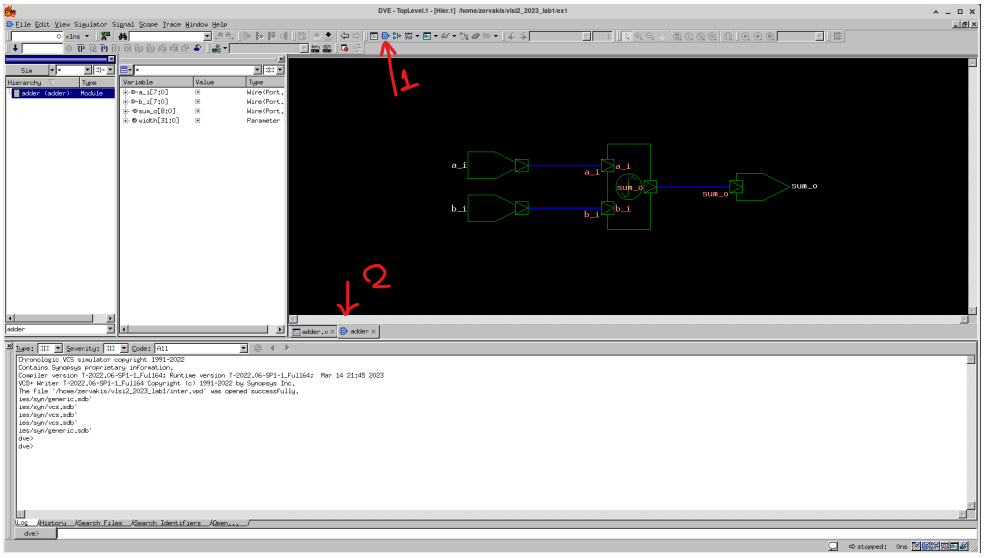
Στο παράδειγμα αυτό θα χρησιμοποιηθεί το design `src/adder.v`. Να περιγράψτε την λειτουργία του. Στη συνέχεια τρέξτε:

```
make example1
```

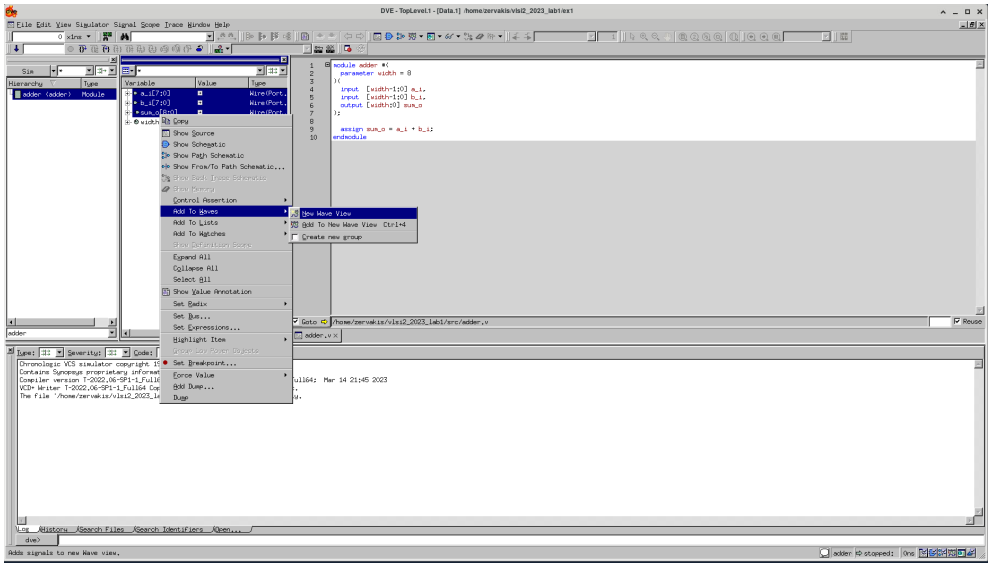
Η εντολή αυτή χρησιμοποιεί το `vcs` για κάνει compile το `src/adder.v` και δημιουργήσει το εκτελέσιμο `exe1`. Οι παράμετροι `-debug_access+all` και `-timescale=10ns/1ns` χρησιμοποιούνται στο compilation. Η δεύτερη μπορεί να χρησιμοποιηθεί αν δεν ορίζεται το timescale στα modules που προσομοιώνονται. Η πρώτη ενεργοποιεί debugging capabilities στο εκτελέσιμο (π.χ. βηματική εκτέλεση στο γραφικό κλπ.). Ανατρέξτε στο user guide για μια πλήρη συλλογή των υποστηριζόμενων παραμέτρων. Αφου δημιουργηθεί το εκτελέσιμο το τρέχουμε με όρισμα `-gui=dve` που υποδηλώνει ότι θέλουμε να κάνουμε χρήση του γραφικού περιβάλλοντος DVE. Το παράθυρο του DVE ανοίγει (Εικόνα 1). Επιλέξτε το module σας και πατήστε *show schematic* για να δείτε το σχηματικό διάγραμμα του κυκλώματος (Εικόνα 2). Στη συνέχεια επιλέξτε τα σήματα που επιθυμείτε και βάλτε τα σε ένα *wave* για να αναλύσετε τις κυματομορφές (Εικόνα 3). Στο νέο παράθυρο επιλέξτε τα σήματα εισόδου και θέστε του τιμές με την *force* (Εικόνα 4). Ορίστε ένα χρονικό βήμα και τρέξτε την προσομοίωση πατώντας *start/continue* (Εικόνα 5). Εξοικειωθείτε με το περιβάλλον, πειραματιστείτε με τις διάφορες λειτουργίες του simulator, επαναλάβετε τη διαδικασία αρκετές φορές και παρατηρείστε την έξοδο σαν συνάρτηση των εισόδων. Επιτελεί το κύκλωμα την αναμενόμενη λειτουργία;



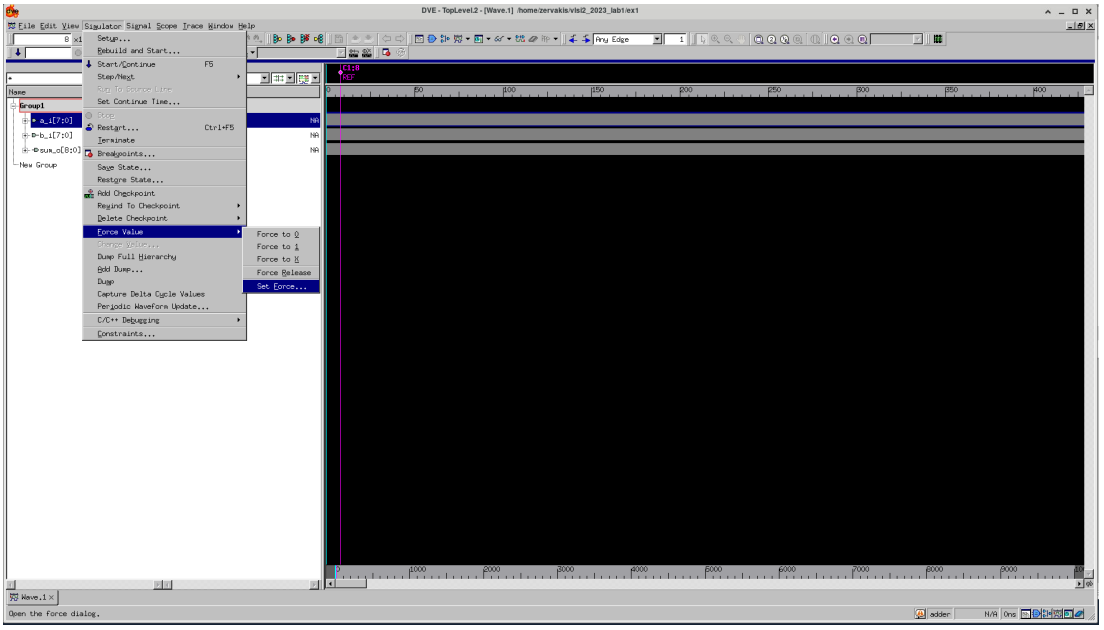
Εικόνα 1



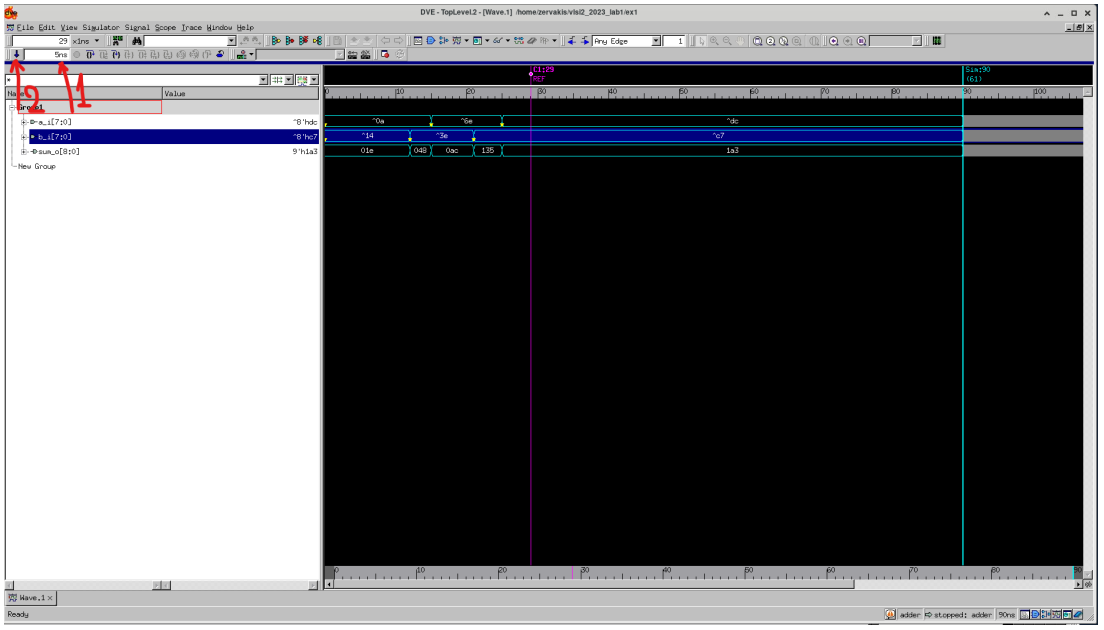
Εικόνα 2



Εικόνα 3



Εικόνα 4



Εικόνα 5

Παράδειγμα 2α

Στο παράδειγμα αυτό θα χρησιμοποιηθεί το design `src/mac.v`. Να περιγράψτε την λειτουργία του. Στη συνέχεια τρέξτε:

```
make example2
```

Επαναλάβετε τα βήματα του παραδείγματος 1. Τι ουσιαστική διαφορά υπάρχει στα κυκλώματα των δύο παραδειγμάτων; Προσέξτε αν υπάρχουν σήματα στο mac module που ίσως να χρήζουν ιδιαίτερης μεταχείρισης. Πειραματιστείτε αρκετά και προχωρήστε στο επόμενο παράδειγμα.

Παράδειγμα 2β

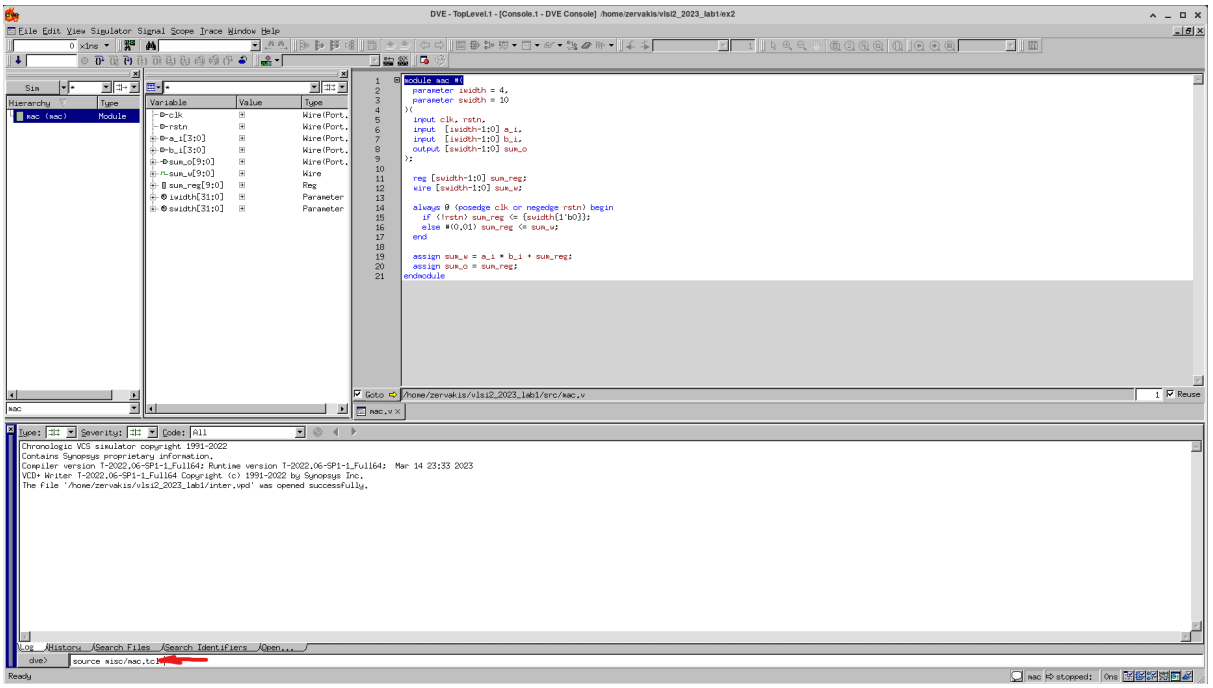
Επαναλάβετε το προηγούμενο παράδειγμα:

```
make example2
```

Αυτή τη φορά, μόλις ανοίξει το DVE, στο command prompt του τρέξτε (Εικόνα 6):

```
source misc/mac.tcl
```

Μελετήστε το `misc/mac.tcl` και να το συγκρίνεται με το αποτέλεσμα της εκτέλεσης της εντολής αυτής. Τι συμπεράσματα βγάζετε;



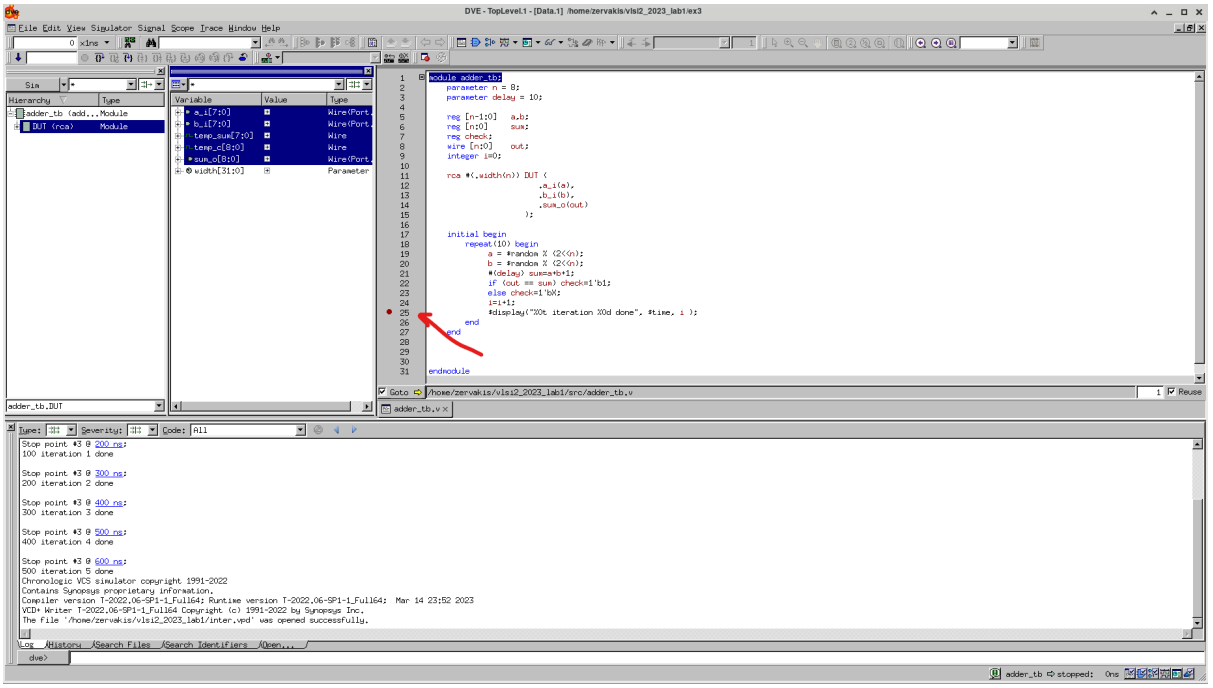
Εικόνα 6

Παράδειγμα 3

Στο παράδειγμα αυτό θα χρησιμοποιηθεί το design `src/rca.v` το οποίο χρησιμοποιεί και το `src/fulladder.v`. Παρατηρείστε τη δομή του και περιγράψτε την λειτουργία του. Επίσης, θα χρησιμοποιηθεί σε αυτήν την περίπτωση και το testbench `src/adder_tb.v` για τον έλεγχο της ορθής λειτουργίας του rca. Μελετήστε το adder_tb και στη συνέχεια τρέξτε την ακόλουθη εντολή. Σημείωση: σε αυτό το παράδειγμα χρησιμοποιήθηκε η το `vlogan` utility του vcs για να κάνει analyze τα verilog designs, ωστόσο θα μπορούσε να χρησιμοποιηθεί απλά όπως και πριν μόνο εντολή `vcs`.

```
make example3
```

Επαναλάβετε τα βήματα του παραδείγματος 1. Αρχικά μελετήστε το σχηματικό του rca. Τι διαφορές παρατηρείτε σε σχέση με το σχηματικό του παραδείγματος 1; Στη συνέχεια να βάλετε ένα break point κάνοντας διπλό κλικ σε μια κατάλληλη για το σκοπό αυτό γραμμή του testbench (Εικόνα 7). Βάλτε τα σήματα που σας ενδιαφέρουν στο wave. Προσοχή, μπορείτε να επιλέξετε τόσο σήματα του `adder_tb` όσο και του `DUT`. Αυτή τη φορά μην ορίσετε κάποιο χρονικό βήμα για το simulation και πατήστε *start/continue*. Τι παρατηρείτε (περισσότερα του ενός συμπεράσματα); Τέλος, ελέγξτε τα modules adder_tb και rca και προχωρήστε σε όποια αλλαγή/διόρθωση χρειάζεται.



Εικόνα 7

Παράδειγμα 4α

Αυτό το παράδειγμα χρησιμοποιεί το ίδιο `DUT` με το παράδειγμα 3 με ένα ελαφρώς διαφορετικό testbench. Μελετήστε `το src/adder_tb2.v`. Τι διαφορές έχει συγκριτικά με το παράδειγμα 3; Εν συνεχεία τρέξτε:

```
make example4
```

Μελετήστε το makefile και προσπαθήστε να εξηγήσετε το παράδειγμα αυτό. Ποια είναι τα συμπεράσματά σας; Σημείωση: σε αυτό το παράδειγμα δεν μπορούμε να δούμε τις κυματομορφές.

Παράδειγμα 4β

Τρέξτε το προηγούμενο παράδειγμα ως εξής (θεωρούμε ότι έχουμε κάνει compile και έχει παραχθεί το εκτελέσιμο ex4 στο προηγούμενο παράδειγμα):

```
./ex4 -ucli -do misc/dump.tcl
dve -full64 -vpd ex4.vpd
```

Σε αυτή την περίπτωση τρέξαμε το simulation σε cli debugging mode και κάναμε `dump` τις τιμές των σημάτων του `DUT` (δείτε το `misc/dump.tcl`) για να μπορούμε να τις αναλύσουμε μετά το πέρας του simulation.

Ζητούμενα

1. Τροποποιήστε κάποιο από τα ανωτέρω παραδείγματα για να ελέγξτε τη λειτουργία ενός 8-bit rca για όλες τις πιθανές εισόδους.
2. Φτιάξτε ένα testbench που θα ελέγχει τη λειτουργία του mac.
3. Περιγράψτε σε Verilog έναν διπλό counter που μετράει ως εξής:
   c0: 0, 1, 2, 3, 4, 5, 6, 7, 1, 2, 3, 4, 5, 6, 7, 1,..
   C1: 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 4,…
   Η έξοδο c0 είναι των 3 bit και η c1 των 16 bit.
   Να περιγράψτε κι ένα testbench που να ελέγχει τη λειτουργία του counter.

Σημείωση: Να ανεβάσετε στο eclass ένα zip που να περιέχει σύντομη αναφορά καθώς και όλους του πηγαίους κώδικες που αναπτύξατε.