Θέματα 4ης άσκησης

1. Σκοπός της άσκησης

Στην άσκηση αυτή συνεχίζουμε στο online Chisel bootcamp. Επισκεφθείτε τη διεύθυνση:

https://mybinder.org/v2/gh/freechipsproject/chisel-bootcamp/master

Σκοπός της άσκησης είναι η εκμάθηση της περιγραφής κυκλωμάτων με τη μέθοδο της «συμπεριφοράς»: αντί να περιγράφουμε ένα κύκλωμα με λογικές πράξεις (λογικές πύλες), επιτυγχάνουμε το ίδιο αποτέλεσμα περιγράφοντας το **πώς πρέπει να συμπεριφερθεί** το κύκλωμα.

Χρησιμοποιήστε το notebook της 4ης άσκησης που θα βρείτε στο site του μαθήματος, ανεβάζοντάς το στο online Chisel bootcamp. Στη συνέχεια συμπληρώστε τα κελιά σύμφωνα με τις οδηγίες.

2. Η δομή when..elsewhen..otherwise.

Μπορούμε να περιγράψουμε τη *συμπεριφορά* ενός κυκλώματος στην Chisel με τη δομή **when..elsewhen..otherwise** ως εξής:

```
when ( condition1 ) {
  // εάν η συνθήκη condition1 είναι αληθής
} . elsewhen ( condition2 ) {
  // αλλιώς, εάν η συνθήκη condition2 είναι αληθής
} . otherwise {
  // σε κάθε άλλη περίπτωση εκτός από τις προηγούμενες
}
```

Μέσα στις αγκύλες { } θα προσθέσετε τον κώδικα Chisel. Θυμηθείτε ότι ο κώδικας αυτός **δεν εκτελείται υπό συνθήκη** (όπως σε μια κανονική γλώσσα προγραμματισμού). Αντιθέτως η Chisel θα κατασκευάσει το κατάλληλο κύκλωμα που **θα υπολογίζει τις τιμές του με διαφορετικό τρόπο, ανάλογα με τη συνθήκη**.

Τα **elsewhen** και **otherwise** είναι προαιρετικά. Μπορείτε να προσθέσετε όσα **elsewhen** απαιτεί η λογική του κυκλώματός σας.

Για τις συνθήκες (conditions) η Chisel παρέχει τους εξής τελεστές:

`===`	ισότητα (τριπλό ίσον!)
`=/=`	ανισότητα
`<, >, >=, <=`	τελεστές σύγκρισης
`!, &&, ||`	λογικοί τελεστές

3. Η δομή switch.

Εναλλακτικά, μπορούμε να περιγράψουμε τη *συμπεριφορά* ενός κυκλώματος με τη δομή **switch**:

```
switch ( x ) {
  is ( value1 ) {
    // εάν x === value1
  }
  is ( value2 ) {
    // εάν x === value2
  }
}
```

Μπορείτε να προσθέσετε όσα **is** απαιτούνται. Επειδή δεν υπάρχει default κλάδος, καλό είναι να αρχικοποιείτε τα σήματα που παράγονται μέσα στα **{ }** με κάποια αρχική τιμή.

4. Ο αποκωδικοποιητής 2-σε-4.

Ο **αποκωδικοποιητής 2-σε-4** είναι ένα κύκλωμα με είσοδο 2 bits (**a[1:0]**) και έξοδο 4 bits (**y[3:0]**). Η λειτουργία του αποκωδικοποιητή περιγράφεται από τον παρακάτω πίνακα αλήθειας:

a(1)	a(0)	y(3)	y(2)	y(1)	y(0)
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Στο **notebook του 4ου εργαστηρίου** θα βρείτε την υλοποίηση του αποκωδικοποιητή α) με λογικές πράξεις (πύλες), β) με τη δομή when και γ) με τη δομή switch. **Μελετήστε** τους 3 τρόπους υλοποίησης και **ελέγξτε την ορθότητα** καθενός από αυτούς.

3. Ο πολυπλέκτης 4-σε-1.

Ο **πολυπλέκτης 4-σε-1** είναι ένα κύκλωμα που επιλέγει μία από 4 εισόδους (**in[3:0]**) και την εμφανίζει στην έξοδο **out**. Η επιλογή γίνεται με βάση την τιμή του σήματος εισόδου **sel[1:0]**. Η λειτουργία αυτή φαίνεται στον παρακάτω πίνακα αλήθειας:

sel(1)	sel(0)	out
0	0	in(0)
0	1	in(1)
1	0	in(2)
1	1	in(3)

Στο **notebook του 4ου εργαστηρίου** θα βρείτε μια πιθανή υλοποίηση του **πολυπλέκτη 4-σε-1** με λογικές πράξεις (πύλες). **Υλοποιήστε** με τη σειρά σας τον ίδιο πολυπλέκτη α) με τη δομή when και β) με τη δομή switch. **Ελέγξτε την ορθότητα** των δύο υλοποιήσεών σας, χρησιμοποιώντας τον κώδικα ελέγχου που δίνεται στο notebook.

4. Τελειώνοντας το εργαστήριο

Όταν ολοκληρώσετε την άσκηση, κατεβάστε (**download**) το notebook με τον κώδικά σας. **Φυλάξτε το για τα επόμενα εργαστήρια!**