ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

# Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Μάθημα:

Ψηφιακά Συστήματα HW σε Χαμηλά Επίπεδα Λογικής Ι

Τίτλος Εργασίας:

Υλοποίηση Συστήματος Αποστολέα-Δέκτη σε Verilog με χρήση Πρωτοκόλλου UART και προβολή σε Οθόνη Τεσσάρων LED 7-Τμημάτων

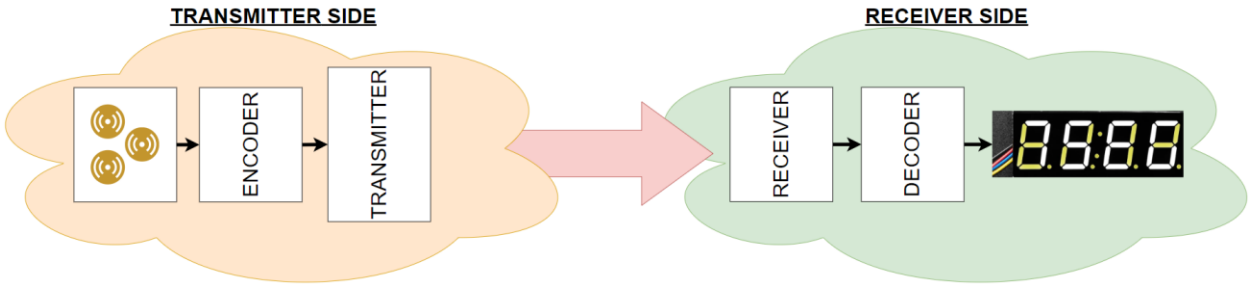
Χειμερινό Εξάμηνο 2022-2023

# Πίνακας Περιεχομένων

## Εισαγωγή

Γενικός στόχος της παρούσας εργασίας είναι η κατανόηση των βασικών στοιχείων της γλώσσας περιγραφής υλικού Verilog και της σχεδίασης ψηφιακών κυκλωμάτων. Η πρακτική εφαρμογή των εννοιών που διδάχτηκαν στη θεωρία θα σας δώσει βαθύτερη κατανόηση των ψηφιακών κυκλωμάτων.

Στην εργασία αυτή θα πρέπει να υλοποιήσετε σε Verilog, ένα ψηφιακό σύστημα αποστολέα-δέκτη, όπως παρουσιάζεται στην Εικόνα 1. Η επικοινωνία θα πραγματοποιηθεί με χρήση του πρωτοκόλλου UART και το μήνυμα που αποσταλεί προαιρετικά θα πρέπει να κωδικοποιηθεί από τον αποστολέα και να αποκωδικοποιηθεί στο δέκτη. Ο δέκτης θα λάβει από το περιβάλλον του τις ενδείξεις κάποιον αισθητήρων και ο δέκτης θα εμφανίζει τις ενδείξεις που λαμβάνει σε μία οθόνη τεσσάρων LED 7-τμημάτων.




*Εικόνα 1*

Η εργασία χωρίζεται σε επιμέρους μέρη, τα οποία πρέπει να δοκιμαστούν το καθένα μεμονωμένα αλλά και συνδυαστικά. Για να αποδείξετε την ορθή λειτουργία θα πρέπει να προσομοιώσετε τη λειτουργία του κάθε τμήματος και τέλος ολόκληρου του συστήματος.

Οι αισθητήρες, που αποτελούν την είσοδο του συστήματος, δεν θα υλοποιηθούν από εσάς. Εσείς θα θεωρήσετε ότι λαμβάνετε τις ενδείξεις τους από το περιβάλλον του αποστολέα. Οι ενδείξεις θα έχουν την ακόλουθη κανονική έκφραση [‘κενό’|‘-’][‘κενό’|1-9] [‘κενό’|0-9] [0-9], π.χ. ‘–120’, ‘ 201’, ‘- 3’. Οι ενδείξεις αυτές προαιρετικά μπορούν να κωδικοποιηθούν και έπειτα να αποσταλούν στον δέκτη με το πρωτόκολλο UART. Το πρωτόκολλο αυτό είναι σειριακό και αποστέλλεται ένα bit τη φορά. Περισσότερες λεπτομέρειες για το πρωτόκολλο UART δίνονται στο Μέρος Β. Ο δέκτης αφού λάβει το μήνυμα θα πρέπει να το προβάλει στην οθόνη τεσσάρων LED και 7-τμημάτων που διαθέτει. Αν το μήνυμα είναι κωδικοποιημένο, ο δέκτης πρέπει να το αποκωδικοποιήσει πριν το προβάλει στην οθόνη.

## Μέρος Α’ – 7-Segment Display Driver

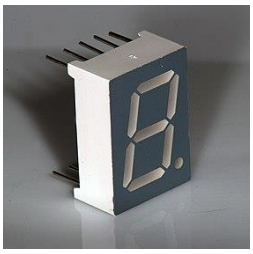
Ο στόχος του Μέρους Α είναι η υλοποίηση ενός οδηγού ενδείξεων 7-τμημάτων τεσσάρων LED για να προβληθεί μία ακολουθία αλφαριθμητικών και ειδικών χαρακτήρων.

### 1 - Οθόνη 7 Τμημάτων

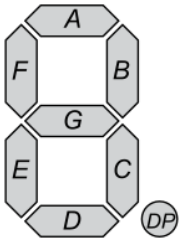
Η οθόνη 7 τμημάτων αποτελεί την απλούστερη μορφή αναπαράστασης ενός αλφαριθμητικού ψηφίου. Το κάθε ψηφίο αποτελείται από τα 7 συν 1 (δεκαδικό σημείο) σήματα A, B, C, D, E, F, G και DP, όπως φαίνεται στην Εικόνα 3. Για να μπορέσουμε να εμφανίσουμε ένα μήνυμα με περισσότερα από ένα

ψηφία, χρησιμοποιούμε περισσότερες οθόνες 7-τμημάτων τη μία δίπλα στην άλλη, όπως φαίνεται στην Εικόνα 5. Για την αποφυγή πολλαπλών σημάτων, οι οθόνες χρησιμοποιούν τα ίδια σήματα A, B, C, D, E, F, G και DP, ωστόσο για να επισημανθεί ποια οθόνη θα προβάλει το ψηφίο την εκάστοτε στιγμή, χρησιμοποιούνται οι άνοδοι AN. Έτσι, για να ανάψει ένα συγκεκριμένο τμήμα LED πρέπει (από το εξής και για το υπόλοιπο της εργασίας θεωρούμε ότι υπάρχουν 4 LED):

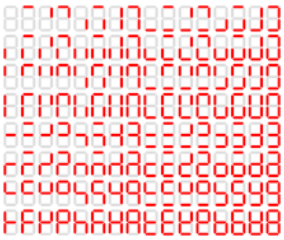
- το σχετικό σήμα ανόδου, AN0-3, που επιλέγει το ψηφίο, να οδηγείται στο μηδέν,
- το σχετικό σήμα τμήματος, A-G/DP, που επιλέγει το τμήμα, να οδηγείται στο μηδέν.



*Εικόνα 2*



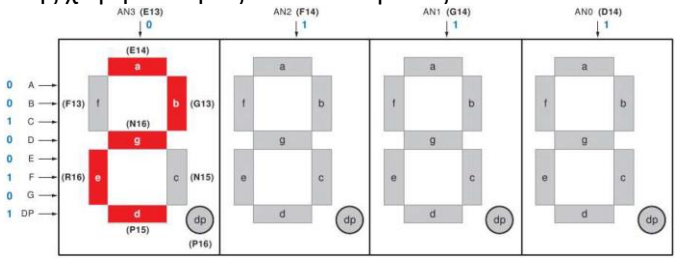

*Εικόνα 3*



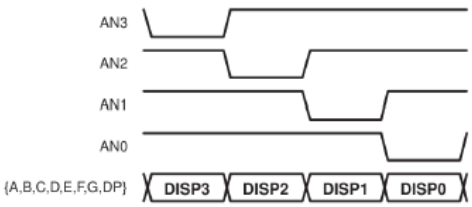
*Εικόνα 4*

Η συχνότητα και η διάρκεια που κάθε οθόνη θα προβάλει έχει να κάνει με τα κατασκευαστικά χαρακτηριστικά της. Στα πλαίσια της εργασίας θα υποθέσουμε ότι η συχνότητα του ρολογιού που χρησιμοποιούμε είναι 50 MHz.

Στην Εικόνα 6 φαίνεται η κατάλληλη πολυπλεξία στον χρόνο που απαιτείται για να εμφανιστούν δεδομένα και στα τέσσερα ψηφία. Οι τέσσερις άνοδοι πρέπει να οδηγούνται εναλλάξ στο μηδέν, ενώ αλλάζουν κατάλληλα τα σήματα των 7 συν 1 ενδείξεων. Η διαδικασία πρέπει να επαναλαμβάνεται διαρκώς. Παρότι τα LED ουσιαστικά αναβοσβήνουν, λόγω της γρήγορης εναλλαγής και της διατήρησης της φωτεινότητας τους, στο ανθρώπινο μάτι φαντάζουν σταθερά. Όταν μια άνοδος, ANx, επιστρέφει στο ένα, οι τιμές των ενδείξεων δεν διατηρούνται, και καθώς το κύκλωμα των LED εκφορτίζεται, τα φωτισμένα τμήματα σβήνουν. Κατά την εναλλάξ διαδικασία σάρωσης, η άνοδος και τα σήματα των τμημάτων πρέπει να μένουν σταθερά για ένα σχετικά μεγάλο χρονικό διάστημα, ως προς το ρολόι της πλακέτας, λόγω της μεγάλης χωρητικότητας του κυκλώματος των LED.




*Εικόνα 5*




*Εικόνα 6*

Ως χρόνος εναλλαγής των ανόδων προτείνονται τα 0.32 μsec, τα οποία είναι πολλαπλάσια του ρολογιού των 50 MHz.

## 2 - Υλοποίηση Αποκωδικοποιητή 7-τμημάτων

Για την αποκωδικοποίηση των 7 συν 1 τμημάτων υλοποιήστε ένα συνδυαστικό αποκωδικοποιητή, ο οποίος θα έχει ως είσοδο το ψηφίο ή χαρακτήρα που θέλετε να εμφανίσετε, και θα παράγει ως έξοδο τις τιμές οδήγησης των 8 τμημάτων. Για παράδειγμα, για την αναπαράσταση 16 ψηφίων απαιτείται οι χρήση αριθμών 4-bit, ενώ τα 7 τμήματα απαιτούν 7 ξεχωριστά bit (το δεκαδικό ψηφίο, DP, μπορεί να μένει σβηστό, οδηγώντας το μόνιμα στο 1). Έτσι, μια μορφή του αποκωδικοποιητή 7-τμημάτων είναι η εξής:

```
module LEDdecoder(char, LED);
input [3:0] char;
output [6:0] LED;
…
endmodule
```

Συμπληρώστε την υλοποίηση του αποκωδικοποιητή και ελέγξτε λειτουργικά, μέσω προσομοίωσης, ότι οι τιμές που προκύπτουν στα ψηφία του LED είναι οι κατάλληλες, και έχουν και την σωστή πολικότητα (0 = αναμμένο και 1 = σβηστό).

Οι χαρακτήρες που υποστηρίζονται είναι οι παρακάτω:
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -, F, κενό (το κενό σημαίνει ότι όλα τα τμήματα είναι σβηστά).

## 3 - Οδήγηση Τεσσάρων Ψηφίων

Όπως εξηγήθηκε νωρίτερα, για τη χρήση των τεσσάρων ψηφίων απαιτείται η εναλλάξ οδήγηση του κάθε ψηφίου με προτεινόμενη ελάχιστη καθυστέρηση φόρτισης τα 0.32 μsec (μπορείτε να το ορίσετε και περισσότερο). Για να μετρηθεί ο χρόνος αυτός, θα χρησιμοποιηθεί μετρητής ο οποίος, κάθε δεκαέξι κύκλους του ρολογιού των 50 MHz, δηλαδή περιόδου 20 ns, θα σηματοδοτεί την αλλαγή των σημάτων των ανόδων και των τμημάτων.

### *3.1 - Οδήγηση των Ανόδων*

Έχοντας διαιρέσει το ρολόι, θα πρέπει να οδηγήσετε τα σήματα των ανόδων όπως στην Εικόνα 6. Τα σήματα των ανόδων δεν θα πρέπει να επικαλύπτονται όσο βρίσκονται στο μηδέν, αλλιώς θα αλλοιωθούν οι ενδείξεις. Έτσι, απαιτείται και ένα περιθώριο ασφαλείας από την επιστροφή ενός σήματος ανόδου στο λογικό ένα, μέχρι την πτώση του επόμενου.

Η μη-επικαλυπτόμενη οδήγηση των ανόδων μπορεί να επιτευχθεί με τη χρήση ενός περιστροφικού μετρητή, όπου κάποιες τιμές του μετρητή θα ενεργοποιούν (στο μηδέν) εναλλάξ τις ανόδους, ενώ οι ενδιάμεσες θα τις επιστρέφουν στο λογικό ένα. Παραδείγματος χάριν, ένας 4-bit μετρητής θα μπορούσε να ενεργοποιεί τις ανόδους όπως φαίνεται στον παρακάτω πίνακα, αρχικοποιημένος στην τιμή 1111.

| Τιμή μετρητή | AN3 | AN2 | AN1 | AN0 |
|---|---|---|---|---|
| 1111 | 1 | 1 | 1 | 1 |
| 1110 | 0 | 1 | 1 | 1 |
| 1101 | 1 | 1 | 1 | 1 |
| 1100 | 1 | 1 | 1 | 1 |
| 1011 | 1 | 1 | 1 | 1 |
| 1010 | 1 | 0 | 1 | 1 |
| 1001 | 1 | 1 | 1 | 1 |
| 1000 | 1 | 1 | 1 | 1 |
| 0111 | 1 | 1 | 1 | 1 |
| 0110 | 1 | 1 | 0 | 1 |
| 0101 | 1 | 1 | 1 | 1 |
| 0100 | 1 | 1 | 1 | 1 |
| 0011 | 1 | 1 | 1 | 1 |
| 0010 | 1 | 1 | 1 | 0 |
| 0001 | 1 | 1 | 1 | 1 |
| 0000 | 1 | 1 | 1 | 1 |

*Εικόνα 7*

Πρακτικά, ο μετρητής χρησιμοποιείται εμμέσως ως ΜΠΚ (Μηχανή Πεπερασμένων Καταστάσεων), όπου η μέτρηση του αντιστοιχεί πρακτικά σε μια κατάσταση. Επιβεβαιώστε ποσοτικά ότι οι κυματομορφές των ανόδων που παράγετε έχουν την ίδια συμπεριφορά με αυτές της Εικόνας 7 και έπειτα ότι ο χρόνος που κάθε άνοδος είναι ενεργή, είναι ίση με 0.32μsec.

Για την οδήγηση των σημάτων των ενδείξεων (δηλαδή των χαρακτήρων που θέλετε να προβάλετε στην κάθε οθόνη) χρησιμοποιείστε πάλι τιμές του μετρητή, έτσι ώστε να υπάρχει χρόνος προετοιμασίας (πρόθεσης - setup) μεταξύ των ανόδων και των δεδομένων των ενδείξεων. Δώστε περιθώριο αλλαγής τουλάχιστον ενός βήματος. Λ.χ. για την οδήγηση της AN3 στο μηδέν στο 1110, αλλάξτε τα δεδομένα στη μέτρηση 0000, για την οδήγηση της AN2 στο μηδέν στο 1010, αλλάξτε τα δεδομένα στο 1100, κοκ. Ο λόγος που απαιτείται και αυτό το περιθώριο είναι η μεγάλη χωρητικότητα των ενδείξεων.

### *3.2 - Αρχικοποίηση*

Για την αρχικοποίηση του κυκλώματος, χρησιμοποιείστε ένα ασύγχρονο σήμα αρχικοποίησης reset σε κάθε ακολουθιακό τμήμα always.

### *3.3 - Λειτουργικός Έλεγχος*

Όταν ολοκληρώσετε την υλοποίηση του Μέρους Α, πραγματοποιείστε έλεγχο λειτουργίας, με προσομοίωση σε επίπεδο συμπεριφοράς, χρησιμοποιώντας ένα testbench που οδηγεί κατάλληλα το ρολόι. Καλείστε να εμφανίσετε μία σειρά από τέσσερις διαδοχικές ενδείξεις από τους αισθητήρες, π.χ. '-194', ' 10', '- 32', 'FFFF'.

Σύνοψη Μέρους Α

1) Υλοποιείστε ένα κωδικοποιητή 4 σε 7 bit, ώστε να κωδικοποιήσετε τον κάθε υποστηριζόμενο χαρακτήρα στα 7 τμήματα του κάθε LED.
2) Υλοποιείστε το FSM οδήγησης των τεσσάρων ανόδων.
3) Δείξτε την ορθή λειτουργία του οδηγού τεσσάρων ενδείξεων 7 τμημάτων κάνοντας προσομοίωση με χρήση κυκλώματος ελέγχου (testbench).

Παραδοτέα

1) Σχηματικό διάγραμμα του κυκλώματος, το οποίο θα εξάγεται από το εργαλείο προσομοίωσης.
2) Σχηματικό των υλοποιημένων FSM, το οποίο θα σχεδιάσετε εσείς, κατά προτίμηση με κάποιο πρόγραμμα (MS Word, MS PowerPoint, Photoshop, draw.io, κτλ.) και όχι στο χαρτί.
3) Κυματομορφές που αποδεικνύουν την ορθή λειτουργία του κυκλώματος, με επισημάνσεις στους ζητούμενους χρονισμούς και σχέσεις μεταξύ των σημάτων.

4) Σχολιασμός της λύσης και περιγραφή των προβλημάτων που παρουσιάστηκαν, καθώς και τις λύσεις που βρήκατε.
5) Κώδικα Verilog και testbench.

## Μέρος Β' – Υλοποίηση Γενικού Ασύγχρονου Δέκτη Αποστολέα (UART)

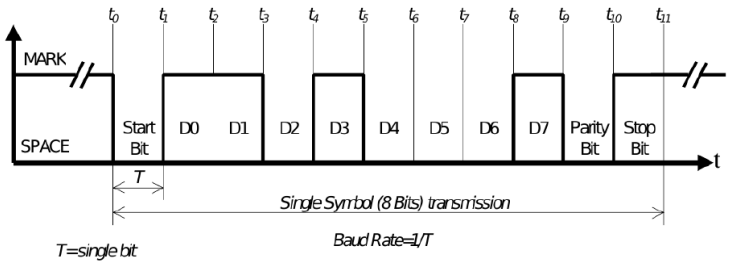
Ο στόχος του Μέρους Β της εργασίας είναι η υλοποίηση ενός συστήματος σειριακής επικοινωνίας, το οποίο θα χρησιμοποιεί το πρωτόκολλο UART (Universal Asynchronous Receiver Transmitter - Γενικού Ασύγχρονου Δέκτη Αποστολέα). Το σύστημα θα αποτελείται από έναν UART Αποστολέα και έναν UART Δέκτη, οι οποίοι μεταφέρουν δεδομένα στη μια κατεύθυνση, από τον Αποστολέα στον Δέκτη, μέσω μιας σειριακής σύνδεσης ενός σήματος.

Το UART που θα υλοποιηθεί, θα χρησιμοποιηθεί για τη σειριακή μεταφορά τουλάχιστον μιας αλληλουχίας τεσσάρων διαφορετικών συμβόλων συνήθως των 8-bit, από τον Αποστολέα στον Δέκτη. Για τα πλαίσια της εργασίας οι ενδείξεις θα αποστέλλονται από τον Αποστολέα στον Δέκτη σε 2 πακέτα των 8-bit. Πιο συγκεκριμένα αν για παράδειγμα η ένδειξη του αισθητήρα είναι '-888' τότε τα δύο πακέτα των 8-bit θα είναι '-8' και '88'. Ανάλογα με την κωδικοποίηση που κάνατε τα δύο πακέτα θα πρέπει να ερμηνευτούν σε συμβολοσειρές από 0 και 1. Π.χ. αν 8 -> 1000 και '-' -> 1010 τότε το πρώτο πακέτο των 8-bit θα είναι το 10101000 ('-8') και το δεύτερο το 10001000 ('88').

### 1 - Το UART

Το UART (Universal Asynchronous Receiver Transmitter - Γενικός Ασύγχρονος Δέκτης και Αποστολέας) είναι ένα σειριακό, ασύγχρονο πρωτόκολλο επικοινωνίας, και σχετικό κύκλωμα που το υλοποιεί, το οποίο επιτρέπει τη μεταφορά δεδομένων μεταξύ δυο (ή περισσοτέρων γενικά) συσκευών, οι οποίες μπορεί να έχουν ανεξάρτητα και ασυσχέτιστα ρολόγια. Το UART είναι ευρύτατα διαδεδομένο (RS232), λόγω απλότητας στην υλοποίηση, πρακτική ευκολία στη χρήση και της γενικής του εφαρμογής. Το πρωτόκολλο μεταφοράς USB, είναι στην ουσία απόγονος του UART.

Η ασύγχρονη επικοινωνία του UART συντελείται μέσω μιας ενσύρματης σύνδεσης ενός bit, μεταξύ του Αποστολέα (TxD), ο οποίος την οδηγεί, και του Δέκτη (RxD), ο οποίος τη δειγματοληπτεί και εξετάζει. Το δεδομένο προς επικοινωνία, συνήθως ονομάζεται σύμβολο, και για να σταλεί σειριακά πρέπει να μετατραπεί στα συναποτελούντα του ψηφία, τα οποία θα σταλούν ένα προς ένα, από το μικρότερο (LSB) στο μεγαλύτερο (MSB). Στην Εικόνα 8, φαίνεται αναλυτικά το χρονοδιάγραμμα επικοινωνίας ενός συμβόλου. Μια και δεν υπάρχει κάποια εγγύηση ως προς τη σχετική συχνότητα και φάση των ρολογιών του Αποστολέα και Δέκτη, στην ασύγχρονη επικοινωνία τα δεδομένα έπονται του bit Εκκίνησης (Start bit). Αυτό σηματοδοτεί για τον Δέκτη την έναρξη της επικοινωνίας. Επίσης, το πέρας του συμβόλου και η λήξη της επικοινωνίας σηματοδοτείται από το bit Παύσης (Stop bit). Έτσι, όσο το κανάλι επικοινωνίας 1-bit του UART μένει αδρανές, και δεν χρησιμοποιείται για μεταφορά δεδομένων, θα πρέπει να βρίσκεται στο λογικό 1, την τιμή δηλαδή του Stop bit. Για επαλήθευση της ορθής επικοινωνίας των δεδομένων, αυτά συνοδεύονται επίσης από ένα bit Ισοτιμίας (Parity), το οποίο καταδεικνύει αν το σύμβολο που μεταβιβάστηκε εμπεριέχει ζυγό ή μονό αριθμό άσσων.




*Εικόνα 8*

Για τον ορθό ρυθμό δειγματοληψίας, έτσι ώστε να μην χάνονται ή να πολλαπλασιάζονται δεδομένα στο πρωτόκολλο UART, ο Αποστολέας και ο Δέκτης προσυμφωνούν την ταχύτητα της μεταξύ τους επικοινωνίας σε μονάδες Baud (bits/sec). Η προσυμφωνία αυτή δεν είναι μέρος του πρωτοκόλλου επικοινωνίας και γίνεται σε υψηλότερο επίπεδο. Στον παρακάτω πίνακα παρουσιάζονται οι ταχύτητες επικοινωνίας του UART, και η σχετική κωδικοποίηση τους από το 3-bit σήμα BAUD_SEL, το οποίο και τις επιλέγει.

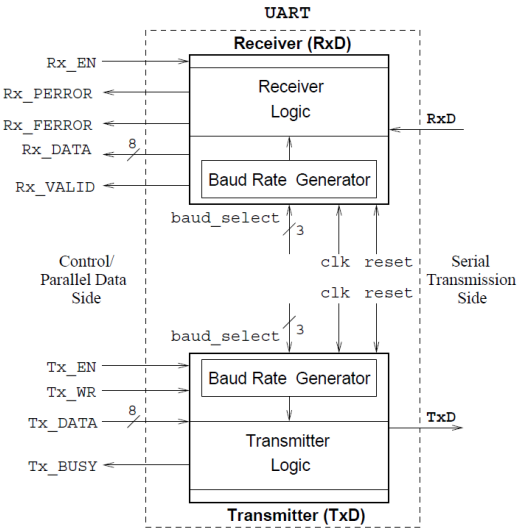
| BAUD_SEL | Baud Rate |
|---|---|
| 000 | 300 bits/sec |
| 001 | 1200 bits/sec |
| 010 | 4800 bits/sec |
| 011 | 9600 bits/sec |
| 100 | 19200 bits/sec |
| 101 | 38400 bits/sec |
| 110 | 57600 bits/sec |
| 111 | 115200 bits/sec |

*Εικόνα 9*

Η περίοδος του κάθε ψηφίου στην Εικόνα 9 αναλογεί σε T = 1/BaudRate. Η λειτουργία και δειγματοληψία του UART γίνεται σε πολλαπλάσιο της συχνότητας του Baud Rate. Η πιο συνήθης συχνότητα δειγματοληψίας είναι ×16, δηλαδή το κάθε μεταδιδόμενο ψηφίο του Αποστολέα, εξετάζεται με 16πλάσιο ρυθμό από τον Δέκτη, για την προσυμφωνημένη ταχύτητα. Δηλαδή, ο Δέκτης πρακτικά λειτουργεί σε ταχύτητα BaudRate × 16. Επιπλέον, ο Δέκτης, αφού συγχρονιστεί με το Start bit, και με την κατάλληλη καθυστέρηση από τη δειγματοληψία του τελευταίου, θα πρέπει να ευθυγραμμίσει τη δειγματοληψία του στο κέντρο της περιόδου, 1/BaudRate.

Στην Εικόνα 10, παρουσιάζεται αναλυτικά η κυκλωματική δομή ενός ζεύγους Αποστολέα, Δέκτη. Και τα δυο κυκλώματα έχουν δυο πλευρές, την πλευρά που επιτελείται η σειριακή επικοινωνία, δηλ. τα σήματα RxD και TxD, τα οποία συνδέονται μεταξύ τους για να δημιουργήσουν το σειριακό κανάλι, και την πλευρά ελέγχου/δεδομένων, στην οποία το οποιοδήποτε σύστημα επικοινωνεί με τις μονάδες του UART, και αποστέλλει ή λαμβάνει τα (παράλληλα) δεδομένα των συμβόλων. Η κάθε μονάδα ενεργοποιείται για επικοινωνία με το σχετικό σήμα, Rx_EN, Tx_EN. Όπως φαίνεται στην Εικόνα 10, η κάθε μονάδα περιλαμβάνει έναν προγραμματιζόμενο ελεγκτή Baud Rate, ο οποίος θέτει τον ρυθμό μετάδοσης πριν την έναρξη, και όχι κατά τη διάρκεια, της επικοινωνίας. Ο ελεγκτής αυτός ορίζει σε ποια ταχύτητα, δηλαδή ποιους χρόνους και τελικά κύκλους ρολογιού, οι μονάδες είναι ενεργές. Όπως ειπώθηκε νωρίτερα, ο Αποστολέας υπολογίζει και στέλνει το ψηφίο Parity, ενώ ο Δέκτης θα πρέπει να επαληθεύσει ότι η ισοτιμία του συμβόλου είναι σωστή. Σε περίπτωση που διαπιστωθεί σφάλμα στην ισοτιμία, ο Δέκτης θα πρέπει να πληροφορήσει την πλευρά του συστήματος ότι τα δεδομένα που παρελήφθησαν δεν είναι σωστά, για να αγνοηθούν σε υψηλότερο επίπεδο. Αυτό επιτυγχάνεται με το σήμα Rx_PERROR. Επιπλέον, στην περίπτωση που ο Δέκτης δε δειγματοληπτήσει το Stop bit στον χρόνο που το περιμένει ή δεν μπορεί να ευθυγραμμιστεί με τη μέση του Start bit, οι οποίες δυο περιπτώσεις σημαίνουν ότι τα

δεδομένα δεν έχουν πλαισιωθεί σωστά (λ.χ. το Baud Rate μπορεί να είναι διαφορετικό, ή ο Δέκτης να ενεργοποιηθεί καθυστερημένα), θα πληροφορήσει το σύστημα για το σφάλμα με το σήμα Rx_FERROR.




*Εικόνα 10*

Επισημαίνεται ότι το πρωτόκολλο UART δεν περιλαμβάνει έλεγχο ροής μεταξύ Δέκτη-Αποστολέα, δηλαδή ο Δέκτης δεν μπορεί να σταματήσει τον Αποστολέα, έτσι και οι δυο θα πρέπει να είναι έτοιμοι για διαρκή επικοινωνία. Στην πλευρά του συστήματος, όπου και οι ταχύτητες μπορεί να είναι πολύ μεγαλύτερες, είναι απαραίτητο ένα απλό πρωτόκολλο επικοινωνίας που να καταδεικνύει: (1) ότι ο Αποστολέας βρίσκεται σε διαδικασία μετάδοσης και δεν μπορεί να λάβει το επόμενο σύμβολο, και (2) ότι ο Δέκτης έλαβε σύμβολο που πρέπει να αναγνωστεί. Αυτές οι δυο απαιτήσεις επιτυγχάνονται με τα σήματα Tx_BUSY και Rx_VALID αντίστοιχα.

## 2 - Ελεγκτής Baud Rate

Ο Ελεγκτής θα χρησιμοποιηθεί εσωτερικά στα κυκλώματα Αποστολέα και Δέκτη. Ο στόχος του είναι να παρέχει το κατάλληλο σήμα δειγματοληψίας, ανάλογα με τον επιλεγμένο Baud Rate. Το σήμα δειγματοληψίας (στον Δέκτη) σας προτείνεται να είναι θετικά ενεργό και να μένει ενεργό για ένα κύκλο, ενώ όπως επεξηγήθηκε παραπάνω θα πρέπει να έχει συχνότητα 16× Baud Rate. Η προτεινόμενη μορφή του Ελεγκτή Baud Rate φαίνεται παρακάτω:

```
module baud_controller(reset, clk, baud_select, sable_ENABLE);
input clk, reset;
input [2:0] baud_select;
output sample_ENABLE;
…
endmodule
```

Για τον Ελεγκτή Baud Rate, σας προτείνεται να χρησιμοποιήσετε το ρολόι των 50 MHz. Για την παραγωγή του σήματος δειγματοληψίας, σας προτείνεται να χρησιμοποιήσετε έναν μετρητή κύκλων ρολογιού των 50 MHz, όπου για κάθε απαιτούμενο Baud Rate, θα ορίσετε μια μέγιστη τιμή του μετρητή,

η οποία θα μετράει, με το μικρότερο δυνατό ποσοστό λάθους, την ποσότητα Tsc = 1/(16×BaudRate) , η οποία αντιστοιχεί στην περίοδο δειγματοληψίας, σε αριθμό κύκλων.

Έτσι, ανάλογα με το απαιτούμενο Baud Rate, όταν ο μετρητής θα φτάνει τη σχετική μέγιστη τιμή, το σήμα sample_ENABLE θα γίνεται 1, και ο μετρητής θα επιστρέφει στο μηδέν. Με αυτόν τον τρόπο, κάθε περίοδο Tsc = 1/(16×BaudRate) , το σήμα sample_ENABLE θα γίνεται 1 για έναν κύκλο. Όταν υπολογίσετε τις απαιτούμενες τιμές του μετρητή, υπολογίστε και το σχετικό σφάλμα στο Baud Rate. Το τελευταίο θα πρέπει να το συμπεριλάβετε και να το σχολιάσετε στην εργαστηριακή αναφορά.

## 3 - Υλοποίηση UART Αποστολέα

Ο UART Αποστολέας θα πρέπει, όπως εξηγήθηκε νωρίτερα, να είναι ενεργοποιήσιμος, να επικοινωνεί με το σύστημα για να παραλαμβάνει το σύμβολο προς μεταφορά, αλλά και να παράγει μια ένδειξη διαθεσιμότητας ή όχι, ανάλογα με το αν εκτελεί μεταφορά ή περιμένει επόμενο δεδομένο. Το σήμα Tx_EN αποτελεί το σήμα ενεργοποίησης του Αποστολέα. Το συγκεκριμένο σήμα θα πρέπει να μένει ενεργό, από το σύστημα (ή το πλαίσιο δοκιμής), όσο ο Αποστολέας πρέπει να είναι ενεργός ή όσο δεν έχει ακόμα ολοκληρώσει την τρέχουσα μεταβίβαση. Το Baud Rate θα πρέπει να έχει τεθεί πριν ενεργοποιηθεί ο Αποστολέας. Για την παραλαβή δεδομένων προς μεταβίβαση, σας προτείνετε να χρησιμοποιήσετε το σήμα Tx_WR, όπου το τελευταίο θα γίνεται 1 για ένα κύκλο, ενώ τα δεδομένα του συμβόλου προς μεταφορά θα βρίσκονται στο Tx_DATA[7:0]. Το σήμα Tx_BUSY, με κατεύθυνση από τον Αποστολέα προς το σύστημα θα σηματοδοτεί, όσο μένει 1, ότι ο Αποστολέας βρίσκεται σε κατάσταση μεταβίβασης, έτσι (1) δεν επιτρέπεται να πέσει το Tx_EN και (2) δεν επιτρέπεται να του μεταβιβαστούν νέα δεδομένα μέσω του ζεύγους Tx_WR, Tx_DATA[7:0].

### *3.1 - Δομή Μονάδας*

Στην υλοποίηση του UART Αποστολέα, θα εμπεριέχεται ο Ελεγκτής Baud Rate που υλοποιήσατε στο πρώτο μέρος. Σας προτείνεται η εξής δομή:

```
module uart_transmitter(reset, clk, Tx_DATA, baud_select, Tx_WR, TX_EN, TxD, TX_BUSY);
input clk, reset;
input [7:0] Tx_DATA;
input [2:0] baud_select;
input TX_EN;
input Tx_WR;
output TxD;
output Tx_BUSY;
…
   baud_controller baud_controller_tx_instance(reset, clk, baud_select, Tx_sample_ENABLE);
…
endmodule
```

### *4.2 - Δομή Αποστολέα*

Ο Αποστολέας μπορεί να βρίσκεται σε μια από δύο καταστάσεις, είτε θα είναι ανενεργός, είτε θα βρίσκεται σε διαδικασία μετάδοσης δεδομένων. Μετά την παραλαβή συμβόλου από το σύστημα, όπως επεξηγήθηκε παραπάνω, και εφόσον είναι ενεργός, ο Αποστολέας θα πρέπει να μπει στην κατάσταση μετάδοσης, και να ξεκινήσει την αποστολή στη σειριακή σύνδεση TxD, στέλνοντας το Start bit. Οι κύκλοι που το σήμα Tx_sample_ENABLE είναι 1, σηματοδοτούν ενεργούς κύκλους για τον Αποστολέα. Έτσι, η μετάδοση δεδομένων, συμπεριλαμβανομένου και του Start bit, θα γίνεται κάθε 16 κύκλους του Tx_sample_ENABLE, το οποίο και εξαρτάται από το επιλεγμένο Baud Rate. Αλλάζοντας το τελευταίο, θα αλλάξουν και οι ενεργοί κύκλοι μετάδοσης του Αποστολέα, οπότε το σειριακό τρένο των δεδομένων είτε θα πλαταίνει, είτε θα συρρικνώνεται.

Για την υλοποίηση του Αποστολέα, σας προτείνεται να χρησιμοποιήσετε ένα μετρητή δεδομένων, ο οποίος κάθε 16 παλμούς του Tx_sample_ENABLE, θα οδηγεί κατάλληλα το TxD. Συνολικά, θα πρέπει μεταδώσετε σειριακά 11 bit (8 δεδομένων, 1 αρχής, 1 τέλους και 1 ισοτιμίας). Ολοκληρώνοντας τη διαδικασία, ο μετρητής δεδομένων μπορεί να επιστρέψει στο μηδέν, και ο Αποστολέας στην ανενεργή κατάσταση.

## 4 - Υλοποίηση UART Δέκτη

Ο Δέκτης, κατά αναλογία με τον Αποστολέα θα πρέπει να είναι ενεργοποιήσιμος με τον ίδιο τρόπο, και επιπλέον να επιδεικνύει στο σύστημα ότι υπάρχει διαθέσιμο σύμβολο προς ανάγνωση. Το σήμα Rx_EN αποτελεί το σήμα ενεργοποίησης του Δέκτη, με ανάλογη συμπεριφορά με του Αποστολέα. Ομοίως λειτουργεί η επιλογή Baud Rate. Κατά την ολοκλήρωση της λήψης από τον Δέκτη, αν διαπιστωθεί σφάλμα στην ισοτιμία ή στη σειριακή πλαισίωση των δεδομένων (όπως εξηγήθηκε νωρίτερα), θα πρέπει να γίνουν 1 τα αντίστοιχα σήματα λάθους, Rx_PERROR και Rx_FERROR.
Εναλλακτικά, αν δεν υπάρχει σφάλμα, το σύμβολο που θα παραληφθεί θα πρέπει να εμφανιστεί στο Rx_DATA[7:0], και για 1 τουλάχιστον κύκλο το σήμα Rx_VALID θα πρέπει να τεθεί στη τιμή 1, το οποίο σηματοδοτεί ότι τα δεδομένα είναι έγκυρα και μπορεί να διαβαστούν από το σύστημα. Σας προτείνεται, για ευκολία στην επικοινωνία με το σύστημα, το τελευταίο ζεύγος σημάτων να μένει έγκυρο, έως ότου ξεκινήσει νέα επικοινωνία από τον Αποστολέα, δηλαδή έως ότου δεν έχει παραληφθεί νέο Start bit. Ομοίως και για τα δυο σήματα σφάλματος.

### *4.1 - Λειτουργία Δέκτη*

Ο Δέκτης επίσης έχει δυο καταστάσεις, την ανενεργή, όπου περιμένει δεδομένα, και την κατάσταση λήψης, όπου έχει ξεκινήσει και είναι ενεργή η σειριακή παραλαβή των bit. Ο Δέκτης μπαίνει στην κατάσταση λήψης με την ανίχνευση του Start bit. Μια και ο Δέκτης είναι ασύγχρονος ως προς τον Αποστολέα, θα πρέπει πριν την οποιαδήποτε δειγματοληψία, να συγχρονιστεί η ασύγχρονη σειριακή είσοδος.

Είναι σημαντικό ο Δέκτης να δειγματοληπτεί σωστά τα σειριακά δεδομένα, έτσι, για μέγιστη αξιοπιστία, η δειγματοληψία του Δέκτη θα πρέπει να πραγματοποιείται στην προβλεπόμενη μέση του επομένου bit. Η συγκεκριμένη πρόβλεψη, και η κατάλληλη ευθυγράμμιση της δειγματοληψίας, γίνεται σε σχέση με το Start bit, μετρώντας τον κατάλληλο αριθμό ενεργών κύκλων, σύμφωνα με το Baud Rate. Στο Δέκτη, οι ενεργοί κύκλοι σηματοδοτούνται από το σχετικό σήμα Rx_sample_ENABLE.

Για την υλοποίηση του Δέκτη, σας προτείνεται να χρησιμοποιήσετε αναλόγως ένα μετρητή δεδομένων, όπου ανάλογα με το bit που αναμένεται, θα γίνονται οι κατάλληλες ενέργειες. Τα bit δεδομένων θα πρέπει να ολισθαίνουν ένα προς ένα σε έναν καταχωρητή, έτσι ώστε να παραδοθούν μετά παράλληλα στο σύστημα. Το πέρας της επικοινωνίας σηματοδοτείται από την δειγματοληψία του Stop bit στον κατάλληλο κύκλο λήψης. Τότε, θα πρέπει να ενεργοποιείται η λειτουργία των Rx_VALID, Rx_DATA[7:0] , όπως εξηγήθηκε παραπάνω.

```
module uart_receiver(reset, clk, Rx_DATA, baud_select, RX_EN, RxD, Rx_FERROR, Rx_PERROR, Rx_VALID);
input clk, reset;
input [2:0] baud_select;
input RX_EN;
output RxD;
output [7:0] Rx_DATA;
output Rx_FERROR; // Framing Error //
output Rx_PERROR; // Parity Error //
output Rx_VALID;  // Rx_DATA is Valid //
…
   baud_controller baud_controller_rx_instance(reset, clk, baud_select, Rx_sample_ENABLE);
…
endmodule
```

## 5 - Σύστημα UART Αποστολέα Δέκτη για Σειριακή Μεταφορά Δεδομένων

Έχοντας υλοποιήσει τον Αποστολέα και Δέκτη, ο στόχος τώρα είναι να συνενωθούν σε ένα πλήρες κανάλι UART. Υλοποιήστε το ζεύγος Αποστολέα-Δέκτη, και το κατάλληλο πλαίσιο δοκιμής για να ελέγξετε ότι η μεταφορά των τεσσάρων διαδοχικών λέξεων γίνεται σωστά, και ότι τα απεσταλμένα δεδομένα παραλαμβάνονται σωστά στην πλευρά του Δέκτη. Η επικοινωνία του πλαισίου δοκιμής με τους Αποστολέα και Δέκτη, θα πρέπει να βασίζεται αποκλειστικά στο πρωτόκολλο των σχετικών σημάτων, λ.χ. Rx_VALID, Tx_BUSY, και όχι σε μέτρηση κύκλων ή απόλυτες καθυστερήσεις.

Σύνοψη Μέρους Β

1) Υλοποιείστε ένα Ελεγκτή Baud Rate.
2) Υλοποιείστε το FSM του Αποστολέα, χρησιμοποιώντας τον Ελεγκτή Baud Rate.
3) Δείξτε την ορθή λειτουργία του Αποστολέα κάνοντας προσομοίωση με χρήση κυκλώματος ελέγχου (testbench).
4) Υλοποιείστε το FSM του Δέκτη, χρησιμοποιώντας τον Ελεγκτή Baud Rate.
5) Δείξτε την ορθή λειτουργία του Δέκτη κάνοντας προσομοίωση με χρήση κυκλώματος ελέγχου (testbench).
    a. Δείχνοντας την λειτουργία του
    b. Αλλά και το πως εντοπίζει τα δύο σφάλματα που αναφέρθηκαν παραπάνω. Τα σφάλματα μπορείτε να τα εισάγεται εσείς μέσω του testbench.
6) Συνενώστε τον Αποστολέα και τον Δέκτη και δείξτε την ορθή επικοινωνία μεταξύ τους προσομοίωση με χρήση κυκλώματος ελέγχου (testbench).
    a. Για διαφορετικά Baud Rates

Παραδοτέα

1) Σχηματικό διάγραμμα του κυκλώματος, το οποίο θα εξάγεται από το εργαλείο προσομοίωσης.
2) Σχηματικό των υλοποιημένων FSM.
3) Κυματομορφές που αποδεικνύουν την ορθή λειτουργία του κυκλώματος, με επισημάνσεις στους ζητούμενους χρονισμούς και σχέσεις μεταξύ των σημάτων.
4) Σχολιασμός της λύσης και περιγραφή των προβλημάτων που παρουσιάστηκαν, καθώς και τις λύσεις που βρήκατε.
5) Κώδικα Verilog και testbench

## Μέρος Γ' – Σύστημα Δέκτη με προβολή του μηνύματος στην οθόνη τεσσάρων LED 7-τμημάτων

Το μέρος αυτό περιλαμβάνει τη συνένωση του Μέρους Α και του Β. Θα πρέπει να συνδεθεί κατάλληλα η έξοδος του Δέκτη με την είσοδο την οθόνης τεσσάρων LED 7-τμημάτων. Με τον τρόπο αυτό οι ενδείξεις που συλλέχθηκαν από τον Αποστολέα, θα προβληθούν από το Δέκτη.

Υπενθυμίζουμε ότι ο Αποστολέας έχει διαχωρίσει την κάθε ένδειξη σε δύο πακέτα των 8-bit, οπότε ο Δέκτης θα πρέπει να παραλάβει τα 2 πακέτα, και αφού αποθηκεύσει τα 16 bit να τα προβάλει στην οθόνη. Στην περίπτωση που ο Δέκτης εντοπίσει κάποιο σφάλμα (FERROR ή PERROR) θα πρέπει να εμφανίσει στην οθόνη το μήνυμα σφάλματος 'FFFF'. Η προβολή της κάθε ένδειξης θα προβάλλεται στην οθόνη μέχρι να προβληθεί η επόμενη ένδειξη ή το μήνυμα σφάλματος.

Σύνοψη Μέρους Γ

1) Υλοποιείστε τη συνένωση του Δέκτη με την οθόνη.
2) Υλοποιείστε, αν είναι απαραίτητο, το FSM για την προβολή του μηνύματος στην οθόνη.
3) Δείξτε την ορθή λειτουργία της σύνδεσης του Δέκτη με την οθόνη κάνοντας προσομοίωση με χρήση κυκλώματος ελέγχου (testbench).

   a. Δείχνοντας τη λειτουργία του
   b. Αλλά και το πως προβάλλονται τα δύο σφάλματα που αναφέρθηκαν παραπάνω. Τα σφάλματα μπορείτε να τα εισάγετε εσείς τροποποιώντας το testbench που κατασκευάσατε για τη Δέκτη στον προηγούμενο μέρος, επεκτείνοντάς το για να περιλαμβάνει και την υλοποίηση της οθόνης.
4) Συνενώστε τον Αποστολέα και τον Δέκτη και δείξτε την ορθή επικοινωνία μεταξύ τους
   a. Για διαφορετικά Baud Rates

Παραδοτέα

1) Σχηματικό διάγραμμα του κυκλώματος, το οποίο θα εξάγεται από το εργαλείο προσομοίωσης.
2) Σχηματικό των υλοποιημένων FSM, το οποίο θα σχεδιάσετε εσείς, κατά προτίμηση με κάποιο πρόγραμμα (MS Word, MS PowerPoint, Photoshop, draw.io, κτλ.) και όχι στο χαρτί.
3) Κυματομορφές που αποδεικνύουν την ορθή λειτουργία του κυκλώματος, με επισημάνσεις στους ζητούμενους χρονισμούς και σχέσεις μεταξύ των σημάτων.
4) Σχολιασμός της λύσης και περιγραφή των προβλημάτων που παρουσιάστηκαν, καθώς και τις λύσεις που βρήκατε.
5) Κώδικα Verilog και testbench

## Προαιρετικό Μέρος Δ – Υλοποίηση Κωδικοποιητή-Αποκωδικοποιητή (+1.5 στον τελικό βαθμό στην περίπτωση που είναι λειτουργικός)

Το μέρος αυτό είναι προαιρετικό και προσθέτει 1.5 βαθμό στον τελικό βαθμό της/του φοιτήτριας/τή. Στόχος του μέρους αυτού είναι η υλοποίηση τόσο του κωδικοποιητή των ενδείξεων που πραγματοποιείται στην πλευρά του Αποστολέα, όσο και την αποκωδικοποίηση του μηνύματος που λαμβάνει ο Δέκτης. Καλείστε να υλοποιήσετε όποιον κωδικοποιητή-αποκωδικοποιητή επιλέξετε και να ελέγξετε την ορθή λειτουργία του τόσο μεμονωμένα όσο και σε συνδυασμό με το σύστημα Αποστολέα-Δέκτη που υλοποιήσατε στα προηγούμενα μέρη της εργασίας. Τα παραδοτέα του συγκεκριμένου μέρους παραμένουν τα ίδια με τα προηγούμενα μέρη, π.χ. κώδικας Verilog, testbench, κυματομορφές, σχεδιαγράμματα, κτλ..

## Βαθμολόγηση

Η βαθμολόγηση θα πραγματοποιηθεί βάση της αναφοράς που θα συντάξετε για όλα τα μέρη της εργασίας, καθώς και πάνω στην ποιότητα του κώδικα και αρχείου ελέγχου (testbench) που θα παραδώσετε. Θα υπάρξει ολιγόλεπτη (10’-15’) προφορική εξέταση πάνω στην υλοποίηση που πραγματοποιήσατε στο τέλος της εργασίας, οπότε θα προκύψει και η συνολική βαθμολόγησή σας. Η Εργασία θα είναι ομαδική, των δύο ατόμων, αλλά η βαθμολόγηση ατομική, βάσει των απαντήσεων που θα δώσετε στην προφορική εξέταση. Υπενθύμιση, η μέγιστη βαθμολογία του υποχρεωτικού μέρους είναι 2 μονάδες επί του τελικού βαθμού, δηλαδή 2 στα 10, και αν υλοποιήσετε σωστά το προαιρετικό Μέρος Δ θα προστεθεί 1.5 βαθμός στον τελικό βαθμό σας. Η ημερομηνία παράδοσης και εξέτασης θα ανακοινωθεί στο elearning.

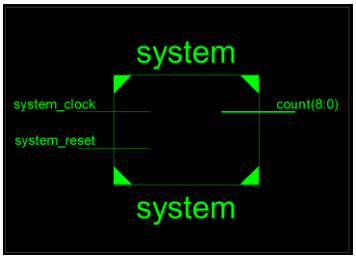
# Καλή Επιτυχία!!!

## Παράρτημα

### Α. Σχηματικό διάγραμμα του κυκλώματος

*Παράδειγμα RTL σε Verilog και σχηματικού διαγράμματος του top-level module ενός αθροιστή 8-bit, ο οποίος αυξάνει κατά ένα την προηγούμενη τιμή της εξόδου του.*

```
module system(
   system_clock,
   system_reset,
   count
);

input system_clock, system_reset;
output [8:0] count;

wire [7:0] reg2adder;
wire [8:0] adder2reg;

adder8 adder_0(
   .in0(reg2adder),
   .in1(8'b00000001),
   .out(adder2reg)
);

reg8 reg_0 (
   .datain(adder2reg),
   .dataout(reg2adder),
   .clock(system_clock),
   .reset(system_reset)
);

assign count=adder2reg;

endmodule
```

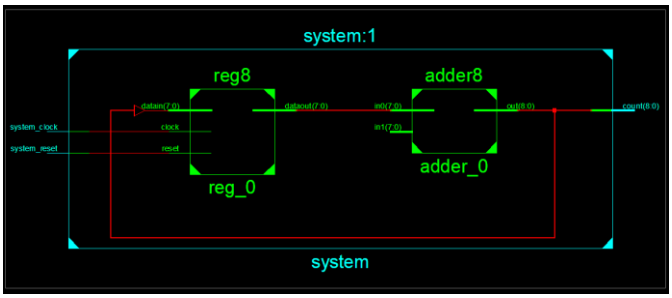



Στο παραπάνω σχηματικό διακρίνονται οι είσοδοι και οι έξοδοι του συστήματός μας. Το σύστημα έχει ως είσοδο το ρολόι (system_clock) και το σήμα αρχικοποίησης του συστήματος (system_reset). Ως έξοδο έχει το σήμα 9-bit count, το οποίο και φέρει κάθε φορά το αποτέλεσμα του αθροιστή.

*Παράδειγμα RTL σε Verilog και σχηματικού διαγράμματος των module που αρχικοποιούνται εντός του παραπάνω αθροιστή.*

```
module adder8(
in0,
in1,
out
    );

input [7:0] in0,in1;
output out;

reg [8:0] out;

always @ (in0 or in1)
begin

out<= in0 + in1;

end

endmodule
```

```
module reg8(
datain,
dataout,
clock,
reset
    );

input [7:0] datain;
output [7:0] dataout;

input clock,reset;


reg [7:0] dataout;

always @ (posedge clock)
begin
   if (reset)
      dataout<=0;
   else
      dataout<=datain;
end


endmodule
```




Απεικονίζοντας τα επιμέρους στοιχεία του συστήματος, διακρίνουμε την συνδεσμολογίας τους σε επίπεδο RTL. Διακρίνουμε ότι η είσοδος του adder_0 είναι η έξοδος του καταχωρητή reg_0. Η δεύτερη είσοδος του adder_0 φαίνεται αποσυνδεδεμένη διότι κατά την αρχικοποίηση του adder_0 (δείτε την RTL του προηγούμενου παραδείγματος) η είσοδος in1 έχει τεθεί σταθερά στην τιμή 1.

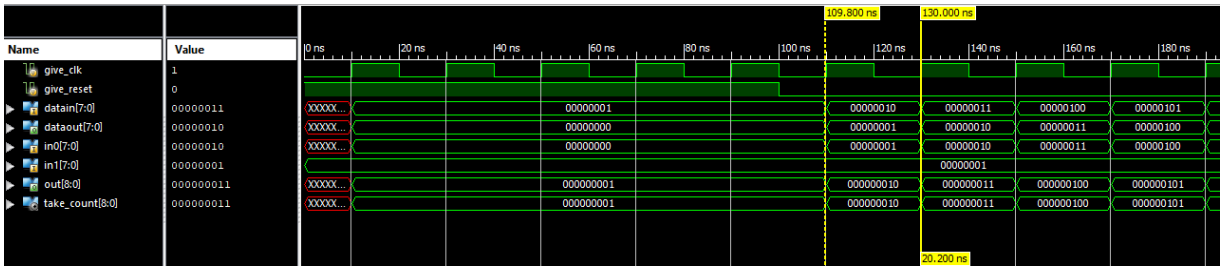
Επιπλέον, στο σχηματικό παρατηρούμε ότι υπάρχει ανάδραση της εξόδου του adder_0 προς τον reg_0 ώστε να αποθηκευτή η τιμή του αποτελέσματος και να χρησιμοποιηθεί στον επόμενο κύκλο

ρολογιού. Είναι σημαντικό να τονιστεί η χρήση του σήματος reset, όπου κάθε φορά που τίθεται σε 1, και το ρολόι είναι στη θετική ακμή (σύγχρονο reset), τότε θέτει τη τιμή του dataout (δηλαδή της εισόδου in0 του adder_0) σε 0, ώστε να ξεκινήσει το σύστημα να αθροίζει από το μηδέν.

## Β. Κυματομορφές

*Παράδειγμα testbench και Κυματομορφών.*

```
module tb;

reg give_clk,give_reset;
wire [8:0] take_count;


system sys0(
   .system_clock(give_clk),
   .system_reset(give_reset),
   .count(take_count)

);

initial begin

   give_clk = 0;
   give_reset = 1;


   #100;

   give_reset = 0;

   #10000 $finish;

end

always #10 give_clk = ~ give_clk;

endmodule
```




Στην εικόνα αυτή μπορείτε να δείτε τις κυματομορφές των ορισμένων σημάτων του συστήματος του αθροιστή, καθώς και το testbench που τις παράγει. Στις κυματομορφές μπορούμε να επιβεβαιώσουμε ότι η περίοδος του ρολογιού είναι 20 ns, όπως μπορούμε να δούμε και στο testbench. Επιπλέον, παρατηρούμε ότι μέχρι το reset να τεθεί σε 0, τα σήματα παραμένουν σταθερά. Όταν το reset τεθεί σε 0, τότε κάθε κύκλο ρολογιού παρατηρούμε την αύξηση της τιμής εξόδου.