ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ψηφιακά Συστήματα HW σε Χαμηλά Επίπεδα Λογικής Ι

7ο’ ΕΞΑΜΗΝΟ

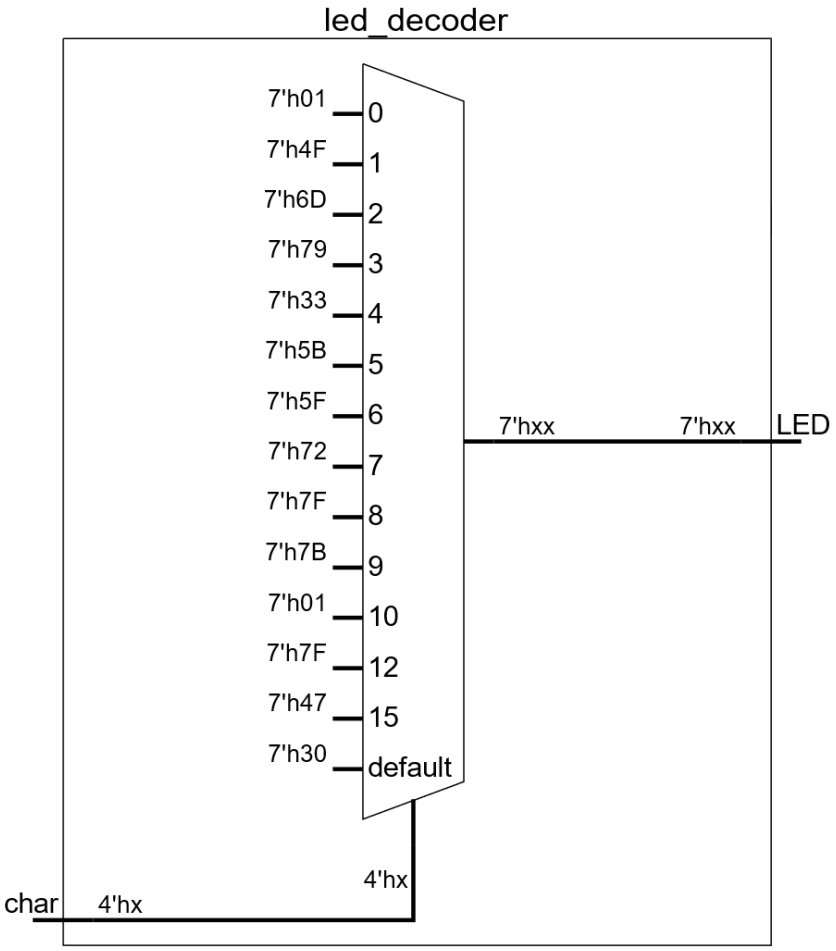
“Υλοποίηση Συστήματος Αποστολέα-Δέκτη σε Verilog με χρήση Πρωτοκόλλου UART και προβολή σε Οθόνη Τεσσάρων LED 7-Τμημάτων”

Φοιτητές:
Λιουλιάκης Νικόλαος – 10058
Παναγιώτης Συσκάκης - 10045

Θεσσαλονίκη
2022-2023

Μέρος Α’ – 7-Segment Display Driver

Στο πρώτο μέρος της εργασίας υλοποιήθηκε το module LEDdecoder, που έχει ως είσοδο έναν 4-bit χαρακτήρα και στην έξοδο ενεργοποιεί τα μέρη του 7-segment-display που εμφανίζουν αυτόν τον χαρακτήρα.
Υποστηρίζονται οι χαρακτήρες 0,1,2,3,4,5,6,7,8,9,F, και ο κενός χαρακτήρας, ενώ εάν στην είσοδο έρθει κάποια άγνωστη σειρά bit, ανάβει στην οθόνη η ένδειξη χαρακτήρα E (“Error”).

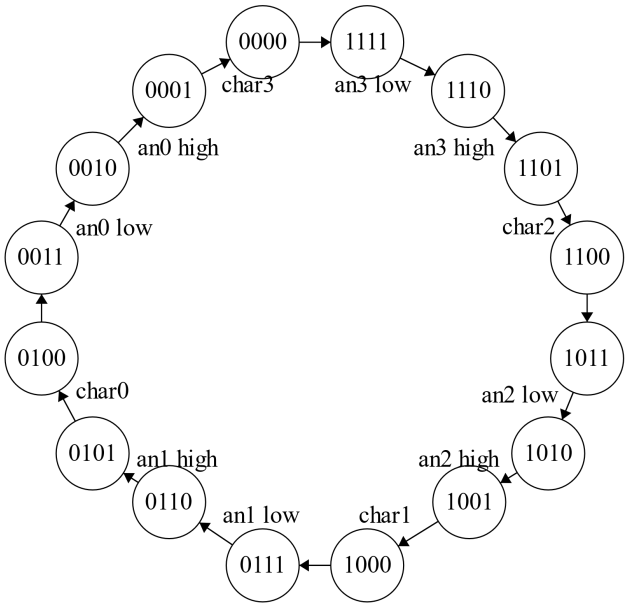


Schematic 1 LED Decoder

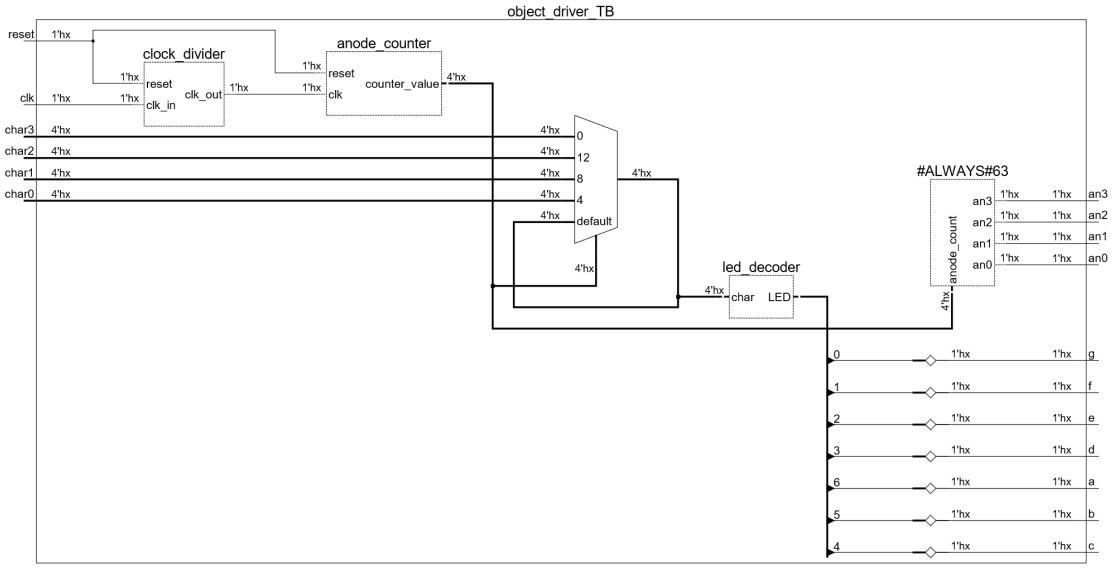
```
case(char)
    4'h0:LED = 7'b0000001;   // 0
    4'h1:LED = 7'b1001111;   // 1
    4'h2:LED = 7'b1101101;   // 2
    4'h3:LED = 7'b1111001;   // 3
    4'h4:LED = 7'b0110011;   // 4
    4'h5:LED = 7'b1011011;   // 5
    4'h6:LED = 7'b1011111;   // 6
    4'h7:LED = 7'b1110010;   // 7
    4'h8:LED = 7'b1111111;   // 8
    4'h9:LED = 7'b1111011;   // 9
    4'ha:LED = 7'b0000001;   // -
    // none ^_^
    4'hc:LED = 7'b1111111;   // " "
    4'hf:LED = 7'b1000111;   // F

    default:LED = 7'b0110000;    // E (error case)

endcase
```

Code Block 1 LED Decoder

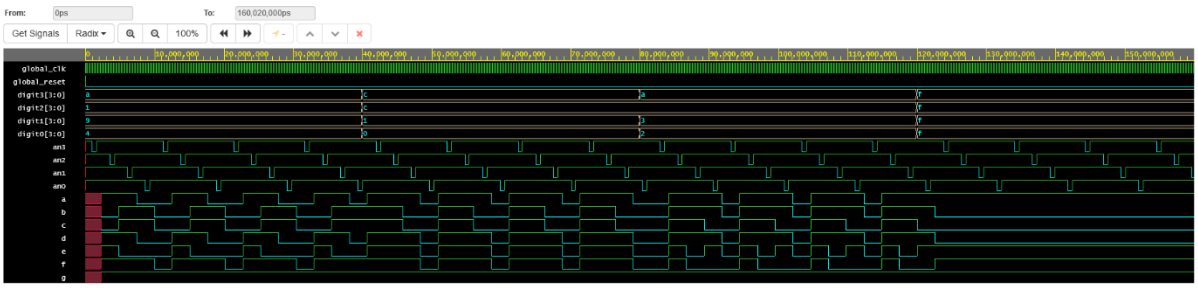
Έπειτα, υλοποιήθηκε το module FourDigitLEDdriver, το οποίο έχει ως είσοδο τους 4 χαρακτήρες που θέλουμε να δείξουμε στην οθόνη. Καθώς η οθόνη αποτελείται από τέσσερα 7-segment displays, στην έξοδο εναλλάσσονται κατάλληλα τα anodes και τα segments (a,b,c,d,e,f,g), έτσι ώστε οι 4 χαρακτήρες να προβάλλονται διαδοχικά, ο καθένας στην αντίστοιχη οθόνη.
Ο χρονισμός για τη φόρτωση των χαρακτήρων και την ενεργοποίηση των ανόδων γίνεται με τη χρήση ενός μετρητή που είναι ισοδύναμος με το παρακάτω FSM.



FSM 1 LED Driver



Schematic 2 LED Driver



Waveform 1 LED Driver

Ορθή Λειτουργία LED Driver

Από το Waveform 1 φαίνεται ότι στην έξοδο του LED Driver τα σήματα που οδηγούν τα τμήματα της οθόνης (a, b, c, d, e, f, g) παίρνουν την κατάλληλη τιμή η οποία εξαρτάται από το ποια από τις 4 οθόνες θα οδηγηθεί.

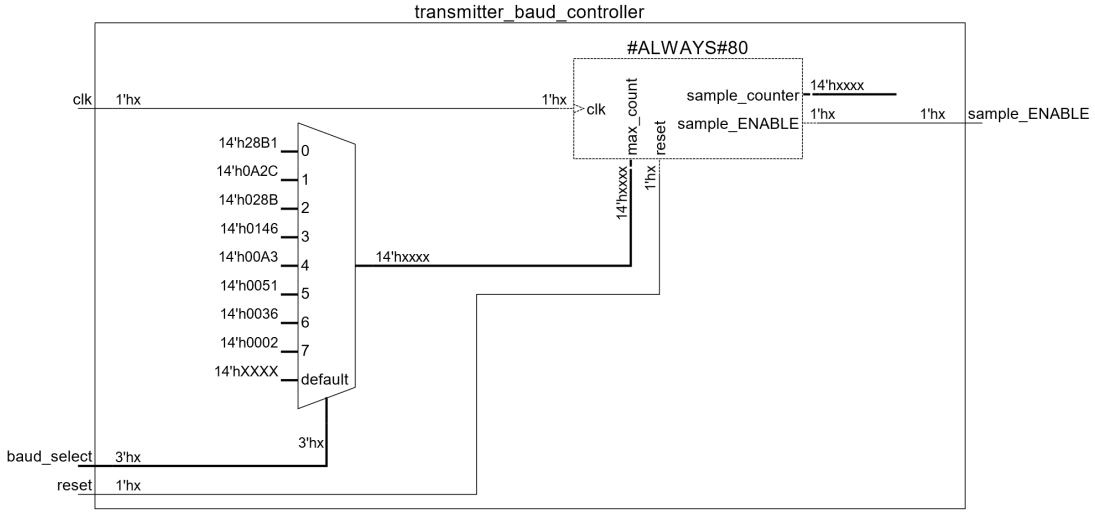
Επιλέξαμε καθυστέρηση φόρτισης 0.64μs (που μπορεί να αλλάξει εύκολα με την παράμετρο RANK στο leddriver.v). Ο κάθε χαρακτήρας φορτώνεται 1.28μs πριν την κάθοδο του επόμενου anode.
Η περίοδος ενός πλήρη κύκλου ενεργοποίησης και των 4ων οθονών είναι 10.24μs (που ισοδυναμεί σε ~100k “frames-per-second”).

Μέρος Β’ – Υλοποίηση Γενικού Ασύγχρονου Δέκτη-Αποστολέα (UART)

Για την υλοποίηση του ζεύγους UART, ήταν αρχικά απαραίτητη η δημιουργία του module BaudController, που δημιουργεί παλμούς κατάλληλου χρονισμού ανάλογα με το Baud rate που έχει επιλεγεί.
Θεωρήθηκε ότι παρέχεται clock γνωστής περιόδου 20ns, έτσι με έναν απλό counter μπορεί να μετρηθεί οποιαδήποτε χρονική διάρκεια πολλαπλάσια του 20ns, και να παραχθεί παλμοσειρά με περίοδο αυτή τη χρονική διάρκεια. Καθώς είναι επιθυμητό ο δέκτης να λειτουργεί με ταχύτητα 16 φορές αυτή του αποστολέα, παράγονται τελικά παλμοσειρές με συχνότητα 16 φορές μεγαλύτερη από αυτή των αντίστοιχων Baud rates.
Επειδή τα baud rates δε μπορούν να παραχθούν ακριβώς με το ρολόι των 20ns θα υπάρχει κάποια απόκλιση από το επιθυμητό baud rate. Έπειτα από τους κατάλληλους υπολογισμούς παρατηρούμε όπως περιμέναμε ότι το σχετικό σφάλμα μεγαλώνει όσο αυξάνει η ταχύτητα επικοινωνίας (Πίνακας 1).

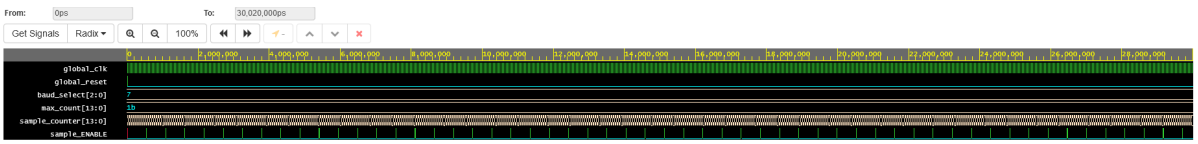
Ideal values			Actual values			
Baud Rate (bits/sec)	Clock pulses between samples	Clock pulses between bits	Clock pulses between samples	Clock pulses between bits	Baud rate (based on sample rate)	Relative error in baud rate
300	10416,667	166666,667	10417	166672	299,990	0,0032%
1200	2604,167	41666,667	2604	41664	1200,077	0,0064%
4800	651,042	10416,667	651	10416	4800,307	0,0064%
9600	325,521	5208,333	326	5216	9585,890	0,1470%
19200	162,760	2604,167	163	2608	19171,779	0,1470%
38400	81,380	1302,083	81	1296	38580,247	0,4694%
57600	54,253	868,056	54	864	57870,370	0,4694%
115200	27,127	434,028	27	432	115740,741	0,4694%

Πίνακας 1

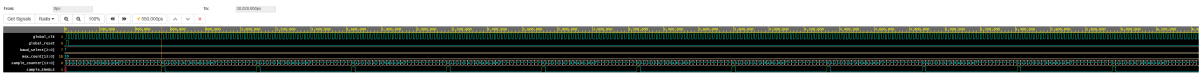


Schematic 3 Baud Controller

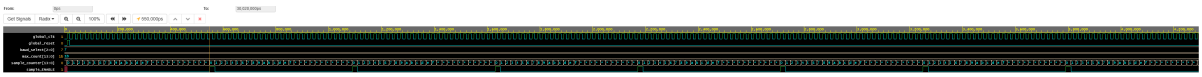
Στις παρακάτω κυματομορφές φαίνεται το ρολόι στα 50MHz (περίοδος 20ns) και το sample_counter που είναι ένας μετρητής που μετράει μέχρι το max_count. Το max_count παίρνει κατάλληλη τιμή ανάλογα με το baud_select (είναι η στήλη Clock pulses between samples του Πίνακας 1). Υπάρχουν πολλές κυματομορφές για 3 διαφορετικά baud_select και με διαφορετική κλίμακα στους χρόνους έτσι ώστε σε κάποιες να φαίνεται η περιοδικότητα των σημάτων και σε κάποιες να φαίνονται οι τιμές του sample_counter για την εύκολη επαλήθευση της ορθής λειτουργίας.



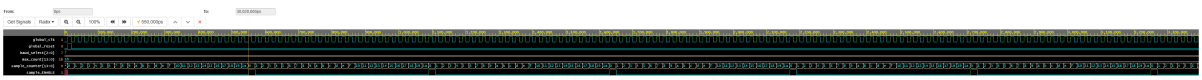
Waveform 2 Baud selection 111 Baud Rate 115200 Max count 27



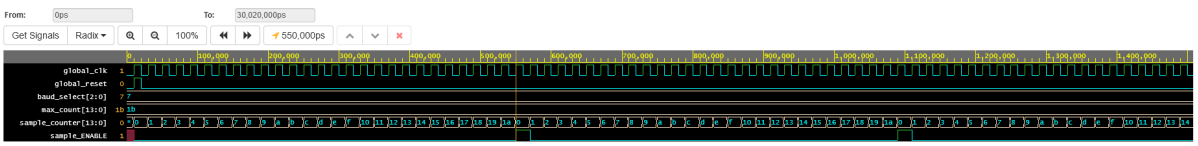
Waveform 3 Baud selection 111 Baud Rate 115200 Max count 27 Sample pulses 11



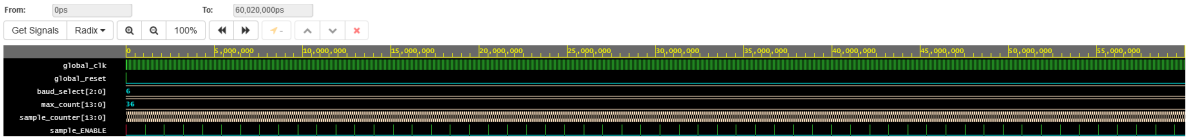
Waveform 4 Baud selection 111 Baud Rate 115200 Max count 27 Sample pulses 7



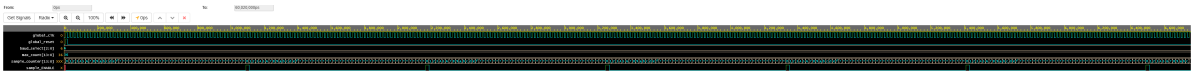
Waveform 5 Baud selection 111 Baud Rate 115200 Max count 27 Sample pulses 6



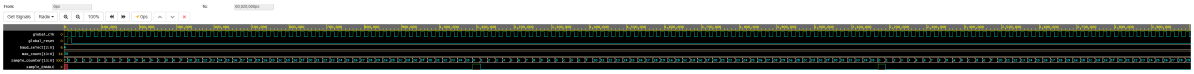
Waveform 6 Baud selection 111 Baud Rate 115200 Max count 27 Sample pulses 2



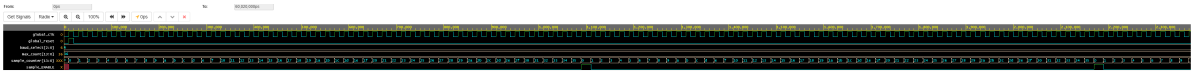
Waveform 7 Baud selection 110 Baud Rate 57600 Max count 54



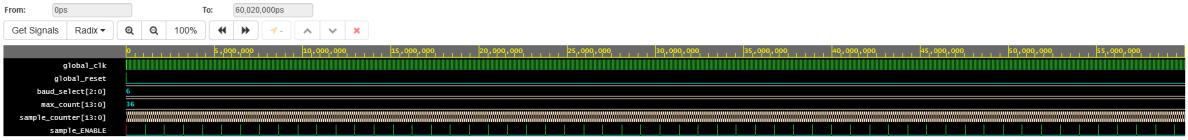
Waveform 8 Baud selection 110 Baud Rate 57600 Max count 54 Sample pulses 6



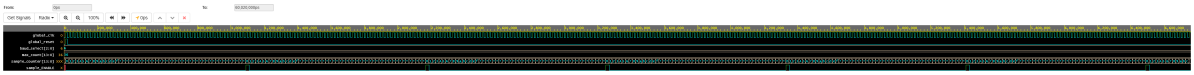
Waveform 9 Baud selection 110 Baud Rate 57600 Max count 54 Sample pulses 3



Waveform 10 Baud selection 110 Baud Rate 57600 Max count 54 Sample pulses 2



Waveform 11 Baud selection 010 Baud Rate 4800 Max count 651



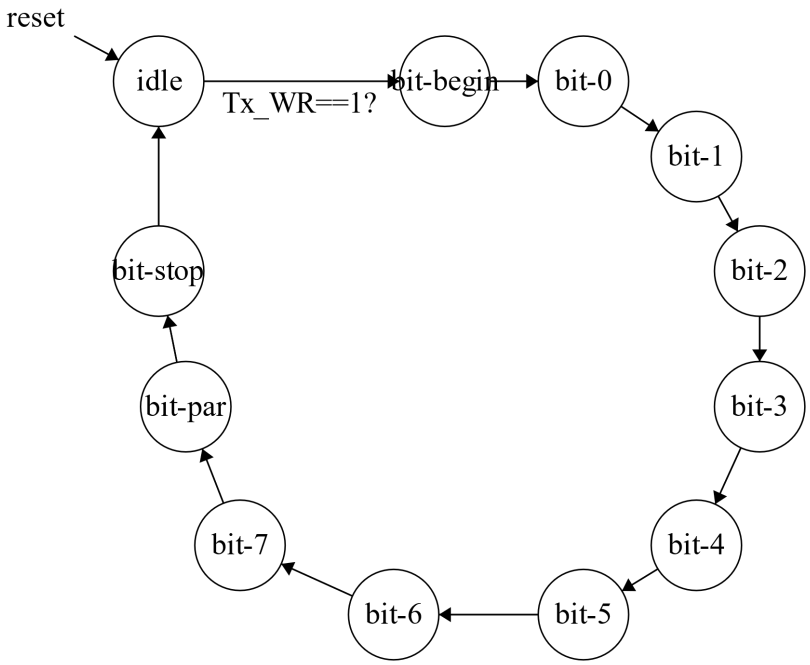
Waveform 12 Baud selection 010 Baud Rate 4800 Max count 651 Sample pulses 1

Ο αποστολέας (UART_transmitter) υλοποιήθηκε ως FSM 12 καταστάσεων, έχοντας μια κατάσταση για κάθε μεταδιδόμενο bit, και επιπλέον μια “idle” κατάσταση. Εσωτερικά, ο αποστολέας διαθέτει έναν BaudController, για κατάλληλο χρονισμό ανάλογα με το επιθυμητό baud rate.

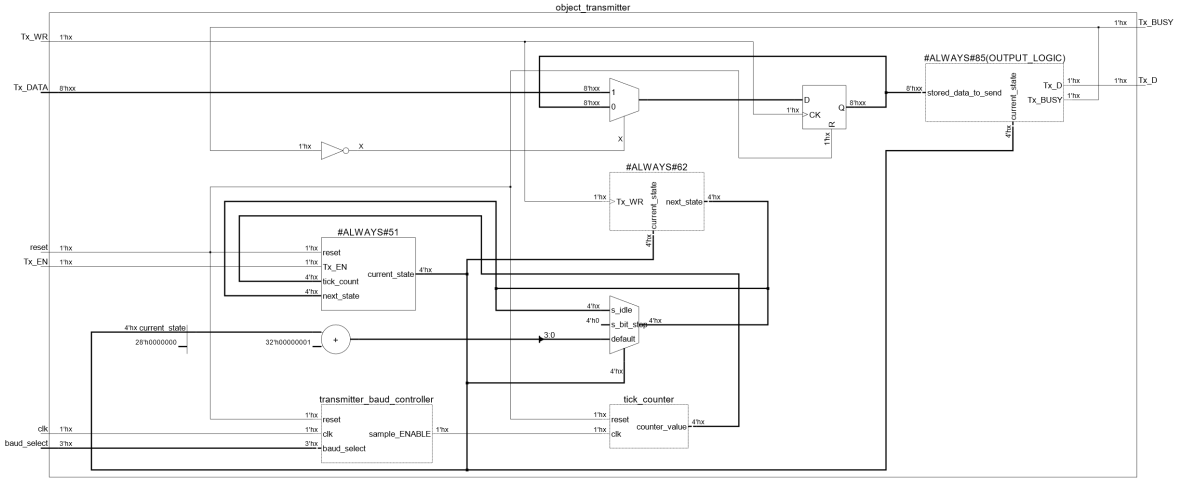
Ο αποστολέας ξεκινά από την idle κατάσταση. Όταν ληφθεί posedge Tx_WR (εφόσον ο αποστολέας δεν είναι Busy), αποθηκεύονται τα δεδομένα που βρίσκονται στην είσοδο του module σε ένα buffer, και ξεκινά η αποστολή τους. Κατά την αποστολή ενεργοποιείται το σήμα Tx_Busy, το οποίο δεν επιτρέπει να ξεκινήσει άλλη αποστολή μέχρι αυτή να τελειώσει.

Καθώς ο BaudController παράγει 16 παλμούς ανά bit, οι εναλλαγές των καταστάσεων στο FSM γίνονται μόνο κάθε 16ο παλμό. Ο αποστολέας έχει μια ιδιοτροπία η οποία μπορεί να καθυστερήσει την εκκίνηση της αποστολής μέχρι ο εσωτερικός μετρητής που έχει να φτάσει έναν αυθαίρετο αριθμό. Η μέγιστη καθυστέρηση που μπορεί να έχει είναι 15 παλμούς. Θα μπορούσε να βελτιστοποιηθεί με λίγη ακόμα λογική ωστόσο θα τον έκανε πιο περίπλοκο.

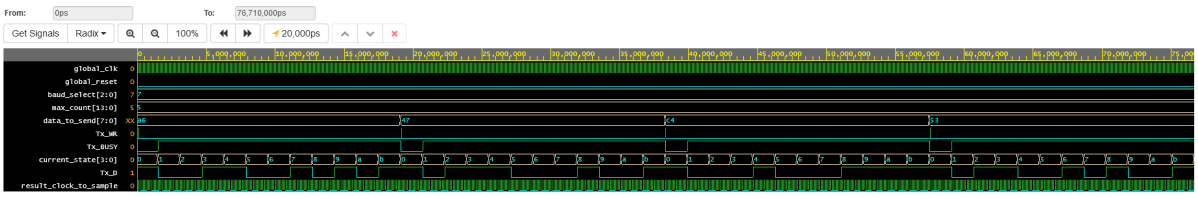
Στις κυματομορφές του transmitter για να είναι δυνατή η προσομοίωση και προβολή τους επιλέχτηκε πιο γρήγορο baud rate (625000 bits/sec).



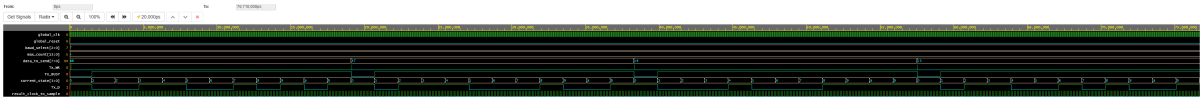
FSM 2 Transmitter



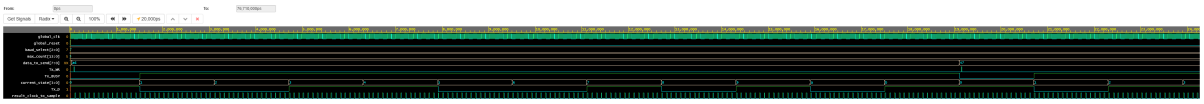
Schematic 4 Transmitter



Waveform 13 Transmitter transmissions 4 (Baud Rate 625000 Max count 5)



Waveform 14 Transmitter transmissions 4 (Baud Rate 625000 Max count 5)



Waveform 15 Transmitter transmissions 1 (Baud Rate 625000 Max count 5)

Ο δέκτης (UART_Receiver) διαθέτει επίσης εσωτερικά ένα BaudController. Ο δέκτης υλοποιήθηκε με τον συνδυασμό δύο FSM. Το πρώτο FSM (FSM 3) υποδεικνύει ποιο bit λαμβάνουμε ανά πάσα στιγμή, και άρα έχει μια “idle” κατάσταση και μία κατάσταση για κάθε λαμβανόμενο bit. Το δεύτερο FSM (FSM 4) λειτουργεί ως μετρητής, και υποδεικνύει τον αριθμό του sample που δειγματοληπτούμε.

Ο δέκτης ξεκινά από την idle κατάσταση. Μόλις εντοπιστεί negedge του Rx_D (δηλαδή το start bit στο σειριακό κανάλι), ξεκινάει η διαδικασία λήψης δεδομένων. Για κάθε bit που περιμένουμε να λάβουμε, κάνουμε 14 δειγματοληψίες (στους μεσαίους από τους 16 παλμούς που παράγει ο BaudController). Εάν τα 14 δείγματα δεν είναι όλα ίδια μεταξύ τους (Rx_FERROR), χρησιμοποιείται η τιμή της πλειοψηφίας ως καλύτερη εκτίμηση *(επειδή δεν είναι περιττό το πλήθος των δειγμάτων η υπάρχει η* περίπτωση *να είναι μισά-μισά που τότε επιλέγεται το μηδέν)*. Η διαδικασία επαναλαμβάνεται για όλα τα 11 bit που περιμένουμε να λάβουμε.

Υπάρχουν δύο δυνατά σφάλματα:

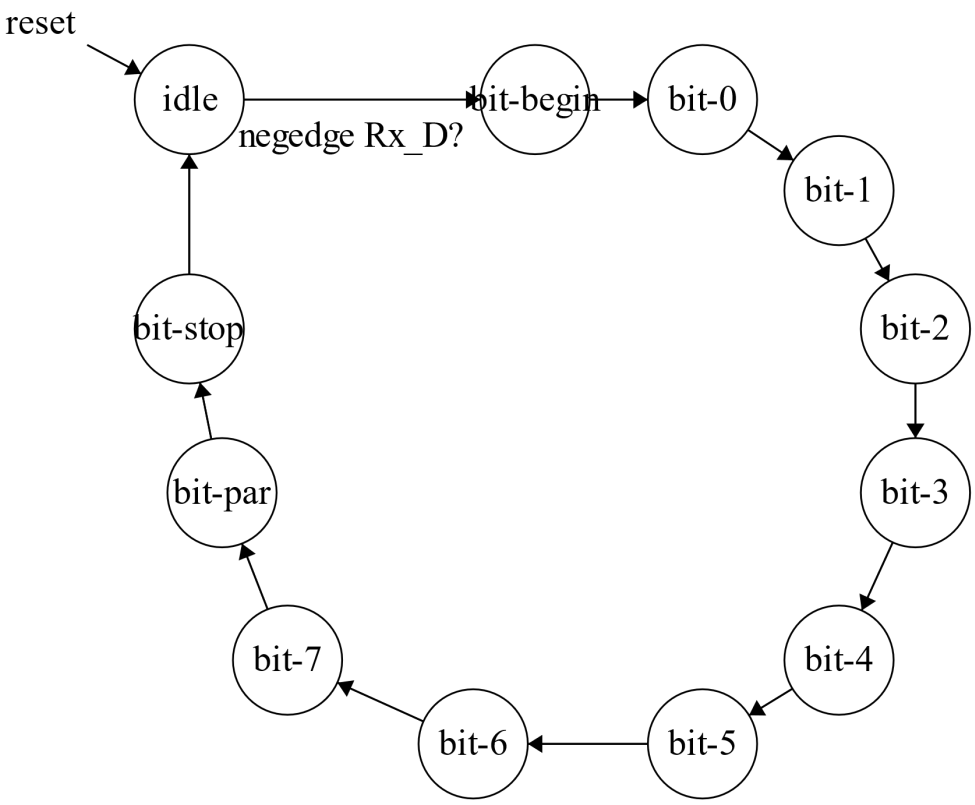
- Parity error, δηλαδή ασυμφωνία μεταξύ του parity bit και του parity check στα δεδομένα που θεωρήθηκε ότι λήφθηκαν
- Frame error, δηλαδή ασυμφωνία των δειγμάτων στη δειγματοληψία κάποιου bit, ή αδυναμία λήψης του stop bit.

Στο τέλος αφού έχουμε λάβει το stop-bit αν δεν υπήρξε Rx_PERROR ή Rx_FERROR ενεργοποιούμε το σήμα Rx_VALID.

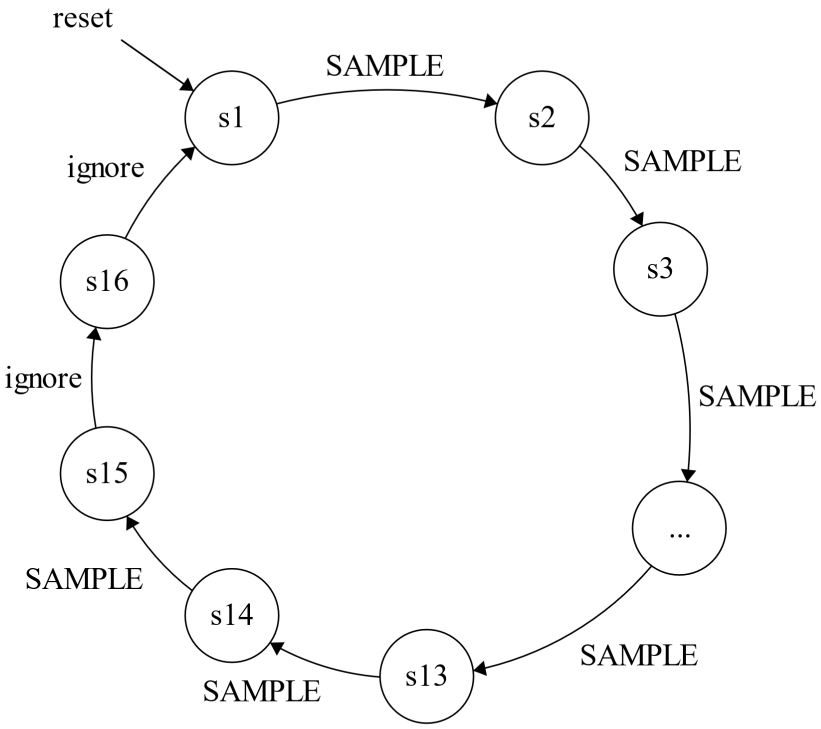
Στην έξοδο, τα δεδομένα ανανεώνονται συνεχόμενα κατά τη διάρκεια της λήψης. Αυτά που δεν έχουν ληφθεί ακόμα, διατηρούν την προηγούμενη τιμή που είχαν. Ακόμα και αν στο τέλος της λήψης υπάρχει κάποιο error, τα δεδομένα στην έξοδο είναι οι πιο πιθανές τιμές που μπορεί να είχαν τα bits (γιατί χρησιμοποιήθηκε η πλειοψηφία για τη λήψη της απόφασης).

Τέλος, ο δέκτης μπαίνει ξανά σε λειτουργία idle (εκτός αν υπάρχει start bit αμέσως μετά το stop bit), αναμένοντας την επόμενη επικοινωνία.

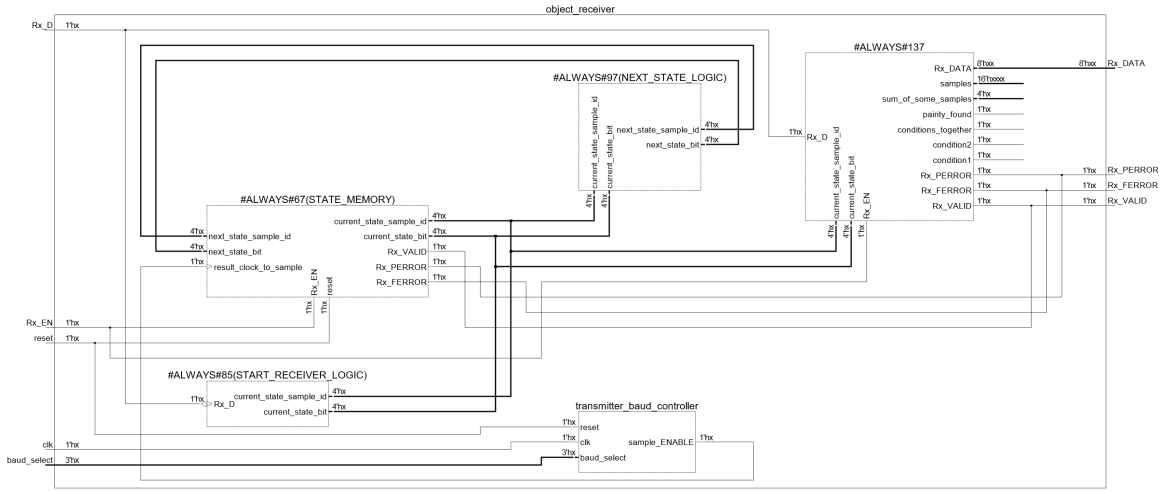
Για τη δημιουργία των κυματομορφών εισόδου του receiver χρησιμοποιήσαμε τον transmitter που υλοποιήσαμε προηγουμένως. Οι προσομοιώσεις έγιναν για δύο διαφορετικά baud rates για την επαλήθευση της ορθής λειτουργίας (του receiver και του transmitter). Για να είναι δυνατή η προσομοίωση και προβολή των κυματομορφών επιλέχτηκε πιο γρήγορο baud rate (625000 bits/sec και 446428,571 bits/sec).



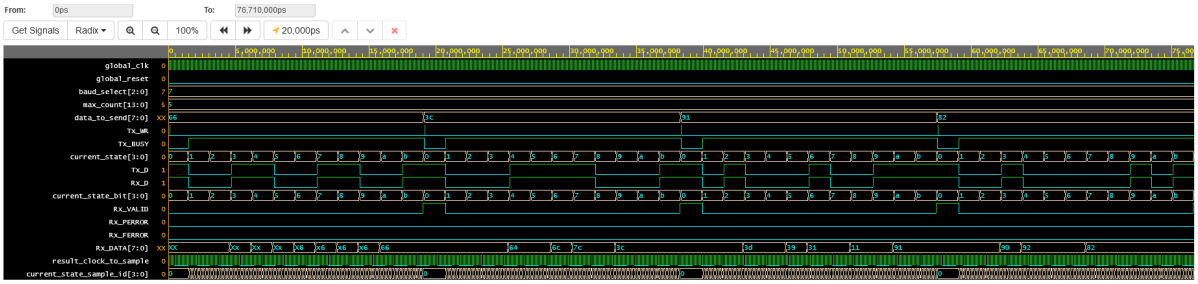
FSM 3 Receiver



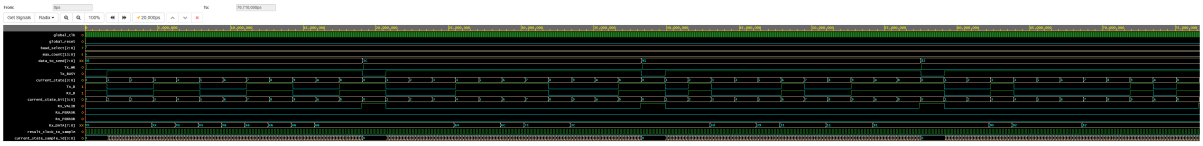
FSM 4 Receiver sampling



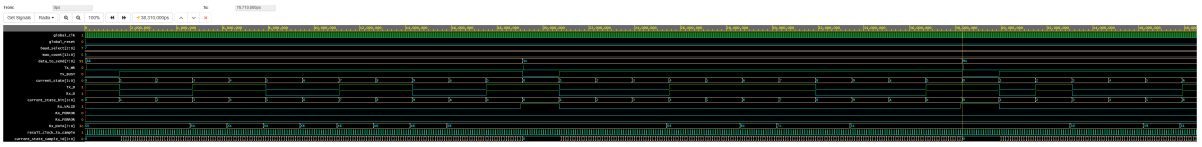
Schematic 5 Receiver



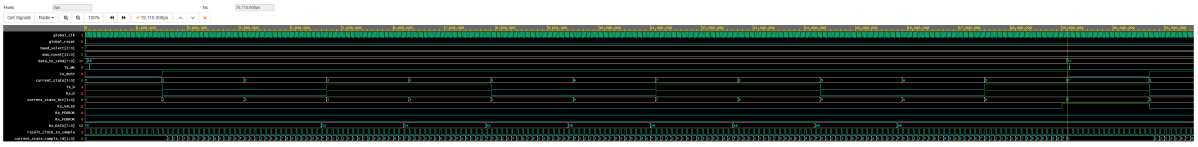
Waveform 16 Receiver transmissions 4 (Baud Rate 625000 Max count 5)



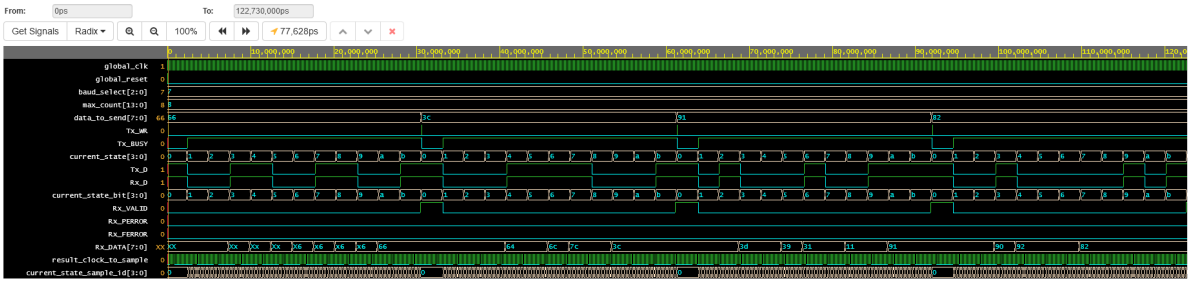
Waveform 17 Receiver transmissions 4 (Baud Rate 625000 Max count 5)



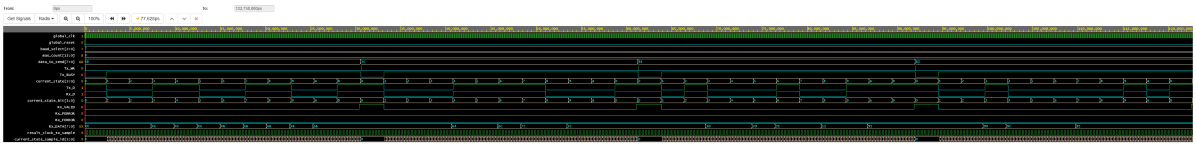
Waveform 18 Receiver transmissions 2 (Baud Rate 625000 Max count 5)



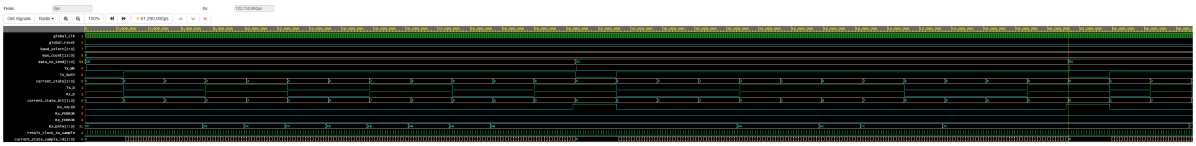
Waveform 19 Receiver transmissions 1 (Baud Rate 625000 Max count 5)



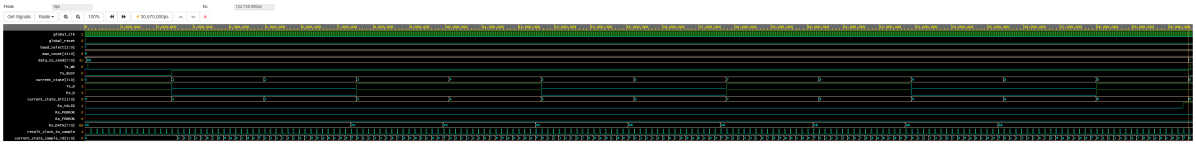
Waveform 20 Receiver transmissions 4 (Baud Rate 446428,571 Max count 7)



Waveform 21 Receiver transmissions 4 (Baud Rate 446428,571 Max count 7)

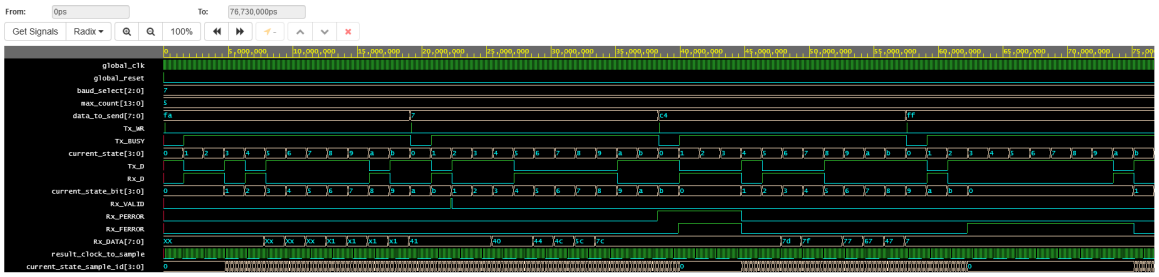


Waveform 22 Receiver transmissions 2 (Baud Rate 446428,571 Max count 7)

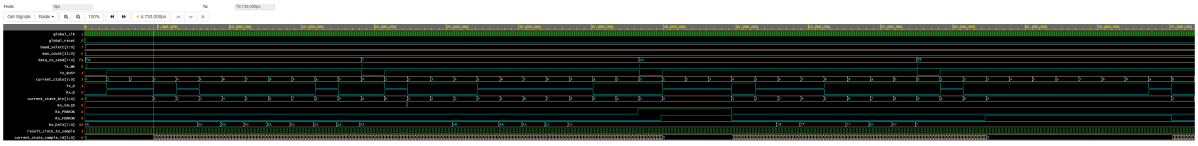


Waveform 23 Receiver transmissions 1 (Baud Rate 446428,571 Max count 7)

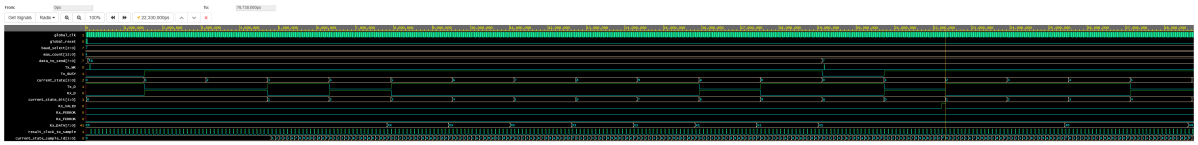
Για την επαλήθευση της σωστής διαχείρισης σφαλμάτων αντιστρέψαμε την έξοδο του transmitter πριν τη συνδέσουμε στην είσοδο του receiver. Έτσι μπορέσαμε να δημιουργήσουμε Valid, Parity Errors και Frame Errors.



Waveform 24 Receiver Error transmissions 3 (Baud Rate 625000 Max count 5)



Waveform 25 Receiver Error transmissions 3 (Baud Rate 625000 Max count 5)



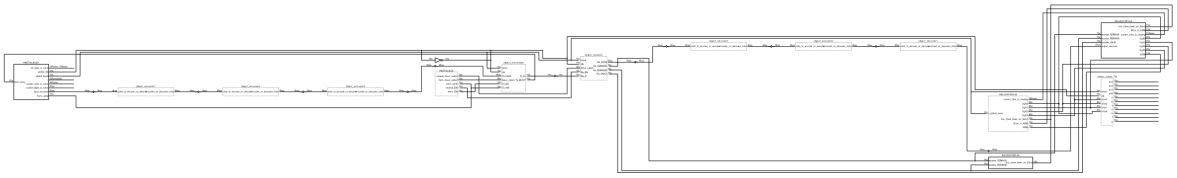
Waveform 26 Receiver Error transmissions 1 (Baud Rate 625000 Max count 5)

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

Ο σκοπός αυτού του μέρους της εργασίας είναι να πραγματοποιηθεί επικοινωνία αποστολέα-δέκτη (με κωδικοποίηση), και τα δεδομένα που λήφθηκαν να προβληθούν στην οθόνη.

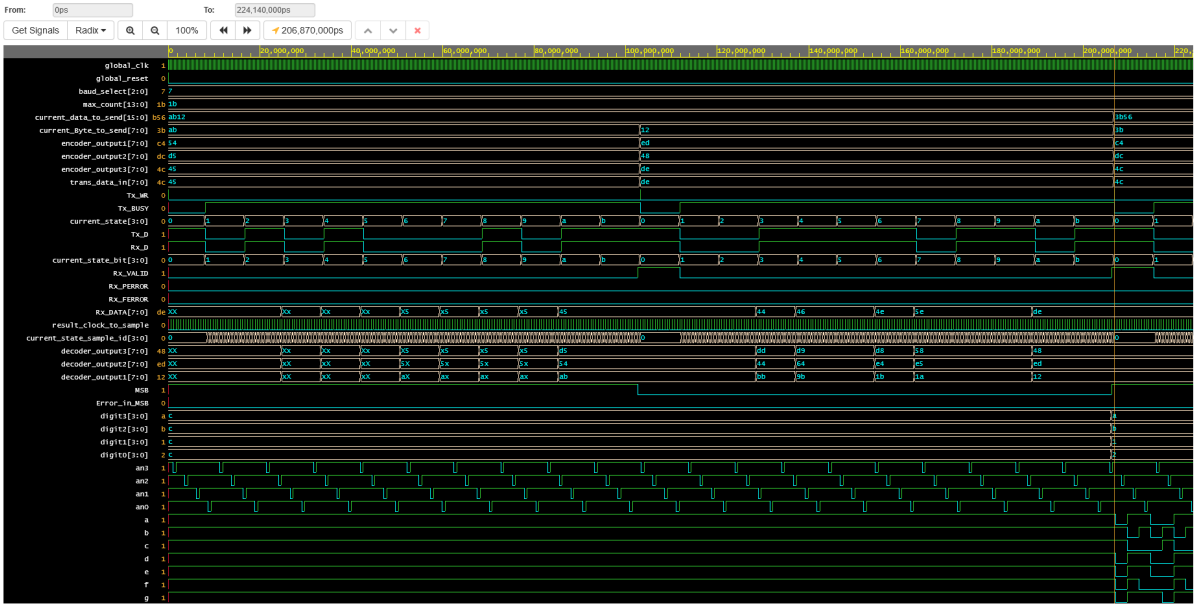
Επειδή οι επικοινωνίες γίνονται σε πακέτα 8 bits αλλά η οθόνη προβάλλει 16 bits (4 χαρακτήρες των 4 bit), απαιτούνται 2 επικοινωνίες για κάθε ένδειξη της οθόνης. Αυτό το πρόβλημα λύθηκε με τη χρήση ενός 16-bit buffer, στον οποίο σε κάθε λήψη αποθηκεύουμε τα 8 bit που λήφθηκαν, εναλλάξ είτε ως 8 MSB είτε ως 8 LSB.

Εάν λήφθηκαν χωρίς σφάλμα 4 χαρακτήρες (16 bits), τότε αυτοί προβάλλονται στην οθόνη. Εάν υπήρξε σφάλμα σε κάποια από τις 2 επικοινωνίες, προβάλλεται η συμβολοσειρά “FFFF”, μέχρι να ληφθούν επιτυχώς τα επόμενα 16 bits. Η οθόνη ενημερώνεται όταν ληφθούν και οι 4 χαρακτήρες αλλιώς διατηρεί τους προηγούμενους.

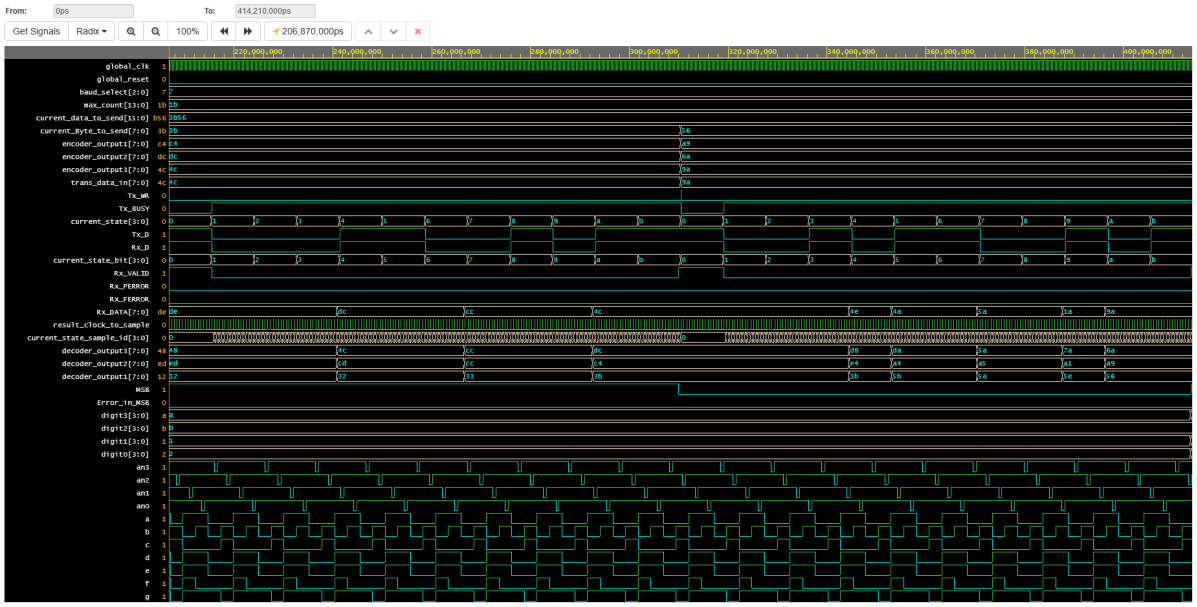


Schematic 6 All modules combined

Στις κυματομορφές Waveform 27 και Waveform 28 το Baud Rate είναι 115200 και το max_count 27. Στο Waveform 27 γίνεται η εκπομπή των πρώτων 4 χαρακτήρων οι οποίοι εφόσον έχουν ληφθεί σωστά εμφανίζονται στην οθόνη στο Waveform 28. Μέχρι να γίνει η επόμενη λήψη χαρακτήρων στη οθόνη συνεχίζουν να εμφανίζονται τα δεδομένα της προηγούμενης λήψης.

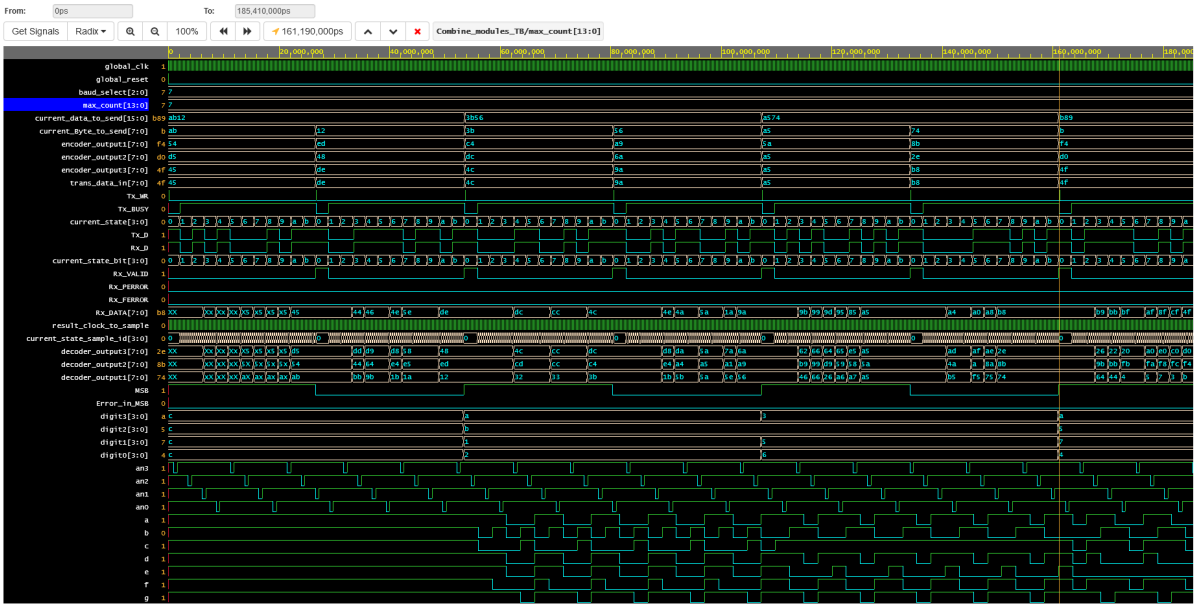


Waveform 27 Combined Modules First 2 transmissions (Baud Rate 115200 Max count 27)

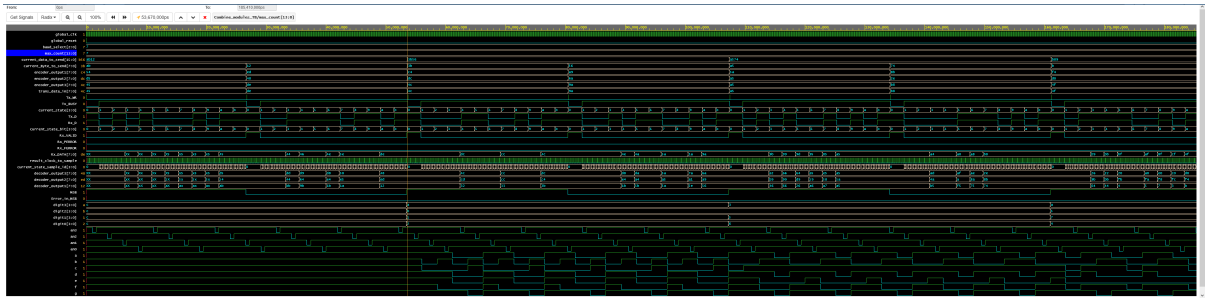


Waveform 28 Combined Modules Second 2 transmissions (Baud Rate 115200 Max count 27)

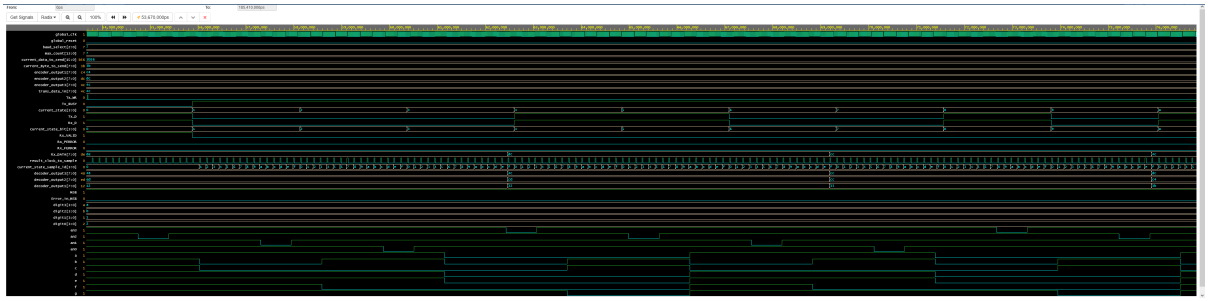
Στις κυματομορφές Waveform 29, Waveform 30, Waveform 31 το Baud Rate είναι 446428,5714 και το max_count 7.



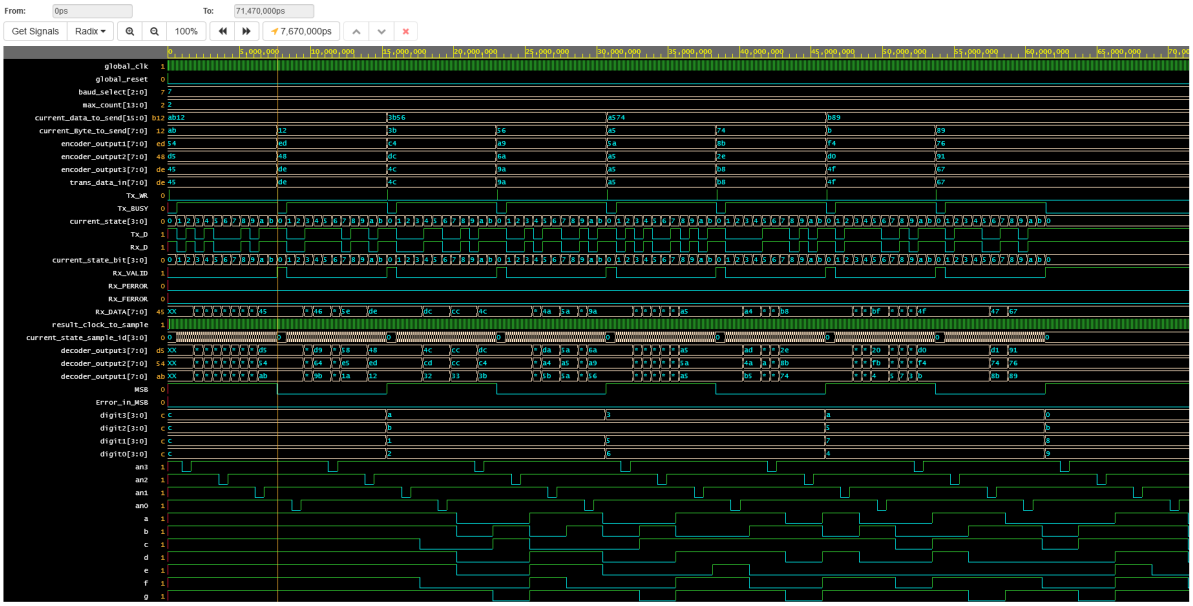
Waveform 29 Combined Modules 3 transmissions (Baud Rate 446428,5714 Max count 7)



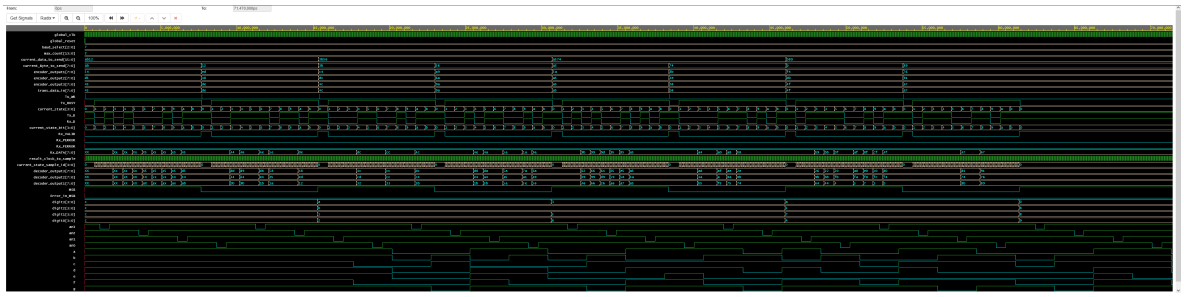
Waveform 30 Combined Modules 3 transmissions (Baud Rate 446428,5714 Max count 7)



Waveform 31 Combined Modules 0.5 transmissions (Baud Rate 446428,5714 Max count 7)

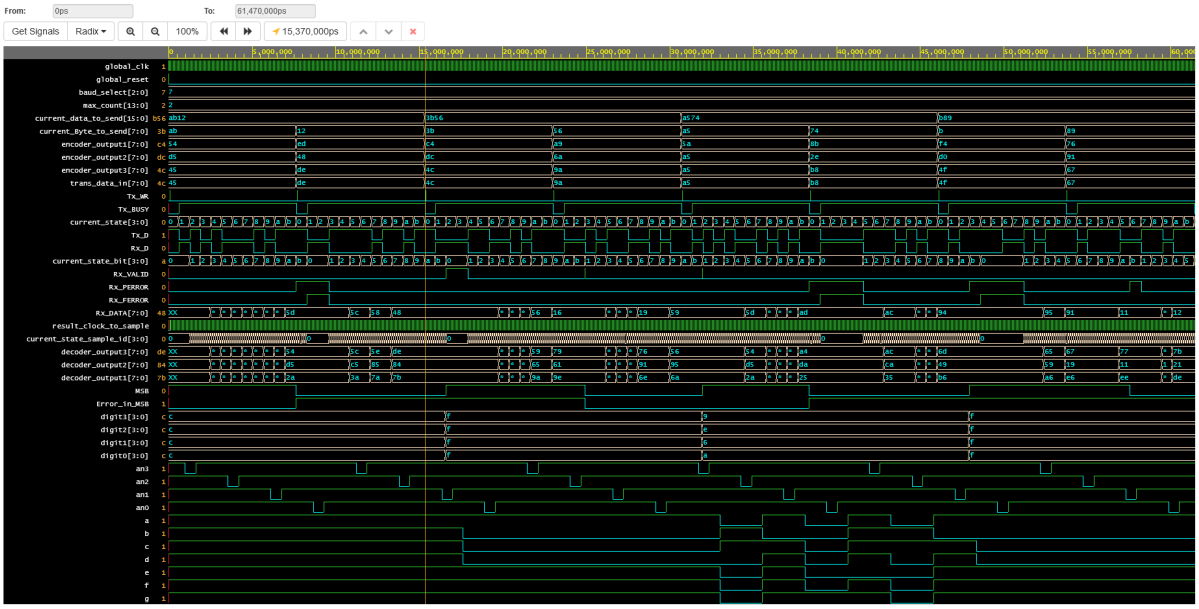


Waveform 32 Combined Modules 4 transmissions (Baud Rate 1562500 Max count 2)

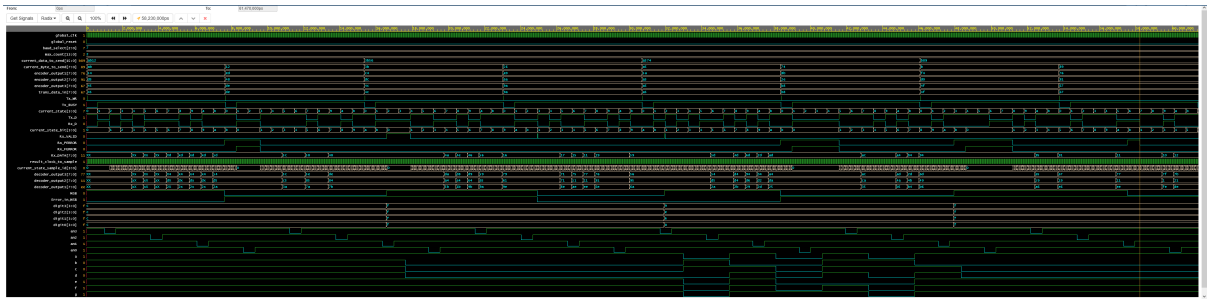


Waveform 33 Combined Modules 4 transmissions (Baud Rate 1562500 Max count 2)

Παρακάτω φαίνονται οι κυματομορφές που έχουμε λανθασμένη λήψη. Για τη δημιουργία σημάτων με σφάλματα αντιστρέψαμε την έξοδο του transmitter πριν τη συνδέσουμε στην είσοδο του receiver. Επομένως ενεργοποιούνται τα σήματα Rx_PERROR, Rx_FERROR ανάλογα το σφάλμα και μετά τη λήψη των 4 χαρακτήρων εμφανίζεται FFFF στην οθόνη.



Waveform 34 Combined Modules with Errors 4 transmissions (Baud Rate 1562500 Max count 2)



Waveform 35 Combined Modules with Errors 4 transmissions (Baud Rate 1562500 Max count 2)

Προαιρετικό Μέρος Δ – Υλοποίηση Κωδικοποιητή-Αποκωδικοποιητή

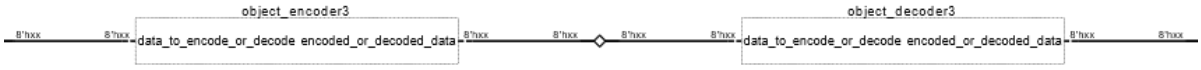
Για αυτό το κομμάτι της εργασίας, υλοποιήθηκαν τα modules encoder-decoder, τα οποία μετασχηματίζουν τα δεδομένα των 8 bit χρησιμοποιώντας πύλες NOT και συμμετρικές αναδιατάξεις των bits. Λόγω της συμμετρίας, ο encoder είναι ταυτόχρονα και decoder. Αυτό γίνεται επειδή:

$$F(F(data)) = data$$

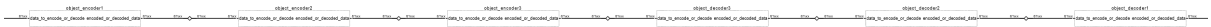
όπου F ο μετασχηματισμός του module encoder-decoder στα 8-bit data.

Επιπλέον, οι τρεις encoder-decoder που επιλέχθηκαν έχουν την ιδιότητα ότι έχουν το ίδιο αποτέλεσμα ανεξαρτήτως της σειράς που θα εφαρμοστούν. Επομένως, αν τους θεωρήσουμε ως συναρτήσεις F1, F2, F3 τότε για τη σύνθεση τους (που είναι ισοδύναμη με τη σύνδεση τους σε σειρά) ισχύει η αντιμεταθετική ιδιότητα.

Τα δεδομένα γίνονται encode πριν τη μετάδοση, και γίνονται decode αφού ληφθούν από τον δέκτη, ώστε να προβληθούν σωστά στην οθόνη.

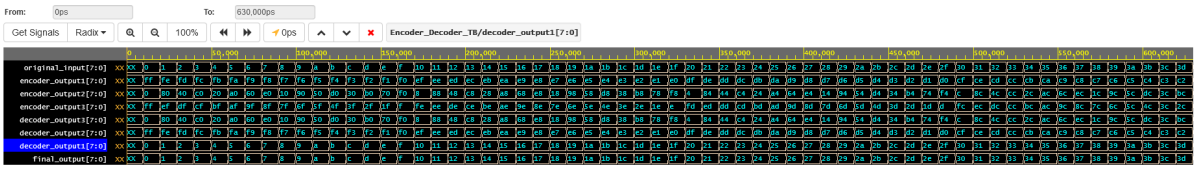


Schematic 7 Encoder-Decoder



Schematic 8 Encoders-Decoders

Στις παρακάτω κυματομορφές, βλέπουμε τρεις encoders συνδεδεμένους σε σειρά, και έπειτα τρεις αντίστοιχους (αλλά αντίστροφα συνδεδεμένους) decoders, όπου επαναφέρουν τελικά το αρχικό δεδομένο:



Waveform 36 Encoding Decoding 3 times

Σύνοψη και γενικά σχόλια

- Η εργασία ήταν αρκετή για να εξοικειωθούμε αρκετά με τη γλώσσα verilog, ώστε να αρχίσουν να μας απασχολούν τα περισσότερο ουσιαστικά προβλήματα της καλής σχεδίασης.
- Έγιναν κάποιες λάθος σχεδιαστικές επιλογές από μέρους μας, συγκεκριμένα:
    - Έλλειψη consistency στις ονομασίες των wire/reg/modules/module instances.
    - Χρήση latches σε σημεία που δεν ήταν απαραίτητο.
- Ήταν πολύ βοηθητικό που τα ερωτήματα ήταν κλιμακούμενης δυσκολίας, καθώς στην αρχή ακόμη και το απλούστερο συνδυαστικό κύκλωμα χρειάστηκε αρκετό trial-and-error μέχρι να λειτουργήσει.
- Χρησιμοποιήσαμε τον online editor “edaplayground” ωστόσο είχε πολλά προβλήματα και συχνά crashes με αποτέλεσμα να χάνεται μη αποθηκευμένος κώδικας.
- Για τη δημιουργία των schematics χρησιμοποιήσαμε το Quartus Prime/Questa αφού είχαμε ήδη οριστικοποιήσει τη λειτουργική σχεδίαση. Όμως κατά την σύνθεση (που επίσης είχε αρκετά crashes) παρατηρήσαμε ότι δεν ήταν βέλτιστη.
- Θα ήταν ωραίο να υπάρχει εύκολος τρόπος δημιουργίας των schematics στο στάδιο ανάπτυξης της λειτουργικής σχεδίασης.