Εργαστήριο Ψηφιακών Συστημάτων
Εργασία 2
Υλοποίηση UART

Ανδρικόπουλος Κωνσταντίνος

Περίληψη

Η εργασία έχει ως σκοπό την υλοποίηση ενός γενικού Ασύγχρονου Δέκτη Αποστολέα. Χωρίζεται σε 4 μέρη.
Στο **Α μέρος** υλοποιείται ο baud controller για τον συγχρονισμό της επικοινωνίας.
Στο **Β μέρος** υλοποιείται ο uart transmitter.
Στο **Γ μέρος** υλοποείται ο uart receiver.
Στο **Δ μέρος** υλοποιείται η συνολική μονάδα UART.

Μέρος Α: Υλοποίηση Baud Controller

Ο Baud Controller χρησιμοποιείται για να συμφωνηθεί ανάμεσα στον αποστολέα και τον δέκτη ένας κοινός ρυθμός επικοινωνίας ώστε να γίνεται χωρίς σφάλματα η αποστολή και η δειγματοληψία δεδομένων.
Ο baud controller έχει υλοποιηθεί για τις παρακάτω ταχύτητες:

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

Table 1: Επιλογές Baud Rate

Για να υλοποιήσω τον baud controller χρησιμοποίησα έναν μετρητή κύκλων ρολογιού ανάλογσα με το baud rate που επιλέγεται. Στην εργασία χρησιμοποιούμε ρολόι 100Mhz. Για να βρούμε μετά από πόσους κύκλους ρολογιού πρέπει να ενεργοποιείται το σήμα του controller πρέπει να κάνουμε την διαίρεση: 100.000.000/baud_rate. Όμως η διαίρεση αυτή για κάθε baud rate δεν είναι ακριβής, οπότε ο baud controller έχει σφάλμα σχετικά με τι συχνότητα θέλουμε να χει και τι έχει όντως.
Αυτό το σχετικό σφάλμα το υπολογίζουμε ως εξής:

$$\Sigma\varphi\lambda\mu\alpha = \frac{f_{real} - f_{baud}}{f_{baud}} \tag{1}$$

f_{real} είναι η ακέραια συχνότητα που χρησιμοποιεί ο baud controller αντί για την σωστή θεωρητικά που περιέχει δεκαδικά. Χρησιμοποιώντας την παραπάνω εξίσωση καταλήγουμε στα παρακάτω σφάλματα:

300:0.0001%
1200:0.0004%
4800:0.0015%
9600:0.0032%
19200:0.00635%
38400:0.00638%
57600:0.006388%
115200:0.00639%

Η μεταβλητή enable_baud χρησιμοποιείται για να συγχρονίζεται σωστά ο αποστολέας και ο δέκτης. Όταν βρίσκονται στην ενεργοποιημένη αλλά αδρανή κατάσταση και ενεργοποιούνται δεν θέλουμε να πετυχαίνουν στον bau controller στην μέση του μετρητή κύκλων.

Επίσης ο baud controller ενεργοποιείται σε συχνότητα $16 \times baud\_rate$.

Μέρος Β: Υλοποίηση UART Transmitter

Ο transmitter είναι υπεύθυνος για την αποστολή του συμβόλου.
Υλοποίησα τον transmitter ως μια moore fsm με βάση το παρακάτω διάγραμμα:

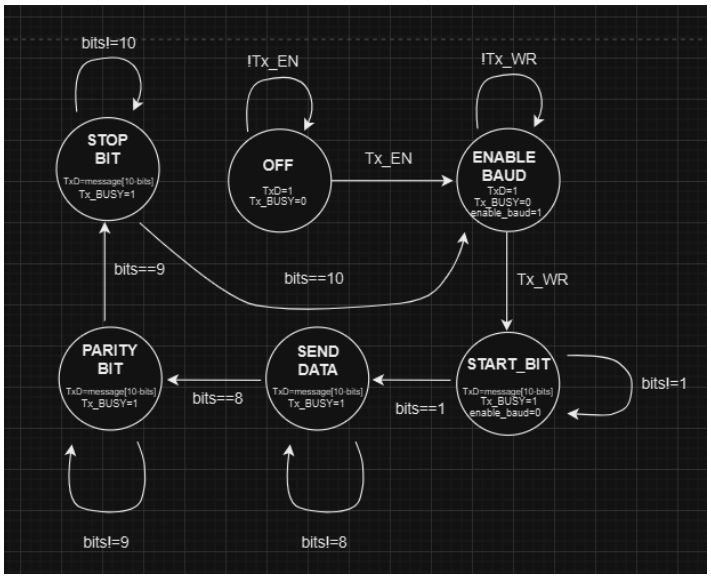

Figure 1: Transmitter FSM diagram

Ο Αποστολέας στέλνει bit δεδομένων σε συχνότητα baud_rate, δηλαδή περιμένει 16 ενεργοποιήσεις του baud controller. Για αυτό έχω υλοποιήσει σε μια always έναν μετρητή που μετράει 16 παλμούς του baud controller και αυξάνει μία μεταβλητή που μετράει πόσα bit έχουν αποσταλεί.

Περιγραφή της λειτουργίας του transmitter:
1)Στην OFF κατάσταση ο Αποστολέα είναι ανενεργός και μόλις λάβει τι σήμα Tx_EN ενεργοποιείται. Όσο το Tx_EN είναι ανενεργό ο Αποστολέας παραμένει στην OFF κατάσταση.
2)Η επόμενη κατάσταση είναι η ENABLE_BAUD στην οποία συγχρονίζεται ο baud controller με την μονάδα και δρα ως αδρανής αλλά ενεργή κατάσταση του Αποστολέα, δηλαδή είναι ενεργός αλλά δεν στέλνει δεδομένα. Αν το σήμα Tx_WR ο Αποστολέας μεταβαίνει στην επόμενη κατάσταση και στην αποστολή δεδομένων.
3)Στην κατάσταση START_BIT στέλνει 0 δηλώνοντας την αρχή της αποστολής. Αν ο μετρητής bits_counter είναι 1 μεταβαίνει ο Αποστολέας στην επόμενη κατάσταση.
4)Στην κατάσταση SEND_DATA στέλνει το κύριως μήνυμα. Περιμένει να στείλει 8 bits και μεταβαίνει στην επόμενη κατάσταση.
5)Στην κατάσταση PARITY_BIT στέλνει το parity bit (αριθμός άσσων στο μήνυμα). Αν στείλει το bit μεταβαίνει στην τελευταία κατάσταση.
6)Στην κατάσταση STOP_BIT στέλνει 1 για αν σηματοδοτήσει το τέλος του μηνύμαρος και αν μετρήσει 10ο bit ο bits_counter μεταβαίνει στην ENABLE_BAUD αδρανή κατάσταση.

Μέρος Γ: Υλοποίηση UART Receiver

Ο Receiver είναι υπεύθυνος για την σωστή ανάγνωση των δεδομένων.
Υλοποίησα τον receiver ως μια mealy fsm με βάση το παρακάτω διάγραμμα:

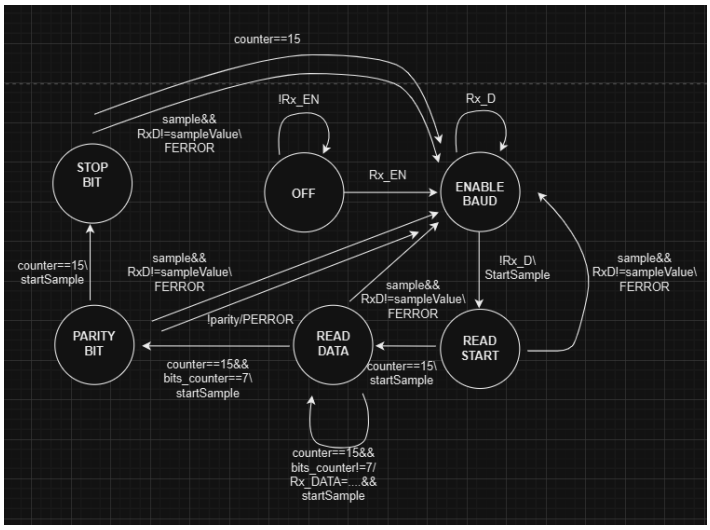

Figure 2: Receiver FSM diagram

Η δειγματοληψία των δεδομένων από τον receiver πραγματοποιείται με μια always η οποία μηδενίζει έναν μετρητή και ένα σήμα δειγματοληψίας όταν παίρνει το σήμα ότι πρέπει να ξεκινήσει την δειγματοληψία από την fsm. Ύστερα σε κάθε παλμό του baud controller αυξάνεται ο μετρητής. Η δειγματοληψία γίνεται 8 φορές για να είναι αξιόπιστη η ανάγνωση των δεδομένων, από την τιμή 4 του μετρητή εώς την τιμή 12. Στην αρχή της δειγματοληψίας ο Δέκτης κρατάει αυτήν την αρχική τιμή και σε κάθε στιγμή της δειγματοληψίας συγκρίνεται με την τιμή που βρίσκεται εκείνη την στιγμή. Αν αυτές οι δύο τιμές είναι διαφορετικές έχει γίνει λάθος στην δειγματοληψία και έχουμε framing error.

Περιγραφή της λειτουργίας του receiver:
1)Στην OFF κατάσταση ο Δέκτης είναι ανενεργός. Όταν είναι ενεργό το σήμα Rx_EN ενεργοποιείται ο Δέκτης και μεταβαίνει στην επόμενη κατάσταση.
2)Η επόμενη κατάσταση είναι η ENABLE_BAUD στην οποία συγχρονίζεται ο baud controller με την μονάδα και δρα ως αδρανής αλλά ενεργή κατάσταση του Δέκτη, δηλαδή είναι ενεργός αλλά δεν διαβάζει δεδομένα. Όταν ο Αποστολέας δεν στέλνει δεδομένα η έξοδός του είναι 1 και το start bit είναι 0. Άρα ο Δέκτης καταλαβαίνει ότι έχει έρθει το start bit και πρέπει να αρχίσει να διαβάζει δεδομένα όταν η είσοδός του RxD μεταβαίνει από 1 σε 0, οπότε και μεταβαίνει στην επόμενη κατάσταση.
3)Στην READ_START ο Δέκτης διαβάζει το start bit, η δειγματοληψία γίνεται με τον τρόπο που περιγράφεται παραπάνω. Αν ο μετρητής της δειγματοληψίας μετρήσει 16 φορές σημαίνει ότι αλλάζει το bit Και μεταβαίνει στην επόμενη κατάσταση.
4)Στην READ_DATA ο Δέκτης διαβάζει τα δεδομένα, η δειγματοληψία γίνεται με τον τρόπο που περιγράφεται παραπάνω. Επίσης, αυτή η κατάσταση έχει έναν μετρητή που μετράει πόσα bit από τα δεδομένα έχουν σταλεί. Αν ο μετρητής της δειγματοληψίας μετρήσει 16 φορές και ο μετρητής δεδομένων μετρήσει 8 bits σημαίνει ότι τα δεδομένα έχουν διαβαστεί σωστά και μεταβαίναι στην επόμενη κατάσταση.
5)Στην READ_PARITY ο Δέκτης διαβάζει το parity bit, η δειγματοληψία γίνεται με τον τρόπο που περιγράφεται παραπάνω.Αν το parity bit δεν αντιστοιχεί με τον αριθμό άσσων που έχει σταλεί τότε βγαίνει στην έξοδο το σήμα PERROR και ο Δέκτης μεταβαίνει στην αδρανή κατάσταση. Αν ο μετρητής της δειγματοληψίας μετρήσει 16 φορές σημαίνει ότι αλλάζει το bit Και μεταβαίνει στην επόμενη κατάσταση.
3)Στην READ_STOP ο Δέκτης διαβάζει το stop bit, η δειγματοληψία γίνεται με τον τρόπο που περιγράφεται παραπάνω. Αν ο μετρητής της δειγματοληψίας μετρήσει 16 φορές σημαίνει ότι η ανάγνωση των δεδομένων έχει γίνει σωστά και ο Δέκτης μεταβαίνει στην αδρανή κατάσταση περιμένοντας νέα δεδομένα από τον Αποστολέα.

Επίσης, χρησιμοποιώ μια always η οποία κρατάει την τιμή σημάτων και διανυσμάτων που χρειάζεται να διατηρηθεί η τιμή τους. Αν διατηρούσα αυτές τιμές μες την FSM δημιουργόντουσαν latches στην σύνθεση.

Μέρος Δ: Υλοποίηση UART

Για την τελική υλοποίηση της UART μονάδας ένωσα τον transmitter και receiver σε ένα top module. Η έξοδος TxD του Αποστολέα και η είσοδος RxD του Δέκτη συνδέθηκαν μεταξύ τους.

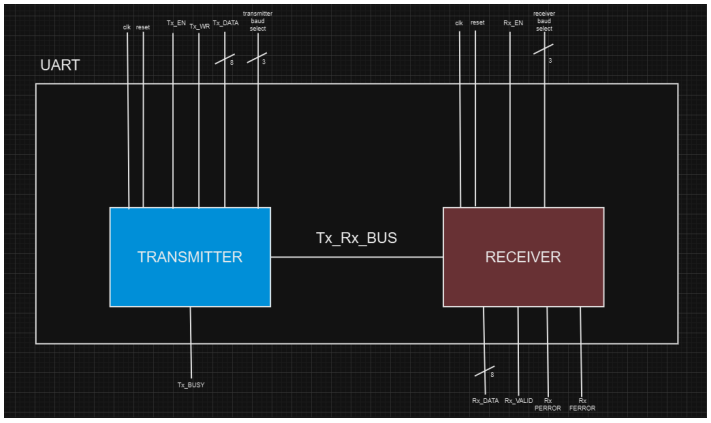

Figure 3: UART top-level diagram

Για να ελέγξω την συνολική λειτουργία της μονάδας υλοποίησα στο testbench μια απλή FSM. Στην αρχή ενεργοποιεί τον Δέκτη και τον Αποστολέα και τους αντίστοιχους baud controller τους. Ύστερα στέλνει τα 4 σύμβολα που προτείνονται στην εκφώνηση. Μετά ελέγχει την λειτουργία του PERROR και τελικά την λειτουργία του FERROR. Σε κάθε κατάσταση της FSM τυπώνονται αντίστοιχα μηνύματα για την επαλήθευση και αυτοματοποίηση του testbench.