Serializer/Deserializer

Στεφανίδης Απόστολος 56447

Περιγραφή:

Ο SerDes (Serializer/Deserializer) είναι ένα κύκλωμα το οποίο μετατρέπει τα παράλληλα δεδομένα σε σειριακά και το αντίστροφο. Χρησιμοποιείται σε συστήματα υψηλής ταχύτητας και έχει ως σκοπό την μείωση των γραμμών μεταφοράς δεδομένων, μετατρέποντάς τα από παράλληλα σε σειριακά.

Ο Serdes αποτελείται από 2 blocks, το parallel to serial (PISO) και το Serial to Parallel (SIPO).

Το PISO block συνήθως αποτελείται από πολλαπλές γραμμές εισόδου, από ένα parallel clock, ένα PLL που δίνει το serial clock και ένα shift register. Ο shift register δέχεται τα δεδομένα εισόδου σε κάθε parallel clock, και τα κάνει shift στην έξοδο σε κάθε serial clock, με το serial clock να έχει υψηλότερη συχνότητα από το parallel.

Το SIPO block διαθέτει τα ίδια στοιχεία, απλά έχει πολλαπλές γραμμές στην έξοδο. Επίσης διαθέτει ένα clock receive line, στο οποίο δέχεται το clock των δεδομένων εισόδου (το προηγούμενο serial clock), το οποίο ανακτά από τα δεδομένα. Στη συνέχεια από το serial clock υπολογίζει το parallel, το οποίο δίνει στην έξοδο.

Υπάρχουν 4 αρχιτεκτονικές SerDes:

Parallel clock:

Αυτή η αρχιτεκτονική χρησιμοποιείται για μεγάλο εύρος δεδομένων. Αντί να γίνουν serialize όλα τα δεδομένα, η κάθε γραμμή αναλαμβάνει ένα διαφορετικό τμήμα των δεδομένων, και τα πολλαπλά σειριακά δεδομένα αποστέλλονται παράλληλα στο δέκτη μαζί με το clock. Για σωστή ανάκτηση των δεδομένων στο δέκτη, τα δεδομένα πρέπει να αποστέλλονται ταυτόχρονα (μικρό skew). Τα πλεονεκτήματά του είναι το χαμηλό κόστος, και το ότι μπορεί να επεξεργαστεί μεγάλο εύρος δεδομένων, ενώ το μειονέκτημα είναι το ότι απαιτεί υψηλή καλωδίωση, εφόσον έχει πολλά ζεύγη PISO/SIPO.

Embedded clock:

Σε αυτή την αρχιτεκτονική, όλα τα δεδομένα γίνονται serialize σε ένα stream, μαζί με το clock. Σε κάθε μετάδοση ενσωματώνονται και 2 clock bits, ένα high και ένα low. Αυτό επιτρέπει υψηλότερο jitter στο clock, καθώς ανακτάται κατευθείαν από τα δεδομένα, και διευκολύνει την ανάκτηση των δεδομένων στο δέκτη, καθώς λειτουργεί ως start/stop bit. Η ύπαρξη του clock bit δημιουργεί μια περιοδική rising edge στα δεδομένα, την οποία αναζητάει ο δέκτης. Όταν την εντοπίσει κλειδώνει στα δεδομένα και ξεκινάει το deserialize. Το βασικό πλεονέκτημα είναι το χαμηλό κόστος του clock, εφόσον το clock jitter δεν

χρειάζεται να είναι αυστηρά μικρό, ενώ το μειονέκτημα είναι το ότι δεν έχει κάποιον τρόπο αποτροπής δημιουργίας DC συνιστώσας.

8/10 bit:

Ο serializer, πριν αρχίσει να μεταδίδει τα bit, μετατρέπει τα 8 bit σε 10 bit, χρησιμοποιώντας μια κωδικοποίηση η οποία εξασφαλίζει πολλαπλές μεταβάσεις 0-1, άρα εύκολη ανάκτηση clock στο δέκτη, και αποφεύγει μεγάλες αλληλουχίες 0 και 1, άρα δεν δημιουργεί DC συνιστώσα. Επίσης μεταδίδεται ένας ειδικός χαρακτήρας ο οποίος δηλώνει την αρχή και το τέλος των δεδομένων, και δεν μπορεί να εμφανιστεί μέσα στα δεδομένα με τη συγκεκριμένη κωδικοποίηση, οπότε εξασφαλίζεται και εύκολο data recovery. Ο δέκτης λαμβάνει τα δεδομένα, και εάν δεν εντοπίσει κάποιο σφάλμα τα αποκωδικοποιεί πίσω σε 8 bit. Το βασικό μειονέκτημα είναι ότι σε περίπτωση που το μήκος των δεδομένων δεν είναι ακέραιο πολλαπλάσιο του 8 (byte) τότε χρειάζεται επιπλέον επεξεργασία για να λειτουργήσει σωστά.

Bit Interleaving:

Σε αυτή την περίπτωση, δεν κάνουμε serialize παράλληλα δεδομένα, αλλά πολυπλέκουμε διαφορετικά δεδομένα από αργές γραμμές σε μια ταχύτερη γραμμή. Τα δεδομένα αυτά μπορεί να έχουν οποιαδήποτε κωδικοποίηση. Ο βασικός περιορισμός είναι ότι απαιτείται πολύ καλός συγχρονισμός (μικρό clock jitter), οπότε το κόστος είναι υψηλό.

Υλοποίηση:

Στην υλοποίηση έχουν γίνει κάποιες απλοποιήσεις:

- Δεν λαμβάνουμε υπόψη το clock and data recovery (cdr). Το clock recovery σε fpga με vhdl γενικά είναι αρκετά περίπλοκο, ενώ αντί για data recovery με κάποια κωδικοποίηση, στέλνουμε μαζί με τα δεδομένα ένα valid_in σήμα.
- Υπάρχει ένα clock. Σε ένα πραγματικό σύστημα θα δημιουργούταν ακόμα ένα με χρήση PLL, αλλά και πάλι θα ήταν αρκετά περίπλοκο στην υλοποίηση.
- Δεν υπάρχει κάποια μέθοδος error detection

Θεωρούμε ότι τα data lines είναι των 8 bit.

Σε αυτό το repository υπάρχουν τα αρχεία vhd.

serializer.vhd

Αυτό το κύκλωμα είναι υπεύθυνο για τη μετατροπή των παράλληλων δεδομένων σε σειριακά. Δέχεται το clock, reset, data_in και valid_in για να δηλώσει ότι τα δεδομένα που έχει στην είσοδο είναι έτοιμα για αποστολή. Δίνει στην έξοδο τα σειριακά δεδομένα, το valid_out για να δηλώσει ότι τα δεδομένα εξόδου είναι σωστά και το ready, για να δηλώσει ότι είναι έτοιμο να λάβει νέα δεδομένα.

Το κύκλωμα ουσιαστικά αποτελείται από ένα shift register και έναν counter. Αρχικά το σύστημα βρίσκεται σε idle state. Όταν το valid_in γίνει ίσο με 1 τότε περνάει στη shift state, όπου περνάει τα data_in στο shift register, και σε κάθε κύκλο τα μετακινεί κατά μία θέση δεξιά για 7 φορές. Η έξοδος είναι πάντα το LSB του shift register, και κατά τη διάρκεια της διαδικασίας τα data_out είναι valid. Μόλις ολοκληρωθεί η διαδικασία, τότε το κύκλωμα είναι έτοιμο να δεχθεί νέα data_in. Περισσότερες λεπτομέρειες για τον κώδικα φαίνονται στα σχόλια.

deserializer.vhd

Αποτελείται επίσης από έναν counter και έναν shift register, μόνο που κάνει την ανάποδη διεργασία. Όσο είναι idle θεωρούμε ότι τα δεδομένα του είναι valid (καθώς είναι τα προηγούμενα που του δόθηκαν). Μόλις δεχθεί valid είσοδο ο shift register τοποθετεί το νέο bit στην MSB θέση, και μετακινεί προς τα δεξιά όλα τα υπόλοιπα. Μόλις αυτό γίνει 8 φορές, τα data_out γίνονται valid.

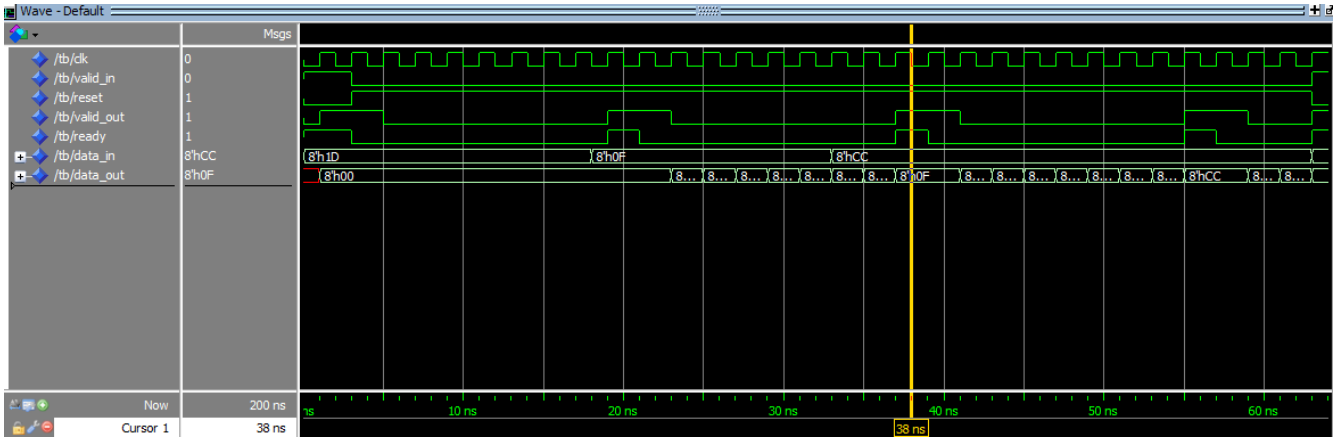
top.vhd

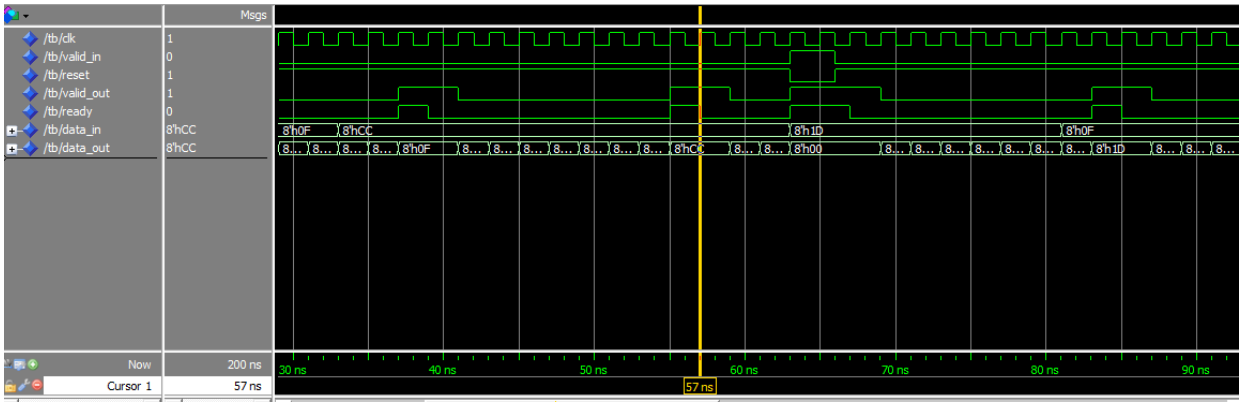
Είναι το top level αρχείο που ουσιαστικά ενώνει τα 2 κυκλώματα. Όλες οι είσοδοι πάνε στον serializer, εκτός από το clock και reset που πάν και στα 2, ενώ όλες οι έξοδοι προέρχονται από τον deserializer, εκτός από το ready. Ο deserializer παίρνει ως data_in τα data_out του serializer, και ως valid_in το vaid_out του serializer.

tb.vhd

Το testbench του κυκλώματος, δίνει τιμές στο clock και σε όλες τις εισόδους, ώστε να παρατηρήσουμε την έξοδο.

Παρακάτω φαίνονται screenshots από την προσομοίωση του tb.vhd





Όπως φαίνεται, όταν το valid_out είναι 1, τα data_out είναι ίσα με τα data_in που δέχθηκε το κύκλωμα στους προηγούμενους κύκλους.

Πηγές:

https://en.wikipedia.org/wiki/SerDes

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.173.4371&rep=rep1&type=pdf