Πανεπιστήμιο Ιωαννίνων

Τμήμα Μηχανικών Ηλεκτρονικών
Υπολογιστών & Πληροφορικής

Δοκιμή και Αξιοπιστία Ηλεκτρονικών Συστημάτων

Εαρινό Εξάμηνο 2024

JTAG 1149.1

Χρήστος Δημητρέσης 4351 cs04351@uoi.gr

Περιεχόμενα

Δομικά Στοιχεία JTAG 1149.1

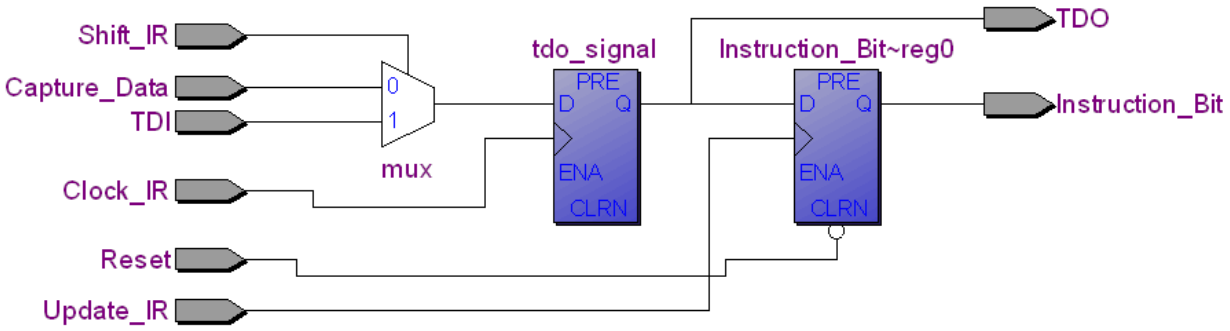
Παρακάτω παρουσιάζονται τα δομικά στοιχεία για την υλοποίηση μιας αλυσίδας Boundary Scan σύμφωνα με το πρότυπο **JTAG 1149.1** . Λόγω της απλότητας των δομικών στοιχείων η επαλήθευση της ορθότητας τους γίνεται με σύνθεση τους στο Quartus και οπτικοποίηση τους σε επίπεδο RTL.

** Τα αρχεία κώδικα σε γλώσσα VHDL που υλοποιούν τα παρακατω δομικά στοιχεία συνοδεύουν την αναφορά στο αρχείο παράδοσης.

** Τα δομικά στοιχεία έχουν υλοποιηθεί απο τη αρχη σε σχέση με την άσκηση 3 ενώ έχουν χρησιμοποιηθεί και πιο κατανοητα ονόματα σημάτων.

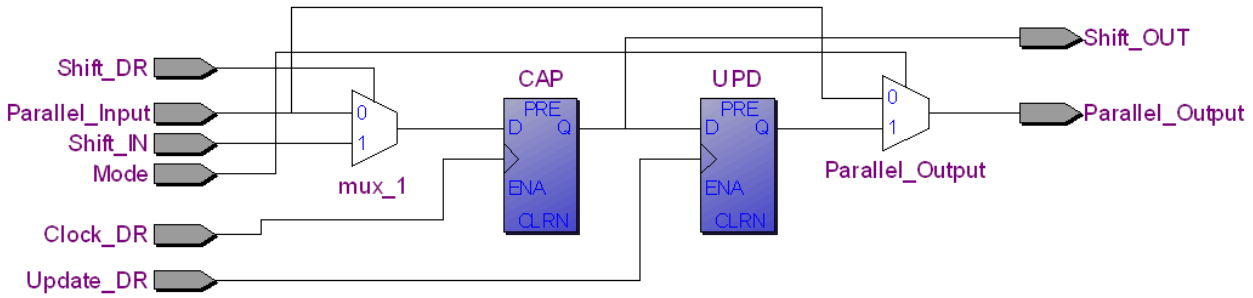
Instruction Register (IR)

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Instruction Register (IR): ***IR_REG_JTAG.vhdl***



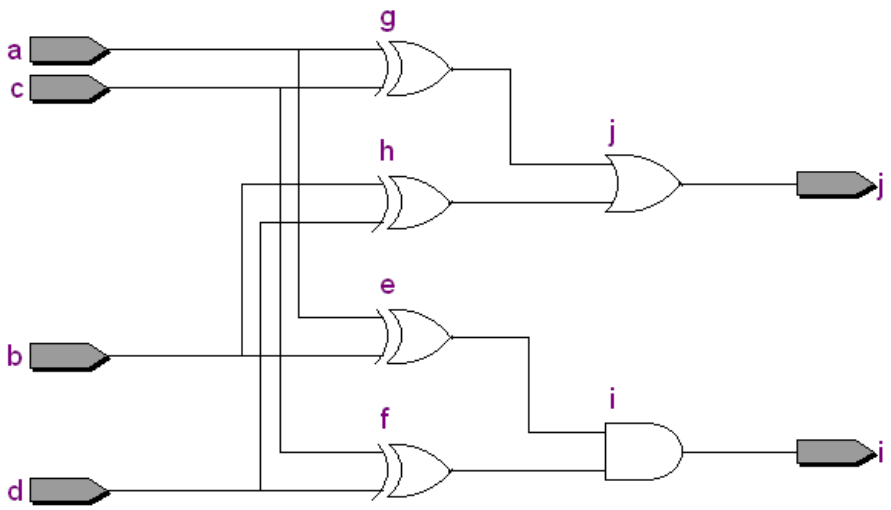
Boundary Register Cell (BRC)

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Boundary Register Cell (BRC):
BR_CELL_JTAG.vhdl



Circuit Under Test (CUT)

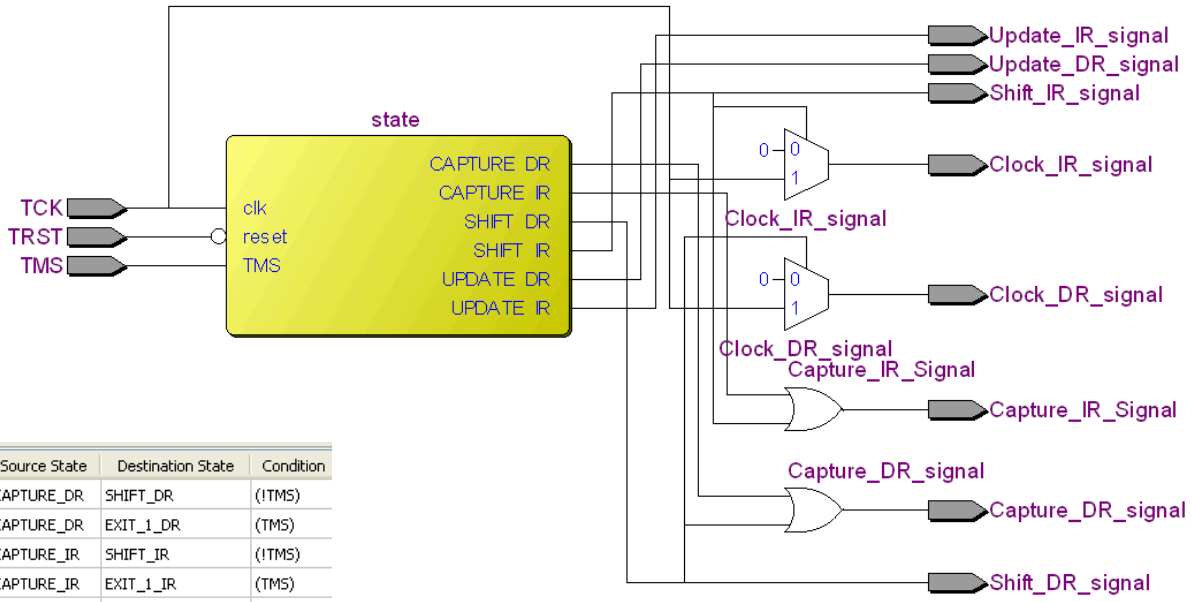
Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Circuit Under Test (CUT):
Circuit_Under_Test.vhdl



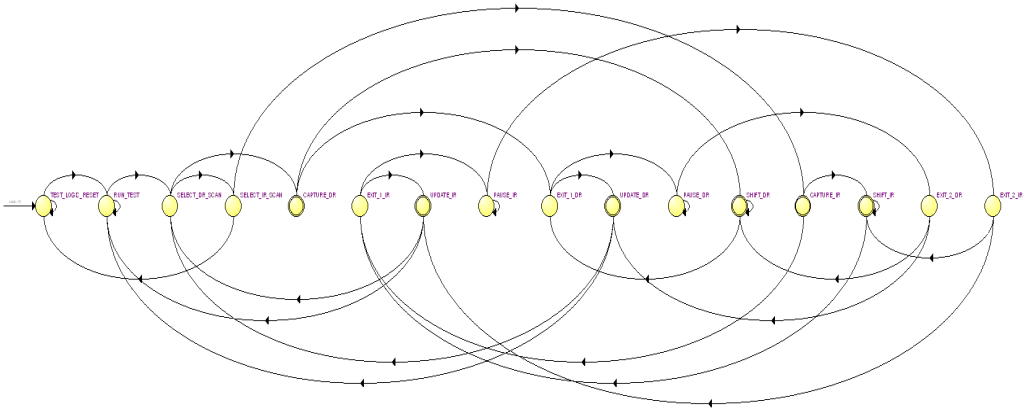
Tap Controller FSM (TAP)

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Tap Controller FSM (TAP):
tap_controler_FSM.vhdl

** Είναι αναγκαίο και το αρχείο **Declare.vhdl** διοτί μέσα σε αυτό ορίζεται ο τύπος **state,** με την χρήση του οποίου αντιστοιχίζουμε ένα label σε φυσική γλώσσα για κάθε κατάσταση του FSM.



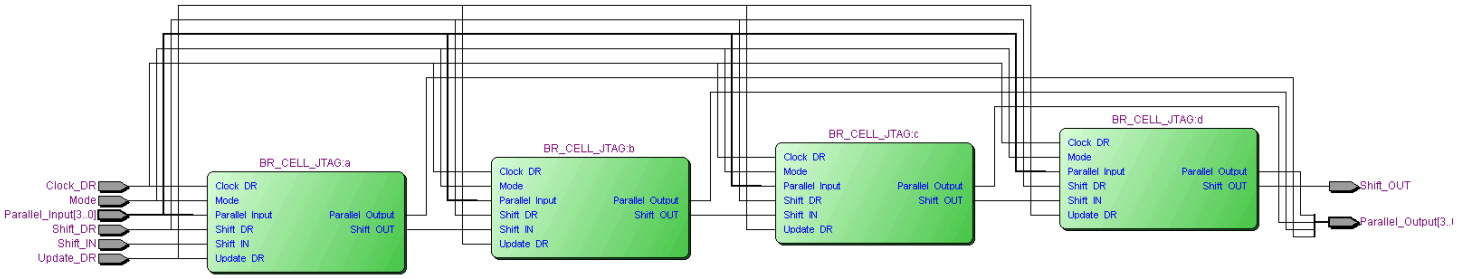
	Source State	Destination State	Condition
1	CAPTURE_DR	SHIFT_DR	(!TMS)
2	CAPTURE_DR	EXIT_1_DR	(TMS)
3	CAPTURE_IR	SHIFT_IR	(!TMS)
4	CAPTURE_IR	EXIT_1_IR	(TMS)
5	EXIT_1_DR	UPDATE_DR	(TMS)
6	EXIT_1_DR	PAUSE_DR	(!TMS)
7	EXIT_1_IR	UPDATE_IR	(TMS)
8	EXIT_1_IR	PAUSE_IR	(!TMS)
9	EXIT_2_DR	UPDATE_DR	(TMS)
10	EXIT_2_DR	SHIFT_DR	(!TMS)
11	EXIT_2_IR	UPDATE_IR	(TMS)
12	EXIT_2_IR	SHIFT_IR	(!TMS)
13	PAUSE_DR	PAUSE_DR	(!TMS)
14	PAUSE_DR	EXIT_2_DR	(TMS)
15	PAUSE_IR	PAUSE_IR	(!TMS)
16	PAUSE_IR	EXIT_2_IR	(TMS)
17	RUN_TEST	SELECT_DR_SCAN	(TMS)
18	RUN_TEST	RUN_TEST	(!TMS)
19	SELECT_DR_...	SELECT_IR_SCAN	(TMS)
20	SELECT_DR_...	CAPTURE_DR	(!TMS)
21	SELECT_IR_...	TEST_LOGIC_RESET	(TMS)
22	SELECT_IR_...	CAPTURE_IR	(!TMS)
23	SHIFT_DR	SHIFT_DR	(!TMS)
24	SHIFT_DR	EXIT_1_DR	(TMS)
25	SHIFT_IR	SHIFT_IR	(!TMS)
26	SHIFT_IR	EXIT_1_IR	(TMS)
27	TEST_LOGIC...	TEST_LOGIC_RESET	(TMS)
28	TEST_LOGIC...	RUN_TEST	(!TMS)
29	UPDATE_DR	SELECT_DR_SCAN	(TMS)
30	UPDATE_DR	RUN_TEST	(!TMS)
31	UPDATE_IR	SELECT_DR_SCAN	(TMS)
32	UPDATE_IR	RUN_TEST	(!TMS)

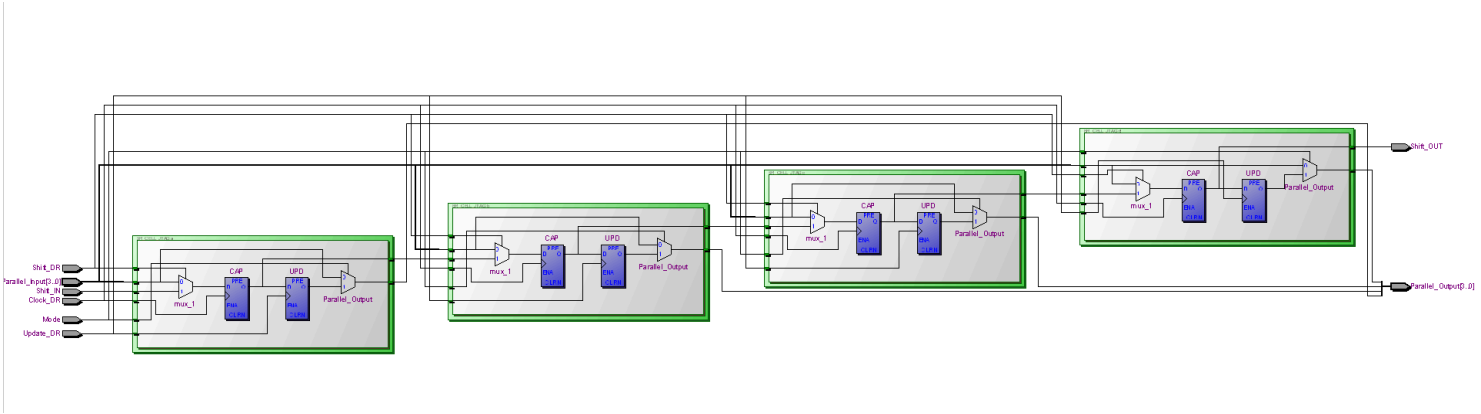


Boundary Register Cells 4 Bits (BRC_4BITS)

Ο καταχωρητής συνδεδεμένος με τις εισόδους του CUT δημιουργημένος απο 4 κύτταρα BRC συνδεδεμένα κατάλληλα.

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Boundary Register Cells 4 Bits (BRC_4BITS): ***BRCell_4bits_JTAG.vhdl***

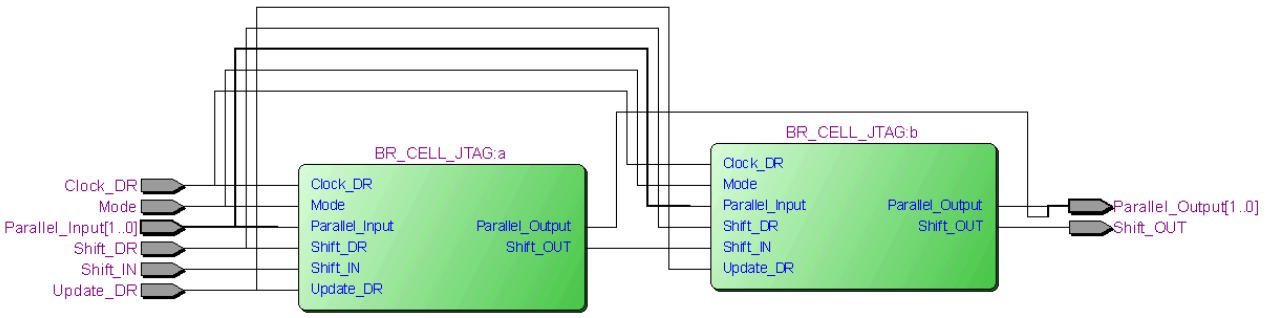


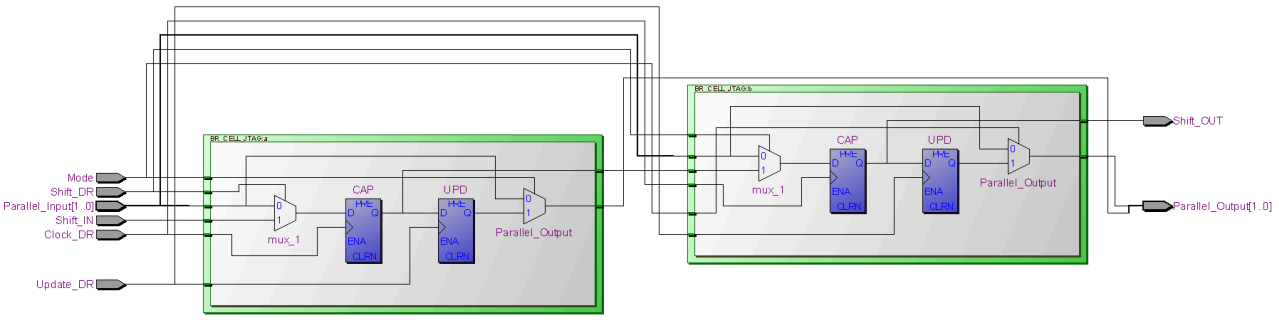


Boundary Register Cells 2 Bits (BRC_2BITS)

Ο καταχωρητής συνδεδεμένος με τις εξόδους του CUT δημιουργημένος απο 2 κύτταρα BRC συνδεδεμένα κατάλληλα.

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Boundary Register Cells 2 Bits (BRC_2BITS): ***BRCell_2bits_JTAG.vhdl***





Instruction Register 2 BITS (IR_2BITS)

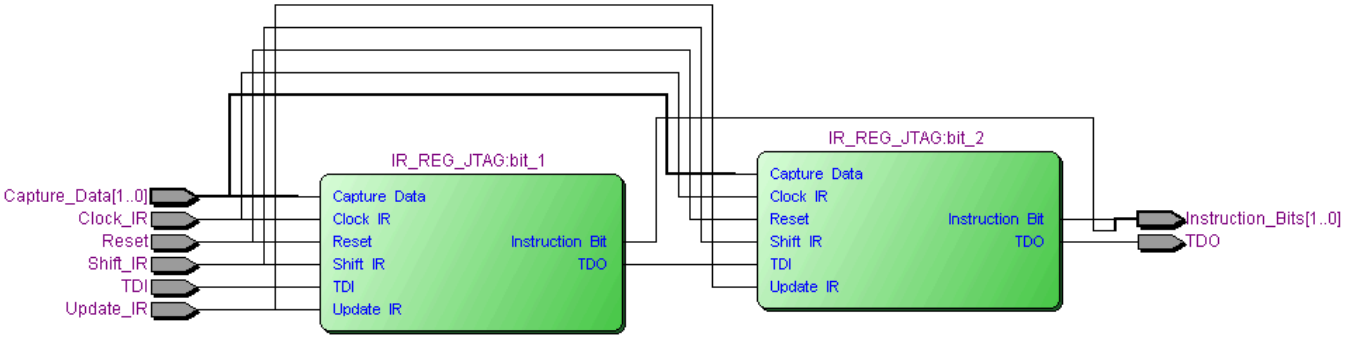
Ο καταχωρητής εντολών που θα χρησιμοποιήσουμε στην υλοποίηση μας αποτελείται από 2 καταχωρητες του δομικού στοιχείου **IR** που παρουσιαστηκε προηγουμένως. 2 bits είναι το ελάχιστο μέγεθος καταχωρητή εντολών που μπορούμε να χρησιμοποιήσουμε προκειμένου να υλοποιηθούν οι 3 υποχρεωτικές εντολές που επιβάλλει το πρότυπο JTAG 1149.1.

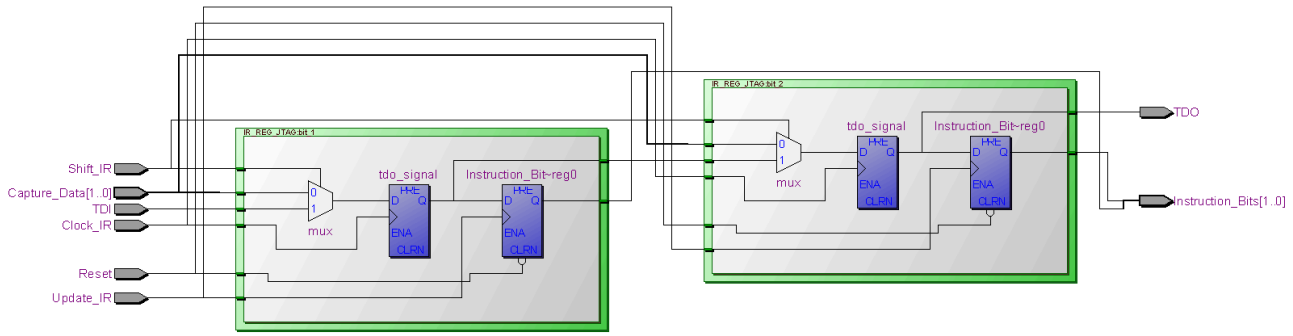
Συγκεκριμένα οι 3εις αυτές εντολές είναι οι εξής :

1) BYPASS
2) SAMPLE/PRELOAD
3) EXTEST

και παραδείγματα εκτέλεσης τους στην υλοποίηση μας θα παρουσιαστούν στην συνέχεια.

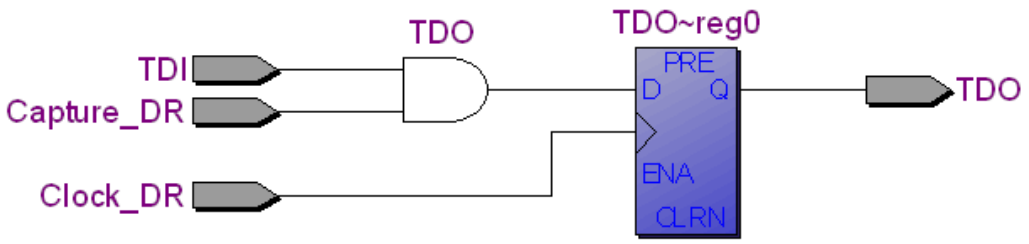
Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Instruction Register 2 BITS (IR_2BITS):
IR_REG_2bits_JTAG.vhdl





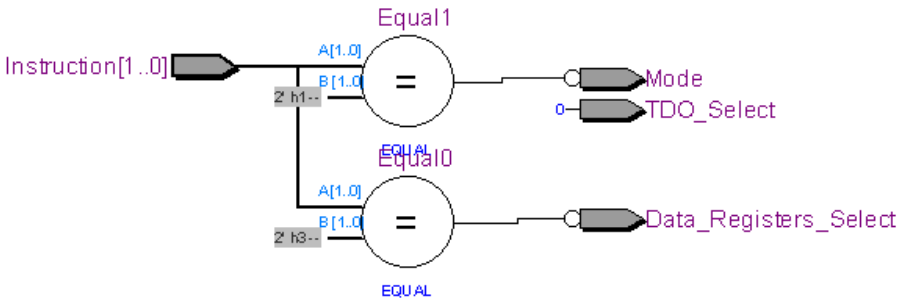
Bypass Register (BYPASS_REG)

Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Bypass Register (BYPASS_REG):
Bypass_REG_JTAG.vhdl

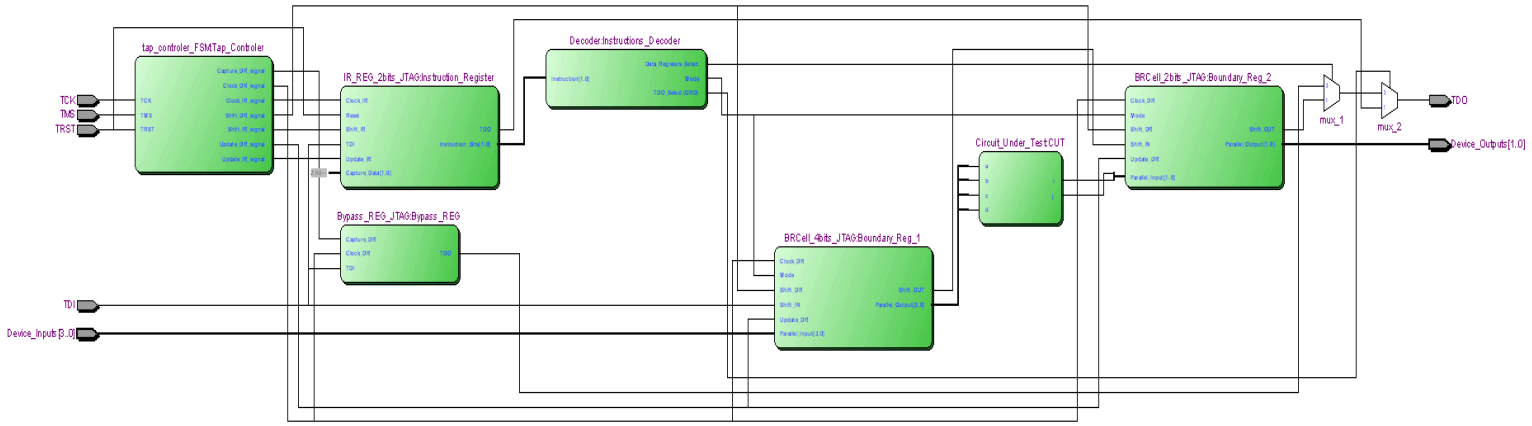


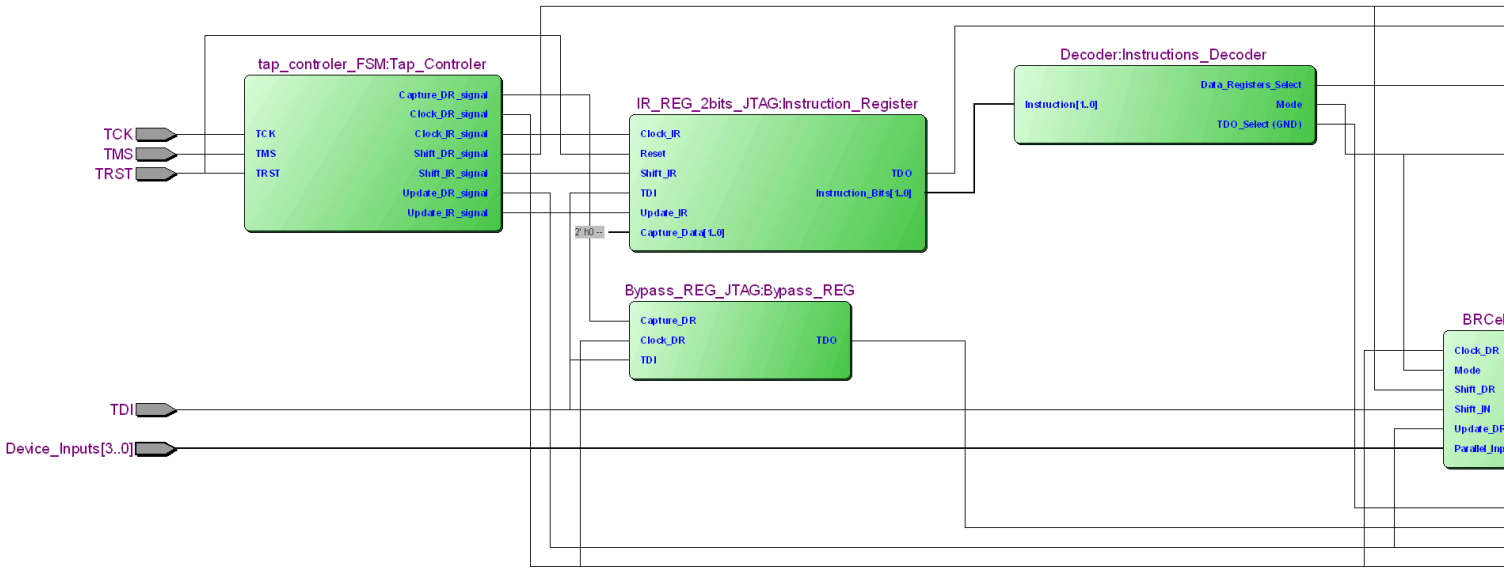
Decoder (DEC)

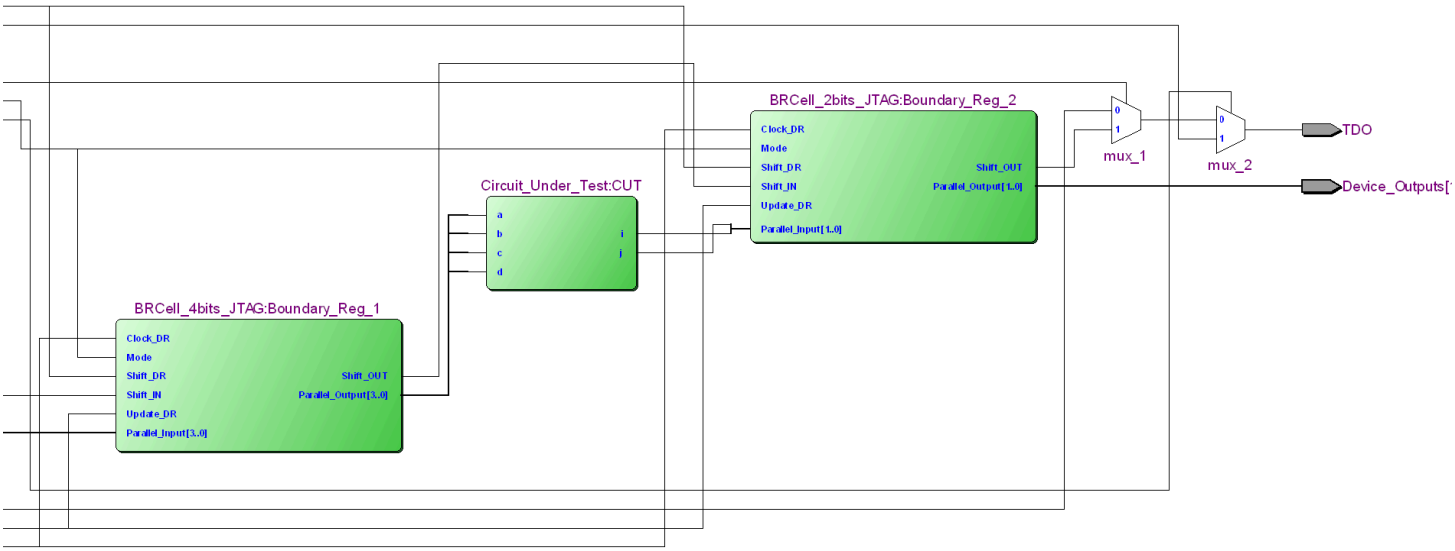
Όνομα αρχείου που υλοποιείται το δομικό στοιχείο Bypass Register Decoder (DEC): ***Decoder.vhdl***



Σύνδεση Δομικών Στοιχείων - Υλοποίηση JTAG



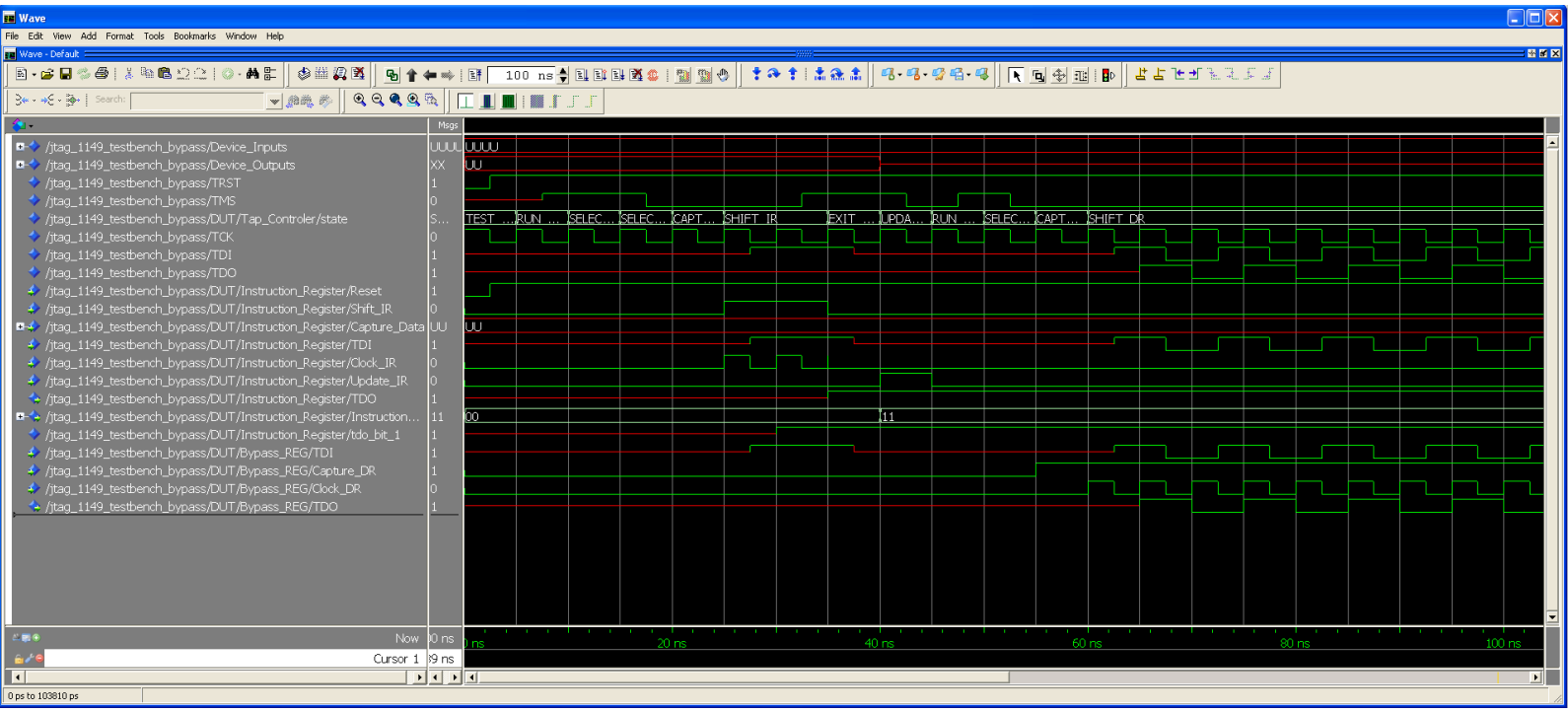


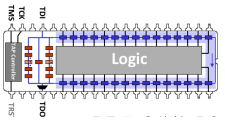


Όνομα αρχείου στο οποίο υλοποιείται το JTAG 1149.1 συνδεόντας κατάλληλα μεταξύ τους τα δομικά στοιχεία που αναφέρθηκαν προηγουμένως : ***JTAG_1149.vhdl***

Υλοποίηση Υποχρεωτικών Εντολών του προτύπου

BYPASS





Επεξήγηση TestBench

Η προσομοίωση ξεκινάει θέτωντας το σήμα TRST στο λογικό 0 για χρόνο 2,5 ns.

Λόγω αυτού αρχικοποιείται η μηχανή καταστάσεων του Tap Controller στην κατάσταση **TEST_LOGIC_RESET** ενώ όλα τα δομικά στοιχεία που έχουν καταχωρητες αρχικοποιούνται και αυτά στο μηδεν διότι στην υλοποίηση μας έχουμε προσθέση το σήμα RESET στα IR , BRC.

Λόγω της παραπάνω αρχικοποίησης βλέπουμε τον καταχωρητή εντολών να περιέχει την τιμή “00” στην αρχή της προσομοίωσης αν και δεν του έχουμε φορτώσει εντολή ακόμα.

Στην συνέχεια φορτώνουμε την εντολή για την λειτουργία του bypass , δηλαδή τα bits “11” μέσω της σειριακής εισόδου TDI του JTAG με κατάλληλο τρόπο.

Η εντολή φορτώνεται σύμφωνα με το πρότυπο, διατρέχοντας τις καταστάσεις στον TAP_CONTROLER.

Η εντολή οπως φαίνεται και στην προσομοίωση έχει φορτωθεί την χρονική στιγμή t = 40ns.

Με την φορτωση της εντολής και λόγω του Decoder οι πολυπλέκτες που συνδέουν την έξοδο του bypass register με την έξοδο TDO του JTAG , έχουν φτιάξει το επιθυμητό μονοπάτι εξόδου.

Αφου φορτώσαμε την εντολή διατρέχουμε πάλι κατάλληλα τις καταστάσεις του TAP controller προκειμένου να μπουμε σε διαδικασία εκτέλεσης της.

Φτάνοντας στην κατάσταση CAPTURE_DR του TAP_CONTROLER το σήμα Capture_DR_Signal τίθεται στο λογικό ‘1’ πράγμα αναγκαίο έτσι ώστε στην AND πυλή του bypass reg να περνάει μέσα οτιδηποτε τιμή έχει το TDI.

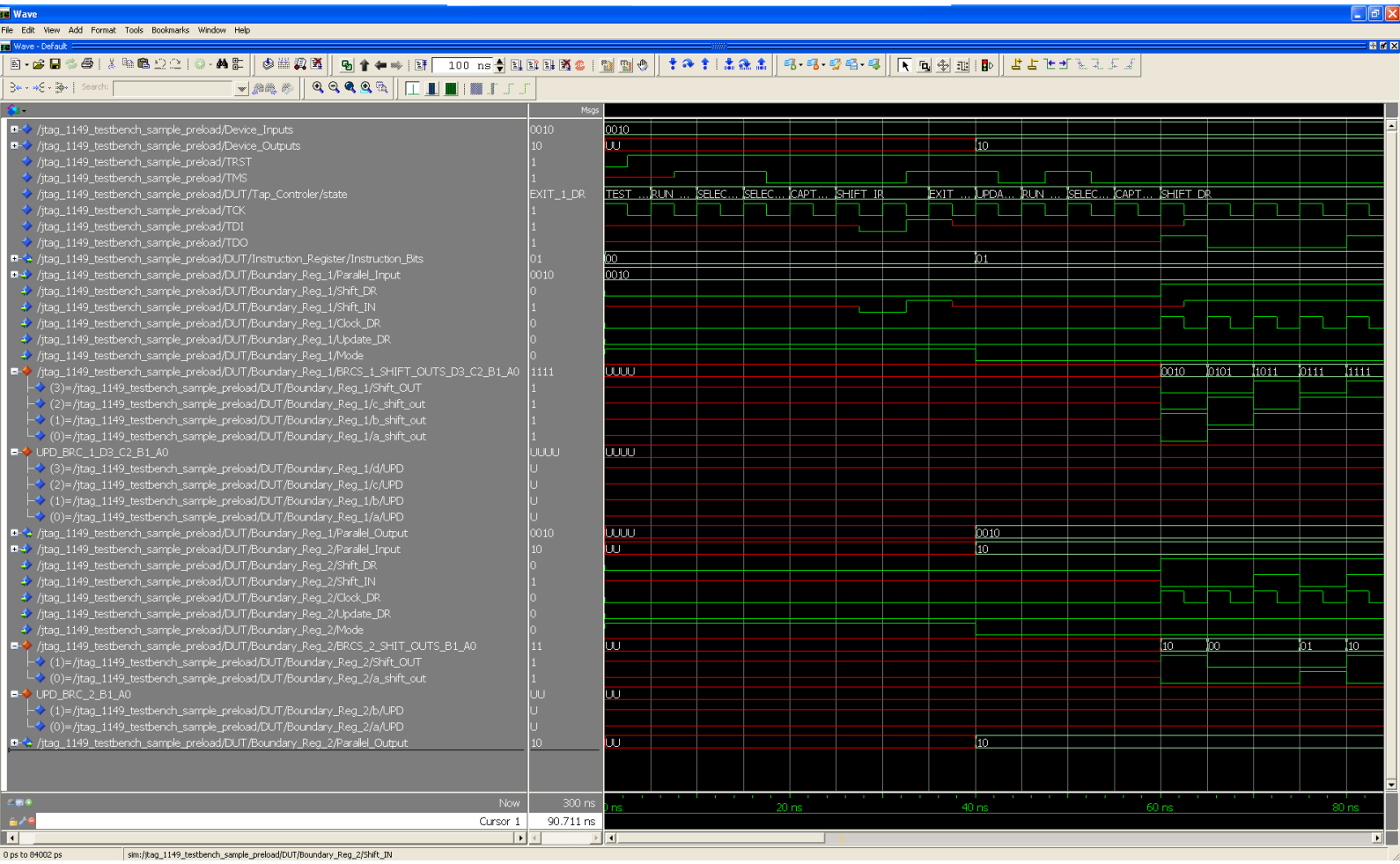
Στην συνέχεια μεταβαίνοντας στην κατάσταση SHIFT_DR και παραμένοντας εκει, το CLOKC_DR, να ακολουθει το TCK , ΤΟ Capture_DR παραμένει στο λογικό ‘1’ και άρα περνάει μέσα στον BYPASS REG η τιμή του TDI και τέλος έχουμε δημιουργήσει το καταλληλο μονοπάτι από τη έξοδο του bypassreg μέχρι την έξοδο TDO του JTAG.

Αρα όσο βρισκόμαστε στην κατάσταση shift σε κάθε θετική ακμή του TCK (το οποίο ισούται ΜΕ CLOCK_DR σε εκεινη την κατάσταση) η έξοδος TDO θα ακολουθεί την είσοδο TDI

Το παραπάνω φαίνεται και στην εξομοίωση μας απο την χρονική στιγμή 65 ns και μετά.

Sample/Preload

Επεξήγηση Testbench



- Φορτώνουμε τον κωδικό “10” ο οποίος αντιστοιχεί στην κωδικοποίηση της εντολής SAMPLE/PRELOAD στον INSTRUCTION REGISTER ακολουθόντας ακριβώς τα ίδια βήματα που έγιναν για την φόρτωση της εντολής bypass. Επειδή είναι ακριβώς η ίδια διαδικασία και για μείωση της πολυπλοκότητας έχουν παραληφθεί τα εσωτερικά σήματα του Instruction Register και έχει κρατηθεί να εμφανίζεται μόνο το σήμα που δείχνει τα bits που έχει αποθηκευμένα ο Instruction Register μέσα του , προκειμένου να επιβεβαιώσουμε ότι φορτώθηκε όρθα η εντολή.

- Την χρονική στιγμή t = 40 ns έχει φορτωθεί η εντολή επιτυχώς.

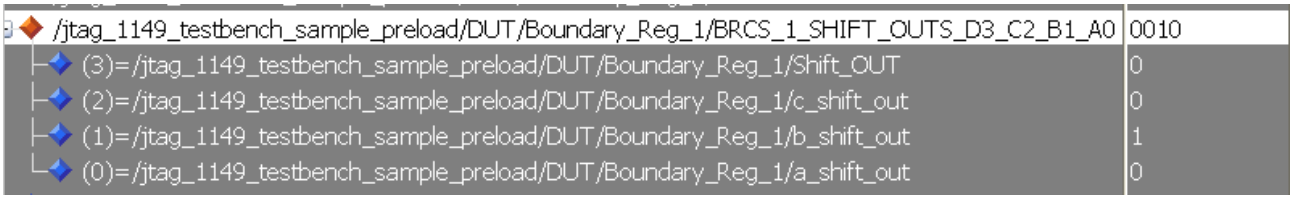
- Στην συνέχεια αφού έχει φορτωθεί όρθα η εντολή, διατρέχουμε τις καταστάσεις του Tap_Controler κατάλληλα προκειμένου να φτάσουμε στην κατάσταση Capture_DR και να εκτελέσουμε την λειτουργία Sample.

- Τονίζεται ότι έχουμε φροντίσει στην αρχή της προσομοίωσης η παράλληλη φόρτωση του JTAG να έχει φορτωθεί με το διάνυσμα “0010” πραγμα που φαίνεται και στη προσομοίωση. Παράλληλη φόρτωση JTAG = DEVICE_INPUTS

- Την χρονική στιγμή t = 60 ns κάνουμε Capture στον boundary_Reg_1 τις τιμές που δέχεται ως είσοδο (από το device_inputs του JTAG) το CUT και στον boundary_Reg_2 κάνουμε capture τις αποκρίσεις του CUT στις τιμες εισόδου αυτες.

- Η απόκριση του CUT που αποθηκεύεται στον Boundary_Reg_Cell_2 είναι η αναμενόμενη δεδομένου του διανύσματος εισόδου.

- Η ορθότητα της διαδικασίας επιβεβαιώνεται απο τα σήματα



και

Τα εξής αντιστοιχούν στα shift_outs του κάθε κυττάρου Boundary_Cell που περιέχονται στον Boundary_Reg_1 και Boundary_Reg_2 αντίστοιχα.

Όπως φαίνεται και στα στιγμιότυπα , έχουν ομαδοποιηθεί **αναποδα** απο την σειρα που βρίσκονται στην πραγματικότητα , δηλαδή ανάποδα απο την φορά που γίνεται το shift_out μεταξύ τους.

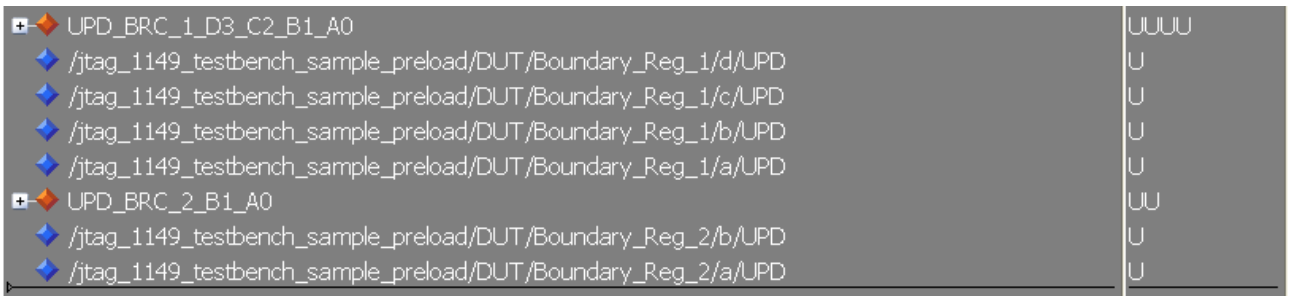
Αυτό έγινε προκειμένου να γίνεται εύκολα οπτικά η επιτυχής επαλήθευση του capture με τη τιμή εισόδου(Κρατάμε το περισσότερο σημαντικό bit αριστερά)

Αρα προσοχή , όταν κάνουμε shift στην συνέχεια , στο ομαδοποιημένο σήμα τα bits θα φαίνεται να γινονται shift από δεξιά προς τα αριστερά.

Στην συνέχεια από την χρονική στιγμή t = 60 ns και για (4+2)*5 = 30 ns παραμένουμε στην κατάσταση SHIFT_DR προκειμένου να ολισθήσουμε εξω απο το JTAG μέσω του TDO σειριακά , την απόκριση του CUT ακολουθούμενη από το διάνυσμα που εφαρμόστηκε στο CUT. Παράλληλα φροντίζουμε κάθως βγάζουμε έξω σειριακά τα παραπάνω να φορτώνεται σειριακά μέσω του TDO το διάνυσμα “1111” με κατάλληλα σήματα στο TDI.

Την χρονική στιγμή t= 95 ns γίνεται update του διανύσματος “1111” που φορτώσαμε σειριακά στον BRC_1 αλλά και του “11” που έχει φορτωθεί σειριακά στον BRC_2 και μεταφέρονται απο το CAP στον καταχωρητή UPD.

Το παραπάνω είναι εμφανές στα σηματα



τα οποία είναι ομαδοποιημένα ακριβώς με την ίδια λογική που ακολουθήθηκε για το shift_out