ΔΙΠΛΩΜΑΤΙΚΗ ΔΙΑΤΡΙΒΗ

Επέκταση μεθόδου Λογικού Φόρτου (Logical-Effort) μέσω ενσωμάτωσης μοντέλου διασύνδεσης

Αθανασόπουλος Παναγιώτης

Α.Μ. 1701012|83

Επιβλέπων Καθηγητής

Κ. Γεώργιος Σταμούλης

2008

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ & ΔΙΚΤΥΩΝ ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΘΕΣΣΑΛΙΑΣ

Περιεχόμενα

Ευχαριστίες

Θα ήθελα πρώτα από όλους να ευχαριστήσω τους γονείς μου για την υποστήριξή τους όλα αυτά τα χρόνια, για τις στερήσεις που επέστησαν ώστε να μπορέσω να πραγματοποιήσω όλα τα όνειρα μου, αλλά κυρίως για το γεγονός ότι ήταν πάντα εκεί τόσο στις καλές όσο και στις δύσκολες στιγμές.

Τον Καθηγητή μου κύριο Σταμούλη για την βοήθεια κάθε είδους όλα αυτά τα χρόνια, για την καθοδήγηση του αλλά κυρίως για το γεγονός ότι αποτελεί διαρκή παράγοντα έμπνευσης.

Τον συμφοιτητή και καλό μου φίλο Ιάσων Κουκά για τη συνεργασία, και διασκέδαση που είχαμε κατά τη διάρκειά αυτής, όλα αυτά τα χρόνια.

Τους καθηγητές μου κύριους Yusuf Leblebici και Paolo Ienne και συνολικά το École Polytechnique Fédérale de Lausanne για την καθοδήγηση, υποστήριξη και την πρόσβαση σε εργαλεία που μου παρείχαν.

Επίσης σε αυτή που μου με έκανε να συνειδητοποιήσω ότι πάντα πίσω από κάτι κακό κρύβεται πάντα κάτι καλό.

Τέλος σε όλους αυτούς που δεν αναφέρονται ονομαστικά, οι οποίοι παρόλα αυτά βοήθησαν, σαν αφανείς ήρωες, με τον τρόπο τους όλα αυτά τα χρόνια.

Τη σημερινή εποχή, ο πολιτισμός μας οφείλει σημαντικό ποσοστό της ανάπτυξης και ευημερίας του στην διάδοση και ανάπτυξη των αποκαλούμενων «ολοκληρωμένων κυκλωμάτων». Τα κυκλώματα αυτά αποτέλεσαν σημαντικό κινητήριο μοχλό ανάπτυξης κατά τις προηγούμενες δεκαετίες γεγονός που αναμένεται να συνεχιστεί και στο μέλλον. Παντού γύρω μας βρίσκονται ολοκληρωμένα κυκλώματα, στους υπολογιστές μας, στα κινητά μας ακόμα και σε σημεία όπου η χρήση τους δεν είναι προφανής, όπως για παράδειγμα αυτοκίνητα, πιστωτικές κάρτες κτλ.

Ολοκληρωμένα Κυκλώματα

Ολοκληρωμένο κύκλωμα (Integrated Circuit IC) αποτελεί ένα ηλεκτρικό κύκλωμα σε σμίκρυνση αποτελούμενο κυρίως από ημιαγωγούς και παθητικά στοιχεία (αντιστάσεις, πυκνωτές κτλ). Τα παραπάνω συνήθως κατασκευάζονται μαζί με τις διασυνδέσεις σε ένα και μόνο κοινό υπόστρωμα.

Τα ολοκληρωμένα κυκλώματα χωρίζονται συνήθως σε διάφορες κατηγορίες ανάλογα με τον αριθμό των ολοκληρωμένων από τα οποία αποτελούνται. Οι κατηγορίες αυτές είναι συνήθως οι ακόλουθες:

Μικρής κλίμακας ολοκλήρωσης (SSI)	> 10 τρανζίστορ
Μεσαίας κλίμακας ολοκλήρωσης (MSI)	> 100 τρανζίστορ
Μεγάλης κλίμακας ολοκλήρωσης (LSI)	> 1,000 τρανζίστορ
Πολύ μεγάλης κλίμακας ολοκλήρωσης (VLSI)	> 10,000 τρανζίστορ
Υπερ-μεγάλης κλίμακας ολοκλήρωσης (ULSI)	> 1,000,000 τρανζίστορ

Αναλογικά κυκλώματα

Αναλογικά κυκλώματα αποκαλούνται τα κυκλώματα τα οποία χρησιμοποιούν σήματα ενός εύρους συνεχής τάσεως (Αναλογικά σήματα) σε αντίθεση με τα διακριτά επίπεδα τάσης που χρησιμοποιούνται στα ψηφιακά κυκλώματα.

Αναλογικό σήμα

Αναλογικό σήμα ονομάζουμε το είδος σήματος το οποίο παίρνει τιμές σε ένα συνεχές φάσμα. Ένα αναλογικό σήμα χρησιμοποιεί κάποια ιδιότητα του μέσου στο οποίο διαδίδεται ώστε να μεταφέρει πληροφορίες. Σαν παράδειγμα θα μπορούσαμε να αναφέρουμε ένα βαρόμετρο κενού (Aneroid Barometer) το οποίο χρησιμοποιεί την γωνιακή θέση σαν σήμα ώστε να μεταβιβάσει την πληροφορία της πίεσης. Για να χρησιμοποιήσουμε ηλεκτρικά σήματα ώστε να αναπαραστήσουμε πληροφορία, μπορούμε να χρησιμοποιήσουμε διάφορες ιδιότητες των σημάτων αυτών όπως την τάση (Voltage), το φορτίο (Current), την συχνότητα (Frequency) ή την ένταση του ρεύματος (Electric charge). Η διακύμανση ενός εκ των παραπάνω μπορεί να αντιστοιχιστεί σε διακύμανση της πληροφορίας την οποία αναπαριστούμε μέσω του ηλεκτρικού σήματος αυτού. Για την μετατροπή ενός φυσικού σήματος (όπως ένταση φωτός, ήχος, θερμοκρασία, πίεση , θέση κτλ) σε ηλεκτρικό σήμα είναι αναγκαία η χρήση ενός είδους μόρφο-μετατροπέα (Transducer). Τα σήματα παίρνουν τιμές σε ένα εύρος τιμών, και κάθε τιμή του σήματος αναπαριστά διαφορετική πληροφορία. Οποιαδήποτε μετατροπή της τιμής του σήματος έχει νόημα, όπως επίσης κάθε στάθμη του σήματος αναπαριστά διαφορετική ένταση του φυσικού φαινομένου το οποίο αναπαριστά. Για παράδειγμα μπορούμε να θεωρήσουμε ότι το

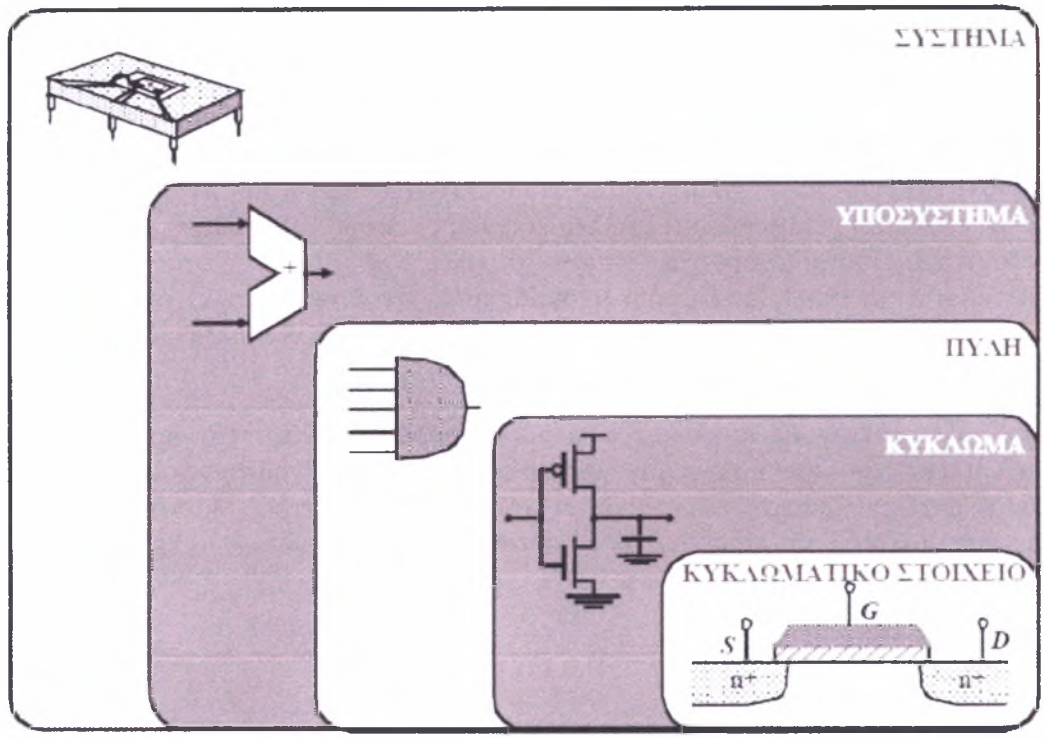
ηλεκτρικό μας σήμα χρησιμοποιείται για να αναπαραστήσει θερμοκρασία. Έστω ότι αντιστοιχούμε 1 Volt τάσης σε 1 βαθμό Celsius. Σε αυτό το παράδειγμα 10 volts θα μπορούσαν να αναπαριστούν 10 βαθμούς Celsius, ενώ 10.1 volts θα μπορούσαν να αναπαριστούν 10.1 βαθμούς Celsius. Ένας διαφορετικός τρόπος να εξαγάγουμε ένα αναλογικό σήμα είναι μέσω διαμόρφωσης (Modulation). Κατά την διαμόρφωση, μεταβάλλεται κάποια ιδιότητα ενός βασικού φέροντος σήματος. Η διαμόρφωση πλάτους (AM) κάνει χρήση της αλλαγής του πλάτους της κυματομορφής της τάσης από την πηγή πληροφορίας. Η διαμόρφωση συχνότητας (FM) κάνει χρήση της αλλαγής συχνότητας της κυματομορφής ώστε να απεικονίσει πληροφορία. Άλλες τεχνικές, όπως η αλλαγή φάσης του φέροντος, χρησιμοποιούνται επίσης συχνά. Καθημερινό παράδειγμα των ανωτέρω αποτελεί η αναλογική ηχογράφηση, κατά την οποία το μικρόφωνο συλλαμβάνει την διακύμανση της πίεσης του ήχου και δημιουργεί αντίστοιχη μεταβολή στο ηλεκτρικό σήμα.

Τα αναλογικά κυκλώματα μερικές φορές ονομάζονται και γραμμικά κυκλώματα, αν και πολλά μη γραμμικά φαινόμενα χρησιμοποιούνται στα αναλογικά κυκλώματα, όπως μείκτες (Mixers), ρυθμιστές (Modulators), κλπ. Χαρακτηριστικά παραδείγματα αναλογικών κυκλωμάτων περιλαμβάνουν ενισχυτές, ταλαντωτές κτλ.

Ψηφιακά κυκλώματα

Ψηφιακά κυκλώματα ονομάζονται τα κυκλώματα τα οποία κάνουν χρήση σημάτων διακριτών στάθμεων (Ψηφιακά σήματα) σε αντίθεση με τα αναλογικά. Τα ψηφιακά κυκλώματα κάνουν χρήση της άλγεβρας Boole (Boolean Algebra). Η άλγεβρα αυτή αναπτύχθηκε από τον George Boole και παρουσιάζει ομοιότητες με την συνήθη άλγεβρα των πραγματικών αριθμών αλλά με σημαντικότερη διαφορά ότι επιτρεπόμενες τιμές αποτελούν μόνο το λογικό μηδέν (0) και λογικό ένα (1). Η δυαδική(ές) τιμή(ές) αυτής της κατηγορίας αναφέρονται συχνά και ως bit(s). Η άλγεβρα αυτή επέτρεψε την υιοθέτηση μιας νέας αφαιρετικής εξαιρετικά πιο απλουστευμένης προσέγγισης κατά τον σχεδιασμό των κυκλωμάτων. Σαν αποτέλεσμα στα ψηφιακά κυκλώματα κάθε είδος πληροφορίας αναπαριστάται με την βοήθεια των δύο αυτών τιμών (λογικό μηδέν και λογικό ένα). Τα παραπάνω επέτρεψαν την υιοθέτηση ενός επιπέδου αφαίρεσης γνωστού ως λογικής πύλης. Οι λογικές πύλες αποτελούνται από τρανζίστορ τα οποία λειτουργούν σαν ένα είδος διακόπτη. Κατά την ψηφιακή λογική τόσο τα τρανζίστορ όσο και κατ'επέκταση οι λογικές πύλες εξομοιώνουν καταστάσεις λογικού ένα και μηδέν με αποτέλεσμα να αποτελούν το δομικό λίθο των ψηφιακών κυκλωμάτων. Στα ψηφιακά κυκλώματα, η ικανότητα αυτή υψηλού επιπέδου αφαίρεσης των ψηφιακών κυκλωμάτων σε επίπεδο πύλης οδήγησε σε εκπληκτική ανάπτυξη και διάδοσή τους.

Παρακάτω παρουσιάζονται τα ευρέως χρησιμοποιούμενα επίπεδα αφαίρεσης:



Επίπεδα αφαίρεσης ψηφιακής λογικής

Μικτά Κυκλώματα

Επιπλέον σήμερα, ορισμένα αναλογικά κυκλώματα συνηθίζεται να χρησιμοποιούν τεχνικές ψηφιακής λογικής ώστε να βελτιώσουν τις επιδόσεις του βασικού κυκλώματος. Αυτός ο τύπος κυκλωμάτων αναφέρεται συνήθως ως "Μικτά Κυκλώματα".

Μερικές φορές μπορεί να είναι δύσκολο να γίνει διαχωρισμός μεταξύ αναλογικών και ψηφιακών κυκλωμάτων, καθώς μεγάλο ποσοστό κυκλωμάτων κάνουν χρήση τόσο γραμμικών όσο και μη γραμμικών φαινομένων και λογικής.

Συνδυαστικά κυκλώματα

Συνδυαστικά κυκλώματα ονομάζονται τα κυκλώματα εκείνα των οποίων οι έξοδοι εξαρτούνται αποκλειστικά και μόνο από τις εισόδους τους, και αγνοούν προηγούμενες καταστάσεις τους. Με λίγα λόγια η συνδυαστική λογική κατά την οποία λειτουργούν τα συνδυαστικά κυκλώματα ορίζει ότι η έξοδος του κυκλώματος αποτελεί συνάρτηση αποκλειστικά της παρούσας εισόδου:

Έξοδοι = f(Εισόδων)

Σχηματική αναπαράσταση της συνδυαστικής λογικής παρουσιάζεται στο παρακάτω σχήμα:

Σχηματική αναπαράσταση συνδυαστικής λογικής

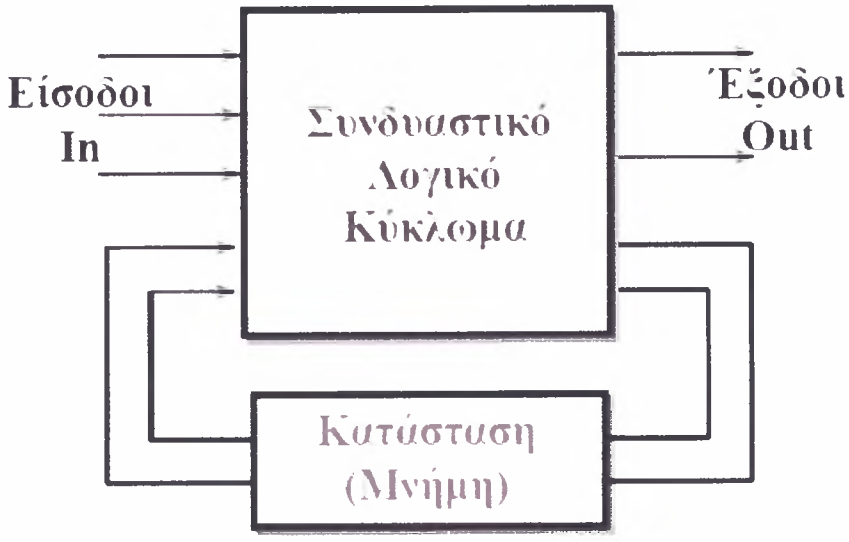
Η συνδυαστική λογική βρίσκει εφαρμογή στην εκτέλεση πράξεων της άλγεβρας Boole, παράδειγμα των οποίων αποτελούν τα διάφορα είδη πυλών.

Ακολουθιακά κυκλώματα

Ακολουθιακά κυκλώματα ονομάζονται τα κυκλώματα εκείνα τα οποίων η έξοδος εξαρτάται τόσο από τις εισόδους τους όσο και από την κατάσταση στην οποία βρίσκονται. Με λίγα λόγια η ακολουθιακή λογική ορίζει ότι η έξοδος του κυκλώματος αποτελεί συνάρτηση της παρούσας εισόδου και της κατάστασης στην οποία βρίσκεται το κύκλωμα κατά την εφαρμογή των εισόδων αυτών:

Έξοδοι = f(Εισόδων, Κατάστασης)

Σχηματική αναπαράσταση της ακολουθιακής λογικής παρουσιάζεται στο παρακάτω σχήμα:



Σχηματική αναπαράσταση ακολουθιακής λογικής

Σαν αποτέλεσμα, τα ακολουθιακά κυκλώματα παρουσιάζουν φαινόμενο μνήμης γεγονός που δεν παρατηρείται στα συνδυαστικά κυκλώματα.

Δομικά Στοιχεία Κυκλωμάτων

Παρακάτω παρουσιάζονται τα βασικά δομικά στοιχεία κυκλωμάτων με τα οποία θα ασχοληθούμε σε αυτή τη διατριβή.

Τρανζίστορ (transistor)

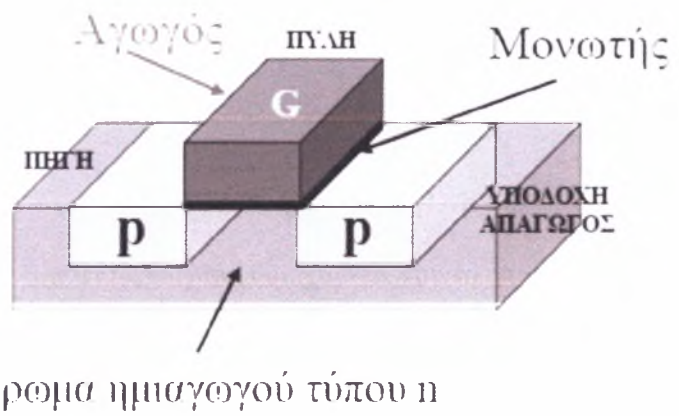
Το τρανζίστορ αποτελεί μια ημιαγωγική διάταξη στερεάς κατάστασης που βρίσκει εφαρμογή στον τομέα της ηλεκτρονικής. Το τρανζίστορ μπορεί, ανάλογα με την τάση με την οποία πολώνεται, να ρυθμίζει τη ροή του ρεύματος που απορροφά από συνδεδεμένη πηγή τάσης. Τα τρανζίστορ αποτελούν είτε ξεχωριστά ηλεκτρονικά εξαρτήματα, είτε μέρη κάποιου ολοκληρωμένου κυκλώματος.

Στη διατριβή αυτή θα ασχοληθούμε μόνο με την τεχνολογία CMOS (Completary MOS) και τα τρανζίστορ τύπου MOSFET.

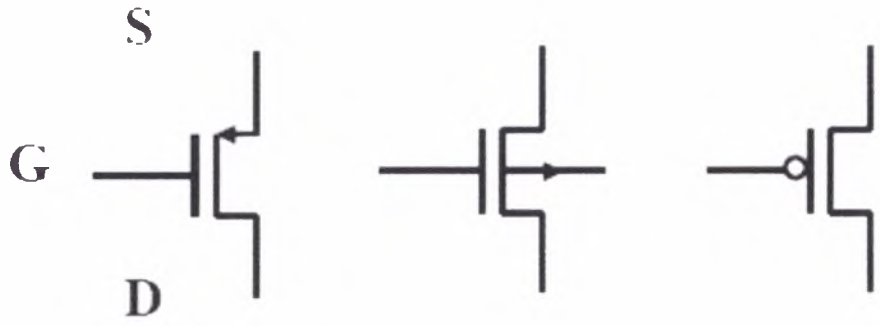
Τρανζίστορ τύπου MOSFET:

p-MOS Τρανζίστορ

Τα τρανζίστορ τύπου p (p-MOS) αποτελούνται από υπόστρωμα τύπου n πάνω στο οποίο κατασκευάζονται δύο περιοχές που ονομάζονται ΠΗΓΗ (Source) και ΥΠΟΔΟΧΗ (Drain) οι οποίες αποτελούνται από υλικό τύπου p. Η παραπάνω δομή παρουσιάζεται στο παρακάτω σχήμα:



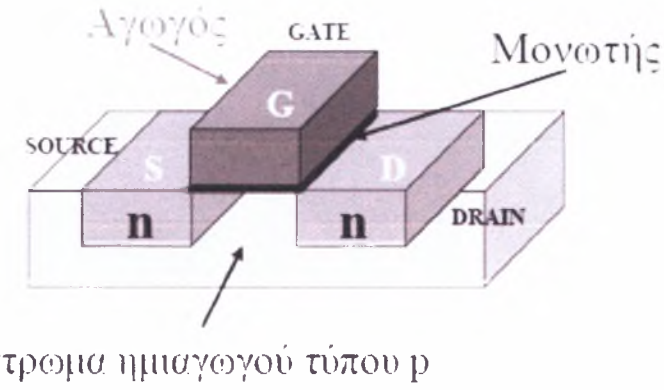
Γραφική απεικόνιση περιοχών τρανζίστορ τύπου p (p-MOS)



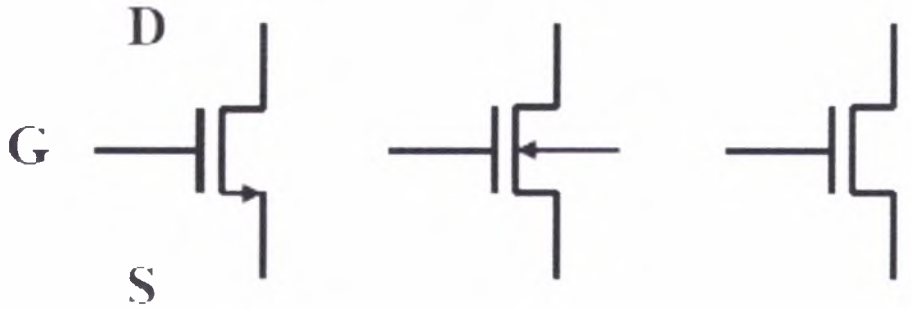
Σχηματικές απεικονίσεις που χρησιμοποιούνται για την αναπαράσταση τρανζίστορ τύπου p (p-MOS)

n-MOS Τρανζίστορ

Τα τρανζίστορ τύπου n (n-MOS) αποτελούνται από υπόστρωμα τύπου p πάνω στο οποίο κατασκευάζονται οι δύο περιοχές πηγής και υποδοχής αποτελούμενες από υλικό τύπου n. Η δομή αυτή παρουσιάζεται στο παρακάτω σχήμα:



Γραφική απεικόνιση περιοχών τρανζίστορ τύπου n (n-MOS)



Σχηματικές απεικονίσεις που χρησιμοποιούνται για την αναπαράσταση τρανζίστορ τύπου n (n-MOS)

Λογικές πύλες

Η λογική πύλη αποτελεί ένα ηλεκτρονικό κύκλωμα το οποίο πραγματοποιεί μία λογική πράξη στις εισόδους της και παράγει μία έξοδο. Οι λογικές πύλες έχουν δημιουργηθεί για να δουλεύουν στο δυαδικό σύστημα. Στα ηλεκτρονικά κυκλώματα ως λογικό μηδέν (0) θεωρείται η τάση εκείνη η οποία είναι κάτω από ένα κατώφλι που έχουν ορίσει οι κατασκευαστές της λογικής πύλης (πχ 0,5V). Αντίστοιχα το λογικό ένα (1) αντιστοιχεί σε τάση η οποία υπερβαίνει κάποια τάση. Με άλλα λόγια το λογικό 0 αντιστοιχεί στην τάση γείωσης και το λογικό 1 σε τάση τροφοδοσίας. Υπάρχουν διάφορες τεχνολογίες κατασκευής πυλών, όπως η CMOS. Οι λογικές πύλες με τις οποίες θα ασχοληθούμε σε αυτή τη διατριβή παρουσιάζονται παρακάτω.

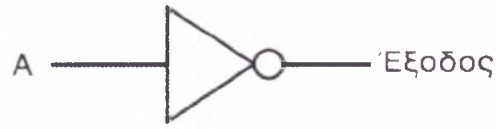
Πύλη NOT

Η πύλη NOT (ΟΧΙ) ή INVERTER (INV) έχει μόνο μία είσοδο και δίνει μόνο μία έξοδο. Η λειτουργία της είναι η αντιστροφή του λογικού σήματος της εισόδου. Ο πίνακας αληθείας της πύλης είναι:

Είσοδος	Έξοδος
A	NOT A
0	1
1	0

Πίνακας αληθείας της πύλης NOT

Το κυκλωματικό σχήμα της NOT είναι το εξής:



Πύλη NOT

Πύλη AND

Η πύλη AND εκτελεί την λογική πράξη AND (ΚΑΙ) μεταξύ των εισόδων της. Η πράξη AND στην άλγεβρα Boole συμβολίζεται με επί (*). Για παράδειγμα εάν η πύλη έχει δύο εισόδους (Α και Β) και μία έξοδο (Γ) θα γίνει η πράξη:

Γ = Α * Β

Ο πίνακας αληθείας της λογικής πύλης AND φαίνεται στο εξής σχήμα:

Είσοδοι		Έξοδος
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Πίνακας αληθείας της πύλης AND δύο εισόδων

Το κυκλωματικό σχήμα της AND δύο εισόδων είναι το εξής:

Πύλη AND δύο εισόδων

Παράδειγμα: έστω ότι Α=1 και Β=0. Η έξοδος Γ θα προκύψει 0.

Γενικότερα, η έξοδος AND δίνει λογική έξοδο 1 όταν όλες οι είσοδοί της βρίσκονται σε λογική κατάσταση 1.

Οι πύλες AND κατασκευάζονται και με περισσότερες των δύο εισόδων (πχ 3,4,5,8 είσοδοι).

Πύλη OR

Η πύλη OR εκτελεί την λογική πράξη OR (Η΄) μεταξύ των εισόδων της. Η πράξη OR στην άλγεβρα Boole συμβολίζεται με το συν (+). Για παράδειγμα εάν η πύλη έχει 2 εισόδους (Α και Β) και μία έξοδο (Γ) θα γίνει η πράξη:

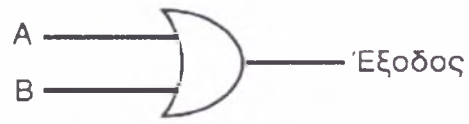
Γ = Α + Β

Ο πίνακας αληθείας της λογικής πύλης OR φαίνεται στο εξής σχήμα:

Είσοδοι		Έξοδος
Α	Β	Α OR Β
0	0	0
0	1	1
1	0	1
1	1	1

Πίνακας αληθείας της πύλης OR δύο εισόδων

Το κυκλωματικό σχήμα της OR δύο εισόδων είναι το εξής:



Πύλη OR δύο εισόδων

Παράδειγμα: Έστω ότι Α=1 και Β=0. Η έξοδος Γ θα προκύψει 1.

Γενικότερα, η πύλη OR δίνει λογικό 1 όταν μία τουλάχιστο είσοδος είναι σε λογικό 1.

Οι πύλες OR κατασκευάζονται και με περισσότερες των δύο εισόδων (πχ 3,4,5,8 είσοδοι).

Πύλη XOR

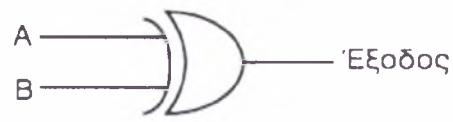
Η πύλη XOR εκτελεί την λογική πράξη XOR (ΑΠΟΚΛΕΙΣΤΙΚΟ Η') μεταξύ των εισόδων της. Η πράξη XOR στην άλγεβρα Boole συμβολίζεται με ένα συν μέσα σε ένα κύκλο (⊕).

Ο πίνακας αληθείας της λογικής πύλης XOR φαίνεται στο εξής σχήμα:

Είσοδοι		Έξοδος
A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

Πίνακας αληθείας της πύλης XOR δύο εισόδων

Το κυκλωματικό σχήμα της XOR δύο εισόδων είναι το εξής:



Πύλη XOR δύο εισόδων

Παράδειγμα: Έστω ότι A=1 και B=0. Η έξοδος Γ θα προκύψει 1.

Γενικά η πύλη XOR ελέγχει την περιττή ισοτιμία, δηλαδή δίνει λογικό 1 όταν περιττός αριθμός εισόδων βρίσκεται σε λογικό 1.

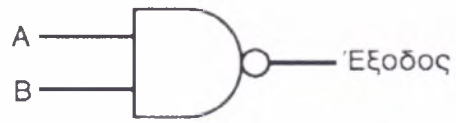
Πύλη NAND

Η πύλη NAND (ΟΧΙ-ΚΑΙ) δίνει την αντίθετη έξοδο από την AND, δηλαδή δίνει λογικό 1 όταν υπάρχει τουλάχιστο ένα λογικό 0 στις εισόδους. Ο πίνακας καταστάσεων είναι ο εξής:

Είσοδοι		Έξοδος
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

Πίνακας αληθείας της πύλης NAND δύο εισόδων

Το κυκλωματικό σχήμα της NAND δύο εισόδων είναι το εξής:



Πύλη NAND δύο εισόδων

Παράδειγμα: Έστω ότι A=1 και B=0. Η έξοδος Γ θα προκύψει 1.

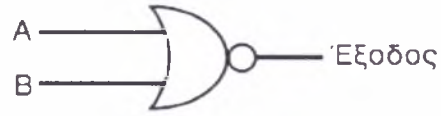
Πύλη NOR

Η πύλη NOR (ΟΧΙ-Η') δίνει την αντίθετη έξοδο από την OR, δηλαδή δίνει λογικό 1 όταν και οι δύο είσοδοι είναι 0. Ο πίνακας καταστάσεων είναι ο εξής:

Είσοδοι		Έξοδος
A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Πίνακας αληθείας της πύλης NOR δύο εισόδων

Το κυκλωματικό σχήμα της NOR δύο εισόδων είναι το εξής:



Πύλη NOR δύο εισόδων

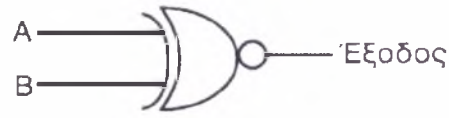
Πύλη XNOR

Η πύλη XNOR δίνει την αντίθετη έξοδο από την XOR, δηλαδή δίνει λογικό 1 όταν οι δύο είσοδοι είναι στην ίδια λογική στάθμη. Ο πίνακας καταστάσεων είναι ο εξής:

Είσοδοι		Έξοδος
A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

Πίνακας αληθείας της πύλης XNOR δύο εισόδων

Το κυκλωματικό σχήμα της XNOR δύο εισόδων είναι το εξής:



Πύλη XNOR δύο εισόδων

Flip Flop

Στα ψηφιακά κυκλώματα, Flip-flop ονομάζουμε μια κατηγορία ηλεκτρικών κυκλωμάτων τα οποία έχουν την ικανότητα να εναλλάσσουν την κατάσταση τους μεταξύ δύο έγκυρων σταθερών καταστάσεων κατά επιλογή. Δεδομένης της ικανότητας τους αυτής, τα κυκλώματα αυτά μπορούν να χρησιμοποιηθούν για αποθήκευση ενός bit δεδομένων. Σειρές αυτών των κυκλωμάτων αντίστοιχα μπορούν να χρησιμοποιηθούν για την αποθήκευση bits. Κάθε flip-flop ελέγχεται συνήθως από ένα ή περισσότερα σήματα ελέγχου ή/και από κάποια λογική πύλη ή σήμα ρολογιού. Σαν έξοδο αυτών των κυκλωμάτων συνήθως λαμβάνουμε πέρα από την φυσιολογική έξοδο και το συμπλήρωμα της φυσιολογικής εξόδου (Με την έννοια του αντίστροφου της εξόδου, Για παράδειγμα αν θεωρήσουμε ότι σαν έξοδο που υποδεικνύει την κατάσταση του flip-flop λαμβάνουμε λογικό ένα, τότε στην έξοδο του flip-flop που αναμένουμε το συμπλήρωμα της φυσιολογικής εξόδου θα λάβουμε μηδέν). Δεδομένο του γεγονότος ότι τα flip-flop αποτελούν ηλεκτρονικά κυκλώματα, με την σειρά τους απαιτούν σύνδεση στην πηγή ενέργειας του κυκλώματος στο οποίο ανήκουν όπως επίσης και σύνδεση στην αντίστοιχη γείωση.

Τα Flip-flop μπορούν να είναι είτε απλά (διαφανή) είτε καθοδηγούμενα από το σήμα του ρολογιού. Τα απλά Flip-flop μπορούν να κατασκευαστούν με χρήση δύο στοιχείων ικανών για αντιστροφή σήματος όπως είναι τα τρανζίστορ, οι πύλες ΌΧΙ-ΚΑΙ (NAND) και οι πύλες ΌΧΙ–Ή (NOR) ίσως επίσης παρόντος κάποιου μηχανισμού ελέγχου που κάνει χρήση λογικών πυλών. Τα καθοδηγούμενα από το σήμα ρολογιού Flip-flop, σχεδιάζονται συγκεκριμένα για χρήση σε συγχρονισμένα κυκλώματα (Κυκλώματα των οποίων η λειτουργία ελέγχεται από το σήμα ρολογιού) οπότε και αγνοούν τις εισόδους εκτός από τις περιπτώσεις μετάβασης του ρολογιού (ακμοπυροδότητα). Κατά την διάρκεια μετάβασης του ρολογιού και ανάλογα με τις εισόδους του Flip-flop αυτό είτε παραμένει στην ίδια κατάσταση που βρισκόταν και προηγουμένως είτε μεταβάλει την κατάστασή του στην άλλη επιτρεπτή κατάσταση. Κάποια από τα Flip-flop αυτά μπορούν να αλλάξουν την έξοδό τους κατά την διάρκεια ανόδου του ρολογιού (θετικά ακμοπυροδότητα), ενώ άλλα κατά την διάρκεια καθόδου του ρολογιού (αρνητικά ακμοπυροδότητα). Τα Flip-flop χωρίζονται περαιτέρω σε υποκατηγορίες οι οποίες χρησιμοποιούνται τόσο σε σύγχρονα όσο και σε ασύγχρονα ακολουθιακά κυκλώματα. Κατηγορίες τέτοιες αποτελούν τα SR (τοποθέτησης-επανατοποθέτησης / set-reset), τα D (καθυστέρησης / Delay), τα T (εναλλαγής / toggle) και JK, καθένα από τα οποία μπορεί να κατασκευαστεί κάνοντας χρήση ενός εκ των (περισσοτέρων) υπόλοιπων και επιπλέον πυλών. Η συμπεριφορά καθενός μπορεί να οριστεί με βάση την επόμενη κατάσταση στην οποία θα βρεθεί, κατά την επόμενη μετάβαση του ρολογιού, δεδομένης της τρέχουσας κατάστασης στην οποία βρίσκεται και των τιμών στις εισόδους του. Στη διατριβή αυτή θα ασχοληθούμε μόνο με τα Flip-flop τύπου D (καθυστέρησης / Delay).

D Flip-flop

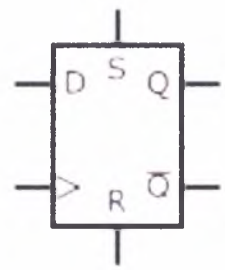
Τα Flip-flop τύπου D ονομάζονται έτσι καθώς έξοδο τους αποτελεί η είσοδός τους D (Data) μετά από καθυστέρηση (Delay) ενός κύκλου ρολογιού.

Στα D Flip-flop η έξοδός τους παίρνει τιμή πάντα κατά την διάρκεια ανόδου του ρολογιού. Κάθε D Flip-flop μπορεί να θεωρηθεί ως στοιχειώδες κελί μνήμης όπως επίσης και ως γραμμή καθυστέρησης. Ακολουθεί ο πίνακας αληθείας:

Ρολόι	Είσοδος D	Επόμενη Κατάσταση Q	Προηγούμενη Κατάσταση $Q_{προηγ}$
Άνοδος	0	0	Οτιδήποτε
Άνοδος	1	1	Οτιδήποτε
Όχι άνοδος	Οτιδήποτε	Σταθερή	

Πίνακας αληθείας Flip-flop τύπου D

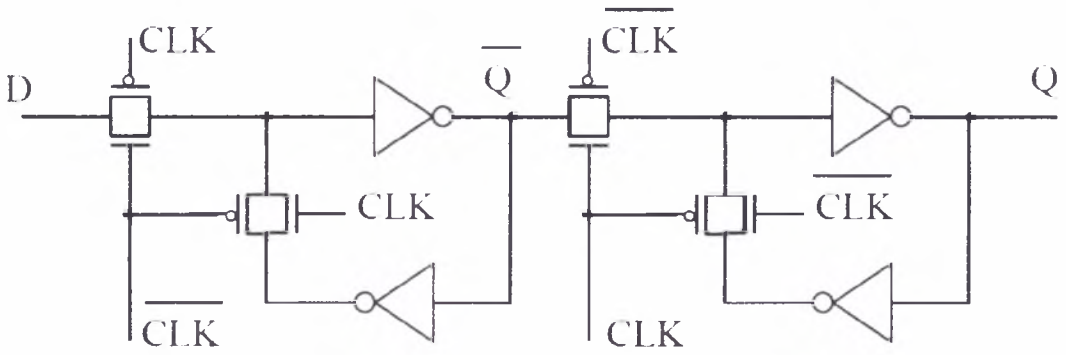
Αντίστοιχα το κυκλωματικό σύμβολο ενός Flip-flop τύπου D παρουσιάζεται παρακάτω:



Κυκλωματικό σύμβολο Flip-flop τύπου D

Η λειτουργία των Flip-flop τύπου D ορίζει ότι αυτά «αιχμαλωτίζουν» το μήνυμα που βρίσκεται στην είσοδό τους κατά την διάρκεια ανόδου του ρολογιού ενώ μεταγενέστερες αλλαγές της εισόδου δεν επηρεάζουν την έξοδο του μέχρι την επόμενη άνοδο του ρολογιού. Τα Flip-flop τύπου D είναι ιδιαίτερα σημαντικά καθώς αποτελούν τη βασική λειτουργική μονάδα από την οποία αποτελούνται οι καταχωρητές μετατόπισης (shift registers) οι οποίοι αποτελούν ουσιώδη υπομονάδα πολλών ηλεκτρονικών κυκλωμάτων. Επιπλέον κάποια εκ του συνόλου των Flip-flop δύναται να έχουν και ένα σήμα εισόδου επανατοποθέτησης (reset) το οποίο μπορεί να επαναφέρει την κατάσταση του Flip-flop στην αρχική (μηδέν) και η οποία μπορεί να είναι είτε σύγχρονη είτε ασύγχρονη με το ρολόι.

Τα Flip-flop τύπου D δύναται να κατασκευαστούν με χρήση στοιχείων c-MOS όπως τρανζίστορ p-MOS και n-MOS και πυλών. Οι πύλες αυτές μπορούν να κατασκευαστούν με την σειρά τους από τρανζίστορ τύπου p-MOS και n-MOS οπότε μπορούμε να θεωρήσουμε ότι δύναται να κατασκευαστούν από τρανζίστορ, για λόγους παρουσίασης παρακάτω παρουσιάζουμε ένα σχηματικό ενός Flip-flop τύπου D με χρήση τρανζίστορ p-MOS και n-MOS και πυλών τύπου αντιστροφέα.



Σχηματικό ακμοπυροδότητου Flip-flop τύπου D με χρήση τρανζίστορ p-MOS και n-MOS και πυλών τύπου αντιστροφέα

Στο παραπάνω σχήμα CLK και $\overline{CLK}$ εννοούμε την είσοδο ρολογιού και την αντίστροφη της εισόδου του ρολογιού αντίστοιχα.

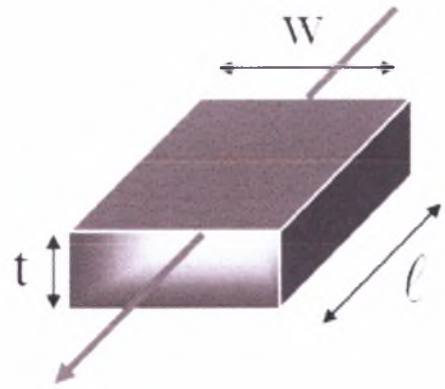
Διασύνδεση ολοκληρωμένων κυκλωμάτων

Στα σύγχρονα ολοκληρωμένα κυκλώματα, σημαντικότατο παράγοντα μελέτης αποτελεί ο τομέας της διασύνδεσης. Αναφερόμενοι στην διασύνδεση, εννοούμε τον τρόπο που συνδέονται, σε ένα ολοκληρωμένο κύκλωμα, μεταξύ τους τα διάφορα τρανζίστορ, τα τρανζίστορ με την πηγή του κυκλώματος, με την γείωση, καθώς και γενικότερα σε οποιαδήποτε σύνδεση υποσυστημάτων που απαρτίζουν το ολοκληρωμένο κύκλωμα. Καθώς μεταβαίνουμε σε διαρκώς μειούμενο μέγεθος τρανζίστορ μέσω της διαρκούς σμίκρυνσης της τεχνολογίας υλοποίησης ο τομέας της διασύνδεσης αποτελεί όλο και σημαντικότερη τροχοπέδη στην προσπάθειά μας για επίτευξη μικρότερης κατανάλωσης και μεγαλύτερης ταχύτητας. Έχει καθιερωθεί πλέον, η διασύνδεση να θεωρείται ως ο πλέον σημαντικός παράγοντας καθυστέρησης και κατανάλωσης στις τεχνολογίες αιχμής υλοποίησης. Δυστυχώς, έχει γίνει πλέον σαφές ότι οποιαδήποτε οφέλη αποκομίζουμε από την διαρκή μείωση των διαστάσεων των τρανζίστορ, εξανεμίζονται αν αγνοήσουμε τον τομέα της διασύνδεσης. Γεγονός αποτελεί ότι παρά την επιτάχυνση και μείωση κατανάλωσης που παρατηρείται από την υιοθέτηση σύγχρονων μεθόδων υλοποίησης, δεν παρατηρείται αντίστοιχη πρόοδος στον τομέα της διασύνδεσης. Παρακάτω παρουσιάζεται η απαραίτητη θεωρία ώστε να γίνουν κατανοητά βασικά μεγέθη που θα χρησιμοποιηθούν στην συνέχεια αυτής της διατριβής.

Αγωγοί διασύνδεσης

Βασικό στοιχείο διασύνδεσης αποτελεί ο αγωγός διασύνδεσης. Χαρακτηριστικά του αγωγού διασύνδεσης αποτελούν πέρα από τα κλασσικά (Μήκος, Πλάτος, Πάχος) και η ειδική αντίσταση αγωγού. Η ειδική αντίσταση του αγωγού εξαρτάται από μια πλειάδα παραγόντων, σημαντικότερο παρόλα αυτά, αποτελεί το είδος του μετάλλου από το οποίο αποτελείται.

Στο παρακάτω σχήμα παρουσιάζονται τα χαρακτηριστικά αυτά:



Σχηματική αναπαράσταση αγωγού διασύνδεσης

Στο παραπάνω σχήμα παρατηρούμε το μήκος του αγωγού (ℓ), το πλάτος (W) και το πάχος (t).

Κυριότερα χαρακτηριστικά της διασύνδεσης που επηρεάζουν την ταχύτητα και την κατανάλωση ενός ολοκληρωμένου κυκλώματος αποτελούν η αντίσταση και η χωρητικότητα των αγωγών διασύνδεσης.

Αντίσταση Αγωγού

Η αντίσταση αγωγού μπορεί να υπολογιστεί από τον παρακάτω τύπο:

$$R = \left(\frac{\rho}{t}\right)\left(\frac{\ell}{W}\right) = R_s\left(\frac{\ell}{W}\right)$$

όπου ρ η ειδική αντίσταση του αγωγού, t το πάχος του αγωγού, ℓ μήκος του αγωγού, W το πλάτος του αγωγού και t το πάχος του αγωγού. Τόσο η ειδική αντίσταση ρ όσο και το πάχος t του αγωγού θεωρούνται ως σταθερές και συνήθως δίνονται από τα χαρακτηριστικά της τεχνολογίας υλοποίησης που χρησιμοποιούμε καθώς και το στρώμα (επίπεδο ή Layer) στο οποίο βρίσκεται ο αγωγός στον οποίο αναφερόμαστε. Έτσι πλέον μπορούμε να χρησιμοποιήσουμε την σταθερά R_s ως το πηλίκο της διαίρεσης $\frac{\rho}{t}$.

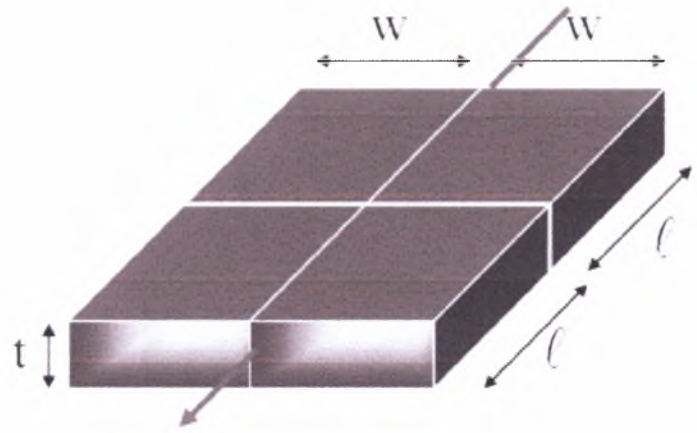
Από τον παραπάνω τύπο γίνεται φανερό ότι η αντίσταση του αγωγού είναι ανάλογη του μήκους ℓ και αντιστρόφως ανάλογη του πλάτους W του αγωγού. Σαν αποτέλεσμα, για να κρατήσουμε την αντίσταση του αγωγού σταθερή καθώς αυξάνει το μήκος του, θα πρέπει να αυξήσουμε το πλάτος του κατά τον ίδιο βαθμό.

Για παράδειγμα για να κρατήσουμε σταθερή την αντίσταση αγωγού αρχικού μήκους ℓ και πλάτους W καθώς διπλασιάζουμε το μήκος του σε 2ℓ πρέπει να

αυξήσουμε αντίστοιχα το πλάτος του σε $2W$ όπως φαίνεται και από τον παρακάτω τύπο:

$$R = R_s\left(\frac{2\ell}{2W}\right) = R_s\left(\frac{\ell}{W}\right)$$

Ο αντίστοιχος αγωγός παρουσιάζεται παρακάτω:

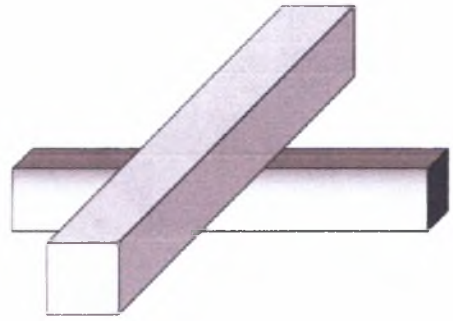


Σχηματική αναπαράσταση αγωγού διασύνδεσης διπλασιασμένου κατά μήκος και πλάτος

Αυτός ο τρόπος όμως δεν αποτελεί βιώσιμη λύση καθώς όπως φαίνεται και από το παραπάνω σχήμα ο αγωγός τετραπλασιάστηκε σε μέγεθος αντί να διπλασιαστεί. Η επιφάνεια του κυκλώματος αποτελεί έναν από τους βασικότερους παράγοντες κόστους κατασκευής του κυκλώματος.

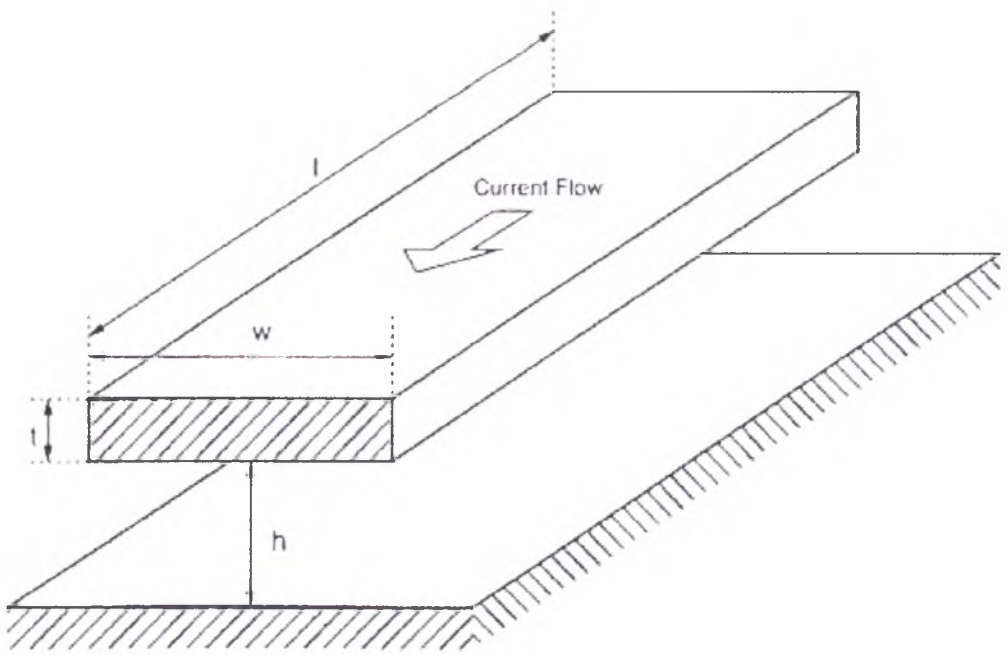
Χωρητικότητα αγωγού

Πέρα της αντίστασης, κάθε αγωγός διασύνδεσης παρουσιάζει και χωρητικότητα. Η χωρητικότητα αυτή αποτελεί παράγοντα που επηρεάζει την απόδοση του ολοκληρωμένου κυκλώματος. Στα σύγχρονα ολοκληρωμένα κυκλώματα ο υπολογισμός αυτών των χωρητικοτήτων (γνωστών και ως παρασιτικών χωρητικοτήτων διασύνδεσης) αποτελεί ένα από τα μεγαλύτερα προβλήματα. Κάθε δομή διασύνδεσης αποτελεί μια τρισδιάστατη δομή που εκτείνεται στο χώρο. Η δομή αυτή μπορεί να λάβει μια πλειάδα μορφών, μηκών, επιφανειών κτλ, καθώς επίσης και να κατασκευαστεί από μια πλειάδα υλικών. Επιπλέον καθεμία από αυτές εκτείνεται σε ένα χώρο στον οποίο βρίσκονται παρούσες άλλες αντίστοιχες δομές όπως φαίνεται στο παρακάτω σχήμα:



Σχηματική αναπαράσταση αγωγών διασύνδεσης στο χώρο

Κύριο παράγοντα εμφάνισης φαινομένων χωρητικοτήτων αποτελεί το γεγονός των αλληλεπιδράσεων των αγωγών διασύνδεσης. Ο υπολογισμός αυτών των φαινομένων αποτελεί εξαιρετικά επίπονη διαδικασία. Στην περίπτωσή μας θα παρουσιάσουμε ένα αρκετά απλουστευμένο, αν και παρόλα αρκετά ακριβές μοντέλο χωρητικότητας αγωγού. Έστω ότι έχουμε τον παρακάτω αγωγό διασύνδεσης:



Στον αγωγό αυτό παρουσιάζεται το πάχος του (thickness) ή t, το πλάτος του (width) ή w, το μήκος του (length) ή l και τέλος το ύψος του από έναν παράλληλο αγωγό (height) ή h. Με βάση αυτές τις διαστάσεις μπορούμε να ορίσουμε τους παρακάτω τύπους που αναπτύχθηκαν από τους Yuan και Trick:

$$C = \left[\frac{w - \frac{t}{2}}{h} + \frac{2\pi}{\ln\left(1 + \frac{2h}{t} + \sqrt{\frac{2h}{t}\left(\frac{2h}{t} + 2\right)}\right)} \right] \varepsilon, για, w \geq \frac{t}{2}$$

και

$$C = \left[\frac{w}{h} + \frac{\pi\left(1 - 0.0543\frac{t}{2h}\right)}{\ln\left(1 + \frac{2h}{t} + \sqrt{\frac{2h}{t}\left(\frac{2h}{t} + 2\right)}\right)} + 1.47 \right] \varepsilon, για, w < \frac{t}{2}$$

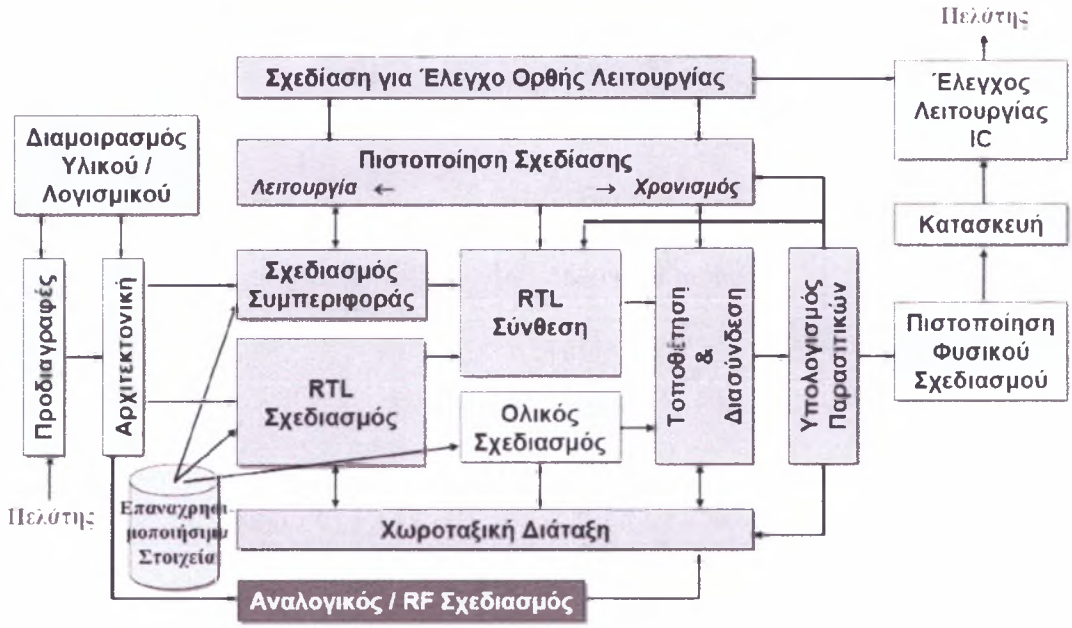
Από τους οποίους μπορούμε να εξάγουμε τις (παρασιτικές) χωρητικότητες *C* και t το πάχος του (thickness), w το πλάτος του (width), l το μήκος του (length), ε η ειδική σταθερά χωρητικότητας που εξαρτάται από το υλικό από το οποίο αποτελείται ο αγωγός και τέλος το ύψος του h από έναν παράλληλο αγωγό (height).

Διάμεσοι (VIAs)

Διάμεσο (VIA) ονομάζουμε το είδος μπλοκ εκείνο το οποίο μας επιτρέπει να συνδέουμε αγωγούς που βρίσκονται σε διαφορετικά επίπεδα (στρώματα-Layers) ενός ολοκληρωμένου κυκλώματος. Ένας διάμεσος κατασκευάζεται δημιουργώντας ένα μικρό άνοιγμα στο μονωτικό στρώμα οξειδίου και στην συνέχεια γεμίζοντάς το με κατάλληλο υλικό ώστε να επιτευχθεί διασύνδεση αγωγών που βρίσκονται σε διαφορετικό επίπεδο του ολοκληρωμένου κυκλώματος.

Σχεδίαση ολοκληρωμένων κυκλωμάτων

Η σύγχρονη σχεδίαση ολοκληρωμένων κυκλωμάτων έχει καθιερωθεί να χωρίζεται σε ορισμένα, διακριτά στάδια σχεδίασης. Ο διαχωρισμός αυτός επέτρεψε σε διάφορες ομάδες μέσα σε κάποια εταιρία να ασχολούνται με ένα ή και περισσότερα στάδια επιτυγχάνοντας την απαραίτητη εξειδίκευση και την παράλληλη εργασία (όχι απαραίτητα στον σχεδιασμό του ιδίου κυκλώματος) όλων των ομάδων αυτών. Τα διάφορα αυτά στάδια, όπως και οι μεταξύ τους μεταβάσεις, παρουσιάζονται παρακάτω:



Χάρτης βημάτων σχεδίασης ολοκληρωμένων κυκλωμάτων

Βήματα σχεδίασης ψηφιακών κυκλωμάτων

Από τα βήματα αυτά θα επικεντρωθούμε μόνο σε αυτά που αποτελούν είδος ψηφιακής σχεδίασης. Τα βήματα αυτά είναι τα παρακάτω:

- Σχεδιασμός Συμπεριφοράς (**E**lectronic **S**ystem **L**evel) ή ESL: Κατά το βήμα αυτό οι σχεδιαστές δημιουργούν την λειτουργική περιγραφή της συμπεριφοράς που πηγάζει από τις προδιαγραφές του συστήματος. Η περιγραφή αυτή πραγματοποιείται μέσω ειδικών γλωσσών υψηλού επιπέδου και εργαλείων όπως SystemC, Simulink κτλ.

- Σχεδιασμός επιπέδου μεταφοράς καταχωρητή (**R**egister **T**ransfer **L**evel) ή RTL: Κατά το βήμα αυτό οι σχεδιαστές περιγράφουν το κύκλωμα με κάθε λεπτομέρεια σε επίπεδο μεταφοράς καταχωρητή. Η περιγραφή αυτή ορίζει με λεπτομέρεια την μετάβαση των σημάτων μεταξύ των διάφορων καταχωρητών, όπως επίσης και τον τρόπο με τον οποίο κάθε υποσύστημα επηρεάζει τα σήματα αυτά.

- Ολικός σχεδιασμός (Custom Design): Κατά το βήμα αυτό πραγματοποιείται η ουσιαστική σχεδίαση του κυκλώματος. Κατά το

βήμα αυτό το κύκλωμα πλέον δεν προσεγγίζουμε το κύκλωμα σαν μια αφαιρετική οντότητα αλλά προχωρούμε στην φυσική του σχεδίαση.

Στη σύγχρονη εποχή δύο είναι οι κυριότερες προσεγγίσεις κατά την φυσική σχεδίαση ολοκληρωμένων κυκλωμάτων. Οι προσεγγίσεις αυτές υποστηρίζονται από μια πλειάδα αυτοματοποιημένων εργαλείων σχεδίασης πολλά από τα οποία υποστηρίζουν και τις δύο μεθόδους.

Πλήρως Προσαρμοσμένη (Full Custom)

Πλήρως προσαρμοσμένη σχεδίαση (Full Custom) ονομάζουμε την μεθοδολογία που ακολουθείται κατά την σχεδίαση ολοκληρωμένων κυκλωμάτων κατά την οποία καθορίζεται με ακρίβεια η φυσική σχεδίαση κάθε επιμέρους τρανζίστορ καθώς κάθε είδους διασύνδεσης μεταξύ τους. Η πλήρως προσαρμοσμένη σχεδίαση αποτελεί την ποιό αποτελεσματική μέθοδο με βάση την ταχύτητα του κυκλώματος, κατανάλωση και μέγεθος του κυκλώματος. Επιπλέον δίνει στο χρήστη την δυνατότητα και ευελιξία να σχεδιάσει το κύκλωμα που επιθυμεί με κάθε λεπτομέρεια. Παρόλα αυτά η πλήρως προσαρμοσμένη σχεδίαση αποτελεί τόσο εξαιρετικά χρονοβόρα διαδικασία ώστε η χρήση της να περιορίζεται μόνο σε περιπτώσεις όπου τα παραπάνω πλεονεκτήματα αποτελούν απαίτηση. Σαν αποτέλεσμα των παραπάνω, η πλήρως προσαρμοσμένη σχεδίαση περιορίζεται πλέον σε ολοκληρωμένα κυκλώματα που πρόκειται να παραχθούν σε πολύ μεγάλες ποσότητες, όπως για παράδειγμα οι επεξεργαστές των σύγχρονων υπολογιστών, όπως επίσης και σε μικρό αριθμό ολοκληρωμένων κυκλωμάτων ειδικών εφαρμογών (Application specific integrated circuits - ASICs).

Ημί-προσαρμοσμένη (Semi Custom)

Ημί-προσαρμοσμένη σχεδίαση (Semi Custom) ονομάζουμε την εναλλακτική σχεδίαση, αντί της πλήρως προσαρμοσμένης, κατά την οποία γίνεται χρήση βιβλιοθηκών τυποποιημένων κελιών. Οι βιβλιοθήκες αυτές παρέχουν ανωτέρου επιπέδου κυκλωματικές δομές όπως λογικές πύλες, αλλά ακόμα και ολόκληρα υποσυστήματα όπως μετρητές και αθροιστές τα οποία έχουν σχεδιαστεί από τον παροχέα της βιβλιοθήκης με χρήση της Πλήρως Προσαρμοσμένης μεθόδου. Οι Βιβλιοθήκες αυτές επιτρέπουν στον σχεδιαστή να επιταχύνει εξαιρετικά την διαδικασία σχεδίασης ενός ολοκληρωμένου κυκλώματος καθώς ο σχεδιαστής πλέον έχει πρόσβαση σε κατάλληλα βασικά υποκυκλώματα από τα οποία μπορούν να προέλθουν όλα τα υπόλοιπα κυκλώματα. Η δουλεία πλέον του σχεδιαστή (ή του αυτοποιημένου εργαλείου το οποίο τον αντικαθιστά) μετατρέπεται σε σύνθεση του επιθυμητού κυκλώματος με χρήση άλλων υποσυστημάτων. Με λίγα λόγια, η ημί-προσαρμοσμένη σχεδίαση μπορεί να προσεγγιστεί σαν την κατασκευή μιας μεγάλης εικόνας με χρήση κομματιών πάζλ.

Τεχνολογίες υλοποίησης

Αναφερόμενοι στις τεχνολογίες υλοποίησης ολοκληρωμένων κυκλωμάτων αναφερόμαστε στο σύνολο των διαδικασιών για την υλοποίηση ενός κυκλώματος σε σιλικόνη. Κάθε τεχνολογία υποστηρίζεται από ένα σύνολο αυτοματοποιημένων εργαλείων. Τα εργαλεία αυτά χρησιμοποιούν ένα είδος σουίτας σχεδιασμού (Design Kit) που περιλαμβάνει τα διαθέσιμα στοιχεία της τεχνολογίας. Κάθε τεχνολογία υλοποίησης συνήθως χαρακτηρίζεται από το τυπικό μήκος των τρανζίστορ από τα οποία αποτελούνται οι πύλες που χρησιμοποιούνται για την κατασκευή κυκλωμάτων στην τεχνολογία αυτή. Οι τεχνολογίες που θα παρουσιάσουμε αντιστοιχούν σε τεχνολογίες 180 νανομέτρων και 65 νανομέτρων. Συχνά όταν αναφερόμαστε στην τεχνολογία υλοποίησης ενός κυκλώματος αναφέρουμε την σουίτα σχεδιασμού (design kit) υπό την οποία αυτό σχεδιάστηκε καθώς πολλές εταιρίες μπορεί να παρέχουν διαφορετικές σουίτες σχεδιασμού για την ίδια τεχνολογία υλοποίησης.

Τεχνολογία υλοποίησης 0.18 μικρόμετρα (UMC 0.18)

Όπως γίνεται φανερό από την ονομασία της τεχνολογίας υλοποίησης αυτής τα τυπικά μήκη τρανζίστορ από τα οποία αποτελούνται οι πύλες που κάνουν χρήση της τεχνολογίας αυτής είναι 0.18 μικρόμετρα ή αντίστοιχα 180 νανόμετρα. Παρόλα αυτά, για να είμαστε ποιό ακριβείς η ονομασία της με αυτή την τιμή έγινε κυρίως λόγω μιας τάσης κλιμάκωσης κατά 70% κάθε δύο με τρία χρόνια της τεχνολογίας υλοποίησης. Η κλιμάκωση αυτή επετεύχθη κατά τα χρόνια 1999-2000 κάνοντας την αντίστοιχη τεχνολογία διαθέσιμη στις εταιρίες κατασκευής και σχεδιασμού. Παρόλα αυτά η ευρεία χρήση της υιοθετήθηκε αρκετά αργότερα όπως γίνεται πάντα στις περιπτώσεις αυτές. Γενικά όταν επιτευχθεί μια κλιμάκωση της τεχνολογίας η αντίστοιχη τεχνολογία και τα εργαλεία αυτής περιορίζονται σε ένα μικρό αριθμό εταιριών οι οποίες έχουν κρίσιμες απαιτήσεις για την ταχύτητα, κατανάλωση κτλ των ολοκληρωμένων τους και οι οποίες μπορούν να επωμιστούν το κόστος των εργαλείων αυτών και των εγκαταστάσεων οι οποίες δύναται να κατασκευάσουν κυκλώματα τέτοιας τεχνολογίας. Καθώς τα χρόνια περνούν και το κόστος της εκάστοτε τεχνολογίας περιορίζεται, σταδιακά υιοθετείται και από άλλες εταιρίες. Την ώρα που γράφονται οι γραμμές αυτές, η εν λόγω τεχνολογία χρησιμοποιείται από ένα μεγάλο αριθμό εταιριών και πανεπιστημιακών ιδρυμάτων λόγω του μεγάλου αριθμού ανεξάρτητων εγκαταστάσεων που δύναται να χρησιμοποιηθούν για την κατασκευή ολοκληρωμένων σε αυτές τις διαστάσεις όπως επίσης και του λογικού κόστους σχεδίασης και κατασκευής ολοκληρωμένων με χρήση αυτής. Η σουίτα σχεδιασμού που θα χρησιμοποιήσουμε για την τεχνολογία αυτή παρέχεται από την εταιρία UMC (United Microelectronics Corporation).

Τεχνολογία υλοποίησης 65 νανομέτρων (ST 65)

Ομοίως από την ονομασία της τεχνολογίας συμπεραίνουμε ότι τα τυπικά μήκη τρανζίστορ από τα οποία αποτελούνται οι πύλες που κάνουν χρήση της τεχνολογίας αυτής είναι 65 νανόμετρα (65nm). Η τεχνολογία αυτή αποτελεί (ως από το 2008) την πλέον προηγμένη τεχνολογία λιθογραφίας που χρησιμοποιείται για την παραγωγή όγκου ολοκληρωμένων κυκλωμάτων. Στην

τεχνολογία αυτή επιτρέπεται η χρήση τρανζίστορ μέχρι και 25 νανομέτρων αν και το τυπικό μήκος των τρανζίστορ που χρησιμοποιούνται στην τεχνολογία αυτή θεωρείται 65 νανόμετρα. Τη στιγμή που γράφονται η γραμμές αυτές η τεχνολογία αυτή αποτελεί την πλέον σύγχρονη τεχνολογία που χρησιμοποιείται κατά το σχεδιασμό και την κατασκευή των σύγχρονων επεξεργαστών με μόνη εξαίρεση τις τελευταίες εκδόσεις του επεξεργαστή Core 2 Duo Penryn και 5400 Xeon της εταιρίας Intel. Η σουίτα σχεδιασμού που θα χρησιμοποιήσουμε για την τεχνολογία αυτή παρέχεται από την εταιρία ST Microelectronics.

Η Μέθοδος λογικού φόρτου (Logical Effort)

Με την φράση «λογικός φόρτος» αποδίδουμε την πρωτότυπη αγγλική έκφραση "logical effort". Μια πιο ακριβής μετάφραση του "logical effort" θα ήταν "λογική προσπάθεια", μετάφραση την οποία αποφεύγουμε εσκεμμένα στην προσπάθεια μας να αποδώσουμε το πραγματικό νόημα της φράσης "logical effort". Ο όρος "logical effort" εισήχθη από τους Ivan E. Sutherland και Robert F. Sproull και αποτελεί μια μέθοδο για την εκτίμηση της καθυστέρησης σε ένα κύκλωμα και σαν επέκταση, αν χρησιμοποιηθεί κατάλληλα, να μας βοηθήσει στην επιλογή κατάλληλων πυλών για την υλοποίηση ενός κυκλώματος. Η επιλογή των πυλών περιλαμβάνει τόσο το είδος των πυλών (π.χ πύλη NAI), τον αριθμό τους (συμπεριλαμβάνοντας και των αριθμό των επιπέδων λογικής), όσο και των διαστάσεων των τρανζίστορ τα οποία τις αποτελούν.

Υπολογισμός καθυστέρησης λογικής πύλης

Η καθυστέρηση μιας CMOS πύλης εξαρτάται από:

- Από τα μεγέθη των τρανζίστορ τα οποία την απαρτίζουν
- Από τον τύπο της πύλης
- Από το φορτίο που οδηγεί στην έξοδό της
- Από την τεχνολογία υλοποίησης

Η καθυστέρηση κάθε πύλης στη μέθοδο του λογικού φόρτου εκφράζεται με βάση την καθυστέρηση ενός αντιστροφέα ο οποίος οδηγεί ένα ακριβώς πανομοιότυπο αντιστροφέα χωρίς καθόλου παρασιτική χωρητικότητα. Ο αντιστροφέας αυτός αποτελεί την μονάδα αναφοράς της μεθόδου. Ακόμα και η καθυστέρηση του συνολικού κυκλώματος υπολογίζεται με βάση τον αντιστροφέα αναφοράς. Ο αντιστροφέας αναφοράς παρουσιάζει καθυστέρηση που υπολογίζεται από τον παρακάτω τύπο:

$$\tau = 3RC$$

Όπου τ η καθυστέρηση του αντιστροφέα αναφοράς και R, C αποτελούν σταθερές τις εκάστοτε τεχνολογίας υλοποίησης. Σε μια τυπική τεχνολογία υλοποίησης 0.6 μικρομέτρων η καθυστέρηση του αντιστροφέα αναφοράς είναι περίπου 50 Πίκο-δευτερόλεπτα (ps) ενώ για τεχνολογία υλοποίησης 0.25 μικρομέτρων η καθυστέρηση του είναι περίπου 20 Πίκο-δευτερόλεπτα (ps).

Σαν αποτέλεσμα μπορούμε πλέον να ορίσουμε την πραγματική (απόλυτη) καθυστέρηση d_{abs} με βάση την καθυστέρηση κάθε πύλης και της καθυστέρησης του αντιστροφέα αναφοράς:

$$d_{abs} = d\tau$$

Όπου τ η καθυστέρηση του αντιστροφέα αναφοράς και d η καθυστέρηση της εκάστοτε πύλης. Η καθυστέρηση του αντιστροφέα αναφοράς τ αναφέρεται συχνά και ως μονάδα καθυστέρησης.

Η καθυστέρηση μιας λογικής πύλης d μπορεί να εκφραστεί ως άθροισμα δύο όρων:

- Της παρασιτικής καθυστέρησης (parasitic delay) ή p (που αποτελεί ένα εγγενές μέγεθος το οποίο δύναται να υπολογιστεί μέσω της εξέτασης της πύλης όταν αυτή δεν οδηγεί φορτίο).
- Της προσπάθειας βαθμίδας (stage effort) ή f (η οποία εξαρτάται από το φορτίο όπως θα φανεί παρακάτω)

Επομένως μπορούμε να ορίσουμε τον τύπο καθυστέρησης της λογικής πύλης ως:

$$d = f + p$$

Όπου d η καθυστέρηση της εκάστοτε πύλης και f, p η προσπάθεια βαθμίδας και η παρασιτική καθυστέρηση αντίστοιχα. Η καθυστέρηση της εκάστοτε πύλης d αναφέρεται συχνά και ως σχετική καθυστέρηση καθώς δεν είναι εκφρασμένη σε μονάδες χρόνου.

Η προσπάθεια βαθμίδας f δύναται να εκφραστεί ως γινόμενο δύο όρων:

- Του λογικού φόρτου (logical effort) ή g ο οποίος αποτελεί το λόγο της χωρητικότητας εισόδου της εκάστοτε πύλης προς την χωρητικότητα εισόδου ενός αντιστροφέα ικανού να προσφέρει το ίδιο φορτίο στην έξοδο. Αντιπροσωπεύει την επίδραση της τοπολογίας της πύλης στην ικανότητά της να δίνει στην έξοδό της ρεύμα και είναι ανεξάρτητος από το μέγεθος των τρανζίστορ της πύλης. Αποτέλεσμα αυτού, είναι το γεγονός ότι ο λογικός φόρτος μπορεί να θεωρηθεί ως σταθερά του τύπου της εκάστοτε πύλης αφού βασίζεται σε εγγενείς ιδιότητες της εκάστοτε κατηγορίας.
- Του ηλεκτρικού φόρτου (electrical effort) ή h ο οποίος αποτελεί το λόγο της χωρητικότητας του εξωτερικού φόρτου της εκάστοτε πύλης προς την χωρητικότητα εισόδου. Ο ηλεκτρικός φόρτος περιγράφει πως το ηλεκτρικό περιβάλλον (φόρτος) της πύλης επηρεάζει την ταχύτητα καθώς και πως το μέγεθος των τρανζίστορ της πύλης καθορίζει την ικανότητά της να οδηγήσει αυτό το φόρτο.

Επομένως μπορούμε να ορίσουμε τον τύπο της προσπάθεια βαθμίδας ως:

$$f = gh$$

Όπου f η προσπάθεια βαθμίδας και g, h ο λογικός και ηλεκτρικός φόρτος αντίστοιχα.

Ο ηλεκτρικός φόρτος (electrical effort) ή h μπορεί να εκφραστεί ως το λόγο της χωρητικότητας του εξωτερικού φόρτου της εκάστοτε πύλης προς την χωρητικότητα εισόδου όπως φαίνεται από τον παρακάτω τύπο:

$$h = \frac{C_{out}}{C_{in}}$$

Όπου h ο ηλεκτρικός φόρτος, C_{in} η χωρητικότητα εισόδου της λογικής πύλης και C_{out} η χωρητικότητα του εξωτερικού φορτίου που οδηγείται από την πύλη αυτή.

Από τον συνδυασμό των παραπάνω εξισώσεων προκύπτει:

$$d = gh + p$$

Παρατηρούμε ότι τόσο ο λογικός όσο και ο ηλεκτρικός φόρτος συμβάλουν κατά τον ίδιο τρόπο στην καθυστέρηση μιας λογικής πύλης. Πρέπει επίσης να σημειωθεί ότι η παρασιτική καθυστέρηση είναι ενδογενής και σε μεγάλο βαθμό ανεξάρτητη του μεγέθους των τρανζίστορ της λογικής πύλης. Επιπλέον ο λογικός φόρτος εξαρτάται αποκλειστικά από την τοπολογία της πύλης και όχι από το φορτίο που οδηγεί ή το μέγεθος των τρανζίστορ της. Ο υπολογισμός του λογικού φόρτου μιας κατηγορίας πυλών είναι τετριμμένος και ιδιαίτερα πολύπλοκος και ξεφεύγει από όρια της διατριβής αυτής. Μπορούμε παρόλα αυτά να θεωρήσουμε την τιμή του γνωστή για κάθε είδος πύλης καθώς είναι δυνατόν να υπολογιστεί για τις πιο ευρέως χρησιμοποιούμενες πύλες, από πολύ απλούς τύπους, οι οποίοι για κάθε είδος πύλης εξαρτώνται μόνο από τον αριθμό των εισόδων της. Ο κάτωθι πίνακας παρέχει τις τιμές του λογικού φόρτου για τις συνηθέστερα χρησιμοποιούμενες πύλες και με σύμβαση ότι το πλάτος κάθε p-MOS τρανζίστορ είναι διπλάσιο του αντίστοιχου n-MOS.

Πύλη	**Αριθμός Εισόδων**					
	1	2	3	4	5	Πλήθος-N
Αντιστροφέας	1					
ΟΧΙ-ΚΑΙ		4/3	5/3	6/3	7/3	(N+2)/3
ΟΧΙ-Ή		5/3	7/3	9/3	11/3	(2N+1)/3
Πολυπλέκτης		2	2	2	2	2
Αποκλειστικό-Ή		4	12	32		

Πίνακας λογικού φόρτου για τους διάφορους τύπους πυλών

Για τον υπολογισμός της καθυστέρησης μιας πύλης απομένει πλέον μόνο ο υπολογισμός της παρασιτικής καθυστέρησης (parasitic delay) ή *p*. Η παρασιτική καθυστέρηση είναι ενδογενής και αμετάβλητη, ανεξάρτητη του μεγέθους της πύλης και του φορτίου που οδηγεί. Αυτό οφείλεται στο ότι μεγαλύτερα τρανζίστορ που παρέχουν μεγαλύτερα ρεύματα παρουσιάζουν μεγαλύτερες χωρητικότητες διάχυσης στις περιοχές της πηγής και απαγωγού. Η παρασιτική καθυστέρηση οφείλεται στην χωρητικότητα των περιοχών πηγής και απαγωγού των τρανζίστορ. Η τυπική τιμή παρασιτικής καθυστέρησης ενός αναστροφέα είναι η μονάδα (1.0). Ο κάτωθι πίνακας μας δίνει τις παρασιτικές καθυστερήσεις του αντιστροφέα, του πολυπλέκτη και των πυλών ΌΧΙ-ΚΑΙ, ΌΧΙ-Ή, Αποκλειστικό-Ή, XNOR. Αυτές δίνονται ως πολλαπλάσια της παρασιτικής καθυστέρησης ($p_{αντιστροφέα}$) του αντιστροφέα αναφοράς, η παρασιτική καθυστέρηση του οποίου εξαρτάται από την τεχνολογία υλοποίησης.

Πύλη	**Παρασιτική Καθυστέρηση**
Αντιστροφέας	$p_{αντιστροφέα}$
Πύλη ΌΧΙ-ΚΑΙ N-εισόδων	$N \times p_{αντιστροφέα}$
Πύλη ΌΧΙ-Ή N-εισόδων	$N \times p_{αντιστροφέα}$
Πολυπλέκτης N-επιλογών	$2 \times N \times p_{αντιστροφέα}$
Αποκλειστικό-Ή	$4 \times p_{αντιστροφέα}$
XNOR	$4 \times p_{αντιστροφέα}$

Πίνακας παρασιτικών καθυστερήσεων για τους διάφορους τύπους πυλών

Στη συνέχεια για τον υπολογισμό της συνολικής καθυστέρησης ενός κυκλώματος θα παρουσιάσουμε δύο μεθόδους του λογικού φόρτου.

Γενικευμένη Μέθοδος

Η γενικευμένη μέθοδος του λογικού φόρτου (logical effort) αποτελεί μια βασική μέθοδο για τον υπολογισμό της καθυστέρησης ενός κυκλώματος. Στην συνέχεια, η μέθοδος μας παρέχει τόσο τον βέλτιστο αριθμό επιπέδων υλοποίησης του κατά εξέταση κυκλώματος όσο και τον τρόπο ελαχιστοποίησης της συνολική καθυστέρηση μέσω της αλλαγής των μεγεθών των τρανζίστορ στις διάφορες πύλες που απαρτίζουν το κύκλωμα κατανέμοντας την καθυστέρηση μεταξύ των επιπέδων. Η μέθοδος του λογικού φόρτου επεκτείνεται εύκολα από ξεχωριστές πύλες σε πολυεπίπεδα μονοπάτια.

Ο λογικός φόρτος ενός μονοπατιού μπορεί να υπολογιστεί πολλαπλασιάζοντας τους λογικούς φόρτους όλων των πυλών κατά μήκος του μονοπατιού. Για την αναπαράσταση του λογικού φόρτου ενός μονοπατιού χρησιμοποιούμε το

κεφαλαίο σύμβολο G (αντί για g που χρησιμοποιούμε για το λογικό φόρτο της πύλης):

$$G = \prod g_i$$

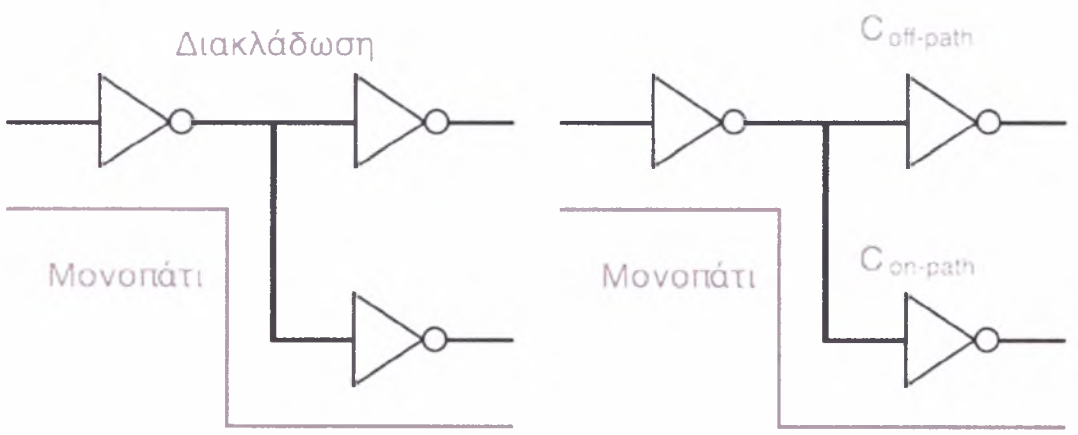
Όπου g_i ο λογικός φόρτος των επιπέδων (και άρα των πυλών) που το απαρτίζουν.

Ο ηλεκτρικός φόρτος ενός μονοπατιού ορίζεται ως ο λόγος της χωρητικότητας του φορτίου στην έξοδο της τελευταίας πύλης του μονοπατιού προς τη χωρητικότητα εισόδου της πρώτης πύλης του μονοπατιού. Για την αναπαράσταση του ηλεκτρικού φόρτου ενός μονοπατιού χρησιμοποιούμε το κεφαλαίο σύμβολο H (αντί για h που χρησιμοποιούμε για τον ηλεκτρικό φόρτο της πύλης):

$$H = \frac{C_{out}}{C_{in}}$$

Όπου C_{out} η χωρητικότητα εξόδου της τελευταίας πύλης και C_{in} η χωρητικότητα της πρώτης πύλης του μονοπατιού.

Εισαγάγουμε τώρα μια καινούρια έννοια φόρτου που αναφέρεται ως φόρτος διακλάδωσης (branching effort) ή b. Η έννοια αυτή αντικατοπτρίζει τον βαθμό διακλάδωσης της εξόδου μιας πύλης σε εισόδους άλλων πυλών. Πλέον σε κάθε επίπεδο, μόνο ένα ποσοστό του συνολικού ρεύματος οδήγησης κατευθύνεται κατά μήκος του συγκεκριμένου μονοπατιού αναφοράς, ενώ το υπόλοιπο κατευθύνεται εκτός αυτού μέσω των διάφορων διακλαδώσεων στο μονοπάτι. Το παραπάνω φαινόμενο παρουσιάζεται στο παρακάτω σχήμα:



Σχηματική αναπαράσταση διακλαδώσεων

Έτσι εισάγουμε την έννοια του φόρτου διακλάδωσης (branching effort) ή *b* στην έξοδο μιας λογικής πύλης ως:

$$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}} = \frac{C_{total}}{C_{useful}}$$

Όπου *b* ο φόρτος διακλάδωσης (branching effort) στην έξοδο της πύλης, $C_{on\text{-}path}$ ο φόρτος εντός μονοπατιού, $C_{off\text{-}path}$ ο φόρτος εκτός μονοπατιού, C_{total} ο συνολικός φόρτος εντός και εκτός μονοπατιού και τέλος $C_{useful} = C_{on\text{-}path}$ ο φόρτος του μονοπατιού που θα είχαμε αν δεν υπήρχαν απώλειες από τις διακλαδώσεις. Σε αυτή την περίπτωση (αν δηλαδή σε ένα μονοπάτι δεν υπάρχουν διακλαδώσεις), ο φόρτος διακλάδωσης είναι ίσος με την μονάδα.

Επεκτεινόμενοι τώρα σε επίπεδο μονοπατιού, ο φόρτος διακλάδωσης μονοπατιού ορίζεται ως το γινόμενο όλων των επιμέρους φόρτων διακλάδωσης κάθε επιπέδου στο μονοπάτι:

$$B = \prod b_i$$

Με χρήση των παραπάνω τύπων είμαστε πλέον σε θέση να εισαγάγουμε μια νέα έννοια φόρτου. Η έννοια αυτή αναφέρεται ως φόρτος μονοπατιού (path effort) ή *F*. Ο φόρτος αυτός είναι ανάλογος της προσπάθειας βαθμίδας, *f* μιας πύλης, αλλά σε επίπεδο μονοπατιού. Ο τύπος του φόρτου μονοπατιού παρουσιάζεται παρακάτω:

$$F = GBH$$

Όπου *F* ο φόρτος μονοπατιού, *G* ο λογικός φόρτος μονοπατιού *B* ο φόρτος διακλάδωσης μονοπατιού και *H* ο ηλεκτρικός φόρτος μονοπατιού. Ο φόρτος μονοπατιού *F* δεν αποτελεί μετρικό καθυστέρησης ενός μονοπατιού, ωστόσο αποτελεί απαιτούμενη έννοια για την ελαχιστοποίηση της καθυστέρησης του.

Η συνολική καθυστέρηση του μονοπατιού, *D* υπολογίζεται ως το άθροισμα των καθυστερήσεων κάθε επιπέδου λογικής του μονοπατιού.

$$D = \sum d_i$$

Αντίστοιχα με την έκφραση της καθυστέρησης λογικής πύλης, και μονοπατιού, θα ορίσουμε την παραπάνω καθυστέρηση ως άθροισμα της καθυστέρησης φόρτου (effort delay) μονοπατιού ή D_F, και της παρασιτικής καθυστέρησης (parasitic delay) μονοπατιού ή P:

$$D = D_F + P$$

Αντίστοιχα η καθυστέρηση φόρτου (effort delay) μονοπατιού ή DF μπορεί να υπολογιστεί ως:

$$D_F = \sum g_i h_i$$

Όπου D_F η καθυστέρηση φόρτου (effort delay) μονοπατιού και g_i , h_i ο λογικός και ηλεκτρικός φόρτος του εκάστοτε επιπέδου.

Ομοίως η παρασιτική καθυστέρηση (parasitic delay) μονοπατιού ή P μπορεί να υπολογιστεί ως:

$$P = \sum p_i$$

Όπου P η παρασιτική καθυστέρηση (parasitic delay) μονοπατιού, και p_i η παρασιτική καθυστέρηση της πύλης του εκάστοτε επιπέδου.

Από τα παραπάνω γίνεται σαφές ότι για να ελαχιστοποιήσουμε την καθυστέρηση ενός μονοπατιού N επιπέδων, οφείλουμε να μειώσουμε την καθυστέρηση φόρτου D_F καθώς η παρασιτική καθυστέρηση εξαρτάται από τον τύπο πυλών και παραμένει σταθερή. Αποδεικνύεται ότι η ελάχιστη καθυστέρηση διαδρομής, σε μια διαδρομή N λογικών επιπέδων, είναι ελάχιστη όταν όλα τα λογικά επίπεδα i σε μια διαδρομή εμφανίζουν τον ίδιο φόρτο επιπέδου f (stage effort). Σαν αποτέλεσμα η ελάχιστη καθυστέρηση διαδρομής επιτυγχάνεται όταν ο φόρτος επιπέδου ισούται με την τιμή:

$$\hat{f} = g_i h_i = F^{\frac{1}{N}} \forall i$$

Όπου $\hat{f}$ ο ιδανικός φόρτος επιπέδου, g_i, h_i ο λογικός και ηλεκτρικός φόρτος της πύλης του εκάστοτε επιπέδου και τέλος F ο φόρτος του μονοπατιού. Από τα παραπάνω δύναται να εξαγάγουμε την έκφραση της ελάχιστης επιτεύξιμης καθυστέρησης $\hat{D}$ κατά μήκος ενός μονοπατιού:

$$\hat{D} = NF^{\frac{1}{N}} + P$$

Όπου $\hat{D}$ η ιδανική (ελάχιστη επιτεύξιμη) καθυστέρηση του μονοπατιού, N ο αριθμός των λογικών επιπέδων, F ο φόρτος του μονοπατιού, και P η παρασιτική καθυστέρηση του μονοπατιού.

Τώρα απομένει να επιλέξουμε κατάλληλα μεγέθη για τα τρανζίστορ σε κάθε επίπεδο λογικής κατά μήκος του μονοπατιού ώστε να επιτύχουμε ελάχιστη καθυστέρηση στο μονοπάτι. Σκοπός είναι να ισοκατανέμουμε τον φόρτο που υπολογίσαμε προηγουμένως στα επίπεδα του μονοπατιού. Από την παραπάνω εξίσωση υπολογισμού του φόρτου επιπέδου (stage effort) που αντιστοιχεί στην ελάχιστη καθυστέρηση, και επιλύοντας ως προς τον ηλεκτρικό φόρτο (electrical effort) προκύπτει:

$$\hat{h}_i = \frac{F^{\frac{1}{N}}}{g_i}$$

Όπου $\hat{h}_i$ ο ιδανικός ηλεκτρικός φόρτος της πύλης, F ο φόρτος του μονοπατιού, N ο αριθμός των λογικών επιπέδων και τέλος g_i ο λογικός φόρτος της εκάστοτε πύλης.

Μέσω του παραπάνω τύπου μπορούμε πλέον να καθορίσουμε τα μεγέθη των τρανζίστορ κατά μήκος του μονοπατιού. Αρχίζοντας από το τέλος του και

κινούμενοι προς τα πίσω κατά μήκος του μονοπατιού, εφαρμόζουμε τον παρακάτω μετασχηματισμό χωρητικότητας:

$$C_{in\,i} = \frac{g_i C_{out_i}}{\hat{f}}$$

Κατά την εφαρμογή της μεθόδου πρέπει να είμαστε σίγουροι ότι η αρχική χωρητικότητα εισόδου του κυκλώματος παραμένει αναλλοίωτη καθώς αυτή αποτελεί μέρος των προδιαγραφών του κυκλώματος. Αποδεικνύεται ότι για να ισχύει αυτός ο περιορισμός πρέπει κατά τη διάρκεια της εφαρμογής του μετασχηματισμού βελτιστοποίησης, να παραμένουν οι εκάστοτε φόρτοι διακλάδωσης (branching efforts) κάθε πύλης που ανήκει στο μονοπάτι σταθερά. Αυτό συνεπάγεται, πως η αλλαγή της χωρητικότητας εισόδου μιας πύλης του μονοπατιού, συνεπάγεται και την κατά τον ίδιο παράγοντα αλλαγή της χωρητικότητας εισόδου όλων των πυλών που βρίσκονται στην ίδια διακλάδωση με την πύλη αυτή.

Επαυξημένη Μέθοδος

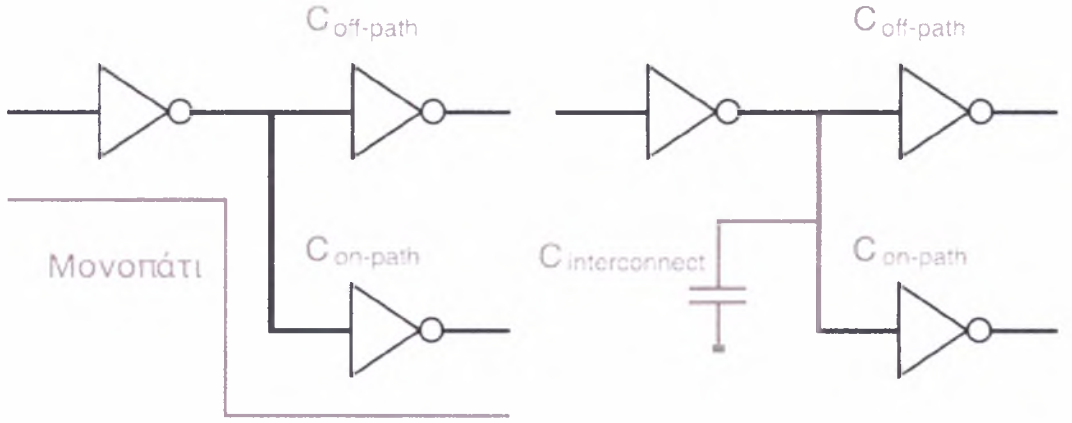
Αν και τα οφέλη που προκύπτουν από την γενικευμένη μέθοδο του λογικού φόρτου είναι σημαντικά, υπάρχει ένα χαρακτηριστικό των σύγχρονων ολοκληρωμένων κυκλωμάτων το οποίο η μέθοδος αυτή αγνοεί. Το χαρακτηριστικό αυτό αποτελεί τη συνήθη τροχοπέδη των σύγχρονων ολοκληρωμένων κυκλωμάτων και δεν είναι άλλο από την διασύνδεση. Όπως αναφέρθηκε προηγουμένως, η διασύνδεση αποτελεί τον κυριότερο παράγοντα καθυστέρησης των σύγχρονων ολοκληρωμένων κυκλωμάτων, γεγονός που έχει σαν αποτέλεσμα η προηγούμενη μέθοδος να κρίνεται ανεπαρκής. Επιπλέον, η ακρίβεια της μεθόδου τίθεται υπό αμφισβήτηση καθώς αυτή δεν λαμβάνει υπόψη της μεγάλο αριθμό φαινομένων με σημαντικότερο όλων αυτό της χωρητικότητας διασύνδεσης.

Αποτέλεσμα των παραπάνω ήταν η προσπάθεια για επέκταση της μεθόδου ώστε αυτή να περιλαμβάνει τα φαινόμενα διασύνδεσης. Παρατηρήθηκε ότι αυτό θα ήταν δυνατό με μια σχετικά μικρή επέκταση μιας εκ των εννοιών του λογικού φόρτου. Η έννοια αυτή είναι η έννοια του φόρτου διακλάδωσης (branching effort). Όπως είδαμε προηγουμένως η έννοια του φόρτου διακλάδωσης αντικατοπτρίζει τον βαθμό διακλάδωσης της εξόδου μιας πύλης σε εισόδους άλλων πυλών και ισούται με:

$$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}} = \frac{C_{total}}{C_{useful}}$$

Όπου b ο φόρτος διακλάδωσης (branching effort) στην έξοδο της πύλης, $C_{on\text{-}path}$ ο φόρτος εντός μονοπατιού, $C_{off\text{-}path}$ ο φόρτος εκτός μονοπατιού, C_{total} ο συνολικός φόρτος εντός και εκτός μονοπατιού και τέλος $C_{useful} = C_{on\text{-}path}$ ο φόρτος του μονοπατιού που θα είχαμε αν δεν υπήρχαν απώλειες από τις διακλαδώσεις. Δεδομένου ότι μόνο ένα ποσοστό του συνολικού ρεύματος οδήγησης κατευθύνεται κατά μήκος του συγκεκριμένου μονοπατιού αναφοράς σε κάθε επίπεδο, ενώ το υπόλοιπο κατευθύνεται εκτός αυτού μέσω των διάφορων

διακλαδώσεων στο μονοπάτι, ο φόρτος διακλάδωσης ουσιαστικά προσφέρει μια ένδειξη των ποσοστών αυτών μέσω των αντίστοιχων χωρητικοτήτων C_{total} και C_{useful}. Έτσι προχωρούμε στην αναδιατύπωση του τύπου του φόρτου διακλάδωσης μέσω της αναθεώρησης του λόγου της συνολικής χωρητικότητας C_{total} προς την εντός μονοπατιού χωρητικότητα C_{useful} ή $C_{on\text{-}path}$. Όπως είδαμε προηγουμένως, οι αγωγοί διασύνδεσης παρουσιάζουν χωρητικότητα, η ύπαρξη της χωρητικότητα αυτής επηρεάζει το κύκλωμα, οπότε προφανώς η διασύνδεση μπορεί να θεωρηθεί ότι αποτελεί παράγοντα που επηρεάζει το ποσοστό του συνολικού ρεύματος οδήγησης το οποίο κατευθύνεται κατά μήκος του συγκεκριμένου μονοπατιού και του ποσοστού του ρεύματος οδήγησης που κατευθύνεται εκτός αυτού. Έτσι με δεδομένη χωρητικότητα διασύνδεσης, έστω $C_{διασύνδεσης}$ ή $C_{interconnect}$, μπορούμε να θεωρήσουμε την νέα ολική χωρητικότητα C_{total} αντί για ίση με $C_{on\text{-}path}+C_{off\text{-}path}$ πλέον ως ίση με $C_{total} = C_{on\text{-}path}+C_{off\text{-}path}+C_{interconnect}$. Τα παραπάνω παρουσιάζονται στο παρακάτω σχήμα:



Σχηματική αναπαράσταση εισαγωγής χωρητικότητας διασύνδεσης στη συνολική χωρητικότητα του κυκλώματος

Έτσι πλέον ο ανανεωμένος τύπος του φόρτου διακλάδωσης μετατρέπεται σε:

$$b = \frac{C_{on-path} + C_{off-path} + C_{interconnect}}{C_{on-path}}$$

Μέσω αυτής της επέκτασης μπορούμε να προχωρήσουμε πλέον σε βελτιστοποίηση κυκλωμάτων με μεγαλύτερη ακρίβεια. Παρόλα αυτά, πλέον στους υπολογισμούς μας εισέρχεται το πρόβλημα του υπολογισμού της εκάστοτε χωρητικότητας διασύνδεσης, η έκταση (για κάθε διακλάδωση, σε κάθε επίπεδο κτλ) και η πολυπλοκότητα του υπολογισμού των οποίων αποτελεί πραγματική πρόκληση.

Παρόλα αυτά στη συνέχεια θα ακολουθήσει μια μεθοδολογία για τον υπολογισμό των χωρητικοτήτων αυτών, όπως και για την εφαρμογή της μεθόδου στο σύνολό της.

Λογισμικό Βελτιστοποίησης

Για τις ανάγκες τις διατριβής αυτής έγινε χρήση καθιερωμένων εργαλείων αυτόματης σχεδίασης όπως επίσης ανάπτυξη μιας νέας σειράς εργαλείων ικανών να υποστηρίξουν τα διάφορα βήματα της μεθόδου βελτιστοποίησης. Στο κεφάλαιο αυτό θα ασχοληθούμε με εργαλεία αυτά. Το κεφάλαιο ξεκινά παρέχοντας στον αναγνώστη μια εισαγωγή στις έννοιες της αυτοματοποιημένης σχεδίασης και αυτοματοποιημένου ηλεκτρονικού σχεδιασμού, συνεχίζει με μια εισαγωγική περιγραφή δύο βασικών βιομηχανικών εφαρμογών, των Design Vision από την εταιρία Synopsys και SoC Encounter από την Cadence, που χρησιμοποιούνται από την μέθοδο βελτιστοποίησης, και τέλος παρουσιάζει με λεπτομέρεια την εφαρμογή που αναπτύχθηκε για την τελική εφαρμογή της επαυξημένης μεθόδου του λογικού φόρτου.

Αυτοματοποιημένη Σχεδίαση

Εισαγωγή

Αυτοματοποιημένη σχεδίαση κυκλωμάτων ονομάζουμε την διαδικασία κατά την οποία κάνουμε χρήση διαφόρων εργαλείων αυτοματισμού ώστε να μειώσουμε το κατά δυνατόν τον φόρτο εργασίας που απαιτείται για τον πλήρη σχεδιασμό ενός κυκλώματος. Τα εργαλεία αυτά έχει καθιερωθεί να αναφέρονται ως εργαλεία αυτοματοποιημένου ηλεκτρονικού σχεδιασμού (**E**lectronic **D**esign **A**utomation) ή κοινώς ως EDA.

Η υιοθέτηση των εργαλείων αυτοματοποιημένου ηλεκτρονικού σχεδιασμού έχει αυξηθεί καθώς η τεχνολογία μεταβαίνει σε μειούμενες συνεχώς διαστάσεις , στις οποίες πέρα από την σημαντική αύξηση του αριθμού των τρανζίστορ και της πολυπλοκότητας που προκύπτει, παρατηρούνται φαινόμενα τα οποία θα ήταν απαγορευτικό να υπολογίζονται «με το χέρι». Το κόστος άδειας χρήσης πολλών εκ των εργαλείων αυτών ανέρχεται σε εξαιρετικά υψηλά επίπεδα της τάξης ακόμα και των εκατομμυρίων ευρώ. Παρόλα αυτά τα οφέλη που προκύπτουν δικαιολογούν το κόστος αυτό είτε λόγω τις εξοικονόμησης σε εργατικό προσωπικό, είτε λόγω της λεπτομέρειας στην οποία μπορούμε να εισέλθουμε κατά την σχεδίαση. Γενικά, σήμερα πλέον απαιτείται η χρήση τέτοιων εργαλείων κατά την σχεδίαση. Το κόστος αδειών χρήσης φαντάζει δυσβάσταχτο για μια νέα εταιρία που θέλει να εισέλθει στην βιομηχανία σχεδίασης ηλεκτρονικών κυκλωμάτων, παρόλα αυτά καθένας πρέπει να έχει κατά νου ότι τα εργαλεία αυτά αποτελούν ίσως την πιο πολύπλοκη έκφραση των σύγχρονων εργαλείων σχεδίασης βοηθούμενης από υπολογιστή (**C**omputer **A**ided **D**esign) ή CAD.

Synopsys Design Vision

Για την εφαρμογή της επαυξημένης μεθόδου Λογικού Φόρτου είναι προηγουμένως απαραίτητη η σύνθεση σε επίπεδο μεταφοράς καταχωρητή (RTL synthesis) των κυκλωμάτων μας. Στη μεθοδολογία μας περιλαμβάνεται ως εργαλείο RTL σύνθεσης το εργαλείο Design Vision της Synopsys. Το εργαλείο αυτό περιλαμβάνει το γνωστό συνθέτη (compiler) Design Compiler και την τελευταία έκδοση αυτού γνωστού και ως Design Compiler Ultra (DC Ultra) και αποτελεί μέρος της πλατφόρμας Galaxy Design Platform. Δεδομένης της περιγραφής της λειτουργικής συμπεριφοράς ενός κυκλώματος σε μια αντίστοιχη γλώσσα, όπως στο παράδειγμά μας Verilog, με χρήση του εργαλείου αυτού δύναται η σύνθεση ενός κυκλώματος που υλοποιεί την λειτουργία που περιγράφεται.

Cadence SoC Encounter

Επόμενο στάδιο της σύνθεσης αποτελεί συνήθως η τοποθέτηση και διασύνδεση (place & route) των στοιχείων που επιλέχτηκαν από την προηγούμενη εφαρμογή ώστε να υλοποιήσουν την λειτουργία του κυκλώματος που ζητήθηκε κατά το στάδιο της σύνθεσης. Για την τοποθέτηση και διασύνδεση των στοιχείων αυτών θα κάνουμε χρήση της εφαρμογής που ονομάζεται SoC Encounter από την εταιρία Cadence. Οι σύγχρονοι σχεδιαστές ολοκληρωμένων κυκλωμάτων βρίσκονται αντιμέτωποι με μια πλειάδα προβλημάτων όπως είναι η διαρκής σμίκρυνση των διαστάσεων των τεχνολογιών υλοποίησης, η αύξηση του μεγέθους των κυκλωμάτων, και η αύξηση της πολυπλοκότητας αυτών. Όλα τα παραπάνω έχουν οδηγήσει στο να θεωρείται πλέον ως πραγματική πρόκληση ο υπολογισμός και ο χειρισμός των φαινομένων διασύνδεσης. Επιπροσθέτως, οι σύγχρονοι κατασκευαστές ηλεκτρονικών κατά την μετάβασή τους στις σύγχρονες τεχνολογίες υλοποίησης απαιτούν συστήματα σχεδίασης τα οποία να είναι σε θέση να προσφέρουν τη μεγαλύτερη δυνατή ποιότητα (ταχύτητα, μέγεθος και κατανάλωση ολοκληρωμένου) και ακρίβεια κατά την μετάβαση της σχεδίασης στη σιλικόνη, επαλήθευση ακριβείας της λειτουργίας του ολοκληρωμένου, διασύνδεση πληρώντας τα απαραίτητα κριτήρια ακεραιότητας σήματος, όπως επίσης εφαρμογή των τελευταίων καινοτομιών χαμηλής κατανάλωσης και απόδοσης παραγωγής (yield). Η εφαρμογή SoC Encounter έχει κατασκευαστεί ώστε να παρέχει τα παραπάνω.

Λογισμικό βελτιστοποίησης επαυξημένης μεθόδου Λογικού Φόρτου

Το λογισμικό βελτιστοποίησης επαυξημένης μεθόδου Λογικού Φόρτου αποτελεί επέκταση μιας εφαρμογής η οποία ξεκίνησε στα πλαίσια μιας έρευνας για απλή εφαρμογή της γενικευμένης μεθόδου του λογικού φόρτου πίσω στο 2003 σε συνεργασία με το συμφοιτητή μου Κουκά Ιάσων. Η εφαρμογή αυτή, άτυπα γνωστή ως CircuitOptimizerLE, συνέχισε την πορεία της μέσα από πολλές εκδόσεις αυτά χρόνια ώστε τελικά να καταλήξει σε μια σουίτα εργαλείων ικανή να παρέχει μια πλειάδα διαφορετικών αυτοματισμών, όπως εξομοίωση

κυκλωμάτων, πιστοποίηση λειτουργίας κτλ. Η έκδοση που αναπτύχθηκε για τις ανάγκες αυτής της διατριβής αποσκοπεί στην βελτιστοποίηση κυκλωμάτων μέσω της επαυξημένης μεθόδου Λογικού Φόρτου που παρουσιάστηκε προηγουμένως. Άτυπα ο όρος COLEI (**C**ircuit **O**ptimizer **L**ogical **E**ffort **I**nterconnect) χρησιμοποιείται ορισμένες φορές αναφερόμενος στην εφαρμογή αυτή. Αφαιρετικά, σκοπός της εφαρμογής είναι η μέτρηση και βελτίωση της καθυστέρησης ενός κυκλώματος CMOS, διατηρώντας τη λειτουργικότητα και τις προδιαγραφές χωρητικότητας εισόδου για κάθε είσοδό του, αναλλοίωτες. Είσοδος της εφαρμογής αποτελεί κυρίως το κύκλωμα προς βελτιστοποίηση σε μορφή SPICE αρχείου ενώ ως έξοδος λαμβάνουμε την αρχική καθυστέρηση του κυκλώματος, την μετά την βελτιστοποίηση καθυστέρηση, τα προτεινόμενα μεγέθη τρανζίστορ στις πύλες καθώς και μια πλειάδα στατιστικών και λεπτομερειών. Εδώ πρέπει να αναφερθεί το γεγονός της αναφοράς των καθυστερήσεων σε μονάδες καθυστέρησης αντιστροφέα αναφοράς για την εκάστοτε τεχνολογία υλοποίησης. Μετατροπή σε απόλυτη καθυστέρηση μπορεί να γίνει με χρήση της τεχνικής που αναφέρθηκε κατά τον υπολογισμό της καθυστέρησης πύλης προηγουμένως.

Στάδια κατά την βελτιστοποίηση μέσω της εφαρμογής

Παρακάτω παρουσιάζονται τα διάφορα στάδια κατά την βελτιστοποίηση μέσω της εφαρμογής. Η εφαρμογή δέχεται σαν παράμετρο το όνομα του αρχείου SPICE το οποίο περιγράφει το προς βελτιστοποίηση κύκλωμα. Η διαδικασία βελτιστοποίησης μπορεί να χωριστεί στα παρακάτω στάδια:

1. Ανάγνωση από το αρχείο που περιγράφει το κύκλωμα
2. Αναπαράσταση του κυκλώματος σε δομή δεδομένων
3. Κατηγοριοποίηση των πυλών σε επίπεδα λογικής
4. Εφαρμογή του αλγορίθμου βελτιστοποίησης
5. Αναίρεση τελευταίων αλλαγών
6. Κβάντιση των μεγεθών των τρανζίστορ
7. Εκτύπωση αποτελεσμάτων.

Πλέον πολύπλοκο στάδιο της βελτιστοποίησης αποτελεί το στάδιο εφαρμογής του αλγορίθμου βελτιστοποίησης. Το στάδιο αυτό θα αναλυθεί σε περισσότερα υποστάδια καθώς κατά τη διάρκεια αυτού γίνεται εκτέλεση πλειάδας ανεξάρτητων μεταξύ τους αλγορίθμων κατά τη διαδικασία βελτιστοποίησης.

Ανάγνωση από το αρχείο που περιγράφει το κύκλωμα

Πρώτο στάδιο αποτελεί η ανάγνωση του αρχείου εισόδου. Το αρχείο αυτό περιέχει την περιγραφή του κυκλώματος σε μορφή SPICE. Δεδομένου ότι ο αλγόριθμός μας εφαρμόζεται σε επίπεδο αφαίρεσης πυλών, το αρχείο πρέπει να περιγράφει το κύκλωμα σε επίπεδο πυλών. Τα αρχεία SPICE υποστηρίζουν τον ορισμό υψηλότερων δομών όπως είναι οι πύλες μέσω της λέξης κλειδιού *.subckt* που αντιστοιχεί στην έννοια του υποκυκλώματος (sub-circuit). Έτσι ορίζοντας όλα τα είδη πυλών που περιέχει το προς περιγραφή κύκλωμα ως

υποκυκλώματα, αυτές μπορούν να χρησιμοποιηθούν επανειλημμένα και να θεωρηθεί πως το κύκλωμα αποτελείται από πύλες. Οι ορισμοί αυτοί κατά την μεθοδολογία μας δίνονται από την βιβλιοθήκη της τεχνολογίας υλοποίησης (design kit) όπως θα δούμε στο κεφάλαιο παρουσίασης της μεθοδολογίας. Εντός ενός υποκυκλώματος (*.subckt*) γίνεται η τοπολογική περιγραφή κάθε υποκυκλώματος (άρα στην περίπτωσή μας πύλης) ενώ καθορίζονται και τα μεγέθη των τρανζίστορ από τα οποία αποτελείται η εκάστοτε πύλη. Από τα υποκυκλώματα αυτά θα εξαγάγουμε τόσο το είδος των πυλών όπως επίσης και τα αρχικά μεγέθη χωρητικοτήτων εισόδου κάθε πύλης. Παρακάτω παρουσιάζουμε ένα παράδειγμα μιας πύλης λογικού Ή ονόματι OR2X4,

.subckt OR2X4 Y A B VDD GND

M0 Y net9 GND GND N_18_MM l=0.18u w=2.4u

M1 VDD net9 Y VDD P_18_MM l=0.18u w=3.6u

...

.ends OR2X4

Στο παραπάνω παράδειγμα οι μεταβλητές w περιγράφουν τα πλάτη των *p-MOS* (P_18_MM) και *n-MOS* (N_18_MM) τρανζίστορ της πύλης αντίστοιχα.

Επιπλέον στο αρχείου εισόδου πρέπει να ορίζονται οι είσοδοι του κυκλώματος. Κάθε είσοδος αντιστοιχίζεται σε έναν αριθμητικό κωδικό και η περιγραφή της αρχίζει με ένα όνομα που αρχίζει με το γράμμα v ακολουθούμενο από τον αριθμητικό κωδικό ο οποίος θα χρησιμοποιηθεί στην περιγραφή της τοπολογίας του υπόλοιπου κυκλώματος. Παρακάτω παρουσιάζουμε ένα παράδειγμα τέτοιων εισόδων.

v1 dat_binary_i[0] 0 pulse(0 5 15n)

Όπου v1 είναι το αναγνωριστικό τις εισόδου, ενώ τα υπόλοιπα χαρακτηριστικά σημαίνουν ότι υπάρχει σήμα από τον κόμβο κυκλώματος με αναγνωριστικό 0 στον κόμβο κυκλώματος με αναγνωριστικό dat_binary_i[0]. Επιπλέον το χαρακτηριστικό pulse(0 5 15n) σημαίνει ότι η είσοδος δέχεται σήμα 0 έως 5 Volts με περίοδο 15 νανοδευτερολέπτων, το τελευταίο όμως αυτό χαρακτηριστικό (pulse) αγνοείται από την εφαρμογή μας και παραμένει απλά λόγω απαίτησης συμβατότητας με τους κανόνες αρχείων SPICE.

Έχοντας ορίσει τα παραπάνω μπορούμε πλέον να προχωρήσουμε κυρίως περιγραφή του κυκλώματος, κατά την οποία η θέση κάθε στοιχείου του κυκλώματος ορίζεται ως εξής. Αρχικά δίνεται ένα αναγνωριστικό του στοιχείου που θα χρησιμοποιείται από το κύκλωμα, ακολουθεί το αναγνωριστικό του κόμβου ο οποίος αποτελεί έξοδό του, ακολουθούν οι είσοδοί του και τέλος το είδος του στοιχείου το οποίο πρέπει να αντιστοιχεί σε ένα στοιχείο που έχει

οριστεί προηγουμένως με τον ορισμό *.subckt*. Εδώ πρέπει αν σημειωθεί ότι συνήθως τα δύο προτελευταία στοιχεία αποτελούν τον κόμβο τροφοδοσίας και γείωσης του στοιχείου αντίστοιχα. Παρακάτω παρουσιάζουμε ένα παράδειγμα τέτοιας περιγραφής:

XU379 ShiftRegNextxDN_107_ ParResponseOutxDI_107_ ShiftRegPrxDP_108_ ResponseShiftEnxSI 1494 1495 ADD3X1HV

Η παραπάνω περιγραφή χρησιμοποιείται για να περιγράψει ένα στοιχείο τύπου ADD3X1HV με το αναγνωριστικό XU379 το οποίο έχει ως έξοδο τον κόμβο ShiftRegNextxDN_107_ και ως εισόδους τους κόμβους ParResponseOutxDI_107_, ShiftRegPrxDP_108_ και ResponseShiftEnxSI. Οι κόμβοι 1494 και 1495 αντιστοιχούν στους κόμβους πηγής και γείωσης αντίστοιχα.

Στην συνέχεια ορίζονται οι χωρητικότητες διασύνδεσης του κυκλώματος. Αυτές χρησιμοποιούν τον προκαθορισμένο τρόπο περιγραφής χωρητικοτήτων. Έτσι κάθε τέτοια περιγραφή ξεκινάει με το αναγνωριστικό της χωρητικότητας, ακολουθεί ο κόμβος στον οποίο καταλήγει η χωρητικότητα, στη συνέχεια βρίσκεται ο κόμβος από τον οποίο ξεκινά η χωρητικότητα και τέλος η τιμή της χωρητικότητας. Παρακάτω παρουσιάζουμε ένα παράδειγμα τέτοιας περιγραφής:

c346 0 ShiftRegNextxDN_41_ 0.000422pf

Η παραπάνω περιγραφή χρησιμοποιείται για να περιγράψει μια χωρητικότητα με το αναγνωριστικό c346 η οποία ξεκινά από τον κόμβο 0 και καταλήγει στον κόμβο ShiftRegNextxDN_41_ με χωρητικότητα 0.000422 Πίκο-φαράντ (pf).

Ομοίως ορίζουμε τις χωρητικότητες εξόδου του κυκλώματος.

<u>Αναπαράσταση του κυκλώματος σε δομή δεδομένων</u>

Αρχικά, κατά τη διάρκεια ανάγνωσης του αρχείου που περιγράφει το κύκλωμα δημιουργείται η δομή δεδομένων που το αναπαριστά. Αρχικά δημιουργούνται οι κόμβοι εισόδου, οι οποίοι τοποθετούνται σε μία αρχική λίστα. Ταυτόχρονα χρησιμοποιείται και μια δομή δυαδικού δένδρου αναζήτησης στο οποίο αποθηκεύονται όλοι οι κόμβοι με κλειδί αναζήτησης το όνομά τους, για γρήγορο εντοπισμό τους

Στη συνέχεια γίνεται ανάγνωση των υποκυκλωμάτων (*.subckt*) ώστε να εντοπιστούν οι δομές (διάφορα στοιχεία και πύλες) που θα συναντήσουμε στο κύκλωμα (αντίγραφα των οποίων θα αποτελούν όλα τα στοιχεία του κυκλώματος). Χάριν ευκολίας θα αναφερόμαστε στις δομές αυτές ως πύλες. Έτσι όταν επιχειρούμε να σχηματίσουμε το κύκλωμα από την περιγραφή των συνδέσεων μεταξύ των πυλών και των κόμβων γίνεται αναζήτηση στο δυαδικό δένδρο για την ύπαρξη του εκάστοτε κόμβου της εκάστοτε πύλης, και εφόσον δεν βρίσκεται ήδη ο κόμβος στο δένδρο (δηλαδή εφόσον είναι η πρώτη

εμφάνισή του στην περιγραφή του κυκλώματος), δημιουργείται και τοποθετείται με βάση το όνομα – κλειδί του. Στον κόμβο εξόδου που δίνεται από την περιγραφή της πύλης, θέτουμε την αντίστοιχη πύλη ως είσοδό του, ενώ στους κόμβους εισόδου προσθέτουμε την πύλη ως έξοδο των κόμβων. Οι συνδέσεις μεταξύ κόμβων και πυλών είναι διπλής κατεύθυνσης, δηλαδή υπάρχει πρόσβαση και από τον κάθε κόμβο προς τις πύλες στις οποίες ο κόμβος αποτελεί είτε είσοδο είτε έξοδο, καθώς και από την κάθε πύλη προς τους κόμβους εισόδου και εξόδου της. Με αυτό το τρόπο πλέον έχουμε αναπαραστήσει όλες τις συνδέσεις και ως εκ τούτου έχουμε αναπαραστήσει όλο το κύκλωμα σε μορφή δεδομένων. Τέλος θέτουμε τις χωρητικότητες των κόμβων που ορίζει το αρχείο του κυκλώματος.

Κατηγοριοποίηση των πυλών σε επίπεδα λογικής

Όπως αναφέρθηκε προηγουμένως, οι κόμβοι εισόδου του κυκλώματος έχουν αποθηκευτεί σε μια αρχική λίστα. Η λίστα αυτή αποτελεί το πρώτο εκ μιας σειράς επιπέδων μιας λίστας η οποία περιέχει το σύνολο των στοιχείων (πυλών) και των κόμβων του κυκλώματος. Θεωρούμε τώρα ότι η λίστα που περιέχει τους κόμβους εισόδου αποτελεί το επίπεδο μηδέν του κυκλώματος. Αντίστοιχα οι πύλες οι οποίες συνδέονται με κόμβους μόνο του επιπέδου μηδέν θεωρούμε ότι ανήκουν στο επίπεδο ένα του κυκλώματος ενώ οι κόμβοι οι οποίοι αποτελούν εξόδους των πυλών αυτών θεωρούμε ότι ανήκουν στο επίπεδο ένα. Με άλλα λόγια, το επίπεδο κάθε πύλης ισούται με το μέγιστο των επιπέδων των κόμβων οι οποίοι αποτελούν εισόδους της πύλης αυτής προσαυξημένο κατά ένα, ενώ το επίπεδο των κόμβων ισούται με το επίπεδο της πύλης στην οποία αποτελούν έξοδο, με εξαίρεση τους κόμβους του επιπέδου μηδέν. Τα παραπάνω ισχύουν για τα συνδυαστικά κυκλώματα καθώς για τα ακολουθιακά η έννοια του επιπέδου μιας πύλης είναι πολύπλοκη. Τα ακολουθιακά κυκλώματα εκτός από λογικές πύλες, περιέχουν στοιχεία μνήμης από τα οποία ξεκινούν βρόχοι ανάδρασης προς πύλες προηγουμένων επιπέδων. Δεδομένου ότι ο παραπάνω τρόπος θα προκαλούσε αέναη επανάληψη, θεωρούμε πως οι κόμβοι εξόδου των στοιχείων μνήμης αποτελούν και αυτοί κύριες εισόδους του κυκλώματος. Με τον τρόπο αυτό πλέον μπορούμε να χειριζόμαστε όλα τα κυκλώματα με συνδυαστική λογική.

Εφαρμογή του αλγορίθμου βελτιστοποίησης

Ο αλγόριθμος βελτιστοποίησης αποτελεί έναν αναδρομικό αλγόριθμο μέχρι την ικανοποίηση κάποιων κριτηρίων τερματισμού. Έτσι ο επαναληπτικός αλγόριθμος σε κάθε αναδρομή εντοπίζει το πιο αργό μονοπάτι του κυκλώματος και ελαχιστοποιεί την καθυστέρησή του μέσω της επαυξημένης μεθόδου του λογικού φόρτου. Στην περίπτωση όπου το μονοπάτι παραμένει το πιο αργό του κυκλώματος μετά την βελτιστοποίηση, ο αλγόριθμος τερματίζεται καθώς το μονοπάτι είναι αδύνατον να γίνει πιο γρήγορο. Στην περίπτωση που το μονοπάτι που βελτιστοποιήσαμε έχει γίνει πιο γρήγορο συνεχίζουμε την εφαρμογή του αλγορίθμου με τον νέο ποιό αργό μονοπάτι εκτός αν η προηγούμενη βελτιστοποίηση προκάλεσε την αύξηση της συνολικής

καθυστέρησης του κυκλώματος (παρότι μείωσε την καθυστέρηση του προηγούμενου χειρότερου μονοπατιού) χειροτερεύοντας την καθυστέρηση ενός άλλου.

Γενικότερα η εφαρμογή του αλγορίθμου βελτιστοποίησης μπορεί να χωριστεί σε παραπάνω στάδια τα οποία παρουσιάζουμε στη συνέχεια:

1. Υπολογισμός των βασικών εννοιών του Λογικού Φόρτου
2. Εύρεση των *N* κρίσιμων μονοπατιών
3. Εφαρμογή επαυξημένης μεθόδου Λογικού Φόρτου στα κρίσιμα μονοπάτια
4. Έλεγχος κριτηρίων τερματισμού

Ο λόγος που στο δεύτερο βήμα βρίσκουμε τα *N* χειρότερα μονοπάτια αντί του ενός χειρότερου, είναι ότι στατιστικά σε ορισμένα κυκλώματα έχουμε καλύτερα αποτελέσματα αν βελτιστοποιούμε αντί για ένα, και τα *N* χειρότερα μονοπάτια σε έναν κύκλο για κάποιες τιμές του *N*. Θα αναφερθούμε με μεγαλύτερη λεπτομέρεια στη βελτιστοποίηση των *N* μονοπατιών παρακάτω.

Μετά από κάθε κύκλο, ελέγχεται αν ικανοποιήθηκαν τα κριτήρια τερματισμού, και αν αυτό συνέβη σταματάει και η επανάληψη του κύκλου εργασιών.

Υπολογισμός των βασικών εννοιών του Λογικού Φόρτου

Κατά τη διάρκεια εφαρμογής του αλγορίθμου βελτιστοποίησης γίνεται υπολογισμός των βασικών εννοιών του Λογικού Φόρτου, δηλαδή εννοιών όπως ο Λογικός Φόρτος της εκάστοτε πύλης, χωρητικότητες εισόδου και εξόδου κτλ. Ο υπολογισμός κάποιων εξ αυτών των εννοιών γίνεται μόνο μια φορά (π.χ Λογικός Φόρτος πύλης) ενώ ο υπολογισμός κάποιων άλλων επαναλαμβάνεται σε κάθε επανάληψη του αλγορίθμου (π.χ χωρητικότητες εισόδου). Τα αποτελέσματά τους αποθηκεύονται στους αντίστοιχους κόμβους και πύλες της δομής του κυκλώματος. Έχοντας πλέον όλες τις βασικές έννοιες και ποσότητες της μεθόδου του Λογικού Φόρτου για κάθε πύλη και κόμβο, μπορούμε πλέον να προχωρήσουμε στον υπολογισμό της συνολικής καθυστέρησης του κάθε μονοπατιού του κυκλώματος.

Εύρεση των *N* κρίσιμων μονοπατιών

Αναφερόμενοι στην καθυστέρηση των σύγχρονων κυκλωμάτων, συνήθως αναφερόμαστε στην καθυστέρηση που εμφανίζει το ποίο αργό μονοπάτι αυτού. Το μονοπάτι αυτό έχει επικρατήσει να αναφέρεται ως κρίσιμο μονοπάτι. Παρόλα αυτά, έχει γενικώς παρατηρηθεί ότι η βελτιστοποίηση αποκλειστικά και μόνο του κρίσιμου μονοπατιού δεν αποφέρει συνήθως τα ίδια ικανοποιητικά αποτελέσματα όπως η βελτιστοποίηση όλων των μονοπατιών. Δεδομένης της πολυπλοκότητας όμως που παρουσιάζουν τα σύγχρονα κυκλώματα η βελτιστοποίηση δεν είναι εφαρμόσιμη σε όλα τα μονοπάτια. Επιπλέον έχει παρατηρηθεί ότι μετά την βελτιστοποίηση ενός αριθμού *N* χειρότερων από πλευράς καθυστέρησης μονοπατιών, το κέρδος από την

περαιτέρω βελτιστοποίηση επιπλέον μονοπατιών αρχίζουν να φθίνουν σημαντικά. Έτσι συνηθίζεται πλέον να γίνεται βελτιστοποίηση ενός αριθμού, έστω *N*, «κρίσιμων μονοπατιών».

Για την εύρεση των *N* αυτών κρίσιμων μονοπατιών, θα χρησιμοποιήσουμε την έννοια της Ελάττωσης Διακλάδωσης (branch slack). Η έννοια αυτή παρουσιάστηκε από τον Yun-Cheng Ju και χρησιμοποιείται από τον αντίστοιχο αλγόριθμο ώστε με βάση ένα αργό μονοπάτι (στην περίπτωσή μας θα χρησιμοποιήσουμε το κρίσιμο μονοπάτι) και με γνωστές τις καθυστερήσεις όλων των πυλών να βρει με πολύ γρήγορους υπολογισμούς το αμέσως λιγότερο αργό μονοπάτι. Επαναλαμβάνοντας αυτή τη διαδικασία *N*-1 φορές, βρίσκοντας σε κάθε βήμα το αμέσως λιγότερο αργό μονοπάτι, καταλήγουμε στα *N* κρισιμότερα μονοπάτια. Ο πίνακας που ακολουθεί αποδεικνύει την υπεροχή της μεθόδου σε σχέση με τους αλγορίθμους αναζήτησης Καλύτερου Πρώτου (Best First) και αναζήτησης κατά βάθος (Depth First). Παρακάτω παρουσιάζονται οι χρόνοι εκτέλεσης σε δευτερόλεπτα κεντρικής μονάδας επεξεργασίας για τρία μεγάλα κυκλώματα αναφοράς καθώς και το ποσοστό των άστοχων

Κύκλωμα	Αλγόριθμος	Αριθμός Μονοπατιών N		
		N = 50	N = 500	N = 5000
		Χρόνος Εκτέλεσης (Ποσοστό Άστοχων Αναζητήσεων)		
C5315	Best First	0.10 (3%)	1.19 (3%)	40.24 (3%)
	Depth First	0.18 (58%)	0.91 (61%)	16.84 (64%)
	Αλγόριθμος 7.1	0.06 (0%)	0.52 (0%)	5.15 (0%)
C6288	Best First	0.22 (4%)	2.46 (5%)	61.30 (6%)
	Depth First	0.64 (70%)	8.06 (78%)	94.83 (83%)
	Αλγόριθμος 7.1	0.17 (0%)	1.49 (0%)	14.52 (0%)
C7552	Best First	0.08 (4%)	1.00 (3%)	32.77 (3%)
	Depth First	0.13 (70%)	1.15 (56%)	11.56 (59%)
	Αλγόριθμος 7.1	0.03 (0%)	0.44 (0%)	4.44 (0%)

αναζητήσεων.

Χρόνοι εκτέλεσης σε δευτερόλεπτα κεντρικής μονάδας επεξεργασίας και ποσοστό άστοχων αναζητήσεων

Εφαρμογή επαυξημένης μεθόδου Λογικού Φόρτου στα κρίσιμα μονοπάτια

Από το προηγούμενο βήμα έχουμε πλέον στη διάθεση μας τα N πιο κρίσιμα μονοπάτια του κυκλώματος. Στη συνέχεια εφαρμόζουμε σε αυτά την επαυξημένη μέθοδο Λογικού Φόρτου ξεκινώντας από το λιγότερο αργό εκ των N μονοπατιών και καταλήγοντας στο πλέον αργό. Ακολουθούμε την στρατηγική αυτή καθώς επιτυγχάνουμε έτσι στατιστικά καλύτερα αποτελέσματα καθώς βελτιστοποιώντας ένα λιγότερο αργό μονοπάτι είναι πιθανό μεν να βλάψουμε ένα ήδη ποιό αργό, όμως προσωρινά, καθώς αργότερα θα ακολουθήσει βελτιστοποίηση του. Έτσι, ακόμα και αν χειροτερέψουμε το πλέον αργό μονοπάτι, μετά τη βελτιστοποίησή του θα γίνει βέλτιστο, αναιρώντας πιθανώς μέρος της βελτιστοποίησης του λιγότερο αργού (το οποίο έχει περιθώρια να γίνει πιο αργό καθώς δεν είναι το πιο αργό μονοπάτι του κυκλώματος) όμως συνολικά μειώνοντας την ολική καθυστέρηση του κυκλώματος. Σε αντίθετη περίπτωση, αν δηλαδή βελτιστοποιούσαμε από το πλέον αργό προς το λιγότερο αργό, το πιθανότερο θα ήταν οι βελτιστοποιήσεις των λιγότερο αργών μονοπατιών που θα γίνονταν μετά τις βελτιστοποιήσεις των περισσότερο αργών, να εξαφάνιζαν τα όποια οφέλη αποκομίσαμε στην συνολική καθυστέρηση του κυκλώματος.

Η εφαρμογή της επαυξημένης μεθόδου Λογικού Φόρτου πραγματοποιεί με σειρά τους κάτωθι υπολογισμούς:

1. Φόρτος Διακλάδωσης Μονοπατιού (B)
2. Λογικός Φόρτος Μονοπατιού (G)
3. Ηλεκτρικός Φόρτος Μονοπατιού (H)
4. Φόρτος Μονοπατιού (F)
5. Ιδανικός Φόρτος Επιπέδου (f)
6. Κατάλληλη αλλαγή των μεγεθών της εκάστοτε πύλης του μονοπατιού και των πυλών των διακλαδώσεων.

Στο βήμα αυτό μέσω της σχέσης παρακάτω σχέσης προχωρούμε στην αλλαγή των χωρητικοτήτων εισόδων κάθε πύλης (εκτός αν είσοδος αυτής αποτελεί είσοδος του κυκλώματος):

$$C_{in_i} = \frac{g_i C_{out_i}}{f}$$

Όπου C_{in_i} η χωρητικότητα εισόδου της εκάστοτε πύλης, g_i ο λογικός φόρτος αυτής, C_{out_i} η χωρητικότητα εξόδου αυτής και f ο ιδανικός φόρτος που υπολογίσαμε προηγουμένως. Εκτός από την αλλαγή της χωρητικότητας εισόδου της εκάστοτε πύλης του μονοπατιού στο τρέχον επίπεδο, πρέπει να προχωρήσουμε σε αλλαγή της χωρητικότητας εισόδου των πυλών που βρίσκονται στην ίδια διακλάδωση με την πύλη που ανήκει στο μονοπάτι και η οποία μόλις άλλαξε μέγεθος. Εάν $C_{πριν}$ είναι η χωρητικότητα εισόδου της πύλης του μονοπατιού πριν την αλλαγή και $C_{μετα}$ η χωρητικότητά της μετά την αλλαγή, η χωρητικότητά των πυλών της διακλάδωσης πρέπει να πολλαπλασιαστεί με

τον παράγοντα $\frac{C_{πριν}}{C_{μετα}}$. Μετά την ολοκλήρωση της διαδικασίας για κάθε επίπεδο λογικής του μονοπατιού, η καθυστέρησή του έχει ελαχιστοποιηθεί χωρίς να έχουμε αλλάξει την χωρητικότητα εισόδου του κυκλώματος.

Έλεγχος κριτηρίων τερματισμού

Μετά το πέρας της βελτιστοποίησης ή εφαρμογή ελέγχει εάν το κύκλωμα βελτιώθηκε μετά την τελευταία αναδρομή. Εάν υπήρξε βελτίωση, επαναλαμβάνουμε την εφαρμογή της βελτιστοποίησης προς αναζήτηση καλύτερης. Εάν όχι, το κύκλωμα που αποτελεί τη βέλτιστη λύση είναι αυτό της προηγούμενης αναδρομής οπότε και ο αλγόριθμος τερματίζει. Παρόλα αυτά σαν βέλτιστη λύση πρέπει να επιστρέψουμε το κύκλωμα της προηγούμενης αναδρομής. Για να γίνει αυτό θα έπρεπε να διατηρούμε ένα αντίγραφο του κυκλώματος της προτελευταίας αναδρομής. Αυτό όμως θα προκαλούσε το σχεδόν διπλασιασμό της χρησιμοποιούμενης μνήμης της εφαρμογής γεγονός μη αποδεκτό καθώς επιθυμούμε η εφαρμογή να διαχειρίζεται ιδιαίτερα μεγάλα κυκλώματα με βέλτιστο τρόπο. Έτσι ευθετήσαμε μια έξυπνη προσέγγιση για την εξαγωγή του βέλτιστου κυκλώματος χωρίς την ανάγκη να διατηρούμε αντίγραφα του κυκλώματος. Η προσέγγιση παρουσιάζεται παρακάτω:

Αναίρεση Τελευταίων Αλλαγών

Η προσέγγιση που ακολουθούμε για την εξαγωγή του βέλτιστου κυκλώματος από την τελευταία αναδρομή του αλγορίθμου αποτελεί ουσιαστικά μια αναίρεση μέρους (ή και του συνόλου) των αλλαγών της τελευταίας αναδρομής. Σε κάθε αναδρομή, πραγματοποιούνται αλλαγές στα *N* μονοπάτια που εντοπίστηκαν ως κρίσιμα. Έτσι, πριν την αλλαγή σε κάποιο μονοπάτι, αποθηκεύουμε μόνο τα μεγέθη των πυλών που πρόκειται να αλλαχθούν και όχι το ολικό κύκλωμα. Οι πύλες αυτές αποτελούν ουσιαστικά τις πύλες που βρίσκονται στο μονοπάτι αλλά και αυτές οι οποίες ανήκουν στις διακλαδώσεις του μονοπατιού. Αφού εντοπιστούν οι πύλες αυτές, στη συνέχεια αποθηκεύονται σε μια λίστα ώστε να είναι δυνατή η αναίρεση των όποιων αλλαγών πραγματοποιηθούν σε αυτές στην επόμενη εφαρμογή του αλγορίθμου. Αντί να διατηρούμε αντίγραφα για όλα τα *N* μονοπάτια, και στη συνέχεια μετά την τελευταία βελτιστοποίηση να επαναφέρουμε τις παλιές τιμές στις πύλες, η αντιγραφή των υποψήφιων πυλών ενός μονοπατιού γίνεται πριν βελτιστοποιηθεί το κύκλωμα αυτό και όχι πριν την έναρξη του συνολικού αλγορίθμου βελτιστοποίησης για οποιοδήποτε μονοπάτι. Έτσι μπορούμε πλέον να αναιρέσουμε τις αλλαγές που προκάλεσε η βελτιστοποίηση των τελευταίων μόνο μονοπατιών και όχι των συνολικά *N* κρίσιμων. Ο λόγος που ακολουθούμε αυτή τη προσέγγιση είναι η πιθανότητα το κύκλωμα να βρέθηκε σε μια ενδιάμεση καλύτερη κατάσταση, την οποία όμως κατάσταση να μετέβαλε η βελτιστοποίηση ενός επόμενου από τα *N* κρίσιμα μονοπάτια.

Αποθηκεύοντας τα αντίγραφα των υποψήφιων πυλών κάθε μονοπατιού αμέσως πριν τη βελτιστοποίηση του, αποθηκεύουμε τις χωρητικότητες των πυλών μετά τις προηγούμενες βελτιστοποιήσεις, με αποτέλεσμα, κάνοντας αναίρεση με αντίστροφη σειρά, να μπορούμε να φέρουμε το κύκλωμα σε οποιαδήποτε από τις καταστάσεις που βρέθηκε προηγουμένως κατά τη διάρκεια της βελτιστοποίησης των N κρίσιμων μονοπατιών.

Κβάντιση των μεγεθών των τρανζίστορ

Όπως αναφέραμε προηγουμένως, η εφαρμογή βελτιώνει την καθυστέρηση που εμφανίζει το κύκλωμα προτείνοντας νέα (διαφορετικά) μεγέθη τρανζίστορ στις πύλες. Λόγο της συνεχής φύσης των εννοιών που εισαγάγαμε για την εφαρμογή της επαυξημένης μεθόδου του λογικού Φόρτου, η εφαρμογή θα μπορούσε να προτείνει τιμή για το μέγεθος τρανζίστορ οποιουδήποτε εύρους. Όμως, τα μεγέθη των τρανζίστορ μιας πύλης δεν είναι δυνατόν να λάβουν οποιαδήποτε πραγματική τιμή, αλλά εξαρτώνται από την τεχνολογία υλοποίησης. Για το λόγο αυτό είναι απαραίτητο να γίνει κβαντισμός των μεγεθών αυτών. Με την έννοια κβαντισμός εννοούμε ένα ελάχιστο βήμα, πολλαπλάσια του οποίου λαμβάνουν όλες οι πύλες. Προφανώς ο κβαντισμός εξαρτάται από την τεχνολογία υλοποίησης. Έτσι, μετά το πέρας του αλγορίθμου, κάνουμε διαπέραση του κυκλώματος και θέτουμε σε όλες τις πύλες, την πλησιέστερη κβαντισμένη τιμή που υποστηρίζει η τεχνολογία, χάνοντας κάποιο ποσοστό από την επιτάχυνση που επετεύχθη αλλά προτείνοντας μεγέθη πολύ κοντά στην πραγματικότητα.

Εκτύπωση αποτελεσμάτων

Κατά τη διάρκεια εκτέλεσης της μεθόδου γίνεται παρουσίαση λεπτομερειών της κατάστασης του κυκλώματος και της μεθόδου. Η εκτύπωση γίνεται σε μορφή κώδικα HTML ώστε να είναι ευανάγνωστη από το χρήστη.

Μεθοδολογία βελτιστοποίησης

Έχοντας πλέον παρουσιάσει τις απαιτούμενες έννοιες, την επαυξημένη μέθοδο Λογικού Φόρτου καθώς και τα διάφορα εργαλεία που μας επιτρέπουν να επιτελέσουμε μια πλειάδα διεργασιών πάνω σε κάποιο κύκλωμα, απομένει να παρουσιάσουμε την ολοκληρωμένη μεθοδολογία για την βελτίωση καθυστέρησης ενός κυκλώματος. Για τις ανάγκες παρουσίασης της μεθοδολογίας θα χρησιμοποιήσουμε ένα μετατροπέα δυαδικού συστήματος σε δεκαδικό σύστημα δυαδικής αναπαράστασης (Binary to Binary Coded Decimal) σαν κύκλωμα επίδειξης. Η λειτουργία του κυκλώματος αυτού είναι άσχετη με την μεθοδολογία που έχουμε αναπτύξει και οποιοδήποτε άλλο κύκλωμα θα μπορούσε να έχει χρησιμοποιηθεί. Επιπλέον σαν τεχνολογία υλοποίησης θα χρησιμοποιήσουμε την UMC 0.18, ενώ σαν μέθοδο σχεδίασης θα χρησιμοποιήσουμε την ημί-προσαρμοσμένη. Στην συνέχεια παρουσιάζεται η μεθοδολογία που χρησιμοποιείται η οποία έχει πάρα πολλά κοινά στοιχεία με την κλασσική «Κορυφή προς κάτω» (top-down) προσέγγιση που χρησιμοποιείται πολλές φορές κατά την υλοποίηση κυκλωμάτων. Αφαιρετικά, η προσέγγιση αυτή τείνει να ξεκινάει με μια βασική περιγραφή του κυκλώματος (κορυφή) και στην συνέχεια να προσεγγίζει κάθε μέρος αυτού όλο και με μεγαλύτερη λεπτομέρεια. Έτσι η μεθοδολογία μας μπορεί να χωριστεί σε ένα σύνολο βημάτων τα οποία μετά την επιτυχή διεκπεραίωσή τους συνθέτουν ένα κύκλωμα βελτιστοποιημένο μέσω της επαυξημένης μεθόδου Λογικού Φόρτου από μια αρχική αφηρημένη περιγραφή του κυκλώματος αυτού. Σε αυτό το σημείο πρέπει να αναφερθεί ότι τα παραδείγματα που θα ακολουθήσουν θα αποτελούν είτε καθαρά φανταστικά ενδεικτικά παραδείγματα (στα ελληνικά) είτε πραγματικά παραδείγματα χαρακτήρες των οποίων αποκρύπτονται με τη χρήση αστερίσκων (*) ώστε να προστατευτούν τα πνευματικά δικαιώματα των εταιριών προϊόντα των οποίων χρησιμοποιούνται, καθώς ο υπογεγραμμένος έχει υπογράψει συμφωνία μη αποκάλυψης (non disclosure agreement) με τις ανωτέρω. Παρόλα αυτά η τρόπος που παρουσιάζεται η μεθοδολογία γίνεται άριστα κατανοητή και επεκτάσιμη από οποιονδήποτε έχει πρόσβαση στα αντίστοιχα εργαλεία είτε ανταγωνιστικά αυτών.

<u>Περιγραφή κυκλώματος σε μορφή μετάβασης επιπέδου καταχωρητή</u>

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, κατά την σχεδίαση του κυκλώματος συνηθίζεται να ορίζεται πρωταρχικά μια αφηρημένη προσέγγιση του κυκλώματος. Η προσέγγιση αυτή είναι συνήθως ο σχεδιασμός επιπέδου μεταφοράς καταχωρητή (**R**egister **T**ransfer **L**evel) ή RTL, όπου οι σχεδιαστές περιγράφουν την μετάβαση των σημάτων μεταξύ των διάφορων καταχωρητών, όπως επίσης και τον τρόπο με τον οποίο κάθε υποσύστημα επηρεάζει τα σήματα αυτά. Συνηθέστερες γλώσσες που χρησιμοποιούνται για την περιγραφή αυτή είναι οι Verilog και VHDL. Δεδομένης της περιγραφής ενός κυκλώματος σε μορφή Verilog ή VHDL μπορούμε πλέον να συνεχίσουμε στην σύνθεση του κυκλώματος.

Σύνθεση κυκλώματος

Έχοντας πλέον την περιγραφή σε επίπεδο μεταφοράς καταχωρητή μπορούμε να συνεχίσουμε προς σύνθεση του κυκλώματος. Έστω ότι η περιγραφή του κυκλώματος είναι στη γλώσσα Verilog και το αρχείο της περιγραφής που περιγράφει το μετατροπέα που χρησιμοποιούμε ως κύκλωμα αναφοράς ονομάζεται «binary_to_bcd.v». Το αρχείο αυτό παρουσιάζεται παρουσιάζεται στο ΠΑΡΑΡΤΗΜΑ. Κατά την σύνθεση του κυκλώματος ένα υποσύστημα του design vision γνωστό και ως design compiler προχωρεί σε σύνθεση του υπό περιγραφή κυκλώματος κάνοντας χρήση των πυλών και στοιχείων που παρέχονται από την τεχνολογία υλοποίησης. Στο σημείο αυτό πρέπει να σημειώσουμε ότι λόγω τις ιδιαιτερότητας της επαυξημένης μεθόδου Λογικού Φόρτου οι πύλες και τα στοιχεία που θα πρέπει να χρησιμοποιηθούν πρέπει να περιοριστούν μόνο σε αυτά τα οποία εφαρμόζεται η επαυξημένη μέθοδος. Ο λόγος που πρέπει να περιορίσουμε τον εύρος των διαθέσιμων επιλογών του «συνθέτη» (design compiler) είναι ότι η σύγχρονες τεχνολογίες υλοποίησης παρέχουν βιβλιοθήκες με πολλά στοιχεία στα οποία δεν δύναται να οριστεί η έννοια του λογικού φόρτου καθώς τα στοιχεία αυτά δεν αποτελούν είδη λογικών πυλών αλλά συνήθως ανώτερες κυκλωματικές δομές με πλειάδα εισόδων, εξόδων και καταστάσεων λειτουργίας. Έτσι πρέπει να θέσουμε στην εφαρμογή τους περιορισμούς κάτω από τους οποίους πρέπει να γίνει η σύνθεση σε ότι αφορά τις διαθέσιμες πύλες. Έτσι το στάδιο αυτό της βελτιστοποίησης αρχίζει μέσω της εκκίνησης την εφαρμογή Design Vision της Synopsys. Μόλις η εκκίνηση της εφαρμογής ολοκληρωθεί, η εφαρμογή πέρα από το γραφικό περιβάλλον διασύνδεσης με το χρήστη παρέχει και μια κονσόλα για την εισαγωγή εντολών. Οι παραπάνω περιορισμοί θα εισαχθούν μέσω της κονσόλας αυτής. Η εφαρμογή υποστηρίζει τον αποκλεισμό κάποιου στοιχείου από τα διαθέσιμα στοιχεία προς χρήση για σύνθεση μέσω εισαγωγής της παρακάτω εντολής στην κονσόλα:

set_dont_use { όνομα-βιβλιοθήκης/όνομα-στοιχείου ... }

Όπου όνομα-βιβλιοθήκης αποτελεί το αναγνωριστικό της βιβλιοθήκης και όνομα-στοιχείου το όνομα του στοιχείου προς αποκλεισμό. Σαν παράδειγμα παρουσιάζουμε την παρακάτω εντολή που αποκλείει τα στοιχεία **CSHCINX2 και **CSHCINX4 της βιβλιοθήκης **_1v8_25c.

*set_dont_use { **_1v8_25c/**CSHCINX2 **_1v8_25c/**CSHCINX4 }*

Στο ΠΑΡΑΡΤΗΜΑ παρουσιάζεται ο συνολικός περιορισμός που χρησιμοποιήθηκε για το συγκεκριμένο παράδειγμα.

Έχοντας πλέον ορίσει περιορισμούς στο είδος των διαθέσιμων προς σχηματισμό πυλών μπορούμε να προχωρήσουμε προς σύνθεση του κυκλώματος. Προς βέλτιστη σύνθεση του κυκλώματός μας πρέπει επίσης να ορίσουμε περιορισμούς (constraints) σε ότι αφορά την καθυστέρηση που θα εμφανίζει το κύκλωμά μας και ίσως όρια στην επιφάνεια που αυτό καταλαμβάνει. Αν δεν οριστούν τα παραπάνω το κύκλωμα που θα συντεθεί θα επιτελεί την λειτουργία η οποία ζητήθηκε αλλά όχι με βέλτιστο τρόπο καθώς ο συνθέτης (Compiler) σε

αυτή την περίπτωση θα προσπαθήσει, ορθώς όπως του ζητήθηκε, να συνθέσει ένα κύκλωμα το οποίο απλά πρέπει να υλοποιεί την εκάστοτε λογική χωρίς περιορισμούς στην ταχύτητα λειτουργίας και στην επιφάνεια την οποία αυτό καταλαμβάνει. Ο καθορισμός περιορισμών (constraints) αποτελεί πρόκληση στη σύγχρονη σχεδίαση κυκλωμάτων καθώς δεν υπάρχει αποτελεσματικός αυτοματοποιημένος τρόπος εξεύρεσης βέλτιστων περιορισμών και έτσι ο ορισμός τους επαφίεται στον εκάστοτε σχεδιαστή. Επιπλέον πρέπει να αναφερθεί ότι οι εκάστοτε περιορισμοί έχουν άμεση σχέση με το προς σύνθεση κύκλωμα και ως εκ τούτου δεν δύναται να γενικευτούν. Ο λεπτομερής τρόπος ανάθεσης περιορισμών καθώς και η τιμές αυτών αποτελούν ολόκληρο κεφάλαιο της σύγχρονης σχεδίασης και ξεφεύγει από τα όρια της διατριβής αυτής. Για το παράδειγμά μας θα χρησιμοποιήσουμε ένα και μόνο ρολόι, έστω συχνότητας 25-μεγάκυκλων (25-Megahertz) και κύκλου εργασίας (duty cycle) 50%. Το ρολόι αυτό πρέπει να συνδεθεί στην αντίστοιχη είσοδο ρολογιού του κυκλώματος «binary_to_bcd.v». Όπως παρατηρούμε από την πλήρη περιγραφή του κυκλώματος που παρουσιάζεται στο ΠΑΡΑΡΤΗΜΑ, η είσοδος αυτή ονομάζεται «clk_i». Έτσι θέτουμε το περιορισμό του ρολογιού στον συνθέτη μέσω της παρακάτω εντολής στην κονσόλα:

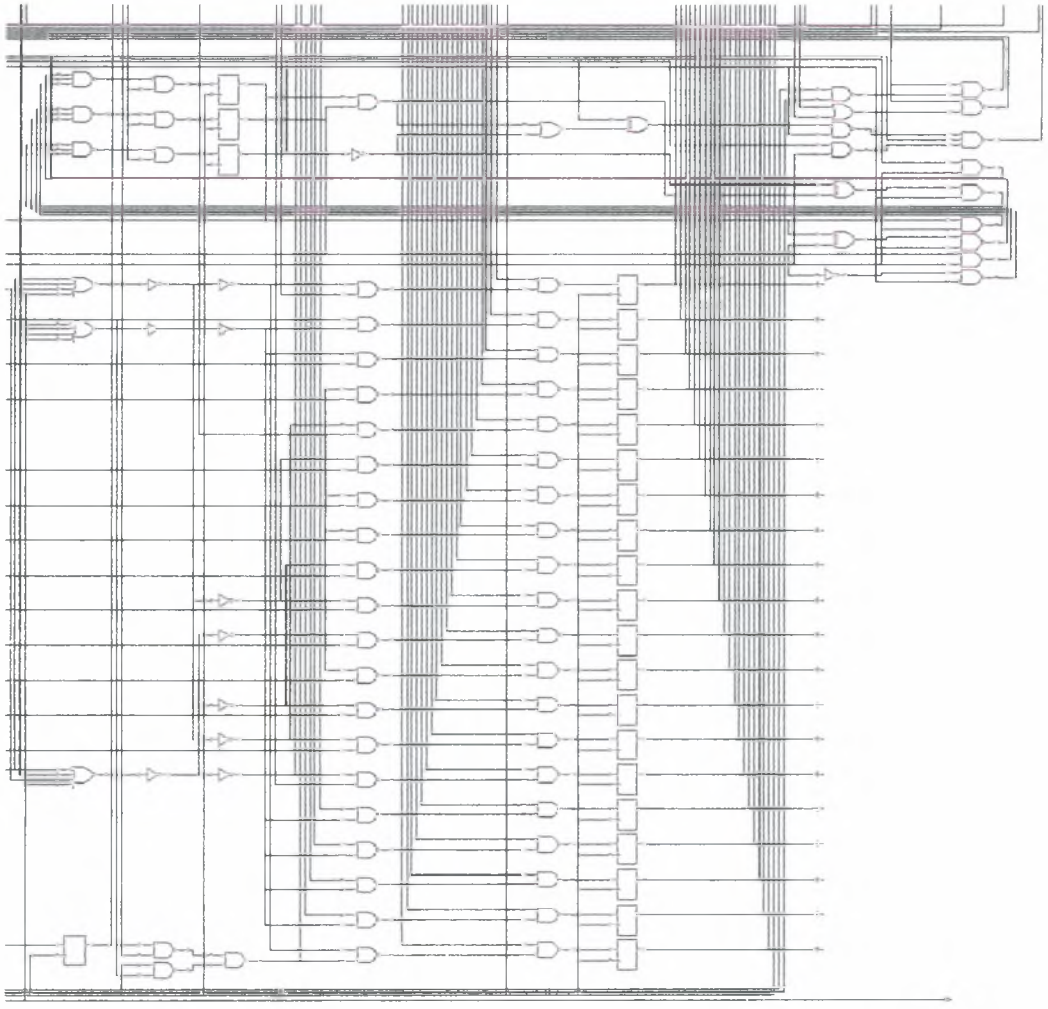
create_clock clk_i –period 40

Εδώ πρέπει να σημειωθεί ότι ο χρήστης δύναται να θέσει τους διάφορους περιορισμούς και να επιτελέσει ίσως το σύνολο των εργασιών και μέσω του γραφικού περιβάλλοντος της εφαρμογής που χρησιμοποιεί αντί της κονσόλας που χρησιμοποιούμε στο παράδειγμά μας. Ο λόγος που χρησιμοποιούμε την κονσόλα κατά την περιγραφή μας είναι ώστε να δώσουμε όσο το δυνατόν μεγαλύτερη διαχρονικότητα στην περιγραφή αυτή της μεθοδολογίας καθώς ενώ το γραφικό περιβάλλον των εφαρμογών αυτών αλλάζει συνήθως με κάθε νέα έκδοση οι εντολές μέσω κονσόλας παραμένουν συνήθως αμετάβλητες.

Επιπλέον, συνήθως ορίζονται πρόσθετοι σημαντικοί περιορισμοί όπως για παράδειγμα μέγιστη καθυστέρηση από είσοδο προς έξοδο (IO-constraints), αλλά δεν θα επεκταθούμε περαιτέρω.

Έχοντας πλέον θέσει τους περιορισμούς του κυκλώματος μένει να προχωρήσουμε σε σύνθεση του. Έτσι στο παράδειγμά μας συνθέτουμε το κύκλωμά μας με χρήση του συνθέτη (Design Compiler). Ομοίως θα μπορούσαμε να χρησιμοποιήσουμε τον Design Ultra οποίος αποτελεί ποιό ανεπτυγμένη έκδοση του προηγουμένου.

Όπως παρατηρούμε στο παρακάτω σχήμα όπου παρουσιάζεται μέρος του σχηματικού του κυκλώματος κατά την σύνθεση έχει γίνει χρήση μόνο των γνωστών πυλών και Flip-flop τύπου D.



Σχηματική αναπαράσταση μέρους του κυκλώματος όπου παρουσιάζεται η αποκλειστική χρήση πυλών και Flip-flop τύπου D

Δημιουργία λίστας στοιχείων κυκλώματος

Έχοντας ολοκληρώσει την σύνθεσή μας, πρέπει να εξαγάγουμε το συντεθειμένο κύκλωμα σε μορφή αρχείου. Η πλέον κατάλληλη μορφή αρχείου που χρησιμοποιείται σε αυτές τις περιπτώσεις είναι η αποκαλούμενη Λίστα Verilog ή Verilog netlist. Το συνολικό κύκλωμα που συντέθηκε παρουσιάζεται στο ΠΑΡΑΡΤΗΜΑ σε μορφή Λίστας Verilog.

Τοποθέτηση και διασύνδεση

Έχοντας πλέον τη Λίστα Verilog του συντεθειμένου κυκλώματος μπορούμε να προχωρήσουμε προς τοποθέτηση και διασύνδεσή του (place and route). Για την τοποθέτηση και διασύνδεση του κυκλώματος θα χρησιμοποιήσουμε την εφαρμογή SoC Encounter της Cadence. Έτσι αφού εκκινήσουμε την εφαρμογή και έχοντας ρυθμίσει κατάλληλα την τεχνολογία υλοποίησης απομένει να ορίσουμε στην εφαρμογή το κύκλωμα προς διασύνδεση και τοποθέτηση. Το κύκλωμα αυτό πρέπει να δοθεί σε μορφή λίστας Verilog όπως το κύκλωμα που σχηματίσαμε πριν. Έτσι επιλέγουμε την εισαγωγή σχεδίου (Design Import) στην εφαρμογή και θέτουμε την Λίστα Verilog του κυκλώματος. Στο παράδειγμά μας το αρχείο αυτό ονομάζεται «binary_to_bcd.v». Εδώ πρέπει να σημειωθεί ότι τα αρχεία Λίστας Verilog και περιγραφής του κυκλώματος ονομάζονται με τον ίδιο τρόπο (binary_to_bcd.v) αν και αποτελούν διαφορετικά αρχεία. Ο λόγος που χρησιμοποιήσαμε την ίδια ονομασία διακινδυνεύοντας παρερμηνεία είναι ότι συνηθίζεται κατά την ονομασία αρχείων που σχετίζονται με την περιγραφή του κυκλώματος να χρησιμοποιούμε το όνομα της ανώτερης δομής του κυκλώματος (binary_to_bcd). Βρίσκεται στην διακριτική επιθυμία του αναγνώστη η χρήση οποιασδήποτε ονοματολογίας αυτός επιθυμεί. Έτσι σαν Verilog Netlist χρησιμοποιούμε το αρχείο binary_to_bcd.v που περιέχει την Λίστα Verilog του κυκλώματος. Επιπλέον ορίζουμε τις βιβλιοθήκες μέγιστου (Max Timing Library) και ελάχιστου χρονισμού (Min Timing Library) τις:

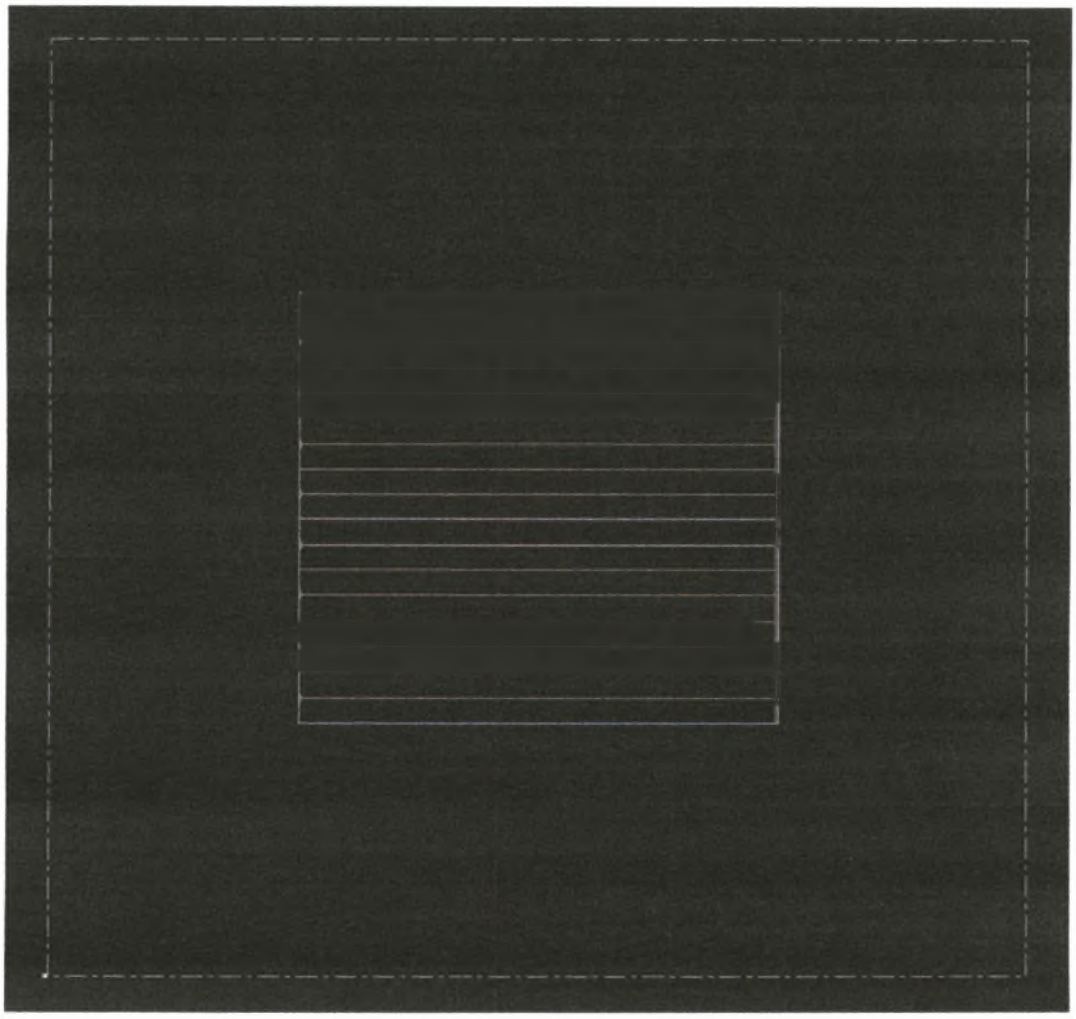
/PAR/TIM/ss_1v62_85c.lib,
/PAR/TIM/ff_1v98_0c.lib

αντίστοιχα για το παράδειγμα αυτό. Επιπλέον ορίζουμε σαν αρχείο φόρμας ανταλλαγής βιβλιοθηκών (Library Exchange Format) ή LEF το:

/PAR/DEX/umc18_61m.lef

Τα αρχεία αυτά (LEF) χρησιμοποιούνται γενικά για να περιγράψουν αφαιρετικά τα υποσυστήματα που περιλαμβάνονται σε μια βιβλιοθήκη. Τα υποσυστήματα αυτά συνήθως περιγράφονται μόνο σε επίπεδο εισόδων και εξόδων και όχι λεπτομερώς με αποτέλεσμα τα αρχεία αυτά να είναι πολύ χρήσιμα καθώς μπορούν να χρησιμοποιηθούν για την ανταλλαγή βιβλιοθηκών.

Μετά τον ορισμό των παραπάνω σχηματίζεται η επιφάνεια του ολοκληρωμένου όπως φαίνεται παρακάτω.

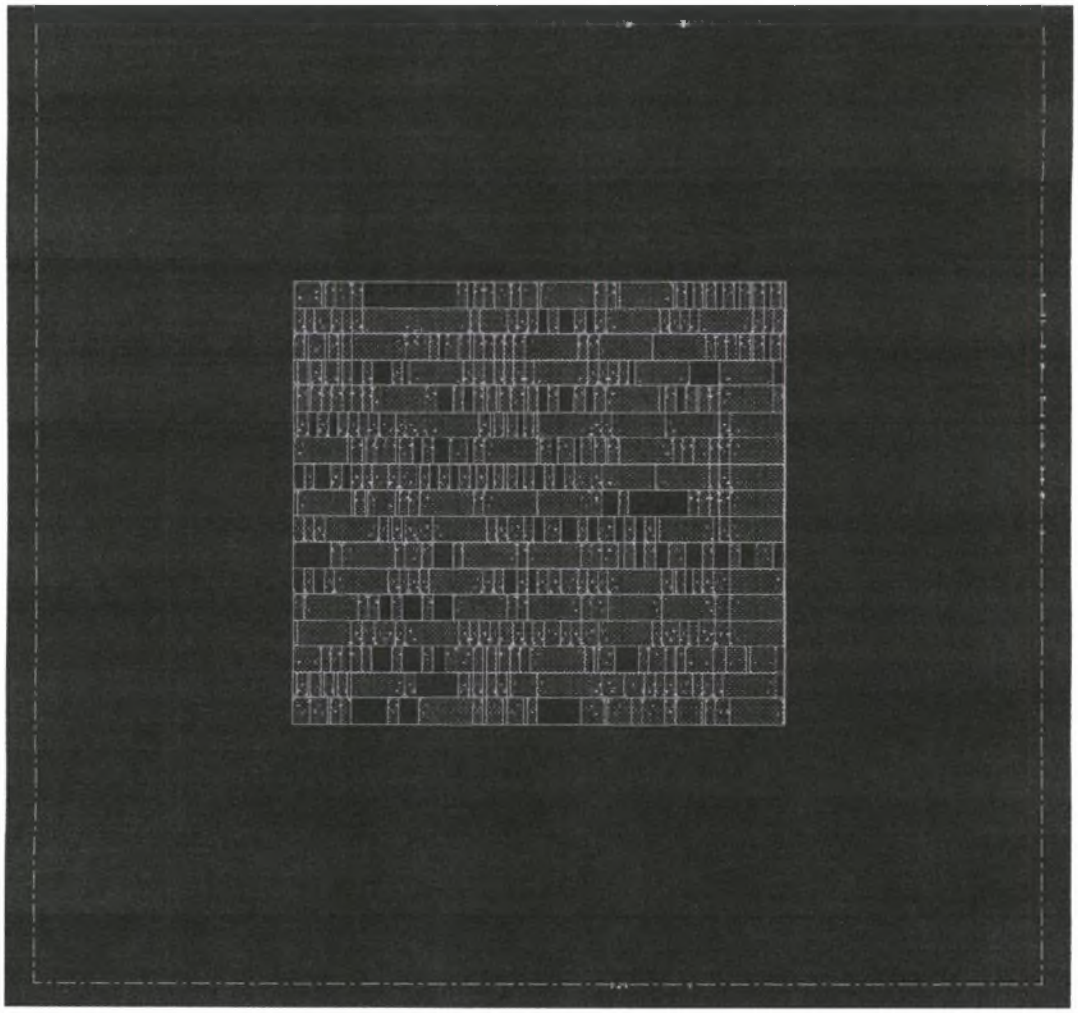


Κάτοψη διαθέσιμης επιφάνειας ολοκληρωμένου κυκλώματος

Παραπάνω παρατηρούμε την κάτοψη της διαθέσιμης επιφάνειας ολοκληρωμένου κυκλώματος. Στο κέντρο της επιφάνειας παρατηρούμε το χώρο που θα χρησιμοποιηθεί για την τοποθέτηση των στοιχείων που απαρτίζουν το κύκλωμά μας. Εδώ πρέπει να σημειωθεί ότι ο εκάστοτε σχεδιαστής δύναται να τροποποιήσει τις διαστάσεις του ολοκληρωμένου.

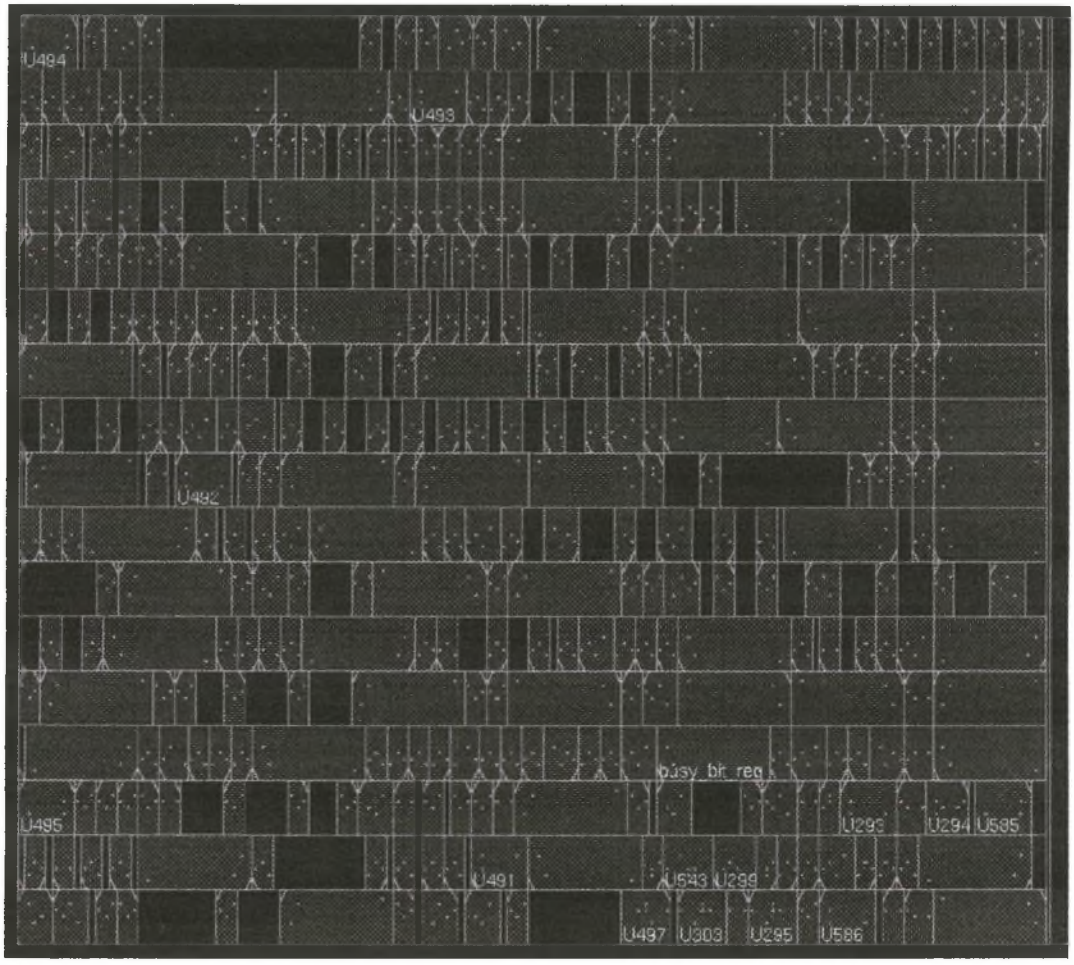
Πλέον είμαστε έτοιμοι να τοποθετήσουμε (place) τα διάφορα στοιχεία που αποτελούν το κύκλωμά μας στην διαθέσιμη επιφάνεια του ολοκληρωμένου. Στο παράδειγμά μας θα χρησιμοποιήσουμε την ημί-προσαρμοσμένη σχεδίαση. Έτσι επιλέγουμε Place (Τοποθέτηση) και στην συνέχεια Standard Cells and Blocks (κελιά και στοιχεία ημι-προσαρμοσμένης σχεδίασης). Με το τρόπο αυτό τοποθετήσαμε τα κελιά από τα οποία απαρτίζεται το κύκλωμά μας.

Η επιφάνεια του ολοκληρωμένου μετά την τοποθέτηση των κελιών παρουσιάζεται παρακάτω:



Επιφάνεια ολοκληρωμένου μετά την τοποθέτηση των κελιών του κυκλώματος

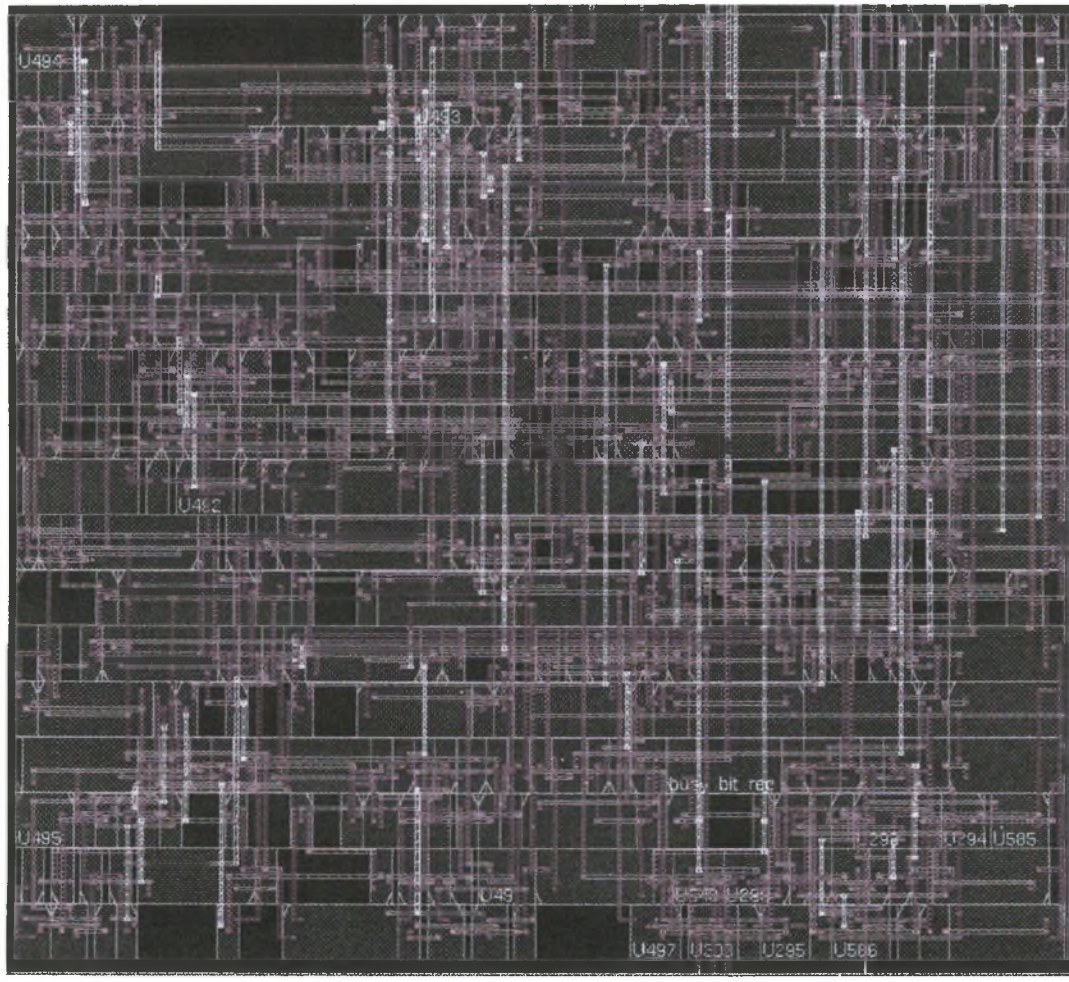
Εστιάζοντας παραπάνω στο κέντρο του ολοκληρωμένου μπορούμε να διακρίνουμε τα κελιά αυτά:



Κεντρική περιοχή ολοκληρωμένου όπου παρουσιάζονται τα κελιά που αποτελούν το κύκλωμα

Έχοντας ολοκληρώσει την τοποθέτηση των κελιών που αποτελούν το κύκλωμά μας προχωρούμε προς διασύνδεση (routing) των κελιών αυτών. Έτσι επιλέγουμε Route (διασύνδεση) και στην συνέχεια Trial Route (δοκιμαστική διασύνδεση). Τα εργαλεία αυτά συνήθως παρέχουν πλειάδα επιλογών και ρυθμίσεων για την διασύνδεση των κελιών. Στο παράδειγμά μας χρησιμοποιούμε την επιλογή της δοκιμαστικής διασύνδεσης (Trial Route) καθώς αυτή αποτελεί καλή επιλογή για τις περιπτώσεις επιδείξεων καθώς προσφέρει ικανοποιητικά αποτελέσματα χωρίς μεγάλο χρόνο εκτέλεσης και την ανάγκη να εισέλθουμε σε λεπτομερείς ρυθμίσεις.

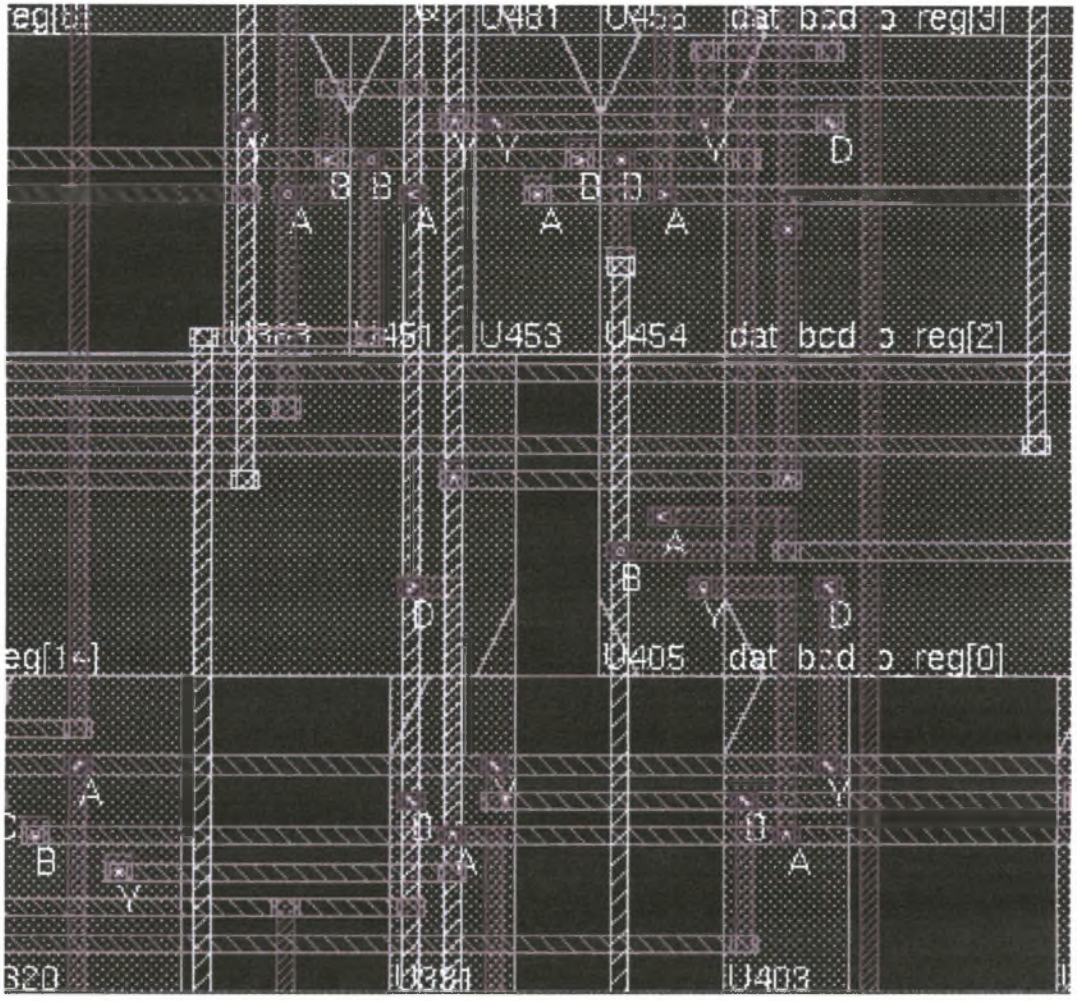
Το αποτέλεσμα της διασύνδεσης παρουσιάζεται παρακάτω:



Κύκλωμα αναφοράς μετά το πέρας της διαδικασίας τοποθέτησης και διασύνδεσης (place & route)

Εστιάζοντας παραπάνω σε μια περιοχή του ολοκληρωμένου μπορούμε να διακρίνουμε τόσο τα επιμέρους κελιά που αντιστοιχούν σε στοιχεία του κυκλώματος που περιγράφονται στη Λίστα Verilog του κυκλώματος αναφοράς, όσο και τις γραμμές διασύνδεσης.

Για παράδειγμα εστιάζουμε στην παρακάτω περιοχή:



Περιοχή εστίασης όπου παρουσιάζονται με μεγαλύτερη ακρίβεια τα στοιχεία και οι συνδέσεις που το αποτελούν

Για παράδειγμα στην παραπάνω περιοχή παρατηρούμε στο κάτω δεξιό άκρο το στοιχείο U403 το οποίο έχει τρείς συνδέσεις (A,B,Y) από τις οποίες για παράδειγμα η πρώτη (A) συνδέεται στην σύνδεση Y του στοιχείου U405. Αναφερόμενοι στην Λίστα Verilog (ΠΑΡΑΡΤΗΜΑ) εντοπίζουμε τα στοιχεία στα οποία αναφέρεται το παράδειγμά μας.

NAND2X1 U403 (.A(n212), .B(n213), .Y(n233));

Και

NAND2X1 U405 (.A(dat_bcd_o[0]), .B(n304), .Y(n212));

Παρατηρώντας τα παραπάνω μπορούμε να εξαγάγουμε ότι η σύνδεση που συνδέει τα A και Y των στοιχείων U403 και U405 αντίστοιχα ονομάζεται n212. Έχοντας πλέον στην διάθεσή μας ένα ολοκληρωμένο κύκλωμα μπορούμε πλέον

να συνεχίσουμε προς τον υπολογισμό των χωρητικοτήτων που παρουσιάζει η διασύνδεσή του.

Εξαγωγή χωρητικοτήτων

Για την εξαγωγή των χωρητικοτήτων της διασύνδεσης του κυκλώματος επιλέγουμε Extract RC και θέτουμε το επιθυμητό όνομα αρχείου χωρητικοτήτων, έστω «binary_to_bcd.cap». Ο λόγος που δεν χρησιμοποιήσαμε το ποιό ακριβές εργαλείο Fire & Ice Extract RC είναι ότι το προηγούμενο κρίνεται αρκετά ικανοποιητικό για τις ανάγκες επίδειξης του παραδείγματος όντας σημαντικά απλούστερο. Το αρχείο που προκύπτει περιέχει τις χωρητικότητες όλων των συνδέσεων του κυκλώματος.

Επεκτείνοντας το παράδειγμα που αναφέρθηκε προηγουμένως εντοπίζουμε στο αρχείο χωρητικοτήτων τη σύνδεση n212 η οποία συνδέει τα A και Y των στοιχείων U403 και U405 αντίστοιχα:

n212 0.000670 0.004014 0.004684 5.240 1.278e-04 1

Η φόρμα του αρχείου χωρητικοτήτων διασύνδεσης είναι:

<όνομα_σύνδεσης> <χωρητικότητα_γραμμής> <χωρητικότητα_διαμέσου>
<συνολική_χωρητικότητα> <μήκος_σύνδεσης>
<χωρητικότητα_γραμμής_ανά_μονάδα_μήκους> <αριθμός_συνδέσεων>

Όπου όλες οι χωρητικότητες παρουσιάζονται σε μονάδες Πίκο-Φαράντ (pf), το μήκος των γραμμών σε μικρόμετρα (microns) και η <χωρητικότητα_γραμμής_ανά_μονάδα_μήκους> αναφέρεται σε Πίκο-Φάραντ ανά μικρόμετρο (pf/micron).

Εξαγωγή αρχείου SPICE από Λίστα Verilog

Δεδομένου της ανάγκης μας για γνώση με ακρίβεια των λεπτομερειών όλων των στοιχείων που απαρτίζουν το κύκλωμά μας, των συνδέσεων και των χωρητικοτήτων αυτών για την εφαρμογή της επαυξημένης μεθόδου Λογικού Φόρτου πρέπει να προχωρήσουμε σε σύνθεση ενός αρχείου τύπου SPICE που να περικλείει όλες τις παραπάνω λεπτομέρειες. Αρχικά ξεκινάμε από την Λίστα Verilog του κυκλώματος. Με βάση αυτή τη Λίστα μπορούμε να εξάγουμε το αρχείο SPICE το οποίο περιέχει τα στοιχεία και τις διαφορές συνδέσεις μεταξύ αυτών. Για την εξαγωγή του αρχείου SPICE από τη Λίστα Verilog χρησιμοποιούμε το εργαλείο v2lvs της σουίτας Calibre από την Mentor Graphics Corp. Για την εξαγωγή του αρχείου του παραδείγματος μας εκτελούμε το εργαλείο v2lvs με ορίσματα τα αρχεία αυτά όπως παρουσιάζεται παρακάτω:

mgc /softs/mentor/calibre/2007.3/bin/v2lvs -lsr
/home/athanaso/private/Thesis_Greece/extras/umc18.cdl -o
/home/athanaso/private/Thesis_Greece/extras/binary_to_bcd.ckt -i -v
/home/athanaso/private/Thesis_Greece/eda/HDL/GATE/binary_to_bcd.v

Όπου umc18.cdl το όνομα της βιβλιοθήκης που περιέχει τις λεπτομέρειες τις τεχνολογίας υλοποίησης, binary_to_bcd.ckt το όνομα του προς σύνθεση SPICE αρχείου και τέλος binary_to_bcd.v το όνομα του αρχείου Λίστας Verilog που περιγράφει το κύκλωμα. Έξοδο της παραπάνω εκτέλεσης αποτελεί το αρχείο binary_to_bcd.ckt τύπου SPICE το οποίο παρουσιάζεται με λεπτομέρεια στο ΠΑΡΑΡΤΗΜΑ.

Εισαγωγή τεχνολογίας υλοποίησης

Έχοντας πλέον το αρχείο SPICE που αναφέρει τα στοιχεία του κυκλώματος από τα οποία αποτελείται καθώς και τις συνδέσεις μεταξύ αυτών, πρέπει πλέον να προσθέσουμε σε αυτό τους ορισμούς των στοιχείων. Με λίγα λόγια πρέπει να προσθέσουμε τους ορισμούς των στοιχείων που περιλαμβάνουν όλες τις λεπτομέρειες καθενός από αυτά όπως για παράδειγμα τα πλάτη των τρανζίστορ που τα αποτελούν κτλ. Οι λεπτομέρειες των στοιχείων αυτών περιλαμβάνονται στην βιβλιοθήκη της εκάστοτε τεχνολογίας υλοποίησης. Έτσι αρκεί να αντιγράψουμε τους ορισμούς όλων των στοιχείων που χρησιμοποιούμε από την βιβλιοθήκη της τεχνολογίας υλοποίησης στην αρχή του αρχείου SPICE. Η λύση της εισαγωγής ολόκληρης της βιβλιοθήκης είναι επίσης αποδεκτή καθώς τα εργαλεία μας έχουν την ικανότητα να αγνοήσουν τους ορισμούς στοιχείων των οποίων δεν γίνεται χρήση. Στην περίπτωσή μας, οι ορισμοί αυτοί περιέχονται στο αρχείο «umc18.cdl» παρουσίαση των περιεχομένων του οποίου δεν γίνεται σε αυτή τη διατριβή καθώς στην περίπτωση αυτή θα παραβιάζαμε πνευματικά δικαιώματα της εταιρίας UMC. Ο αναγνώστης αρκεί να αντιγράψει τα περιεχόμενα της εκάστοτε βιβλιοθήκης της τεχνολογίας υλοποίησης που χρησιμοποιεί στην αρχή του αρχείου SPICE.

Εισαγωγή εισόδων κυκλώματος

Όπως αναφέραμε κατά την παρουσίαση της εφαρμογής βελτιστοποίησης επαυξημένης μεθόδου Λογικού Φόρτου υπάρχει η απαίτηση για ορισμό των εισόδων του κυκλώματος μας. Σε αντίθετη περίπτωση η εφαρμογή της μεθόδου θα οδηγήσει σε αλλαγή των μεγεθών των τρανζίστορ πυλών που ανήκουν, πέρα των άλλων, στο πρώτο επίπεδο με άμεσο αποτέλεσμα να αλλοιώσουμε τη χωρητικότητα εισόδου του κυκλώματος και κατ'επέκταση τις λειτουργικές προδιαγραφές αυτού. Ο ορισμός των εισόδων του κυκλώματος μπορεί να γίνει στην αρχή του αρχείου SPICE. Για τη εισαγωγή των εισόδων μπορούμε να λειτουργήσουμε είτε χειρονακτικά είτε για λόγους ταχύτητας και πολυπλοκότητας αρχείων να χρησιμοποιήσουμε την εφαρμογή (inputs_to_spice). Χειρονακτικά παρατηρούμε στην περιγραφή του κυκλώματος στο αρχείο Λίστας Verilog «binary_to_bcd.v» τον ορισμό των εισόδων της ανώτερης δομής αφαίρεσης, άρα κυκλώματος, ως:

input [15:0] dat_binary_i;
input clk_i, ce_i, rst_i, start_i;

Αυτοματοποιημένα αρκεί η εκτέλεση της εφαρμογής inputs_to_spice με ορίσματα τα αρχεία της Λίστας Verilog και του αρχείου SPICE.

Για το παράδειγμα μας η εκτέλεση έχει την παρακάτω μορφή:

/home/athanaso/private/Thesis_Greece/extras/Utils/inputs_to_spice
/home/athanaso/private/Thesis_Greece/eda/HDL/GATE/binary_to_bcd.v
/home/athanaso/private/Thesis_Greece/extras/binary_to_bcd.ckt

Όπου binary_to_bcd.v και binary_to_bcd.ckt τα αρχεία της Λίστας Verilog και του αρχείου SPICE αντίστοιχα. Η εκτέλεση της εφαρμογής αυτής οδηγεί στην προσθήκη των παρακάτω γραμμών στην αρχή του αρχείου SPICE «binary_to_bcd.ckt» οι οποίες θα έπρεπε αλλιώς να οριστούν χειρονακτικά:

v1 dat_binary_i[0] 0 pulse(0 5 15n)
v2 dat_binary_i[1] 0 pulse(0 5 15n)
v3 dat_binary_i[2] 0 pulse(0 5 15n)
v4 dat_binary_i[3] 0 pulse(0 5 15n)
v5 dat_binary_i[4] 0 pulse(0 5 15n)
v6 dat_binary_i[5] 0 pulse(0 5 15n)
v7 dat_binary_i[6] 0 pulse(0 5 15n)
v8 dat_binary_i[7] 0 pulse(0 5 15n)
v9 dat_binary_i[8] 0 pulse(0 5 15n)
v10 dat_binary_i[9] 0 pulse(0 5 15n)
v11 dat_binary_i[10] 0 pulse(0 5 15n)
v12 dat_binary_i[11] 0 pulse(0 5 15n)
v13 dat_binary_i[12] 0 pulse(0 5 15n)
v14 dat_binary_i[13] 0 pulse(0 5 15n)
v15 dat_binary_i[14] 0 pulse(0 5 15n)
v16 dat_binary_i[15] 0 pulse(0 5 15n)
v17 ce_i 0 pulse(0 5 15n)
v18 rst_i 0 pulse(0 5 15n)
v19 start_i 0 pulse(0 5 15n)
v20 clk_i 0 pulse(0 5 15n)

Σε αυτό το σημείο πρέπει να σημειωθεί ότι μας ενδιαφέρει αποκλειστικά ο ορισμός των εισόδων αυτών ως εισόδων του κυκλώματος και όχι οι τιμές τους κτλ.

Εισαγωγή χωρητικοτήτων διασύνδεσης

Το τελευταίο στάδιο για την ολοκλήρωση του αρχείου SPICE που θα περιέχει το σύνολο των απαιτούμενων πληροφοριών για την εκτέλεση της επαυξημένης μεθόδου Λογικού Φόρτου είναι η εισαγωγή των χωρητικοτήτων των γραμμών διασύνδεσης του κυκλώματος. Για την ορθή εκτέλεση της μεθόδου αρκεί οι χωρητικότητες αυτές να εισαχθούν ως χωρητικότητες στους αντίστοιχους κόμβους του κυκλώματος. Για την εισαγωγή αυτή μπορούμε είτε να δράσουμε χειρονακτικά είτε αυτοποιημένα. Στην περίπτωση των χωρητικοτήτων διασύνδεσης προτείνεται η αυτοματοποιημένη εκδοχή λόγω του μεγάλου αριθμού των συνδέσεων. Για την αυτομοτοποιημένη εκδοχή χρησιμοποιούμε την εφαρμογή, cap_to_spice, που αναπτύξαμε για το λόγο αυτό. Η εφαρμογή αυτή αναλύει το αρχείο χωρητικοτήτων που εξαγάγαμε από την εφαρμογή SoC Encounter και στην συνέχεια εισαγάγει τις χωρητικότητες

διασύνδεσης στο αρχείο SPICE που ορίζεται από τον χρήστη. Συνεπώς, για την ορθή εκτέλεση της εφαρμογής αυτής ο χρήστης πρέπει να ορίσει το αρχείο χωρητικοτήτων, το αρχείο SPICE καθώς και το όνομα του αρχείου που επιθυμεί να λάβει το νέο αρχείο που θα σχηματιστεί. Για το παράδειγμά μας χρησιμοποιούμε την εφαρμογή όπως παρουσιάζεται παρακάτω:

/home/athanaso/private/Thesis_Greece/extras/Utils/cap_to_spice
/home/athanaso/private/Thesis_Greece/extras/binary_to_bcd.cap
/home/athanaso/private/Thesis_Greece/extras/binary_to_bcd.ckt >
/home/athanaso/private/Thesis_Greece/extras/binary_to_bcd.ckt

Όπου binary_to_bcd.cap και binary_to_bcd.ckt τα αρχεία των χωρητικοτήτων και SPICE αντίστοιχα, ενώ μετά το πέρας της εκτέλεσης το νέο αρχείο SPICE που θα προκύψει θα αντικαταστήσει το παλαιό binary_to_bcd.ckt. Το συνολικό αρχείο που προκύπτει πλην των πληροφοριών της τεχνολογίας υλοποίησης παρουσιάζεται στο ΠΑΡΑΡΤΗΜΑ.

Εκτέλεση επαυξημένης μεθόδου Λογικού Φόρτου

Έχοντας πλέον το αρχείο που περιέχει όλες τις λεπτομέρειες που χρειαζόμαστε για την εφαρμογή της επαυξημένης μεθόδου του Λογικού Φόρτου μπορούμε να προχωρήσουμε πλέον προς την εφαρμογή της βελτιστοποίησης. Για την βελτιστοποίηση μας θα εκτελέσουμε την εφαρμογή που παρουσιάσαμε προηγουμένως (COLEI). Έτσι για το παράδειγμα μας θα εκτελέσουμε την ανωτέρω εφαρμογή με ορίσματα το αρχείο εισόδου SPICE, των αριθμό των κρίσιμων μονοπατιών που επιθυμούμε να βελτιστοποιηθούν και τέλος το αρχείο εξόδου. Ο τύπος του αρχείου εισόδου είναι γλώσσα HTML. Έτσι για το παράδειγμα μας εκτελούμε την εφαρμογή όπως παρουσιάζεται παρακάτω:

./colei binary_to_bcd.ckt 10 > binary_to_bcd.html

Όπου binary_to_bcd.ckt το αρχείο SPICE που περιέχει την ολοκληρωμένη περιγραφή του κυκλώματος, 10 ο αριθμός των κρίσιμων μονοπατιών που επιθυμούμε να βελτιστοποιηθούν και τέλος binary_to_bcd.html το όνομα του αρχείου εξόδου της εφαρμογής. Τα αποτελέσματα της βελτιστοποίησης θα παρουσιαστούν με λεπτομέρεια στο κεφάλαιο στατιστικών αποτελεσμάτων.

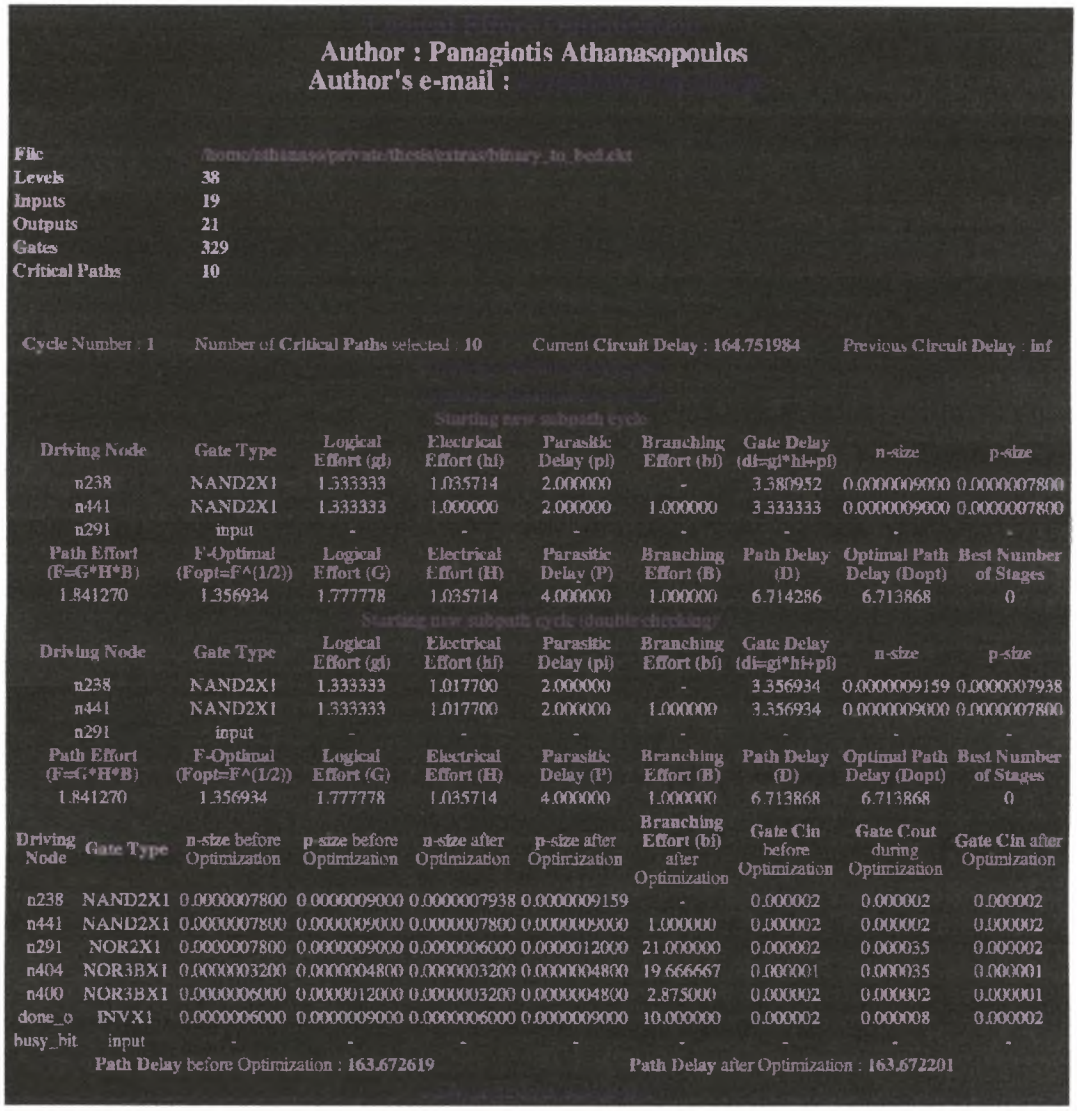
Πειραματικά αποτελέσματα

Στο κεφάλαιο αυτό παρουσιάζουμε τα αποτελέσματα της εφαρμογής της μεθοδολογίας μας. Η παρουσίαση ξεκινά παρουσιάζοντας λεπτομερώς τα αποτελέσματα του παραδείγματος για δεδομένη τεχνολογία και αριθμό κρίσιμων μονοπατιών και συνεχίζει με αφαιρετική παρουσίαση στατιστικών διαφόρων κυκλωμάτων και ρυθμίσεων.

Πειραματικά αποτελέσματα παραδείγματος

Όπως αναφέραμε προηγουμένως το κύκλωμά μας υλοποιείται σε τεχνολογία 180 νανομέτρων και η βελτιστοποίηση εκτελείται με αριθμό κρίσιμων μονοπατιών ίσο με δέκα (10). Παρακάτω παρουσιάζεται η αρχή του αρχείου εξόδου τύπου HTML. Το μέγεθος του αρχείου αυτού δεν μας επιτρέπει την πλήρη αναπαράστασή του, γι'αυτό θα περιοριστούμε μόνο στην παρουσίαση της βελτιστοποίησης της πρώτης αναδρομής του πρώτου από τα N-κρίσιμα μονοπάτια και στην τελική βελτιστοποίηση και στα αποτελέσματα της μεθόδου.

Μετά την ολοκλήρωση εφαρμογής της επαυξημένης μεθόδου Λογικού Φόρτου στο κύκλωμα του παραδείγματός μας λαμβάνουμε την παρακάτω έξοδο:



Author : Panagiotis Athanasopoulos
Author's e-mail :

File	/home/athanaso/private/thesis/extras/binary_to_bcd.ckt
Levels	38
Inputs	19
Outputs	21
Gates	329
Critical Paths	10

Cycle Number : 1 Number of Critical Paths selected : 10 Current Circuit Delay : 164.751984 Previous Circuit Delay : inf

Starting new subpath cycle

Driving Node	Gate Type	Logical Effort (gi)	Electrical Effort (hi)	Parasitic Delay (pi)	Branching Effort (bi)	Gate Delay (di=gi*hi+pi)	n-size	p-size
n238	NAND2X1	1.333333	1.035714	2.000000	-	3.380952	0.0000009000	0.0000007800
n441	NAND2X1	1.333333	1.000000	2.000000	1.000000	3.333333	0.0000009000	0.0000007800
n291	input	-	-	-	-	-	-	-

Path Effort (F=G*H*B)	F-Optimal (Fopt=F^(1/2))	Logical Effort (G)	Electrical Effort (H)	Parasitic Delay (P)	Branching Effort (B)	Path Delay (D)	Optimal Path Delay (Dopt)	Best Number of Stages
1.841270	1.356934	1.777778	1.035714	4.000000	1.000000	6.714286	6.713868	0

Starting new subpath cycle (double checking)

Driving Node	Gate Type	Logical Effort (gi)	Electrical Effort (hi)	Parasitic Delay (pi)	Branching Effort (bi)	Gate Delay (di=gi*hi+pi)	n-size	p-size
n238	NAND2X1	1.333333	1.017700	2.000000	-	3.356934	0.0000009159	0.0000007938
n441	NAND2X1	1.333333	1.017700	2.000000	1.000000	3.356934	0.0000009000	0.0000007800
n291	input	-	-	-	-	-	-	-

Path Effort (F=G*H*B)	F-Optimal (Fopt=F^(1/2))	Logical Effort (G)	Electrical Effort (H)	Parasitic Delay (P)	Branching Effort (B)	Path Delay (D)	Optimal Path Delay (Dopt)	Best Number of Stages
1.841270	1.356934	1.777778	1.035714	4.000000	1.000000	6.713868	6.713868	0

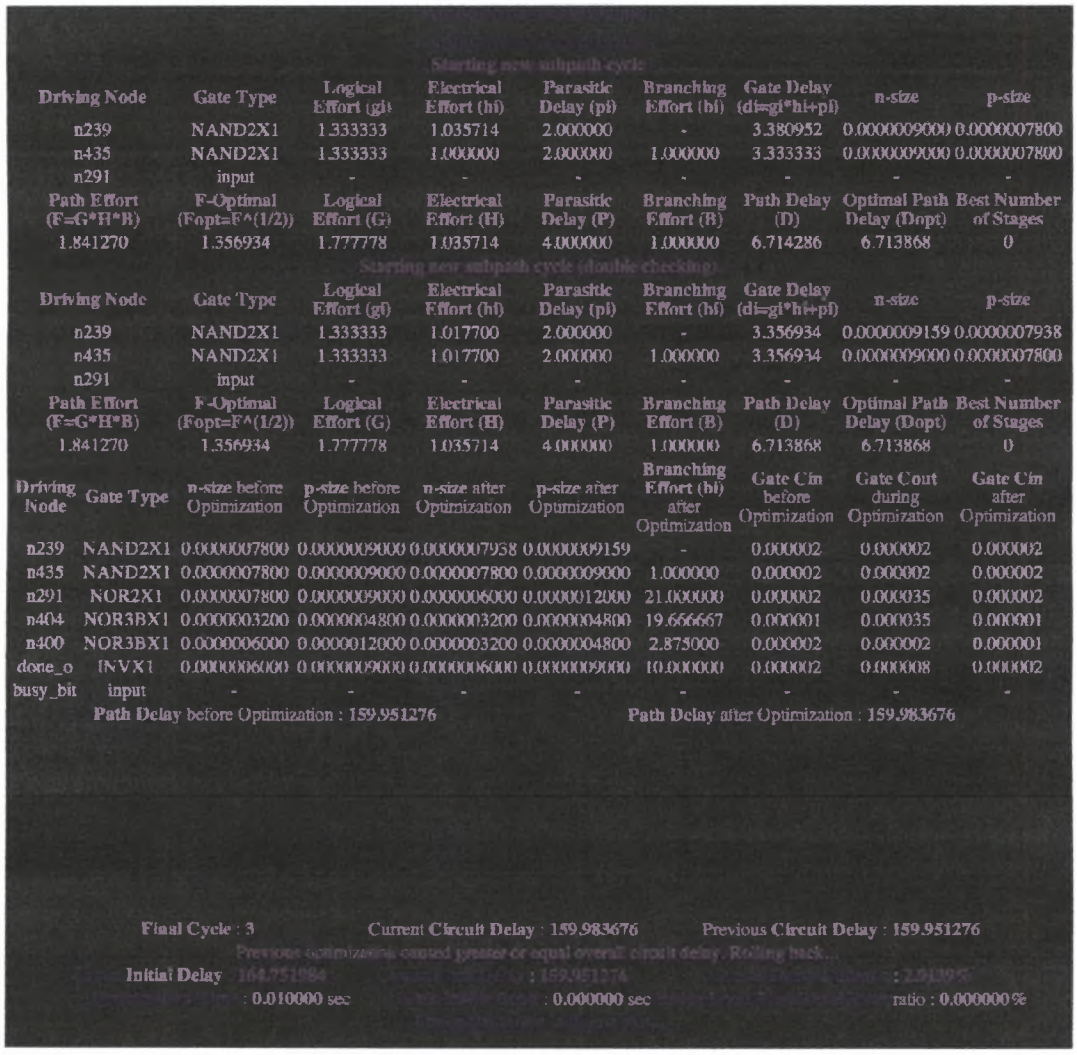
Driving Node	Gate Type	n-size before Optimization	p-size before Optimization	n-size after Optimization	p-size after Optimization	Branching Effort (bi) after Optimization	Gate Cin before Optimization	Gate Cout during Optimization	Gate Cin after Optimization
n238	NAND2X1	0.0000007800	0.0000009000	0.0000007938	0.0000009159	-	0.000002	0.000002	0.000002
n441	NAND2X1	0.0000007800	0.0000009000	0.0000007800	0.0000009000	1.000000	0.000002	0.000002	0.000002
n291	NOR2X1	0.0000007800	0.0000009000	0.0000006000	0.0000012000	21.000000	0.000002	0.000035	0.000002
n404	NOR3BX1	0.0000003200	0.0000004800	0.0000003200	0.0000004800	19.666667	0.000001	0.000035	0.000001
n400	NOR3BX1	0.0000006000	0.0000012000	0.0000003200	0.0000004800	2.875000	0.000002	0.000002	0.000001
done_o	INVX1	0.0000006000	0.0000009000	0.0000006000	0.0000009000	10.000000	0.000002	0.000008	0.000002
busy_bit	input	-	-	-	-	-	-	-	-

Path Delay before Optimization : 163.672619 Path Delay after Optimization : 163.672201

Πρώτο μέρος εξόδου εφαρμογής βελτιστοποίησης

Στο πρώτο μέρος της εξόδου μπορούμε να διακρίνουμε κάποιες λεπτομέρειες του κυκλώματος όπως για παράδειγμα αριθμός επιπέδων, εισόδων, εξόδων, πυλών κτλ. Στην συνέχεια αναφέρεται η αρχική καθυστέρηση του κυκλώματος. Οι καθυστερήσεις όπως προαναφέραμε, αναφέρονται σε μονάδες καθυστέρησης αντιστροφέα αναφοράς τεχνολογίας υλοποίησης. Στην προκειμένη περίπτωση η καθυστέρηση αυτή είναι 164,751984 φορές η καθυστέρηση του αντιστροφέα αναφοράς. Σε κάθε εκτέλεση παρουσιάζονται οι λεπτομέρειες για τις διάφορες έννοιες που συμμετέχουν στον υπολογισμό της επαυξημένης μεθόδου Λογικού Φόρτου όπως για παράδειγμα ο λογικός και ηλεκτρικός φόρτος της εκάστοτε πύλης κτλ. Μετά το πέρας της εκάστοτε βελτιστοποίησης μονοπατιού η εφαρμογή συνεχίζει με την βελτιστοποίηση του επόμενου μονοπατιού ή με νέα αναδρομή. Μετά το πέρας πεπερασμένου αριθμού τέτοιων βελτιστοποιήσεων η

μέθοδος συγκλίνει σε μια βέλτιστη καθυστέρηση. Το τέλος της βελτιστοποίησης παρουσιάζεται παρακάτω όπου η τελευταία βελτιστοποίηση (159,983676 μονάδες καθυστέρησης αντιστροφέα αναφοράς) οδήγησε σε μεγαλύτερη συνολική καθυστέρηση του κυκλώματος σε σχέση με την αμέσως προηγούμενη (159,951276) οπότε γίνεται αναίρεση των τελευταίων αλλαγών, όπως αναφέραμε στο κεφάλαιο που παρουσιάζεται η εφαρμογή, και ως βέλτιστη λύση θεωρείται η προτελευταία (159,951276 μονάδες καθυστέρησης αντιστροφέα αναφοράς).



Starting new subpath cycle

Driving Node	Gate Type	Logical Effort (gi)	Electrical Effort (hi)	Parasitic Delay (pi)	Branching Effort (bi)	Gate Delay (di=gi*hi+pi)	n-size	p-size
n239	NAND2X1	1.333333	1.035714	2.000000	-	3.380952	0.0000009000	0.0000007800
n435	NAND2X1	1.333333	1.000000	2.000000	1.000000	3.333333	0.0000009000	0.0000007800
n291	input	-	-	-	-	-	-	-

Path Effort (F=G*H*B)	F-Optimal (Fopt=F^(1/2))	Logical Effort (G)	Electrical Effort (H)	Parasitic Delay (P)	Branching Effort (B)	Path Delay (D)	Optimal Path Delay (Dopt)	Best Number of Stages
1.841270	1.356934	1.777778	1.035714	4.000000	1.000000	6.714286	6.713868	0

Starting new subpath cycle (double checking)

Driving Node	Gate Type	Logical Effort (gi)	Electrical Effort (hi)	Parasitic Delay (pi)	Branching Effort (bi)	Gate Delay (di=gi*hi+pi)	n-size	p-size
n239	NAND2X1	1.333333	1.017700	2.000000	-	3.356934	0.0000009159	0.0000007938
n435	NAND2X1	1.333333	1.017700	2.000000	1.000000	3.356934	0.0000009000	0.0000007800
n291	input	-	-	-	-	-	-	-

Path Effort (F=G*H*B)	F-Optimal (Fopt=F^(1/2))	Logical Effort (G)	Electrical Effort (H)	Parasitic Delay (P)	Branching Effort (B)	Path Delay (D)	Optimal Path Delay (Dopt)	Best Number of Stages
1.841270	1.356934	1.777778	1.035714	4.000000	1.000000	6.713868	6.713868	0

Driving Node	Gate Type	n-size before Optimization	p-size before Optimization	n-size after Optimization	p-size after Optimization	Branching Effort (bi) after Optimization	Gate Cin before Optimization	Gate Cout during Optimization	Gate Cin after Optimization
n239	NAND2X1	0.0000007800	0.0000009000	0.0000007938	0.0000009159	-	0.000002	0.000002	0.000002
n435	NAND2X1	0.0000007800	0.0000009000	0.0000007800	0.0000009000	1.000000	0.000002	0.000002	0.000002
n291	NOR2X1	0.0000007800	0.0000009000	0.0000006000	0.0000012000	21.000000	0.000002	0.000035	0.000002
n404	NOR3BX1	0.0000003200	0.0000004800	0.0000003200	0.0000004800	19.666667	0.000001	0.000035	0.000001
n400	NOR3BX1	0.0000006000	0.0000012000	0.0000003200	0.0000004800	2.875000	0.000002	0.000002	0.000001
done_o	INVX1	0.0000009600	0.0000009000	0.0000006000	0.0000009000	10.000000	0.000002	0.000008	0.000002
busy_bit	input	-	-	-	-	-	-	-	-

Path Delay before Optimization : 159.951276

Path Delay after Optimization : 159.983676

Final Cycle : 3 Current Circuit Delay : 159.983676 Previous Circuit Delay : 159.951276

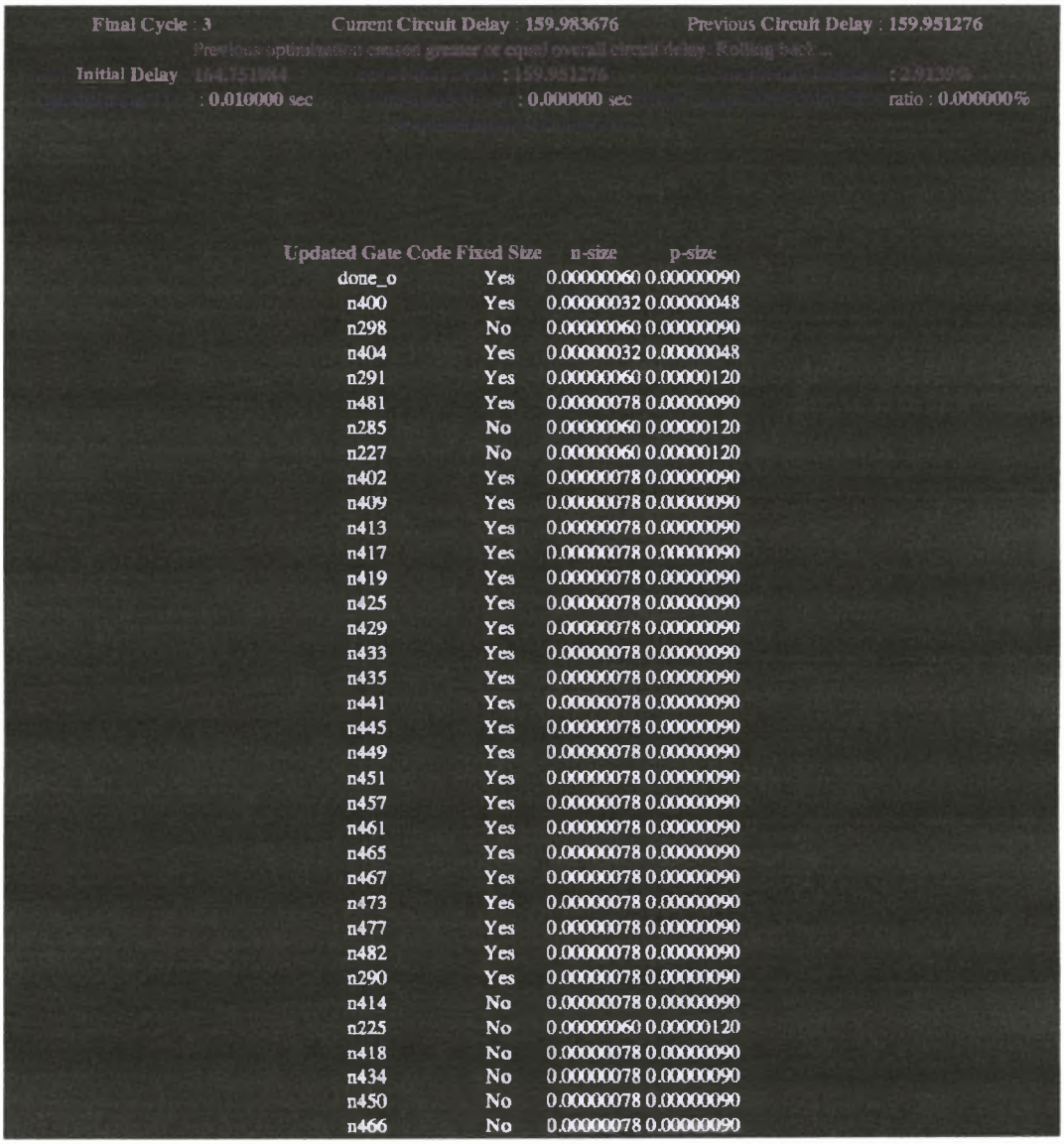
Previous optimization caused greater or equal overall circuit delay. Rolling back...

Initial Delay 164.751984 : 159.951276 : 2.913955

: 0.010000 sec : 0.000000 sec ratio : 0.000000 %

Τελευταία εκτέλεση (3) αναδρομής τελευταίου (πλέον) κρίσιμου μονοπατιού

Επιπλέον στατιστικά στοιχεία παρουσιάζονται μετά το πέρας της εκτέλεσης όπως το συνολικό ποσοστό επιτάχυνσης που επετεύχθη (στην προκειμένη περίπτωση 2,9139) κτλ. Τέλος ακολουθεί η ουσιαστική έξοδος της βελτιστοποίησης η οποία είναι η αναφορά των ανανεωμένων ή μη μεγεθών των τρανζίστορ:



Final Cycle : 3 Current Circuit Delay : 159.983676 Previous Circuit Delay : 159.951276

Previous optimization cannot greater or equal overall circuit delay. Rolling back ...

Initial Delay : 164.751884 : 159.951276 : 2.91394%

: 0.010000 sec : 0.000000 sec ratio : 0.000000%

Updated Gate Code	Fixed Size	n-size	p-size
done_o	Yes	0.00000060	0.00000090
n400	Yes	0.00000032	0.00000048
n298	No	0.00000060	0.00000090
n404	Yes	0.00000032	0.00000048
n291	Yes	0.00000060	0.00000120
n481	Yes	0.00000078	0.00000090
n285	No	0.00000060	0.00000120
n227	No	0.00000060	0.00000120
n402	Yes	0.00000078	0.00000090
n409	Yes	0.00000078	0.00000090
n413	Yes	0.00000078	0.00000090
n417	Yes	0.00000078	0.00000090
n419	Yes	0.00000078	0.00000090
n425	Yes	0.00000078	0.00000090
n429	Yes	0.00000078	0.00000090
n433	Yes	0.00000078	0.00000090
n435	Yes	0.00000078	0.00000090
n441	Yes	0.00000078	0.00000090
n445	Yes	0.00000078	0.00000090
n449	Yes	0.00000078	0.00000090
n451	Yes	0.00000078	0.00000090
n457	Yes	0.00000078	0.00000090
n461	Yes	0.00000078	0.00000090
n465	Yes	0.00000078	0.00000090
n467	Yes	0.00000078	0.00000090
n473	Yes	0.00000078	0.00000090
n477	Yes	0.00000078	0.00000090
n482	Yes	0.00000078	0.00000090
n290	Yes	0.00000078	0.00000090
n414	No	0.00000078	0.00000090
n225	No	0.00000060	0.00000120
n418	No	0.00000078	0.00000090
n434	No	0.00000078	0.00000090
n450	No	0.00000078	0.00000090
n466	No	0.00000078	0.00000090

Ανανεωμένα ή μη μεγέθη τρανζίστορ

Η πλήρης αναφορά των ανανεωμένων μεγεθών των τρανζίστορ παρουσιάζεται στον ΠΑΡΑΡΤΗΜΑ.

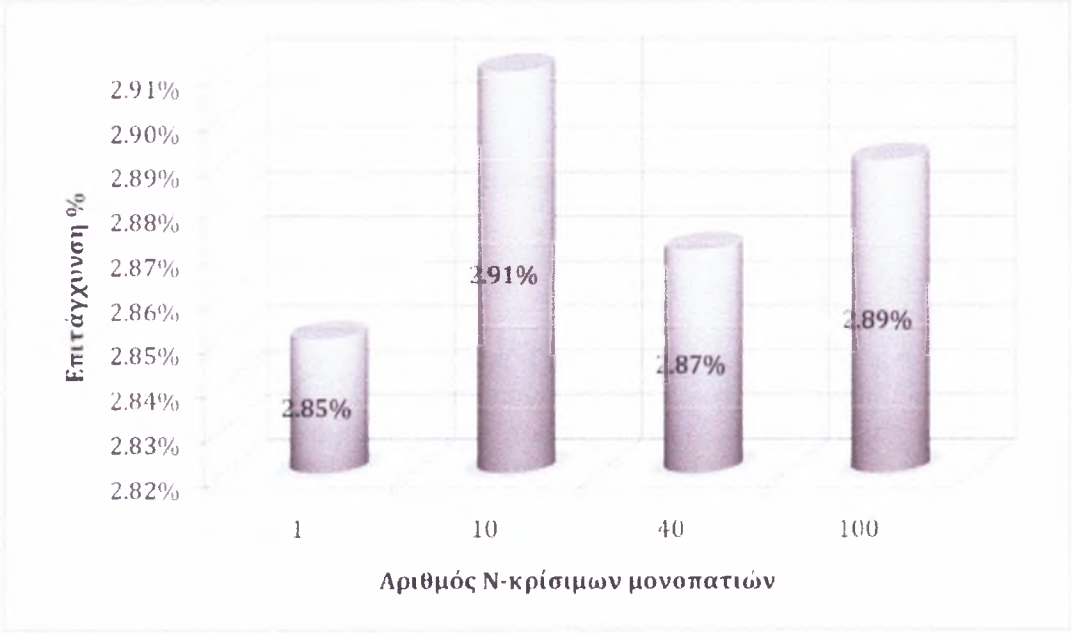
Επίδραση διαφόρων παραγόντων

Έχοντας παρουσιάσει με λεπτομέρεια τα αποτελέσματα του παραδείγματός μας για δεδομένη τεχνολογία υλοποίησης και αριθμό κρίσιμων μονοπατιών, απομένει πλέον να μελετήσουμε πώς αυτοί οι δύο παράγοντες επιδρούν στην βελτίωση. Για τις ανάγκες παρουσίασης της επίδρασης του αριθμού κρίσιμων μονοπατιών στην βελτιστοποίηση, θα χρησιμοποιήσουμε δύο κυκλώματα. Το πρώτο αποτελεί το κύκλωμα του παραδείγματος το οποίο αποτελείται από ένα μεγάλο αριθμό πυλών συνδεδεμένων σε σειρά οι οποίες ως εκ τούτου βρίσκονται πάντα εντός του κρίσιμου μονοπατιού ακολουθούμενες από μια σειρά τελικών διακλαδώσεων. Το δεύτερο κύκλωμα αποτελεί μέρος μια μονάδας «Carry Lookahead Unit» και ως εκ τούτου παρουσιάζει μεγάλο βαθμό παραλληλίας. Στην συνέχεια θα χρησιμοποιήσουμε δύο τεχνολογίες υλοποίησης, της UMC 180 nm και ST 65 nm καθώς και την απομόνωση της διασύνδεσης για να μελετήσουμε την επίδραση της διασύνδεσης στην μέθοδο του Λογικού Φόρτου.

Μελέτη επιπτώσεων επιλογής αριθμού κρίσιμων μονοπατιών

Σε αυτό το σημείο παρουσιάζουμε τις επιπτώσεις της επιλογής αριθμού κρισίμων μονοπατιών. Σε μια ιδανική περίπτωση ο αριθμός αυτός θα ήταν τετριμμένος και πιθανότατα ίσος με τη μονάδα. Το γεγονός αυτό θα απλοποιούσε σημαντικά την εφαρμογή μας και θα περιόριζε των αριθμό των μεταβλητών που απαιτούνται για την εφαρμογή της μεθόδου μας. Όπως θα παρατηρήσουμε παρακάτω, αυτή δεν είναι η πραγματικότητα.

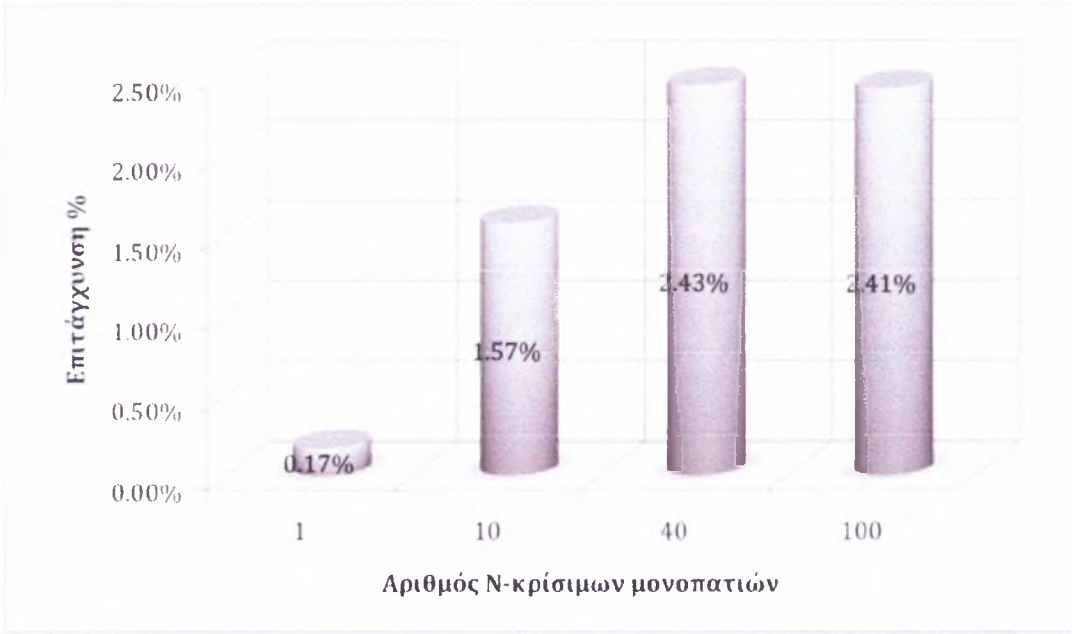
Για την ανάγκη επίδειξης θα χρησιμοποιήσουμε τα δύο κυκλώματα που παρουσιάστηκαν παραπάνω. Τα αποτελέσματα της εφαρμογής της βελτιστοποίησης παρουσιάζονται σε αριθμό κρίσιμων μονοπατιών ίσο με 1, 10, 40, 100. Αρχικά παρουσιάζονται τα αποτελέσματα για το κύκλωμα «binary_to_bcd».



Ποσοστά επιτάχυνσης σε σχέση με τον αριθμό κρίσιμων μονοπατιών N

Στο παραπάνω σχεδιάγραμμα παρατηρούμε ότι στην περίπτωση του πρώτου κυκλώματος δεν παρατηρούμε αξιοσημείωτη μεταβολή της βελτιστοποίησης.

Στη συνέχεια παρουσιάζουμε τα αντίστοιχα αποτελέσματα για το δεύτερο κύκλωμα «Carry_lookahead»:



Ποσοστά επιτάχυνσης σε σχέση με τον αριθμό κρίσιμων μονοπατιών N

Στο παραπάνω σχεδιάγραμμα παρατηρούμε ότι στην περίπτωση του δεύτερου κυκλώματος παρατηρείται αξιοσημείωτη μεταβολή της βελτιστοποίησης. Ειδικότερα παρατηρούμε ότι για ένα και μόνο κρίσιμο μονοπάτι η βελτίωση είναι μηδαμινή. Ο λόγος που συμβαίνει αυτό είναι το γεγονός ότι στο δεύτερο κύκλωμα υπάρχει μεγάλος παραλληλισμός μονοπατιών με αποτέλεσμα, στην περίπτωση που εκτελούμε βελτιστοποίηση με αριθμό κρίσιμων μονοπατιών ίσο με την μονάδα, με το που βελτιστοποιείται ένα μονοπάτι στην αμέσως επόμενη αναδρομή του αλγορίθμου υπάρχει ένα άλλο μονοπάτι, πιθανώς παράλληλο του αρχικού το οποίο ομοίως παρουσιάζει παραπλήσια καθυστέρηση του αρχικού. Μοιραία, ύστερα από κάποια αναδρομή, η βελτιστοποίηση ενός μονοπατιού θα οδηγήσει σε συνολική αύξηση της καθυστέρησης του κυκλώματος καθώς θα αλλοιωθούν τα χαρακτηριστικά κάποιου άλλου μονοπατιού το οποίο είχε καθυστέρηση παραπλήσια του υπό βελτίωση. Τα παραπάνω δεν συμβαίνουν στην περίπτωση του πρώτου κυκλώματος «binary_to_bcd» καθώς αυτό αποτελείται από ένα πολύ μεγάλο υπό-μονοπάτι το οποίο αποτελεί πάντα μέρος του κρίσιμου μονοπατιού και ως εκ τούτου μια ενδεχόμενη βελτιστοποίηση αυτού βελτιώνει σημαντικά την συνολική καθυστέρηση του κυκλώματος. Σε αντίθεση, καθώς στο δεύτερο κύκλωμα αυξάνουμε τον αριθμό των N-κρίσιμων μονοπατιών η επιτάχυνση που παρατηρείται γίνεται όλο και μεγαλύτερη μέχρι κάποιο σημείο πέρα από το οποίο δεν επιτυγχάνουμε επιπλέον επιτάχυνση. Ο λόγος που συμβαίνει αυτό είναι ότι στην επαυξημένη μέθοδο Λογικού Φόρτου, στις περιπτώσεις αριθμού N-κρίσιμων μονοπατιών μεγαλύτερου της μονάδας, ξεκινάμε από την βελτιστοποίηση των N-κρίσιμων μονοπατιών από αυτό που παρουσιάζει την μικρότερη καθυστέρηση προς αυτό με την μεγαλύτερη με αποτέλεσμα όταν βελτιστοποιήσουμε το μονοπάτι με την χειρότερη καθυστέρηση, προκαλώντας ενδεχομένως αύξηση στην καθυστέρηση σε κάποιο

αρχικής παραπλήσιας αυτό να έχει ήδη βελτιστοποιηθεί προηγουμένως καταλήγοντας έτσι σε συνολικά μικρότερη καθυστέρηση από αυτή που είχε πριν την εκκίνηση της αναδρομής.

Από τα παραπάνω γίνεται κατανοητός ο λόγος για τον οποίο ο αριθμός των N-κρίσιμων μονοπατιών αποτελεί μεταβλητή εκτέλεσης. Η επιλογή του αριθμού αυτού αποτελεί ευθύνη του εκάστοτε χρήστη ο οποίος θα πρέπει να είτε να δοκιμάσει διαφορετικές τιμές αυτού είτε να έχει αντίληψη των μονοπατιών του κυκλώματος πριν αποφανθεί για την επιτεύξιμη επιτάχυνση.

Μελέτη επιπτώσεων διασύνδεσης

Για την μελέτη της επίπτωσης της διασύνδεσης στην επιτεύξιμη επιτάχυνση μέσω της επαυξημένης μεθόδου λογικού φόρτου χρησιμοποιούμε δύο διαφορετικές τεχνολογίες υλοποίησης την UMC 180nm και ST 65nm καθώς και απομόνωση της διασύνδεσης. Η μελέτη αυτή θα γίνει για τα δύο κυκλώματα που αναφέρθηκαν προηγουμένως. Η επιτάχυνση του καθενός θα παρουσιαστεί για τις δύο διαφορετικές τεχνολογίες υλοποίησης καθώς και χωρίς διασύνδεση. Παρόλα αυτά πρέπει να σημειωθεί ότι το περιβάλλον βελτιστοποίησης είναι διαφορετικό για τα δύο κυκλώματα, αναφερόμενοι στον αριθμό των N-κρίσιμων μονοπατιών, καθώς σκοπός μας σε αυτή τη μελέτη είναι να επιδείξουμε τις επιπτώσεις της διασύνδεσης. Έτσι, αρχικά παρουσιάζουμε το πρώτο κύκλωμα «binary_to_bcd» για τις τρείς διαφορετικές εκδοχές διασύνδεσης. Στο κύκλωμα αυτό όπως είδαμε προηγουμένως επιτυγχάνουμε μέγιστη επιτάχυνση για εκτέλεση της βελτιστοποίησης με αριθμό N-κρίσιμων μονοπατιών ίσο με δέκα (10). Έτσι για το πρώτο κύκλωμα προκύπτουν τα παρακάτω πειραματικά δεδομένα:

Ποσοστά επιτάχυνσης σε σχέση με την τεχνολογία υλοποίησης

Στη συνέχεια παρουσιάζουμε τα αντίστοιχα αποτελέσματα για το δεύτερο κύκλωμα «Carry_lookahead». Το κύκλωμα αυτό παρουσιάζει βέλτιστη επιτάχυνση για επιλογή αριθμού N-κρίσιμων μονοπατιών ίσο με σαράντα (40):

Ποσοστά επιτάχυνσης σε σχέση με την τεχνολογία υλοποίησης

Από τα παραπάνω διαγράμματα παρατηρούμε ότι καθώς κινούμαστε προς συνεχή σμίκρυνση της τεχνολογίας υλοποίησης η επιτάχυνση που επιτυγχάνεται από την εφαρμογή της επαυξημένης μεθόδου λογικού φόρτου περιορίζεται. Το γεγονός αυτό είναι αναμενόμενο καθώς είναι ευρέως γνωστό ότι κινούμενοι προς σμίκρυνση της τεχνολογίας υλοποίησης η καθυστέρηση της διασύνδεσης υπερτερεί αυτής των λογικών πυλών στις οποίες εμείς επεμβαίνουμε.

Συμπεράσματα

Παρατηρώντας τα αποτελέσματα μας γίνεται κατανοητό ότι παρά τις υπερσύγχρονες τεχνολογίες υλοποίησης που χρησιμοποιήσαμε επιτυγχάνουμε πάντα κάποιο ποσοστό επιτάχυνσης. Η επιτάχυνση αυτή, αν και περιορίζεται συχνά από μια πλειάδα παραγόντων, παραμένει σημαντική καθώς ο υπολογισμός της έχει πραγματοποιηθεί μετά από τοποθέτηση και διασύνδεση των στοιχείων που αποτελούν το κύκλωμα (post place & route). Το στάδιο αυτό αποτελεί ουσιαστικά το στάδιο μετά το οποίο μπορεί κανείς να αποφανθεί για τα χαρακτηριστικά λειτουργίας ενός κυκλώματος, καθώς αυτό αποτελεί το τελευταίο στάδιο που πραγματοποιούνται σημαντικές αλλαγές στα κυκλώματα πριν την κατασκευή τους σε σιλικόνη. Η περισσότερες μέθοδοι βελτιστοποίησης τείνουν να αγνοούν τον παράγοντα διασύνδεσης, παράγοντας ο οποίος αποτελεί το σημαντικότερο παράγοντα καθυστέρησης σε ένα σύγχρονο κύκλωμα. Παρόλα αυτά η μέθοδός μας επιτυγχάνει βελτιστοποίηση παρούσας της διασύνδεσης, αλλά δεν σταματά εκεί καθώς προχωράει την μελέτη αυτή σε μια υπερσύγχρονη τεχνολογία υλοποίησης (65nm) πρόσβαση στην οποία έχουν ελάχιστες εταιρίες την ώρα που γράφονται οι γραμμές αυτές. Σκοπός μας ήταν να αποδείξουμε ότι ακόμα και στην τεχνολογία αυτή όπου η διασύνδεση κυριαρχεί όλων των άλλων παραγόντων σε ότι αφορά την καθυστέρηση, εξακολουθούμε να παρέχουμε βελτίωση της καθυστέρησης. Όπως γνωρίζουμε σήμερα, στόχος της σχεδίασης ολοκληρωμένων κυκλωμάτων αποτελεί η έστω και παραμικρή βελτίωση των επιδόσεων. Επιπλέον λόγω του διαρκώς αυξανόμενου ποσοστού της καθυστέρησης που οφείλεται στην διασύνδεση, καθώς μεταβαίνουμε σε διαρκώς μειούμενο μέγεθος υλοποίησης, μεγάλο μέρος της έρευνας μεταφέρεται στη βελτίωση του τρόπου διασύνδεσης. Ένας μεγάλος αριθμός μεθόδων για τον περιορισμό της καθυστέρησης αυτής που παρουσιάζει η διασύνδεση έχει προταθεί η παρουσίαση των οποίων ξεπερνά τους στόχους της διατριβής αυτής. Επιπλέον σε αυτό το σημείο πρέπει να αναφερθεί ότι κατά τον υπολογισμό των αποτελεσμάτων έγινε χρήση αυτόματα συντιθέμενης διασύνδεσης με κριτήρια επίδειξης (δοκιμαστική διασύνδεση – Trial route). Η ανάπτυξη διασύνδεσης με κριτήρια επίτευξης μέγιστης απόδοσης, στην οποία στοχεύουν οι σύγχρονες εταιρίες ολοκληρωμένων κυκλωμάτων θα είχε σαν αποτέλεσμα το σημαντικό περιορισμό της καθυστέρησης που αυτή παρουσιάζει, και ως εκ τούτου ακόμα μεγαλύτερη επιτάχυνση. Η ανάπτυξη διασύνδεσης με τέτοια κριτήρια ξεφεύγει από τους σκοπούς αυτής της διατριβής καθώς απαιτεί την εργασία μιας πλειάδας επαγγελματιών μηχανικών και ένα μεγάλο χρονικό διάστημα για την επίτευξή της.

Βιβλιογραφία

1. ***Logical Effort, Designing Fast CMOS Circuits*** *των Ivan Sutherland, Bob Sproull και David Harris.*
2. ***Σχεδίαση Ολοκληρωμένων Κυκλωμάτων CMOS VLSI*** *των N. H. Weste και K. Eshraghian σε μετάφραση – επιμέλεια των Κ. ΠΕΚΜΕΣΤΖΗ, Δ. ΣΟΥΝΤΡΗ και Κ. ΓΚΟΥΤΗ.*
3. ***Σχεδίαση Ψηφιακών Συστημάτων με τη Γλώσσα VHDL*** *των Stephen Brown και Zvonko Vranesic σε μετάφραση – επιμέλεια των Νικόλαου Ι. Μάργαρη, Παύλου Χρ. Κούρου, Χρήστου Β. Τζίκα και Ιωάννη Πεταλά.*
4. ***High Speed CMOS Design Styles*** *των Kerry Bernstein K.M. Carrig Christopher M. Durham Patrick R. Hansen David Hogenmiller Edward J. Nowak Norman J.*
5. ***CMOS: Circuit Design, Layout, and Simulation, 2nd Edition*** *του R. Jacob Baker.*
6. ***Incremental Circuit Simulation and Timing Analysis Techniques*** *του YUN-CHENG JU.*
7. *Η παρουσίαση:* ***Logical Effort: Επιλογή του Μεγέθους των Πυλών*** *του καθηγητή κ. Γεώργιου Σταμούλη.*
8. *Η παρουσίαση:* ***Τεχνικές σχεδίασης κυκλωμάτων CMOS*** *του καθηγητή κ. Γεώργιου Τσατούχα.*
9. *Η online βιβλιοθήκη Wikipedia: www.wikipedia.org*
10. *Τα εγχειρίδια χρήσης των εφαρμογών Design Vision και SoC Encounter.*

ΠΑΡΑΡΤΗΜΑ

Στο παράρτημα αυτό παρουσιάζονται τα πλήρη αρχεία, εντολές κτλ που χρησιμοποιούνται κατά το παράδειγμα της μεθοδολογίας

Αρχείο «binary_to_bcd.v»

```
//----------------------------------------------------------------------------
// Binary to BCD converter, serial implementation, 1 clock per input bit.
//
//
// Description: See description below (which suffices for IP core
//                      specification document.)
//
// Copyright (C) 2002 John Clayton and OPENCORES.ORG (this Verilog version)
//
// This source file may be used and distributed without restriction provided
// that this copyright statement is not removed from the file and that any
// derivative work contains the original copyright notice and the associated
// disclaimer.
//
// This source file is free software; you can redistribute it and/or modify
// it under the terms of the GNU Lesser General Public License as published
// by the Free Software Foundation;  either version 2.1 of the License, or
// (at your option) any later version.
//
// This source is distributed in the hope that it will be useful, but WITHOUT
// ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
// FITNESS FOR A PARTICULAR PURPOSE.  See the GNU Lesser General Public
// License for more details.
//
// You should have received a copy of the GNU Lesser General Public License
// along with this source.
// If not, download it from http://www.opencores.org/lgpl.shtml
//
//----------------------------------------------------------------------------
//
// Author: John Clayton
// Date  : Nov. 19, 2003
// Update: Nov. 19, 2003  Copied this file from "led_display_driver.v" and
//                        modified it.
// Update: Nov. 24, 2003  Fixed bcd_asl function, tested module.  It works!
// Update: Nov. 25, 2003  Changed bit_counter and related logic so that long
//                        start pulses produce correct results at the end of
//                        the pulse.
//
//----------------------------------------------------------------------------
// Description:
//
// This module takes a binary input, and converts it into BCD output, with each
// binary coded decimal digit of course occupying 4-bits.
// The user can specify the number of input bits separately from the number of
```

```
// output digits.  Be sure that you have specified enough output digits to
// represent the largest number you expect on the binary input, or else the
// most significant digits of the result will be cut off.
//
//-----------------------------------------------------------------------------


module binary_to_bcd (
 clk_i,
 ce_i,
 rst_i,
 start_i,
 dat_binary_i,
 dat_bcd_o,
 done_o
 );
parameter BITS_IN_PP       = 16; // # of bits of binary input
parameter BCD_DIGITS_OUT_PP  = 5;  // # of digits of BCD output
parameter BIT_COUNT_WIDTH_PP = 4;  // Width of bit counter

// I/O declarations
input  clk_i;               // clock signal
input  ce_i;                // clock enable input
input  rst_i;               // synchronous reset
input  start_i;              // initiates a conversion
input  [BITS_IN_PP-1:0] dat_binary_i;      // input bus
output [4*BCD_DIGITS_OUT_PP-1:0] dat_bcd_o;  // output bus
output done_o;                 // indicates conversion is done

reg [4*BCD_DIGITS_OUT_PP-1:0] dat_bcd_o;

// Internal signal declarations

reg  [BITS_IN_PP-1:0] bin_reg;
reg  [4*BCD_DIGITS_OUT_PP-1:0] bcd_reg;
wire [BITS_IN_PP-1:0] bin_next;
reg  [4*BCD_DIGITS_OUT_PP-1:0] bcd_next;
reg  busy_bit;
reg  [BIT_COUNT_WIDTH_PP-1:0] bit_count;
wire bit_count_done;

//-----------------------------------------------------------------------------
// Functions & Tasks
//-----------------------------------------------------------------------------

function [4*BCD_DIGITS_OUT_PP-1:0] bcd_asl;
 input [4*BCD_DIGITS_OUT_PP-1:0] din;
 input newbit;
 integer k;
 reg cin;
 reg [3:0] digit;
 reg [3:0] digit_less;
 begin
  cin = newbit;
```

```
  for (k=0; k<BCD_DIGITS_OUT_PP; k=k+1)
  begin
   digit[3] = din[4*k+3];
   digit[2] = din[4*k+2];
   digit[1] = din[4*k+1];
   digit[0] = din[4*k];
   digit_less = digit - 5;
   if (digit > 4'b0100)
   begin
    bcd_asl[4*k+3] = digit_less[2];
    bcd_asl[4*k+2] = digit_less[1];
    bcd_asl[4*k+1] = digit_less[0];
    bcd_asl[4*k+0] = cin;
    cin = 1'b1;
   end
   else
   begin
    bcd_asl[4*k+3] = digit[2];
    bcd_asl[4*k+2] = digit[1];
    bcd_asl[4*k+1] = digit[0];
    bcd_asl[4*k+0] = cin;
    cin = 1'b0;
   end

  end // end of for loop
 end
endfunction

//-----------------------------------------------------------------------------
// Module code
//-----------------------------------------------------------------------------

// Perform proper shifting, binary ASL and BCD ASL
assign bin_next = {bin_reg,1'b0};
always @(bcd_reg or bin_reg)
begin
 bcd_next <= bcd_asl(bcd_reg,bin_reg[BITS_IN_PP-1]);
end

// Busy bit, input and output registers
always @(posedge clk_i)
begin
 if (rst_i)
 begin
  busy_bit <= 0;  // Synchronous reset
  dat_bcd_o <= 0;
 end
 else if (start_i && ~busy_bit)
 begin
  busy_bit <= 1;
  bin_reg <= dat_binary_i;
  bcd_reg <= 0;
 end
 else if (busy_bit && ce_i && bit_count_done && ~start_i)
```

```
 begin
  busy_bit <= 0;
  dat_bcd_o <= bcd_next;
 end
 else if (busy_bit && ce_i && ~bit_count_done)
 begin
  bcd_reg <= bcd_next;
  bin_reg <= bin_next;
 end
end
assign done_o = ~busy_bit;

// Bit counter
always @(posedge clk_i)
begin
 if (~busy_bit) bit_count <= 0;
 else if (ce_i && ~bit_count_done) bit_count <= bit_count + 1;
end
assign bit_count_done = (bit_count == (BITS_IN_PP-1));

endmodule
```

Εντολή περιορισμού επιτρεπόμενων στοιχείων κατά τη σύνθεση

Παρακάτω ακολουθεί ο συνολικός περιορισμός των επιτρεπόμενων στοιχείων κατά τη σύνθεση του κυκλώματος του παραδείγματος της μεθοδολογίας:

```
set_dont_use        {        **_1v8_25c/**AFCSHCINX2        **_1v8_25c/**AFCSHCINX4
**_1v8_25c/**AFCSHCONX2     **_1v8_25c/**AFCSHCONX4     **_1v8_25c/**AFHCINX2
**_1v8_25c/**AFHCINX4       **_1v8_25c/**AFHCONX2       **_1v8_25c/**AFHCONX4
**_1v8_25c/**AHHCINX2       **_1v8_25c/**AHHCINX4       **_1v8_25c/**AHHCONX2
**_1v8_25c/**AHHCONX4        **_1v8_25c/**AOI211X1       **_1v8_25c/**AOI211X2
**_1v8_25c/**AOI211X4        **_1v8_25c/**AOI211XL        **_1v8_25c/**AOI21X1
**_1v8_25c/**AOI21X2         **_1v8_25c/**AOI21X4         **_1v8_25c/**AOI21XL
**_1v8_25c/**AOI221X1        **_1v8_25c/**AOI221X2       **_1v8_25c/**AOI221X4
**_1v8_25c/**AOI221XL        **_1v8_25c/**AOI222X1       **_1v8_25c/**AOI222X2
**_1v8_25c/**AOI222X4        **_1v8_25c/**AOI222XL        **_1v8_25c/**AOI22X1
**_1v8_25c/**AOI22X2         **_1v8_25c/**AOI22X4         **_1v8_25c/**AOI22XL
**_1v8_25c/**AOI2BB1X1      **_1v8_25c/**AOI2BB1X2      **_1v8_25c/**AOI2BB1X4
**_1v8_25c/**AOI2BB1XL      **_1v8_25c/**AOI2BB2X1      **_1v8_25c/**AOI2BB2X2
**_1v8_25c/**AOI2BB2X4       **_1v8_25c/**AOI2BB2XL       **_1v8_25c/**AOI31X1
**_1v8_25c/**AOI31X2         **_1v8_25c/**AOI31X4         **_1v8_25c/**AOI31XL
**_1v8_25c/**AOI32X1         **_1v8_25c/**AOI32X2         **_1v8_25c/**AOI32X4
**_1v8_25c/**AOI32XL         **_1v8_25c/**AOI33X1         **_1v8_25c/**AOI33X2
**_1v8_25c/**AOI33X4          **_1v8_25c/**AOI33XL          **_1v8_25c/**BENCX1
**_1v8_25c/**BENCX2           **_1v8_25c/**BENCX4           **_1v8_25c/**BMXX1
**_1v8_25c/**CLKBUFX1       **_1v8_25c/**CLKBUFX12      **_1v8_25c/**CLKBUFX16
**_1v8_25c/**CLKBUFX2        **_1v8_25c/**CLKBUFX20       **_1v8_25c/**CLKBUFX3
**_1v8_25c/**CLKBUFX4        **_1v8_25c/**CLKBUFX8        **_1v8_25c/**CLKBUFXL
**_1v8_25c/**CLKINVX1       **_1v8_25c/**CLKINVX12       **_1v8_25c/**CLKINVX16
```

_1v8_25c/CLKINVX2 **_1v8_25c/**CLKINVX20 **_1v8_25c/**CLKINVX3
_1v8_25c/CLKINVX4 **_1v8_25c/**CLKINVX8 **_1v8_25c/**CLKINVXL
_1v8_25c/CMPR22X1 **_1v8_25c/**CMPR32X1 **_1v8_25c/**CMPR42X1
_1v8_25c/CMPR42X2 **_1v8_25c/**DLY1X1 **_1v8_25c/**DLY2X1
_1v8_25c/DLY3X1 **_1v8_25c/**DLY4X1 **_1v8_25c/**HOLDX1
_1v8_25c/JKFFRX1 **_1v8_25c/**JKFFRX2 **_1v8_25c/**JKFFRX4
_1v8_25c/JKFFRXL **_1v8_25c/**JKFFSRX1 **_1v8_25c/**JKFFSRX2
_1v8_25c/JKFFSRX4 **_1v8_25c/**JKFFSRXL **_1v8_25c/**JKFFSX1
_1v8_25c/JKFFSX2 **_1v8_25c/**JKFFSX4 **_1v8_25c/**JKFFSXL
_1v8_25c/JKFFX1 **_1v8_25c/**JKFFX2 **_1v8_25c/**JKFFX4 **_1v8_25c/**JKFFXL
_1v8_25c/MX2X1 **_1v8_25c/**MX2X2 **_1v8_25c/**MX2X4 **_1v8_25c/**MX2XL
_1v8_25c/MX4X1 **_1v8_25c/**MX4X2 **_1v8_25c/**MX4X4 **_1v8_25c/**MX4XL
_1v8_25c/MXI2X1 **_1v8_25c/**MXI2X2 **_1v8_25c/**MXI2X4
_1v8_25c/MXI2XL **_1v8_25c/**MXI4X1 **_1v8_25c/**MXI4X2
_1v8_25c/MXI4X4 **_1v8_25c/**MXI4XL **_1v8_25c/**OAI211X1
_1v8_25c/OAI211X2 **_1v8_25c/**OAI211X4 **_1v8_25c/**OAI211XL
_1v8_25c/OAI21X1 **_1v8_25c/**OAI21X2 **_1v8_25c/**OAI21X4
_1v8_25c/OAI21XL **_1v8_25c/**OAI221X1 **_1v8_25c/**OAI221X2
_1v8_25c/OAI221X4 **_1v8_25c/**OAI221XL **_1v8_25c/**OAI222X1
_1v8_25c/OAI222X2 **_1v8_25c/**OAI222X4 **_1v8_25c/**OAI222XL
_1v8_25c/OAI22X1 **_1v8_25c/**OAI22X2 **_1v8_25c/**OAI22X4
_1v8_25c/OAI22XL **_1v8_25c/**OAI2BB1X1 **_1v8_25c/**OAI2BB1X2
_1v8_25c/OAI2BB1X4 **_1v8_25c/**OAI2BB1XL **_1v8_25c/**OAI2BB2X1
_1v8_25c/OAI2BB2X2 **_1v8_25c/**OAI2BB2X4 **_1v8_25c/**OAI2BB2XL
_1v8_25c/OAI31X1 **_1v8_25c/**OAI31X2 **_1v8_25c/**OAI31X4
_1v8_25c/OAI31XL **_1v8_25c/**OAI32X1 **_1v8_25c/**OAI32X2
_1v8_25c/OAI32X4 **_1v8_25c/**OAI32XL **_1v8_25c/**OAI33X1
_1v8_25c/OAI33X2 **_1v8_25c/**OAI33X4 **_1v8_25c/**OAI33XL
_1v8_25c/RF1R1WX2 **_1v8_25c/**RF2R1WX2 **_1v8_25c/**RFRDX1
_1v8_25c/RFRDX2 **_1v8_25c/**RFRDX4 **_1v8_25c/**RSLATNX1
_1v8_25c/RSLATNX2 **_1v8_25c/**RSLATNX4 **_1v8_25c/**RSLATNXL
_1v8_25c/RSLATX1 **_1v8_25c/**RSLATX2 **_1v8_25c/**RSLATX4
_1v8_25c/RSLATXL **_1v8_25c/**TIEHI **_1v8_25c/**TIELO
_1v8_25c/TLATNRX1 **_1v8_25c/**TLATNRX2 **_1v8_25c/**TLATNRX4
_1v8_25c/TLATNRXL **_1v8_25c/**TLATNSRX1 **_1v8_25c/**TLATNSRX2
_1v8_25c/TLATNSRX4 **_1v8_25c/**TLATNSRXL **_1v8_25c/**TLATNSX1
_1v8_25c/TLATNSX2 **_1v8_25c/**TLATNSX4 **_1v8_25c/**TLATNSXL
_1v8_25c/TLATNX1 **_1v8_25c/**TLATNX2 **_1v8_25c/**TLATNX4
_1v8_25c/TLATNXL **_1v8_25c/**TLATRX1 **_1v8_25c/**TLATRX2
_1v8_25c/TLATRX4 **_1v8_25c/**TLATRXL **_1v8_25c/**TLATSRX1
_1v8_25c/TLATSRX2 **_1v8_25c/**TLATSRX4 **_1v8_25c/**TLATSRXL
_1v8_25c/TLATSX1 **_1v8_25c/**TLATSX2 **_1v8_25c/**TLATSX4
_1v8_25c/TLATSXL **_1v8_25c/**TLATX1 **_1v8_25c/**TLATX2
_1v8_25c/TLATX4 **_1v8_25c/**TLATXL **_1v8_25c/****LATX1
_1v8_25c/**LATX2 **_1v8_25c/****LATX4 **_1v8_25c/****LATXL
_1v8_25c/DFFNRX1 **_1v8_25c/**DFFNRX2 **_1v8_25c/**DFFNRX4
_1v8_25c/DFFNRXL **_1v8_25c/**DFFNSRX1 **_1v8_25c/**DFFNSRX2
_1v8_25c/DFFNSRX4 **_1v8_25c/**DFFNSRXL **_1v8_25c/**DFFNSX1
_1v8_25c/DFFNSX2 **_1v8_25c/**DFFNSX4 **_1v8_25c/**DFFNSXL
_1v8_25c/DFFNX1 **_1v8_25c/**DFFNX2 **_1v8_25c/**DFFNX4
_1v8_25c/DFFNXL **_1v8_25c/**DFFRHQX1 **_1v8_25c/**DFFRHQX2
_1v8_25c/DFFRHQX4 **_1v8_25c/**DFFRHQXL **_1v8_25c/**DFFRX1
_1v8_25c/DFFRX2 **_1v8_25c/**DFFRX4 **_1v8_25c/**DFFRXL
_1v8_25c/DFFSHQX1 **_1v8_25c/**DFFSHQX2 **_1v8_25c/**DFFSHQX4

```
**_1v8_25c/**DFFSHQXL     **_1v8_25c/**DFFSRHQX1     **_1v8_25c/**DFFSRHQX2
**_1v8_25c/**DFFSRHQX4     **_1v8_25c/**DFFSRHQXL     **_1v8_25c/**DFFSRX1
**_1v8_25c/**DFFSRX2       **_1v8_25c/**DFFSRX4       **_1v8_25c/**DFFSRXL
**_1v8_25c/**DFFSX1        **_1v8_25c/**DFFSX2        **_1v8_25c/**DFFSX4
**_1v8_25c/**DFFSXL        **_1v8_25c/**DFFTRX1       **_1v8_25c/**DFFTRX2
**_1v8_25c/**DFFTRX4       **_1v8_25c/**DFFTRXL       **_1v8_25c/**DFFX1
**_1v8_25c/**DFFX2         **_1v8_25c/**DFFX4         **_1v8_25c/**DFFXL
**_1v8_25c/**EDFFTRX1      **_1v8_25c/**EDFFTRX2      **_1v8_25c/**EDFFTRX4
**_1v8_25c/**EDFFTRXL      **_1v8_25c/**EDFFX1        **_1v8_25c/**EDFFX2
**_1v8_25c/**EDFFX4        **_1v8_25c/**EDFFXL        **_1v8_25c/**SDFFHQX1
**_1v8_25c/**SDFFHQX2      **_1v8_25c/**SDFFHQX4      **_1v8_25c/**SDFFHQXL
**_1v8_25c/**SDFFNRX1      **_1v8_25c/**SDFFNRX2      **_1v8_25c/**SDFFNRX4
**_1v8_25c/**SDFFNRXL      **_1v8_25c/**SDFFNSRX1     **_1v8_25c/**SDFFNSRX2
**_1v8_25c/**SDFFNSRX4     **_1v8_25c/**SDFFNSRXL     **_1v8_25c/**SDFFNSX1
**_1v8_25c/**SDFFNSX2      **_1v8_25c/**SDFFNSX4      **_1v8_25c/**SDFFNSXL
**_1v8_25c/**SDFFNX1       **_1v8_25c/**SDFFNX2       **_1v8_25c/**SDFFNX4
**_1v8_25c/**SDFFNXL      **_1v8_25c/**SDFFRHQX1     **_1v8_25c/**SDFFRHQX2
**_1v8_25c/**SDFFRHQX4     **_1v8_25c/**SDFFRHQXL     **_1v8_25c/**SDFFRX1
**_1v8_25c/**SDFFRX2       **_1v8_25c/**SDFFRX4       **_1v8_25c/**SDFFRXL
**_1v8_25c/**SDFFSHQX1     **_1v8_25c/**SDFFSHQX2     **_1v8_25c/**SDFFSHQX4
**_1v8_25c/**SDFFSHQXL    **_1v8_25c/**SDFFSRHQX1    **_1v8_25c/**SDFFSRHQX2
**_1v8_25c/**SDFFSRHQX4    **_1v8_25c/**SDFFSRHQXL    **_1v8_25c/**SDFFSRX1
**_1v8_25c/**SDFFSRX2      **_1v8_25c/**SDFFSRX4      **_1v8_25c/**SDFFSRXL
**_1v8_25c/**SDFFSX1       **_1v8_25c/**SDFFSX2       **_1v8_25c/**SDFFSX4
**_1v8_25c/**SDFFSXL       **_1v8_25c/**SDFFTRX1      **_1v8_25c/**SDFFTRX2
**_1v8_25c/**SDFFTRX4      **_1v8_25c/**SDFFTRXL      **_1v8_25c/**SDFFX1
**_1v8_25c/**SDFFX2        **_1v8_25c/**SDFFX4        **_1v8_25c/**SDFFXL
**_1v8_25c/**SEDFFHQX1     **_1v8_25c/**SEDFFHQX2     **_1v8_25c/**SEDFFHQX4
**_1v8_25c/**SEDFFHQXL     **_1v8_25c/**SEDFFTRX1     **_1v8_25c/**SEDFFTRX2
**_1v8_25c/**SEDFFTRX4     **_1v8_25c/**SEDFFTRXL     **_1v8_25c/**SEDFFX1
**_1v8_25c/**SEDFFX2 **_1v8_25c/**SEDFFX4 **_1v8_25c/**SEDFFXL }
```

Γενικά σκοπός μας είναι να επιτρέψουμε την χρήση κατά την σύνθεση μόνο των πυλών που παρουσιάστηκαν κατά την μέθοδο του Λογικού Φόρτου και μόνο ενός τύπου Flip-flop (D Flip-flop).

Αρχείο Λίστας Verilog συντεθημένου κυκλώματος binary_to_bcd.v

```
module binary_to_bcd ( clk_i, ce_i, rst_i, start_i, dat_binary_i, dat_bcd_o,
    done_o );
 input [15:0] dat_binary_i;
 output [19:0] dat_bcd_o;
 input clk_i, ce_i, rst_i, start_i;
 output done_o;
 wire  \bin_next[15] , \bin_next[14] , \bin_next[13] , \bin_next[12] ,
     \bin_next[11] , \bin_next[10] , \bin_next[9] , \bin_next[8] ,
     \bin_next[7] , \bin_next[6] , \bin_next[5] , \bin_next[4] ,
```

```
    \bin_next[3] , \bin_next[2] , \bin_next[1] , N7, N8, N9, N11, N12,
    N13, N15, N16, N17, N19, N20, N21, N23, N24, N25, N26, busy_bit, N101,
    N102, N103, N104, n32, n33, n34, n35, n36, n37, n38, n39, n40, n41,
    n42, n43, n44, n45, n46, n47, n48, n49, n50, n51, n52, n53, n54, n55,
    n56, n57, n58, n59, n60, n61, n62, n63, n64, n65, n66, n67, n68, n69,
    n70, n71, n72, n73, n74, n75, n76, n77, n78, n79, n80, n81, n83, n84,
    n85, n86, n87, n88, n89, n90, n91, n92, n93, n94, n95, n96, n97, n98,
    n99, n100, n101, n102, n103, n104, n105, n106, n107, n108, n109, n110,
    n111, n112, n113, n114, n115, n116, n117, n118, n119, n120, n121,
    n122, n123, n124, n125, n126, n127, n128, n129, n130, n131, n132,
    n133, n134, n135, n136, n137, n138, n139, n140, n141, n142, n143,
    n144, n145, n146, n147, n148, n149, n150, n151, n152, n153, n154,
    n155, n156, n157, n158, n159, n160, n161, n162, n163, n164, n165,
    n166, n167, n168, n169, n170, n171, n172, n173, n174, n175, n176,
    n177, n178, n179, n180, n181, n182, n183, n184, n185, n186, n187,
    n188, n189, n190, n191, n192, n193, n194, n195, n196, n197, n198,
    n199, n200, n201, n202, n203, n204, n205, n206, n207, n208, n209,
    n210, n211, n212, n213, n214, n215, n216, n217, n218, n219, n220,
    n221, n222, n223, n224, n225, n226, n227, n228, n229, n230, n231,
    n232, n233, n234, n235, n236, n237, n238, n239, n240, n241, n242,
    n243, n244, n245, n246, n247, n248, n249, n250, n251, n252, n253,
    n254, n255, n256, n257, n258, n259, n260, n261, n262, n263, n264,
    n265, n266, n267, n268, n269, n270, n271, n272, n273, n274, n275,
    n276, n277, n278, n279, n280, n281, n282, n283, n284, n285, n286,
    n287, n288, n289, n290, n291, n292, n293, n294, n295, n296, n297,
    n298, n299, n300, n301, n302, n303, n304, n305, n306, n307, n308,
    n309, n310, n311, n312, n313, n314, n315, n316, n317, n318, n319,
    n320, n321, n322, n323, n324, n325, n326, n327, n328, n329, n330,
    n331, n332, n333, n334, n335;
wire  [19:0] bcd_reg;
wire  [19:0] bcd_next;
wire  [3:0] bit_count;

DFFHQX1 \dat_bcd_o_reg[19]  ( .D(n252), .CK(clk_i), .Q(dat_bcd_o[19]) );
DFFHQX1 \dat_bcd_o_reg[18]  ( .D(n251), .CK(clk_i), .Q(dat_bcd_o[18]) );
DFFHQX1 \dat_bcd_o_reg[17]  ( .D(n250), .CK(clk_i), .Q(dat_bcd_o[17]) );
DFFHQX1 \dat_bcd_o_reg[16]  ( .D(n249), .CK(clk_i), .Q(dat_bcd_o[16]) );
DFFHQX1 \dat_bcd_o_reg[15]  ( .D(n248), .CK(clk_i), .Q(dat_bcd_o[15]) );
DFFHQX1 \dat_bcd_o_reg[14]  ( .D(n247), .CK(clk_i), .Q(dat_bcd_o[14]) );
DFFHQX1 \dat_bcd_o_reg[13]  ( .D(n246), .CK(clk_i), .Q(dat_bcd_o[13]) );
DFFHQX1 \dat_bcd_o_reg[12]  ( .D(n245), .CK(clk_i), .Q(dat_bcd_o[12]) );
DFFHQX1 \dat_bcd_o_reg[11]  ( .D(n244), .CK(clk_i), .Q(dat_bcd_o[11]) );
DFFHQX1 \dat_bcd_o_reg[10]  ( .D(n243), .CK(clk_i), .Q(dat_bcd_o[10]) );
DFFHQX1 \dat_bcd_o_reg[9]  ( .D(n242), .CK(clk_i), .Q(dat_bcd_o[9]) );
DFFHQX1 \dat_bcd_o_reg[8]  ( .D(n241), .CK(clk_i), .Q(dat_bcd_o[8]) );
DFFHQX1 \dat_bcd_o_reg[7]  ( .D(n240), .CK(clk_i), .Q(dat_bcd_o[7]) );
DFFHQX1 \dat_bcd_o_reg[6]  ( .D(n239), .CK(clk_i), .Q(dat_bcd_o[6]) );
DFFHQX1 \dat_bcd_o_reg[5]  ( .D(n238), .CK(clk_i), .Q(dat_bcd_o[5]) );
DFFHQX1 \dat_bcd_o_reg[4]  ( .D(n237), .CK(clk_i), .Q(dat_bcd_o[4]) );
DFFHQX1 \dat_bcd_o_reg[3]  ( .D(n236), .CK(clk_i), .Q(dat_bcd_o[3]) );
DFFHQX1 \dat_bcd_o_reg[2]  ( .D(n235), .CK(clk_i), .Q(dat_bcd_o[2]) );
DFFHQX1 \dat_bcd_o_reg[1]  ( .D(n234), .CK(clk_i), .Q(dat_bcd_o[1]) );
DFFHQX1 \dat_bcd_o_reg[0]  ( .D(n233), .CK(clk_i), .Q(dat_bcd_o[0]) );
DFFHQX1 \bcd_reg_reg[3]  ( .D(n269), .CK(clk_i), .Q(bcd_reg[3]) );
```

```
DFFHQX1 \bcd_reg_reg[7] ( .D(n265), .CK(clk_i), .Q(bcd_reg[7]) );
DFFHQX1 \bcd_reg_reg[11] ( .D(n261), .CK(clk_i), .Q(bcd_reg[11]) );
DFFHQX1 \bcd_reg_reg[15] ( .D(n257), .CK(clk_i), .Q(bcd_reg[15]) );
DFFHQX1 \bcd_reg_reg[19] ( .D(n253), .CK(clk_i), .Q(bcd_reg[19]) );
DFFHQX1 \bin_reg_reg[0] ( .D(n288), .CK(clk_i), .Q(\bin_next[1] ) );
DFFHQX1 \bin_reg_reg[1] ( .D(n287), .CK(clk_i), .Q(\bin_next[2] ) );
DFFHQX1 \bin_reg_reg[2] ( .D(n286), .CK(clk_i), .Q(\bin_next[3] ) );
DFFHQX1 \bin_reg_reg[3] ( .D(n285), .CK(clk_i), .Q(\bin_next[4] ) );
DFFHQX1 \bin_reg_reg[4] ( .D(n284), .CK(clk_i), .Q(\bin_next[5] ) );
DFFHQX1 \bin_reg_reg[5] ( .D(n283), .CK(clk_i), .Q(\bin_next[6] ) );
DFFHQX1 \bin_reg_reg[6] ( .D(n282), .CK(clk_i), .Q(\bin_next[7] ) );
DFFHQX1 \bin_reg_reg[7] ( .D(n281), .CK(clk_i), .Q(\bin_next[8] ) );
DFFHQX1 \bin_reg_reg[8] ( .D(n280), .CK(clk_i), .Q(\bin_next[9] ) );
DFFHQX1 \bin_reg_reg[9] ( .D(n279), .CK(clk_i), .Q(\bin_next[10] ) );
DFFHQX1 \bin_reg_reg[10] ( .D(n278), .CK(clk_i), .Q(\bin_next[11] ) );
DFFHQX1 \bin_reg_reg[11] ( .D(n277), .CK(clk_i), .Q(\bin_next[12] ) );
DFFHQX1 \bin_reg_reg[12] ( .D(n276), .CK(clk_i), .Q(\bin_next[13] ) );
DFFHQX1 \bin_reg_reg[13] ( .D(n275), .CK(clk_i), .Q(\bin_next[14] ) );
DFFHQX1 \bin_reg_reg[14] ( .D(n274), .CK(clk_i), .Q(\bin_next[15] ) );
DFFHQX1 \bin_reg_reg[15] ( .D(n273), .CK(clk_i), .Q(bcd_next[0]) );
DFFHQX1 \bcd_reg_reg[4] ( .D(n268), .CK(clk_i), .Q(bcd_reg[4]) );
DFFHQX1 \bcd_reg_reg[8] ( .D(n264), .CK(clk_i), .Q(bcd_reg[8]) );
DFFHQX1 \bcd_reg_reg[12] ( .D(n260), .CK(clk_i), .Q(bcd_reg[12]) );
DFFHQX1 \bcd_reg_reg[16] ( .D(n256), .CK(clk_i), .Q(bcd_reg[16]) );
DFFHQX1 \bcd_reg_reg[0] ( .D(n272), .CK(clk_i), .Q(bcd_reg[0]) );
DFFHQX1 \bcd_reg_reg[5] ( .D(n267), .CK(clk_i), .Q(bcd_reg[5]) );
DFFHQX1 \bcd_reg_reg[9] ( .D(n263), .CK(clk_i), .Q(bcd_reg[9]) );
DFFHQX1 \bcd_reg_reg[13] ( .D(n259), .CK(clk_i), .Q(bcd_reg[13]) );
DFFHQX1 \bcd_reg_reg[17] ( .D(n255), .CK(clk_i), .Q(bcd_reg[17]) );
DFFHQX1 \bcd_reg_reg[1] ( .D(n271), .CK(clk_i), .Q(bcd_reg[1]) );
DFFHQX1 \bcd_reg_reg[6] ( .D(n266), .CK(clk_i), .Q(bcd_reg[6]) );
DFFHQX1 \bcd_reg_reg[10] ( .D(n262), .CK(clk_i), .Q(bcd_reg[10]) );
DFFHQX1 \bcd_reg_reg[14] ( .D(n258), .CK(clk_i), .Q(bcd_reg[14]) );
DFFHQX1 \bcd_reg_reg[18] ( .D(n254), .CK(clk_i), .Q(bcd_reg[18]) );
DFFHQX1 \bcd_reg_reg[2] ( .D(n270), .CK(clk_i), .Q(bcd_reg[2]) );
DFFHQX1 \bit_count_reg[3] ( .D(n230), .CK(clk_i), .Q(bit_count[3]) );
DFFHQX1 \bit_count_reg[2] ( .D(n231), .CK(clk_i), .Q(bit_count[2]) );
DFFHQX1 \bit_count_reg[1] ( .D(n232), .CK(clk_i), .Q(bit_count[1]) );
DFFHQX1 busy_bit_reg ( .D(n289), .CK(clk_i), .Q(busy_bit) );
DFFHQX1 \bit_count_reg[0] ( .D(n290), .CK(clk_i), .Q(bit_count[0]) );
NOR2X1 U278 ( .A(n306), .B(n305), .Y(n291) );
NOR2X1 U279 ( .A(n197), .B(n198), .Y(n229) );
INVX2 U280 ( .A(n294), .Y(n295) );
INVX2 U281 ( .A(n307), .Y(n306) );
INVX2 U282 ( .A(n78), .Y(n307) );
INVX2 U283 ( .A(n297), .Y(n300) );
INVX2 U284 ( .A(n297), .Y(n301) );
INVX2 U285 ( .A(n294), .Y(n296) );
INVX2 U286 ( .A(n297), .Y(n298) );
INVX2 U287 ( .A(n297), .Y(n299) );
INVX2 U288 ( .A(bcd_next[16]), .Y(n97) );
INVX2 U289 ( .A(bcd_next[12]), .Y(n112) );
INVX2 U290 ( .A(bcd_next[8]), .Y(n127) );
INVX2 U291 ( .A(bcd_next[4]), .Y(n218) );
```

```
INVX2 U292 ( .A(N26), .Y(n81) );
XOR2X1 U293 ( .A(n333), .B(n334), .Y(N103) );
NAND4X1 U294 ( .A(bit_count[3]), .B(bit_count[2]), .C(bit_count[1]), .D(
    bit_count[0]), .Y(n197) );
NOR4BX1 U295 ( .AN(n197), .B(n198), .C(done_o), .D(rst_i), .Y(n78) );
BUFX2 U296 ( .A(n36), .Y(n304) );
NOR2X1 U297 ( .A(n295), .B(rst_i), .Y(n36) );
BUFX2 U298 ( .A(n152), .Y(n305) );
NOR3BX1 U299 ( .AN(start_i), .B(busy_bit), .C(rst_i), .Y(n152) );
INVX2 U300 ( .A(ce_i), .Y(n198) );
INVX2 U301 ( .A(busy_bit), .Y(done_o) );
INVX2 U302 ( .A(n292), .Y(n294) );
NOR4BX1 U303 ( .AN(n229), .B(rst_i), .C(start_i), .D(done_o), .Y(n292) );
NAND2X1 U304 ( .A(n87), .B(n88), .Y(n255) );
NAND2X1 U305 ( .A(n78), .B(n69), .Y(n88) );
NAND2X1 U306 ( .A(n291), .B(bcd_reg[17]), .Y(n87) );
NAND2X1 U307 ( .A(n102), .B(n103), .Y(n259) );
NAND2X1 U308 ( .A(n78), .B(n58), .Y(n103) );
NAND2X1 U309 ( .A(n291), .B(bcd_reg[13]), .Y(n102) );
NAND2X1 U310 ( .A(n117), .B(n118), .Y(n263) );
NAND2X1 U311 ( .A(n306), .B(n47), .Y(n118) );
NAND2X1 U312 ( .A(n291), .B(bcd_reg[9]), .Y(n117) );
NAND2X1 U313 ( .A(n132), .B(n133), .Y(n267) );
NAND2X1 U314 ( .A(n306), .B(n35), .Y(n133) );
NAND2X1 U315 ( .A(n291), .B(bcd_reg[5]), .Y(n132) );
NAND3X1 U316 ( .A(n149), .B(n150), .C(n151), .Y(n273) );
NAND2X1 U317 ( .A(dat_binary_i[15]), .B(n305), .Y(n150) );
NAND2X1 U318 ( .A(\bin_next[15] ), .B(n306), .Y(n149) );
NAND2X1 U319 ( .A(bcd_next[0]), .B(n291), .Y(n151) );
NAND3X1 U320 ( .A(n153), .B(n154), .C(n155), .Y(n274) );
NAND2X1 U321 ( .A(dat_binary_i[14]), .B(n305), .Y(n154) );
NAND2X1 U322 ( .A(\bin_next[14] ), .B(n78), .Y(n153) );
NAND2X1 U323 ( .A(\bin_next[15] ), .B(n291), .Y(n155) );
NAND3X1 U324 ( .A(n156), .B(n157), .C(n158), .Y(n275) );
NAND2X1 U325 ( .A(dat_binary_i[13]), .B(n305), .Y(n157) );
NAND2X1 U326 ( .A(\bin_next[13] ), .B(n306), .Y(n156) );
NAND2X1 U327 ( .A(\bin_next[14] ), .B(n291), .Y(n158) );
NAND3X1 U328 ( .A(n159), .B(n160), .C(n161), .Y(n276) );
NAND2X1 U329 ( .A(dat_binary_i[12]), .B(n305), .Y(n160) );
NAND2X1 U330 ( .A(\bin_next[12] ), .B(n78), .Y(n159) );
NAND2X1 U331 ( .A(\bin_next[13] ), .B(n291), .Y(n161) );
NAND3X1 U332 ( .A(n162), .B(n163), .C(n164), .Y(n277) );
NAND2X1 U333 ( .A(dat_binary_i[11]), .B(n305), .Y(n163) );
NAND2X1 U334 ( .A(\bin_next[11] ), .B(n78), .Y(n162) );
NAND2X1 U335 ( .A(\bin_next[12] ), .B(n291), .Y(n164) );
NAND3X1 U336 ( .A(n165), .B(n166), .C(n167), .Y(n278) );
NAND2X1 U337 ( .A(dat_binary_i[10]), .B(n305), .Y(n166) );
NAND2X1 U338 ( .A(\bin_next[10] ), .B(n78), .Y(n165) );
NAND2X1 U339 ( .A(\bin_next[11] ), .B(n291), .Y(n167) );
NAND3X1 U340 ( .A(n168), .B(n169), .C(n170), .Y(n279) );
NAND2X1 U341 ( .A(dat_binary_i[9]), .B(n305), .Y(n169) );
NAND2X1 U342 ( .A(\bin_next[9] ), .B(n78), .Y(n168) );
NAND2X1 U343 ( .A(\bin_next[10] ), .B(n291), .Y(n170) );
NAND3X1 U344 ( .A(n171), .B(n172), .C(n173), .Y(n280) );
```

```
NAND2X1 U345 ( .A(dat_binary_i[8]), .B(n305), .Y(n172) );
NAND2X1 U346 ( .A(\bin_next[8] ), .B(n78), .Y(n171) );
NAND2X1 U347 ( .A(\bin_next[9] ), .B(n291), .Y(n173) );
NAND3X1 U348 ( .A(n174), .B(n175), .C(n176), .Y(n281) );
NAND2X1 U349 ( .A(dat_binary_i[7]), .B(n305), .Y(n175) );
NAND2X1 U350 ( .A(\bin_next[7] ), .B(n78), .Y(n174) );
NAND2X1 U351 ( .A(\bin_next[8] ), .B(n291), .Y(n176) );
NAND3X1 U352 ( .A(n177), .B(n178), .C(n179), .Y(n282) );
NAND2X1 U353 ( .A(dat_binary_i[6]), .B(n305), .Y(n178) );
NAND2X1 U354 ( .A(\bin_next[6] ), .B(n306), .Y(n177) );
NAND2X1 U355 ( .A(\bin_next[7] ), .B(n291), .Y(n179) );
NAND3X1 U356 ( .A(n180), .B(n181), .C(n182), .Y(n283) );
NAND2X1 U357 ( .A(dat_binary_i[5]), .B(n305), .Y(n181) );
NAND2X1 U358 ( .A(\bin_next[5] ), .B(n306), .Y(n180) );
NAND2X1 U359 ( .A(\bin_next[6] ), .B(n291), .Y(n182) );
NAND3X1 U360 ( .A(n183), .B(n184), .C(n185), .Y(n284) );
NAND2X1 U361 ( .A(dat_binary_i[4]), .B(n305), .Y(n184) );
NAND2X1 U362 ( .A(\bin_next[4] ), .B(n306), .Y(n183) );
NAND2X1 U363 ( .A(\bin_next[5] ), .B(n291), .Y(n185) );
NAND3X1 U364 ( .A(n186), .B(n187), .C(n188), .Y(n285) );
NAND2X1 U365 ( .A(dat_binary_i[3]), .B(n305), .Y(n187) );
NAND2X1 U366 ( .A(\bin_next[3] ), .B(n306), .Y(n186) );
NAND2X1 U367 ( .A(\bin_next[4] ), .B(n291), .Y(n188) );
NAND3X1 U368 ( .A(n189), .B(n190), .C(n191), .Y(n286) );
NAND2X1 U369 ( .A(dat_binary_i[2]), .B(n305), .Y(n190) );
NAND2X1 U370 ( .A(\bin_next[2] ), .B(n306), .Y(n189) );
NAND2X1 U371 ( .A(\bin_next[3] ), .B(n291), .Y(n191) );
NAND3X1 U372 ( .A(n192), .B(n193), .C(n194), .Y(n287) );
NAND2X1 U373 ( .A(dat_binary_i[1]), .B(n305), .Y(n193) );
NAND2X1 U374 ( .A(\bin_next[1] ), .B(n78), .Y(n192) );
NAND2X1 U375 ( .A(\bin_next[2] ), .B(n291), .Y(n194) );
NAND2BX1 U376 ( .AN(n305), .B(n199), .Y(n289) );
NAND2X1 U377 ( .A(n304), .B(busy_bit), .Y(n199) );
NAND2X1 U378 ( .A(n227), .B(n228), .Y(n237) );
NAND2X1 U379 ( .A(bcd_next[4]), .B(n303), .Y(n228) );
NAND2X1 U380 ( .A(dat_bcd_o[4]), .B(n304), .Y(n227) );
NAND2X1 U381 ( .A(n43), .B(n44), .Y(n241) );
NAND2X1 U382 ( .A(bcd_next[8]), .B(n303), .Y(n44) );
NAND2X1 U383 ( .A(dat_bcd_o[8]), .B(n304), .Y(n43) );
NAND2X1 U384 ( .A(n54), .B(n55), .Y(n245) );
NAND2X1 U385 ( .A(bcd_next[12]), .B(n303), .Y(n55) );
NAND2X1 U386 ( .A(dat_bcd_o[12]), .B(n304), .Y(n54) );
INVX2 U387 ( .A(n302), .Y(n303) );
NAND2X1 U388 ( .A(n65), .B(n66), .Y(n249) );
NAND2X1 U389 ( .A(bcd_next[16]), .B(n303), .Y(n66) );
NAND2X1 U390 ( .A(dat_bcd_o[16]), .B(n304), .Y(n65) );
NAND2X1 U391 ( .A(n136), .B(n137), .Y(n268) );
NAND2X1 U392 ( .A(bcd_next[4]), .B(n78), .Y(n137) );
NAND2X1 U393 ( .A(n291), .B(bcd_reg[4]), .Y(n136) );
NAND2X1 U394 ( .A(n91), .B(n92), .Y(n256) );
NAND2X1 U395 ( .A(n78), .B(bcd_next[16]), .Y(n92) );
NAND2X1 U396 ( .A(n291), .B(bcd_reg[16]), .Y(n91) );
NAND2X1 U397 ( .A(n106), .B(n107), .Y(n260) );
NAND2X1 U398 ( .A(n306), .B(bcd_next[12]), .Y(n107) );
```

```
NAND2X1 U399 ( .A(n291), .B(bcd_reg[12]), .Y(n106) );
NAND2X1 U400 ( .A(n121), .B(n122), .Y(n264) );
NAND2X1 U401 ( .A(n306), .B(bcd_next[8]), .Y(n122) );
NAND2X1 U402 ( .A(n291), .B(bcd_reg[8]), .Y(n121) );
NAND2X1 U403 ( .A(n212), .B(n213), .Y(n233) );
NAND2X1 U404 ( .A(bcd_next[0]), .B(n303), .Y(n213) );
NAND2X1 U405 ( .A(dat_bcd_o[0]), .B(n304), .Y(n212) );
NAND2X1 U406 ( .A(n147), .B(n148), .Y(n272) );
NAND2X1 U407 ( .A(bcd_next[0]), .B(n78), .Y(n148) );
NAND2X1 U408 ( .A(bcd_reg[0]), .B(n291), .Y(n147) );
NAND2X1 U409 ( .A(n195), .B(n196), .Y(n288) );
NAND2X1 U410 ( .A(dat_binary_i[0]), .B(n305), .Y(n196) );
NAND2X1 U411 ( .A(\bin_next[1] ), .B(n291), .Y(n195) );
NAND2X1 U412 ( .A(n59), .B(n60), .Y(n247) );
NAND2X1 U413 ( .A(n295), .B(n61), .Y(n60) );
NAND2X1 U414 ( .A(dat_bcd_o[14]), .B(n304), .Y(n59) );
NAND2X1 U415 ( .A(n76), .B(n77), .Y(n253) );
NAND2X1 U416 ( .A(n306), .B(n75), .Y(n77) );
NAND2X1 U417 ( .A(bcd_reg[19]), .B(n291), .Y(n76) );
NAND2X1 U418 ( .A(n83), .B(n84), .Y(n254) );
NAND2X1 U419 ( .A(n306), .B(n72), .Y(n84) );
NAND2X1 U420 ( .A(n291), .B(bcd_reg[18]), .Y(n83) );
NAND2X1 U421 ( .A(n108), .B(n109), .Y(n261) );
NAND2X1 U422 ( .A(n306), .B(n53), .Y(n109) );
NAND2X1 U423 ( .A(bcd_reg[11]), .B(n291), .Y(n108) );
NAND2X1 U424 ( .A(n113), .B(n114), .Y(n262) );
NAND2X1 U425 ( .A(n306), .B(n50), .Y(n114) );
NAND2X1 U426 ( .A(n291), .B(bcd_reg[10]), .Y(n113) );
NAND2X1 U427 ( .A(n123), .B(n124), .Y(n265) );
NAND2X1 U428 ( .A(n306), .B(n42), .Y(n124) );
NAND2X1 U429 ( .A(bcd_reg[7]), .B(n291), .Y(n123) );
NAND2X1 U430 ( .A(n128), .B(n129), .Y(n266) );
NAND2X1 U431 ( .A(n306), .B(n39), .Y(n129) );
NAND2X1 U432 ( .A(n291), .B(bcd_reg[6]), .Y(n128) );
NAND2X1 U433 ( .A(n138), .B(n139), .Y(n269) );
NAND2X1 U434 ( .A(n306), .B(n140), .Y(n139) );
NAND2X1 U435 ( .A(bcd_reg[3]), .B(n291), .Y(n138) );
NAND2X1 U436 ( .A(n141), .B(n142), .Y(n270) );
NAND2X1 U437 ( .A(n306), .B(n143), .Y(n142) );
NAND2X1 U438 ( .A(bcd_reg[2]), .B(n291), .Y(n141) );
NAND2X1 U439 ( .A(n144), .B(n145), .Y(n271) );
NAND2X1 U440 ( .A(n306), .B(n146), .Y(n145) );
NAND2X1 U441 ( .A(bcd_reg[1]), .B(n291), .Y(n144) );
NAND2X1 U442 ( .A(n93), .B(n94), .Y(n257) );
NAND2X1 U443 ( .A(n78), .B(n64), .Y(n94) );
NAND2X1 U444 ( .A(bcd_reg[15]), .B(n291), .Y(n93) );
NAND2X1 U445 ( .A(n98), .B(n99), .Y(n258) );
NAND2X1 U446 ( .A(n78), .B(n61), .Y(n99) );
NAND2X1 U447 ( .A(n291), .B(bcd_reg[14]), .Y(n98) );
NAND2X1 U448 ( .A(n214), .B(n215), .Y(n234) );
NAND2X1 U449 ( .A(n299), .B(n146), .Y(n215) );
NAND2X1 U450 ( .A(dat_bcd_o[1]), .B(n304), .Y(n214) );
NAND2X1 U451 ( .A(n219), .B(n220), .Y(n235) );
NAND2X1 U452 ( .A(n300), .B(n143), .Y(n220) );
```

```
NAND2X1 U453 ( .A(dat_bcd_o[2]), .B(n304), .Y(n219) );
NAND2X1 U454 ( .A(n223), .B(n224), .Y(n236) );
NAND2X1 U455 ( .A(n301), .B(n140), .Y(n224) );
NAND2X1 U456 ( .A(dat_bcd_o[3]), .B(n304), .Y(n223) );
NAND2X1 U457 ( .A(n32), .B(n33), .Y(n238) );
NAND2X1 U458 ( .A(n296), .B(n35), .Y(n33) );
NAND2X1 U459 ( .A(dat_bcd_o[5]), .B(n304), .Y(n32) );
NAND2X1 U460 ( .A(n37), .B(n38), .Y(n239) );
NAND2X1 U461 ( .A(n296), .B(n39), .Y(n38) );
NAND2X1 U462 ( .A(dat_bcd_o[6]), .B(n304), .Y(n37) );
NAND2X1 U463 ( .A(n40), .B(n41), .Y(n240) );
NAND2X1 U464 ( .A(n298), .B(n42), .Y(n41) );
NAND2X1 U465 ( .A(dat_bcd_o[7]), .B(n304), .Y(n40) );
NAND2X1 U466 ( .A(n48), .B(n49), .Y(n243) );
NAND2X1 U467 ( .A(n300), .B(n50), .Y(n49) );
NAND2X1 U468 ( .A(dat_bcd_o[10]), .B(n304), .Y(n48) );
NAND2X1 U469 ( .A(n51), .B(n52), .Y(n244) );
NAND2X1 U470 ( .A(n301), .B(n53), .Y(n52) );
NAND2X1 U471 ( .A(dat_bcd_o[11]), .B(n304), .Y(n51) );
NAND2X1 U472 ( .A(n62), .B(n63), .Y(n248) );
NAND2X1 U473 ( .A(n296), .B(n64), .Y(n63) );
NAND2X1 U474 ( .A(dat_bcd_o[15]), .B(n304), .Y(n62) );
NAND2X1 U475 ( .A(n67), .B(n68), .Y(n250) );
NAND2X1 U476 ( .A(n296), .B(n69), .Y(n68) );
NAND2X1 U477 ( .A(dat_bcd_o[17]), .B(n304), .Y(n67) );
NAND2X1 U478 ( .A(n70), .B(n71), .Y(n251) );
NAND2X1 U479 ( .A(n298), .B(n72), .Y(n71) );
NAND2X1 U480 ( .A(dat_bcd_o[18]), .B(n304), .Y(n70) );
NAND2X1 U481 ( .A(n73), .B(n74), .Y(n252) );
NAND2X1 U482 ( .A(n299), .B(n75), .Y(n74) );
NAND2X1 U483 ( .A(dat_bcd_o[19]), .B(n304), .Y(n73) );
NAND2X1 U484 ( .A(n45), .B(n46), .Y(n242) );
NAND2X1 U485 ( .A(n296), .B(n47), .Y(n46) );
NAND2X1 U486 ( .A(dat_bcd_o[9]), .B(n304), .Y(n45) );
NAND2X1 U487 ( .A(n56), .B(n57), .Y(n246) );
NAND2X1 U488 ( .A(n303), .B(n58), .Y(n57) );
NAND2X1 U489 ( .A(dat_bcd_o[13]), .B(n304), .Y(n56) );
NAND2X1 U490 ( .A(bit_count[1]), .B(bit_count[0]), .Y(n334) );
XOR2X1 U491 ( .A(N23), .B(bcd_reg[17]), .Y(N24) );
XOR2X1 U492 ( .A(N19), .B(bcd_reg[13]), .Y(N20) );
XOR2X1 U493 ( .A(N15), .B(bcd_reg[9]), .Y(N16) );
XOR2X1 U494 ( .A(N11), .B(bcd_reg[5]), .Y(N12) );
XOR2X1 U495 ( .A(N7), .B(bcd_reg[1]), .Y(N8) );
INVX2 U496 ( .A(n34), .Y(n302) );
NOR4BX1 U497 ( .AN(n229), .B(rst_i), .C(start_i), .D(done_o), .Y(n34) );
NAND2X1 U498 ( .A(n104), .B(n105), .Y(n58) );
NAND2X1 U499 ( .A(bcd_next[16]), .B(N19), .Y(n105) );
NAND2X1 U500 ( .A(bcd_reg[12]), .B(n97), .Y(n104) );
NAND2X1 U501 ( .A(n119), .B(n120), .Y(n47) );
NAND2X1 U502 ( .A(bcd_next[12]), .B(N15), .Y(n120) );
NAND2X1 U503 ( .A(bcd_reg[8]), .B(n112), .Y(n119) );
NAND2X1 U504 ( .A(n134), .B(n135), .Y(n35) );
NAND2X1 U505 ( .A(bcd_next[8]), .B(N11), .Y(n135) );
NAND2X1 U506 ( .A(bcd_reg[4]), .B(n127), .Y(n134) );
```

```
NAND2X1 U507 ( .A(n95), .B(n96), .Y(n64) );
NAND2X1 U508 ( .A(bcd_reg[14]), .B(n97), .Y(n95) );
NAND2X1 U509 ( .A(N21), .B(bcd_next[16]), .Y(n96) );
NAND2X1 U510 ( .A(n325), .B(n324), .Y(N21) );
NAND2X1 U511 ( .A(n100), .B(n101), .Y(n61) );
NAND2X1 U512 ( .A(N20), .B(bcd_next[16]), .Y(n101) );
NAND2X1 U513 ( .A(bcd_reg[13]), .B(n97), .Y(n100) );
NAND2X1 U514 ( .A(n110), .B(n111), .Y(n53) );
NAND2X1 U515 ( .A(bcd_reg[10]), .B(n112), .Y(n110) );
NAND2X1 U516 ( .A(N17), .B(bcd_next[12]), .Y(n111) );
NAND2X1 U517 ( .A(n320), .B(n319), .Y(N17) );
NAND2X1 U518 ( .A(n115), .B(n116), .Y(n50) );
NAND2X1 U519 ( .A(N16), .B(bcd_next[12]), .Y(n116) );
NAND2X1 U520 ( .A(bcd_reg[9]), .B(n112), .Y(n115) );
NAND2X1 U521 ( .A(n125), .B(n126), .Y(n42) );
NAND2X1 U522 ( .A(bcd_reg[6]), .B(n127), .Y(n125) );
NAND2X1 U523 ( .A(N13), .B(bcd_next[8]), .Y(n126) );
NAND2X1 U524 ( .A(n315), .B(n314), .Y(N13) );
NAND2X1 U525 ( .A(n130), .B(n131), .Y(n39) );
NAND2X1 U526 ( .A(N12), .B(bcd_next[8]), .Y(n131) );
NAND2X1 U527 ( .A(bcd_reg[5]), .B(n127), .Y(n130) );
NAND2X1 U528 ( .A(n225), .B(n226), .Y(n140) );
NAND2X1 U529 ( .A(bcd_reg[2]), .B(n218), .Y(n225) );
NAND2X1 U530 ( .A(N9), .B(bcd_next[4]), .Y(n226) );
NAND2X1 U531 ( .A(n310), .B(n309), .Y(N9) );
NAND2X1 U532 ( .A(n89), .B(n90), .Y(n69) );
NAND2X1 U533 ( .A(N26), .B(N23), .Y(n90) );
NAND2X1 U534 ( .A(bcd_reg[16]), .B(n81), .Y(n89) );
NAND2X1 U535 ( .A(n79), .B(n80), .Y(n75) );
NAND2X1 U536 ( .A(bcd_reg[18]), .B(n81), .Y(n79) );
NAND2X1 U537 ( .A(N25), .B(N26), .Y(n80) );
NAND2X1 U538 ( .A(n330), .B(n329), .Y(N25) );
NAND2X1 U539 ( .A(n85), .B(n86), .Y(n72) );
NAND2X1 U540 ( .A(N24), .B(N26), .Y(n86) );
NAND2X1 U541 ( .A(bcd_reg[17]), .B(n81), .Y(n85) );
INVX2 U542 ( .A(n293), .Y(n297) );
NOR4BX1 U543 ( .AN(n229), .B(rst_i), .C(start_i), .D(done_o), .Y(n293) );
NAND3X1 U544 ( .A(N8), .B(N7), .C(bcd_reg[2]), .Y(n310) );
NAND3X1 U545 ( .A(N24), .B(N23), .C(bcd_reg[18]), .Y(n330) );
NAND3X1 U546 ( .A(N20), .B(N19), .C(bcd_reg[14]), .Y(n325) );
NAND3X1 U547 ( .A(N16), .B(N15), .C(bcd_reg[10]), .Y(n320) );
NAND3X1 U548 ( .A(N12), .B(N11), .C(bcd_reg[6]), .Y(n315) );
INVX2 U549 ( .A(bcd_reg[0]), .Y(N7) );
INVX2 U550 ( .A(bcd_reg[16]), .Y(N23) );
INVX2 U551 ( .A(bcd_reg[12]), .Y(N19) );
INVX2 U552 ( .A(bcd_reg[8]), .Y(N15) );
INVX2 U553 ( .A(bcd_reg[4]), .Y(N11) );
NAND2BX1 U554 ( .AN(bcd_reg[18]), .B(n328), .Y(n329) );
NAND2X1 U555 ( .A(N24), .B(N23), .Y(n328) );
NAND2BX1 U556 ( .AN(bcd_reg[14]), .B(n323), .Y(n324) );
NAND2X1 U557 ( .A(N20), .B(N19), .Y(n323) );
NAND2BX1 U558 ( .AN(bcd_reg[10]), .B(n318), .Y(n319) );
NAND2X1 U559 ( .A(N16), .B(N15), .Y(n318) );
NAND2BX1 U560 ( .AN(bcd_reg[6]), .B(n313), .Y(n314) );
```

```
 NAND2X1 U561 ( .A(N12), .B(N11), .Y(n313) );
 NAND2X1 U562 ( .A(n221), .B(n222), .Y(n143) );
 NAND2X1 U563 ( .A(N8), .B(bcd_next[4]), .Y(n222) );
 NAND2X1 U564 ( .A(bcd_reg[1]), .B(n218), .Y(n221) );
 NAND2X1 U565 ( .A(n216), .B(n217), .Y(n146) );
 NAND2X1 U566 ( .A(N7), .B(bcd_next[4]), .Y(n217) );
 NAND2X1 U567 ( .A(bcd_reg[0]), .B(n218), .Y(n216) );
 NAND2BX1 U568 ( .AN(bcd_reg[2]), .B(n308), .Y(n309) );
 NAND2X1 U569 ( .A(N8), .B(N7), .Y(n308) );
 NAND2BX1 U570 ( .AN(bcd_reg[19]), .B(n332), .Y(N26) );
 NAND2X1 U571 ( .A(bcd_reg[18]), .B(n331), .Y(n332) );
 OR2X1 U572 ( .A(bcd_reg[17]), .B(bcd_reg[16]), .Y(n331) );
 NAND2BX1 U573 ( .AN(bcd_reg[3]), .B(n312), .Y(bcd_next[4]) );
 NAND2X1 U574 ( .A(bcd_reg[2]), .B(n311), .Y(n312) );
 OR2X1 U575 ( .A(bcd_reg[1]), .B(bcd_reg[0]), .Y(n311) );
 NAND2BX1 U576 ( .AN(bcd_reg[15]), .B(n327), .Y(bcd_next[16]) );
 NAND2X1 U577 ( .A(bcd_reg[14]), .B(n326), .Y(n327) );
 OR2X1 U578 ( .A(bcd_reg[13]), .B(bcd_reg[12]), .Y(n326) );
 NAND2BX1 U579 ( .AN(bcd_reg[11]), .B(n322), .Y(bcd_next[12]) );
 NAND2X1 U580 ( .A(bcd_reg[10]), .B(n321), .Y(n322) );
 OR2X1 U581 ( .A(bcd_reg[9]), .B(bcd_reg[8]), .Y(n321) );
 NAND2BX1 U582 ( .AN(bcd_reg[7]), .B(n317), .Y(bcd_next[8]) );
 NAND2X1 U583 ( .A(bcd_reg[6]), .B(n316), .Y(n317) );
 OR2X1 U584 ( .A(bcd_reg[5]), .B(bcd_reg[4]), .Y(n316) );
 XOR2X1 U585 ( .A(bit_count[1]), .B(bit_count[0]), .Y(N102) );
 XOR2X1 U586 ( .A(bit_count[3]), .B(n335), .Y(N104) );
 NOR2X1 U587 ( .A(n334), .B(n333), .Y(n335) );
 INVX2 U588 ( .A(bit_count[2]), .Y(n333) );
 INVX2 U589 ( .A(bit_count[0]), .Y(N101) );
 AND2X1 U590 ( .A(n203), .B(busy_bit), .Y(n230) );
 NAND3X1 U591 ( .A(n204), .B(n197), .C(n205), .Y(n203) );
 NAND2X1 U592 ( .A(bit_count[3]), .B(n198), .Y(n205) );
 NAND2X1 U593 ( .A(N104), .B(ce_i), .Y(n204) );
 AND2X1 U594 ( .A(n206), .B(busy_bit), .Y(n231) );
 NAND3X1 U595 ( .A(n207), .B(n197), .C(n208), .Y(n206) );
 NAND2X1 U596 ( .A(bit_count[2]), .B(n198), .Y(n208) );
 NAND2X1 U597 ( .A(N103), .B(ce_i), .Y(n207) );
 AND2X1 U598 ( .A(n209), .B(busy_bit), .Y(n232) );
 NAND3X1 U599 ( .A(n210), .B(n197), .C(n211), .Y(n209) );
 NAND2X1 U600 ( .A(bit_count[1]), .B(n198), .Y(n211) );
 NAND2X1 U601 ( .A(N102), .B(ce_i), .Y(n210) );
 AND2X1 U602 ( .A(n200), .B(busy_bit), .Y(n290) );
 NAND3X1 U603 ( .A(n201), .B(n197), .C(n202), .Y(n200) );
 NAND2X1 U604 ( .A(bit_count[0]), .B(n198), .Y(n202) );
 NAND2X1 U605 ( .A(N101), .B(ce_i), .Y(n201) );
endmodule
```

Αρχείο χωρητικοτήτων διασύνδεσης «binary_to_bcd.cap»

```
#################################################################
###################
# FirstEncounter Capacitance File                               #
```

```
#    Format: <netName> <wireCap> <pinCap> <totalCap> <netLength> <wireCapPerUnitLen>
<nrFanout> #
#  The wireCap, pinCap and totalCap are in the unit of pF and                #
#  the netLength is in the unit micron.                                      #
#  the capPerUnitLen is in pF per micron.                                    #
#  scaling factor: 1.000000.                                                 #
##############################################################################
###################
#
#  Print out the nets with capacitance in range: 0.000000 - 1000.000000 pf.
#
dat_binary_i[15] 0.007342      0.004014      0.011356      57.560  1.276e-04     1
dat_binary_i[14] 0.006662      0.004014      0.010676      52.003  1.281e-04     1
dat_binary_i[13] 0.007227      0.004014      0.011241      56.423  1.281e-04     1
dat_binary_i[12] 0.006577      0.004014      0.010591      51.343  1.281e-04     1
dat_binary_i[11] 0.007342      0.004014      0.011356      57.560  1.276e-04     1
dat_binary_i[10] 0.006698      0.004014      0.010712      52.520  1.275e-04     1
dat_binary_i[9]  0.008630      0.004014      0.012644      67.640  1.276e-04     1
dat_binary_i[8]  0.006698      0.004014      0.010712      52.520  1.275e-04     1
dat_binary_i[7]  0.006698      0.004014      0.010712      52.520  1.275e-04     1
dat_binary_i[6]  0.006698      0.004014      0.010712      52.520  1.275e-04     1
dat_binary_i[5]  0.006662      0.004014      0.010676      52.003  1.281e-04     1
dat_binary_i[4]  0.006782      0.004014      0.010796      53.180  1.275e-04     1
dat_binary_i[3]  0.006698      0.004014      0.010712      52.520  1.275e-04     1
dat_binary_i[2]  0.007084      0.004014      0.011098      55.303  1.281e-04     1
dat_binary_i[1]  0.006974      0.004014      0.010988      54.443  1.281e-04     1
dat_binary_i[0]  0.006720      0.004014      0.010734      52.463  1.281e-04     1
dat_bcd_o[19]    0.008398      0.004014      0.012412      65.563  1.281e-04     2
dat_bcd_o[18]    0.007596      0.004014      0.011610      59.540  1.276e-04     2
dat_bcd_o[17]    0.009549      0.004014      0.013563      74.563  1.281e-04     2
dat_bcd_o[16]    0.008144      0.004014      0.012158      63.583  1.281e-04     2
dat_bcd_o[15]    0.009371      0.004014      0.013385      73.160  1.281e-04     2
dat_bcd_o[14]    0.008611      0.004014      0.012625      67.490  1.276e-04     2
dat_bcd_o[13]    0.009000      0.004014      0.013014      70.640  1.274e-04     2
dat_bcd_o[12]    0.010627      0.004014      0.014641      83.240  1.277e-04     2
dat_bcd_o[11]    0.009508      0.004014      0.013522      74.240  1.281e-04     2
dat_bcd_o[10]    0.007036      0.004014      0.011050      55.160  1.276e-04     2
dat_bcd_o[9]     0.008777      0.004014      0.012791      68.840  1.275e-04     2
dat_bcd_o[8]     0.009124      0.004014      0.013138      71.203  1.281e-04     2
dat_bcd_o[7]     0.008788      0.004014      0.012802      68.623  1.281e-04     2
dat_bcd_o[6]     0.008796      0.004014      0.012810      68.683  1.281e-04     2
dat_bcd_o[5]     0.008895      0.004014      0.012909      69.800  1.274e-04     2
dat_bcd_o[4]     0.011926      0.004014      0.015940      93.043  1.282e-04     2
dat_bcd_o[3]     0.008148      0.004014      0.012162      63.583  1.281e-04     2
dat_bcd_o[2]     0.008545      0.004014      0.012559      66.683  1.281e-04     2
dat_bcd_o[1]     0.010063      0.004014      0.014077      78.560  1.281e-04     2
dat_bcd_o[0]     0.008147      0.004014      0.012161      63.583  1.281e-04     2
clk_i            0.079851      0.177815      0.257665      624.540 1.279e-04     61
ce_i             0.009788      0.021907      0.031695      76.720  1.276e-04     5
rst_i            0.009976      0.028325      0.038301      78.180  1.276e-04     6
start_i          0.008585      0.017091      0.025676      67.300  1.276e-04     4
done_o           0.009522      0.019823      0.029345      74.620  1.276e-04     5
bin_next[15]     0.001061      0.008028      0.009089      8.280   1.282e-04     2
bin_next[14]     0.001192      0.008028      0.009220      9.300   1.281e-04     2
bin_next[13]     0.001620      0.008028      0.009648      12.660  1.280e-04     2
bin_next[12]     0.001536      0.008028      0.009564      12.000  1.280e-04     2
bin_next[11]     0.000936      0.008028      0.008964      7.320   1.279e-04     2
bin_next[10]     0.001327      0.008028      0.009355      10.380  1.279e-04     2
bin_next[9]      0.002095      0.008028      0.010123      16.380  1.279e-04     2
```

bin_next[8]	0.001621	0.008028	0.009649	12.720	1.274e-04	2
bin_next[7]	0.001457	0.008028	0.009485	11.400	1.278e-04	2
bin_next[6]	0.002500	0.008028	0.010528	19.500	1.282e-04	2
bin_next[5]	0.002251	0.008028	0.010279	17.580	1.280e-04	2
bin_next[4]	0.001791	0.008028	0.009819	14.040	1.276e-04	2
bin_next[3]	0.001747	0.008028	0.009775	13.680	1.277e-04	2
bin_next[2]	0.002823	0.008028	0.010851	22.040	1.281e-04	2
bin_next[1]	0.002036	0.008028	0.010064	15.900	1.280e-04	2
N7	0.002202	0.018075	0.020277	17.240	1.277e-04	4
N8	0.004462	0.012762	0.017224	34.880	1.279e-04	3
N9	0.000312	0.004014	0.004326	2.440	1.279e-04	1
N11	0.007188	0.017829	0.025017	56.380	1.275e-04	4
N12	0.003288	0.012762	0.016050	25.680	1.280e-04	3
N13	0.000988	0.004014	0.005002	7.720	1.280e-04	1
N15	0.004579	0.017829	0.022408	35.800	1.279e-04	4
N16	0.001630	0.012762	0.014392	12.760	1.278e-04	3
N17	0.000312	0.004014	0.004326	2.440	1.278e-04	1
N19	0.001923	0.017829	0.019752	15.040	1.279e-04	4
N20	0.002201	0.012762	0.014963	17.240	1.276e-04	3
N21	0.002024	0.004014	0.006038	15.820	1.280e-04	1
N23	0.002877	0.017829	0.020706	22.520	1.278e-04	4
N24	0.002313	0.012762	0.015075	18.100	1.278e-04	3
N25	0.000583	0.004014	0.004597	4.580	1.274e-04	1
N26	0.002379	0.018385	0.020764	18.600	1.279e-04	4
busy_bit	0.006650	0.023671	0.030321	51.980	1.279e-04	7
N101	0.001217	0.004014	0.005231	9.520	1.278e-04	1
N102	0.001941	0.004014	0.005955	15.140	1.282e-04	1
N103	0.000828	0.004014	0.004842	6.460	1.281e-04	1
N104	0.001393	0.004014	0.005407	10.880	1.280e-04	1
n32	0.000585	0.004014	0.004599	4.580	1.278e-04	1
n33	0.002802	0.003768	0.006570	21.900	1.279e-04	1
n34	0.013034	0.022854	0.035888	101.840	1.280e-04	6
n35	0.005681	0.007536	0.013217	44.400	1.279e-04	2
n36	0.026424	0.079374	0.105798	206.500	1.280e-04	21
n37	0.002277	0.004014	0.006291	17.800	1.279e-04	1
n38	0.007042	0.003768	0.010810	55.000	1.280e-04	1
n39	0.002613	0.007536	0.010149	20.420	1.280e-04	2
n40	0.000312	0.004014	0.004326	2.440	1.278e-04	1
n41	0.007998	0.003768	0.011766	62.460	1.281e-04	1
n42	0.003192	0.007536	0.010728	24.940	1.280e-04	2
n43	0.001837	0.004014	0.005851	14.320	1.283e-04	1
n44	0.002628	0.003768	0.006396	20.520	1.281e-04	1
n45	0.000752	0.004014	0.004766	5.900	1.274e-04	1
n46	0.001195	0.003768	0.004963	9.360	1.277e-04	1
n47	0.000764	0.007536	0.008300	6.000	1.274e-04	2
n48	0.000312	0.004014	0.004326	2.440	1.280e-04	1
n49	0.001470	0.003768	0.005238	11.500	1.278e-04	1
n50	0.004187	0.007536	0.011723	32.800	1.277e-04	2
n51	0.000312	0.004014	0.004326	2.440	1.278e-04	1
n52	0.003311	0.003768	0.007079	25.860	1.281e-04	1
n53	0.001580	0.007536	0.009116	12.340	1.280e-04	2
n54	0.000787	0.004014	0.004801	6.160	1.278e-04	1
n55	0.001593	0.003768	0.005361	12.440	1.281e-04	1
n56	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n57	0.002640	0.003768	0.006408	20.720	1.274e-04	1
n58	0.003614	0.007536	0.011150	28.180	1.283e-04	2
n59	0.000312	0.004014	0.004326	2.440	1.280e-04	1
n60	0.008350	0.003768	0.012118	65.240	1.280e-04	1
n61	0.003456	0.007536	0.010992	27.040	1.278e-04	2

n62	0.001856	0.004014	0.005870	14.500	1.280e-04	1
n63	0.003369	0.003768	0.007137	26.300	1.281e-04	1
n64	0.005909	0.007536	0.013445	46.160	1.280e-04	2
n65	0.000585	0.004014	0.004599	4.580	1.278e-04	1
n66	0.008226	0.003768	0.011994	64.240	1.281e-04	1
n67	0.000904	0.004014	0.004918	7.060	1.281e-04	1
n68	0.007432	0.003768	0.011200	58.080	1.280e-04	1
n69	0.002033	0.007536	0.009569	15.920	1.277e-04	2
n70	0.000735	0.004014	0.004749	5.740	1.280e-04	1
n71	0.006158	0.003768	0.009926	48.300	1.275e-04	1
n72	0.005104	0.007536	0.012640	39.980	1.277e-04	2
n73	0.001040	0.004014	0.005054	8.140	1.278e-04	1
n74	0.007902	0.003768	0.011670	61.640	1.282e-04	1
n75	0.002217	0.007536	0.009753	17.400	1.274e-04	2
n76	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n77	0.001388	0.003768	0.005156	10.840	1.280e-04	1
n78	0.048778	0.140540	0.189318	381.340	1.279e-04	36
n79	0.000583	0.004014	0.004597	4.580	1.274e-04	1
n80	0.001024	0.003768	0.004792	8.040	1.274e-04	1
n81	0.002024	0.011304	0.013328	15.800	1.281e-04	3
n83	0.000651	0.004014	0.004665	5.080	1.281e-04	1
n84	0.001618	0.003768	0.005386	12.660	1.278e-04	1
n85	0.000785	0.004014	0.004799	6.160	1.274e-04	1
n86	0.000156	0.003768	0.003924	1.220	1.279e-04	1
n87	0.000481	0.004014	0.004495	3.760	1.280e-04	1
n88	0.000512	0.003768	0.004280	4.020	1.274e-04	1
n89	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n90	0.000596	0.003768	0.004364	4.680	1.274e-04	1
n91	0.000499	0.004014	0.004513	3.920	1.274e-04	1
n92	0.001297	0.003768	0.005065	10.180	1.274e-04	1
n93	0.000565	0.004014	0.004579	4.420	1.279e-04	1
n94	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n95	0.000785	0.004014	0.004799	6.160	1.274e-04	1
n96	0.001111	0.003768	0.004879	8.700	1.277e-04	1
n97	0.001260	0.011304	0.012564	9.860	1.278e-04	3
n98	0.000565	0.004014	0.004579	4.420	1.279e-04	1
n99	0.000917	0.003768	0.004685	7.160	1.280e-04	1
n100	0.000565	0.004014	0.004579	4.420	1.279e-04	1
n101	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n102	0.001177	0.004014	0.005191	9.200	1.279e-04	1
n103	0.000682	0.003768	0.004450	5.340	1.277e-04	1
n104	0.000954	0.004014	0.004968	7.480	1.276e-04	1
n105	0.000325	0.003768	0.004093	2.540	1.281e-04	1
n106	0.000735	0.004014	0.004749	5.740	1.281e-04	1
n107	0.003940	0.003768	0.007708	30.720	1.283e-04	1
n108	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n109	0.002984	0.003768	0.006752	23.400	1.275e-04	1
n110	0.000312	0.004014	0.004326	2.440	1.278e-04	1
n111	0.001356	0.003768	0.005124	10.640	1.274e-04	1
n112	0.002045	0.011304	0.013349	16.020	1.276e-04	3
n113	0.000584	0.004014	0.004598	4.580	1.275e-04	1
n114	0.001274	0.003768	0.005042	9.960	1.279e-04	1
n115	0.000499	0.004014	0.004513	3.920	1.274e-04	1
n116	0.000325	0.003768	0.004093	2.540	1.279e-04	1
n117	0.000785	0.004014	0.004799	6.160	1.274e-04	1
n118	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n119	0.000396	0.004014	0.004410	3.100	1.278e-04	1
n120	0.000410	0.003768	0.004178	3.200	1.280e-04	1
n121	0.001092	0.004014	0.005106	8.540	1.278e-04	1

n122	0.000512	0.003768	0.004280	4.020	1.274e-04	1
n123	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n124	0.000512	0.003768	0.004280	4.020	1.274e-04	1
n125	0.000668	0.004014	0.004682	5.240	1.274e-04	1
n126	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n127	0.002043	0.011304	0.013347	16.020	1.275e-04	3
n128	0.002701	0.004014	0.006715	21.100	1.280e-04	1
n129	0.001024	0.003768	0.004792	8.040	1.274e-04	1
n130	0.001326	0.004014	0.005340	10.360	1.280e-04	1
n131	0.000410	0.003768	0.004178	3.200	1.280e-04	1
n132	0.001037	0.004014	0.005051	8.140	1.274e-04	1
n133	0.000851	0.003768	0.004619	6.660	1.278e-04	1
n134	0.000650	0.004014	0.004664	5.080	1.279e-04	1
n135	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n136	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n137	0.000851	0.003768	0.004619	6.660	1.278e-04	1
n138	0.000499	0.004014	0.004513	3.920	1.274e-04	1
n139	0.000940	0.003768	0.004708	7.380	1.274e-04	1
n140	0.004344	0.007536	0.011880	33.920	1.281e-04	2
n141	0.001037	0.004014	0.005051	8.140	1.274e-04	1
n142	0.000680	0.003768	0.004448	5.340	1.274e-04	1
n143	0.004672	0.007536	0.012208	36.520	1.279e-04	2
n144	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n145	0.000156	0.003768	0.003924	1.220	1.279e-04	1
n146	0.003290	0.007536	0.010826	25.720	1.279e-04	2
n147	0.000836	0.004014	0.004850	6.560	1.274e-04	1
n148	0.000579	0.003768	0.004347	4.520	1.282e-04	1
n149	0.000508	0.004734	0.005242	3.960	1.283e-04	1
n150	0.007223	0.004599	0.011822	56.360	1.282e-04	1
n151	0.000429	0.004401	0.004830	3.360	1.278e-04	1
n152	0.019982	0.066351	0.086333	156.080	1.280e-04	18
n153	0.001099	0.004734	0.005833	8.600	1.278e-04	1
n154	0.008259	0.004599	0.012858	64.540	1.280e-04	1
n155	0.000853	0.004401	0.005254	6.660	1.280e-04	1
n156	0.000254	0.004734	0.004988	1.980	1.283e-04	1
n157	0.007548	0.004599	0.012147	59.020	1.279e-04	1
n158	0.000410	0.004401	0.004811	3.200	1.282e-04	1
n159	0.002794	0.004734	0.007528	21.780	1.283e-04	1
n160	0.004347	0.004599	0.008946	34.000	1.278e-04	1
n161	0.001257	0.004401	0.005658	9.800	1.283e-04	1
n162	0.001015	0.004734	0.005749	7.940	1.278e-04	1
n163	0.007347	0.004599	0.011946	57.440	1.279e-04	1
n164	0.000326	0.004401	0.004727	2.540	1.282e-04	1
n165	0.000253	0.004734	0.004987	1.980	1.278e-04	1
n166	0.009922	0.004599	0.014521	77.440	1.281e-04	1
n167	0.000682	0.004401	0.005083	5.340	1.278e-04	1
n168	0.000442	0.004734	0.005176	3.460	1.278e-04	1
n169	0.007483	0.004599	0.012082	58.520	1.279e-04	1
n170	0.000156	0.004401	0.004557	1.220	1.278e-04	1
n171	0.000254	0.004734	0.004988	1.980	1.283e-04	1
n172	0.012346	0.004599	0.016945	96.340	1.282e-04	1
n173	0.000156	0.004401	0.004557	1.220	1.278e-04	1
n174	0.000253	0.004734	0.004987	1.980	1.278e-04	1
n175	0.011632	0.004599	0.016231	90.880	1.280e-04	1
n176	0.000495	0.004401	0.004896	3.860	1.282e-04	1
n177	0.001328	0.004734	0.006062	10.360	1.282e-04	1
n178	0.010949	0.004599	0.015548	85.420	1.282e-04	1
n179	0.000579	0.004401	0.004980	4.520	1.282e-04	1
n180	0.000357	0.004734	0.005091	2.800	1.274e-04	1

n181	0.014758	0.004599	0.019357	115.120	1.282e-04	1
n182	0.000155	0.004401	0.004556	1.220	1.274e-04	1
n183	0.000252	0.004734	0.004986	1.980	1.274e-04	1
n184	0.010512	0.004599	0.015111	82.120	1.280e-04	1
n185	0.000410	0.004401	0.004811	3.200	1.281e-04	1
n186	0.000253	0.004734	0.004987	1.980	1.278e-04	1
n187	0.008303	0.004599	0.012902	64.780	1.282e-04	1
n188	0.000664	0.004401	0.005065	5.180	1.282e-04	1
n189	0.000423	0.004734	0.005157	3.300	1.281e-04	1
n190	0.007897	0.004599	0.012496	61.700	1.280e-04	1
n191	0.000325	0.004401	0.004726	2.540	1.279e-04	1
n192	0.000507	0.004734	0.005241	3.960	1.281e-04	1
n193	0.004768	0.004599	0.009367	37.280	1.279e-04	1
n194	0.000156	0.004401	0.004557	1.220	1.278e-04	1
n195	0.000585	0.004014	0.004599	4.580	1.278e-04	1
n196	0.006817	0.003768	0.010585	53.260	1.280e-04	1
n197	0.005962	0.025303	0.031265	46.560	1.280e-04	6
n198	0.004618	0.024021	0.028639	36.100	1.279e-04	6
n199	0.001061	0.003933	0.004994	8.280	1.282e-04	1
n200	0.001745	0.002031	0.003776	13.620	1.281e-04	1
n201	0.000253	0.004734	0.004987	1.980	1.278e-04	1
n202	0.000156	0.004401	0.004557	1.220	1.278e-04	1
n203	0.001497	0.002031	0.003528	11.680	1.282e-04	1
n204	0.001183	0.004734	0.005917	9.260	1.278e-04	1
n205	0.001028	0.004401	0.005429	8.040	1.279e-04	1
n206	0.001406	0.002031	0.003437	10.980	1.281e-04	1
n207	0.000254	0.004734	0.004988	1.980	1.283e-04	1
n208	0.000514	0.004401	0.004915	4.020	1.278e-04	1
n209	0.000481	0.002031	0.002512	3.760	1.278e-04	1
n210	0.000254	0.004734	0.004988	1.980	1.283e-04	1
n211	0.002036	0.004401	0.006437	15.920	1.279e-04	1
n212	0.000670	0.004014	0.004684	5.240	1.278e-04	1
n213	0.004201	0.003768	0.007969	32.760	1.282e-04	1
n214	0.000755	0.004014	0.004769	5.900	1.279e-04	1
n215	0.008733	0.003768	0.012501	68.140	1.282e-04	1
n216	0.000311	0.004014	0.004325	2.440	1.274e-04	1
n217	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n218	0.001800	0.011304	0.013104	14.080	1.279e-04	3
n219	0.000312	0.004014	0.004326	2.440	1.278e-04	1
n220	0.009169	0.003768	0.012937	71.580	1.281e-04	1
n221	0.001177	0.004014	0.005191	9.200	1.279e-04	1
n222	0.001024	0.003768	0.004792	8.040	1.274e-04	1
n223	0.000585	0.004014	0.004599	4.580	1.278e-04	1
n224	0.009761	0.003768	0.013529	76.200	1.281e-04	1
n225	0.000395	0.004014	0.004409	3.100	1.274e-04	1
n226	0.000512	0.003768	0.004280	4.020	1.274e-04	1
n227	0.000787	0.004014	0.004801	6.160	1.278e-04	1
n228	0.005192	0.003768	0.008960	40.500	1.282e-04	1
n229	0.001927	0.008025	0.009952	15.040	1.281e-04	3
n230	0.001888	0.002120	0.004008	14.740	1.281e-04	1
n231	0.001329	0.002120	0.003449	10.360	1.282e-04	1
n232	0.000397	0.002120	0.002517	3.100	1.281e-04	1
n233	0.000358	0.002120	0.002478	2.800	1.278e-04	1
n234	0.002810	0.002120	0.004930	21.940	1.281e-04	1
n235	0.000762	0.002120	0.002882	5.940	1.283e-04	1
n236	0.000612	0.002120	0.002732	4.780	1.280e-04	1
n237	0.000254	0.002120	0.002374	1.980	1.283e-04	1
n238	0.000781	0.002120	0.002901	6.100	1.280e-04	1
n239	0.000254	0.002120	0.002374	1.980	1.281e-04	1

n240	0.001015	0.002120	0.003135	7.940	1.278e-04	1
n241	0.001289	0.002120	0.003409	10.060	1.282e-04	1
n242	0.000254	0.002120	0.002374	1.980	1.281e-04	1
n243	0.001522	0.002120	0.003642	11.880	1.281e-04	1
n244	0.000930	0.002120	0.003050	7.280	1.278e-04	1
n245	0.001270	0.002120	0.003390	9.920	1.281e-04	1
n246	0.001437	0.002120	0.003557	11.220	1.281e-04	1
n247	0.001691	0.002120	0.003811	13.200	1.281e-04	1
n248	0.001373	0.002120	0.003493	10.740	1.278e-04	1
n249	0.000611	0.002120	0.002731	4.780	1.279e-04	1
n250	0.000930	0.002120	0.003050	7.280	1.278e-04	1
n251	0.001097	0.002120	0.003217	8.600	1.275e-04	1
n252	0.001438	0.002120	0.003558	11.240	1.280e-04	1
n253	0.001440	0.002120	0.003560	11.220	1.283e-04	1
n254	0.001368	0.002120	0.003488	10.740	1.274e-04	1
n255	0.001185	0.002120	0.003305	9.240	1.283e-04	1
n256	0.001355	0.002120	0.003475	10.560	1.283e-04	1
n257	0.000761	0.002120	0.002881	5.940	1.281e-04	1
n258	0.000441	0.002120	0.002561	3.460	1.274e-04	1
n259	0.000252	0.002120	0.002372	1.980	1.274e-04	1
n260	0.000357	0.002120	0.002477	2.800	1.274e-04	1
n261	0.000254	0.002120	0.002374	1.980	1.281e-04	1
n262	0.000252	0.002120	0.002372	1.980	1.274e-04	1
n263	0.001184	0.002120	0.003304	9.240	1.281e-04	1
n264	0.000252	0.002120	0.002372	1.980	1.274e-04	1
n265	0.001184	0.002120	0.003304	9.240	1.281e-04	1
n266	0.000423	0.002120	0.002543	3.300	1.281e-04	1
n267	0.000423	0.002120	0.002543	3.300	1.281e-04	1
n268	0.001185	0.002120	0.003305	9.240	1.283e-04	1
n269	0.001185	0.002120	0.003305	9.240	1.283e-04	1
n270	0.000423	0.002120	0.002543	3.300	1.283e-04	1
n271	0.001101	0.002120	0.003221	8.580	1.283e-04	1
n272	0.000357	0.002120	0.002477	2.800	1.274e-04	1
n273	0.004888	0.002120	0.007008	38.160	1.281e-04	1
n274	0.001334	0.002120	0.003454	10.420	1.281e-04	1
n275	0.000996	0.002120	0.003116	7.780	1.280e-04	1
n276	0.000800	0.002120	0.002920	6.260	1.278e-04	1
n277	0.000969	0.002120	0.003089	7.580	1.278e-04	1
n278	0.001485	0.002120	0.003605	11.580	1.282e-04	1
n279	0.001927	0.002120	0.004047	15.040	1.281e-04	1
n280	0.001498	0.002120	0.003618	11.760	1.274e-04	1
n281	0.000573	0.002120	0.002693	4.480	1.278e-04	1
n282	0.000907	0.002120	0.003027	7.120	1.274e-04	1
n283	0.000466	0.002120	0.002586	3.660	1.274e-04	1
n284	0.000825	0.002120	0.002945	6.460	1.277e-04	1
n285	0.000657	0.002120	0.002777	5.140	1.279e-04	1
n286	0.000910	0.002120	0.003030	7.120	1.278e-04	1
n287	0.001229	0.002120	0.003349	9.600	1.280e-04	1
n288	0.000254	0.002120	0.002374	1.980	1.281e-04	1
n289	0.000442	0.002120	0.002562	3.460	1.278e-04	1
n290	0.000397	0.002120	0.002517	3.100	1.281e-04	1
n291	0.048491	0.138600	0.187091	379.100	1.279e-04	36
n292	0.014308	0.028316	0.042624	111.880	1.279e-04	7
n293	0.018821	0.032112	0.050933	146.980	1.281e-04	8
n308	0.000823	0.003933	0.004756	6.460	1.274e-04	1
n309	0.000918	0.003768	0.004686	7.160	1.282e-04	1
n310	0.000410	0.004014	0.004424	3.200	1.281e-04	1
n311	0.000940	0.003768	0.004708	7.380	1.274e-04	1
n312	0.002002	0.003933	0.005935	15.660	1.278e-04	1

n313	0.000466	0.003933	0.004399	3.660	1.274e-04	1
n314	0.000856	0.003768	0.004624	6.720	1.274e-04	1
n315	0.000240	0.004014	0.004254	1.880	1.274e-04	1
n316	0.001024	0.003768	0.004792	8.040	1.274e-04	1
n317	0.000552	0.003933	0.004485	4.320	1.278e-04	1
n318	0.000466	0.003933	0.004399	3.660	1.274e-04	1
n319	0.000917	0.003768	0.004685	7.160	1.280e-04	1
n320	0.000240	0.004014	0.004254	1.880	1.274e-04	1
n321	0.001363	0.003768	0.005131	10.680	1.277e-04	1
n322	0.001501	0.003933	0.005434	11.740	1.278e-04	1
n323	0.000571	0.003933	0.004504	4.480	1.274e-04	1
n324	0.000408	0.003768	0.004176	3.200	1.274e-04	1
n325	0.000324	0.004014	0.004338	2.540	1.274e-04	1
n326	0.000596	0.003768	0.004364	4.680	1.274e-04	1
n327	0.000636	0.003933	0.004569	4.980	1.278e-04	1
n328	0.000722	0.003933	0.004655	5.640	1.280e-04	1
n329	0.001003	0.003768	0.004771	7.820	1.282e-04	1
n330	0.000410	0.004014	0.004424	3.200	1.281e-04	1
n331	0.000155	0.003768	0.003923	1.220	1.274e-04	1
n332	0.001248	0.003933	0.005181	9.760	1.279e-04	1
n333	0.000736	0.009627	0.010363	5.740	1.281e-04	2
n334	0.001536	0.010983	0.012519	12.000	1.280e-04	2
n335	0.002267	0.006751	0.009018	17.700	1.281e-04	1
bcd_reg[19]	0.000895	0.006144	0.007039	7.020	1.275e-04	2
bcd_reg[18]	0.004698	0.018327	0.023025	36.700	1.280e-04	5
bcd_reg[17]	0.002830	0.017033	0.019863	22.180	1.276e-04	4
bcd_reg[16]	0.005116	0.017239	0.022355	39.940	1.281e-04	4
bcd_reg[15]	0.002209	0.006144	0.008353	17.280	1.278e-04	2
bcd_reg[14]	0.004728	0.018327	0.023055	37.000	1.278e-04	5
bcd_reg[13]	0.005040	0.017033	0.022073	39.400	1.279e-04	4
bcd_reg[12]	0.003383	0.017239	0.020622	26.480	1.277e-04	4
bcd_reg[11]	0.002586	0.006144	0.008730	20.220	1.279e-04	2
bcd_reg[10]	0.006153	0.018327	0.024480	48.060	1.280e-04	5
bcd_reg[9]	0.004449	0.017033	0.021482	34.840	1.277e-04	4
bcd_reg[8]	0.004272	0.017239	0.021511	33.460	1.277e-04	4
bcd_reg[7]	0.001826	0.006144	0.007970	14.280	1.279e-04	2
bcd_reg[6]	0.003617	0.018327	0.021944	28.320	1.277e-04	5
bcd_reg[5]	0.006737	0.017033	0.023770	52.660	1.279e-04	4
bcd_reg[4]	0.006166	0.017239	0.023405	48.320	1.276e-04	4
bcd_reg[3]	0.001402	0.006144	0.007546	10.980	1.277e-04	2
bcd_reg[2]	0.005164	0.018573	0.023737	40.320	1.281e-04	5
bcd_reg[1]	0.003056	0.017279	0.020335	23.880	1.280e-04	4
bcd_reg[0]	0.005005	0.017485	0.022490	39.140	1.279e-04	4
bcd_next[16]	0.005589	0.026167	0.031756	43.720	1.278e-04	6
bcd_next[12]	0.008410	0.026167	0.034577	65.800	1.278e-04	6
bcd_next[8]	0.011162	0.026167	0.037329	87.260	1.279e-04	6
bcd_next[4]	0.008136	0.026167	0.034303	63.600	1.279e-04	6
bcd_next[0]	0.007689	0.012042	0.019731	59.980	1.282e-04	3
bit_count[3]	0.003282	0.014204	0.017486	25.620	1.281e-04	3
bit_count[2]	0.005456	0.015340	0.020796	42.580	1.281e-04	3
bit_count[1]	0.002922	0.018253	0.021175	22.820	1.281e-04	4
bit_count[0]	0.007370	0.025682	0.033052	57.560	1.280e-04	5

Αρχείο περιγραφής κυκλώματος σε μορφή SPICE «binary_to_bcd.ckt»

$ Spice netlist generated by v2lvs
$ v2007.3_18.11 Wed Aug 22 13:24:38 PDT 2007

```
.SUBCKT binary_to_bcd clk_i ce_i rst_i start_i dat_binary_i[15]
+ dat_binary_i[14] dat_binary_i[13] dat_binary_i[12] dat_binary_i[11]
+ dat_binary_i[10] dat_binary_i[9] dat_binary_i[8] dat_binary_i[7]
+ dat_binary_i[6] dat_binary_i[5] dat_binary_i[4] dat_binary_i[3]
+ dat_binary_i[2] dat_binary_i[1] dat_binary_i[0] dat_bcd_o[19] dat_bcd_o[18]
+ dat_bcd_o[17] dat_bcd_o[16] dat_bcd_o[15] dat_bcd_o[14] dat_bcd_o[13]
+ dat_bcd_o[12] dat_bcd_o[11] dat_bcd_o[10] dat_bcd_o[9] dat_bcd_o[8]
+ dat_bcd_o[7] dat_bcd_o[6] dat_bcd_o[5] dat_bcd_o[4] dat_bcd_o[3] dat_bcd_o[2]
+ dat_bcd_o[1] dat_bcd_o[0] done_o
Xdat_bcd_o_reg[19] dat_bcd_o[19] clk_i n252 1000 1001 DFFHQX1
Xdat_bcd_o_reg[18] dat_bcd_o[18] clk_i n251 1002 1003 DFFHQX1
Xdat_bcd_o_reg[17] dat_bcd_o[17] clk_i n250 1004 1005 DFFHQX1
Xdat_bcd_o_reg[16] dat_bcd_o[16] clk_i n249 1006 1007 DFFHQX1
Xdat_bcd_o_reg[15] dat_bcd_o[15] clk_i n248 1008 1009 DFFHQX1
Xdat_bcd_o_reg[14] dat_bcd_o[14] clk_i n247 1010 1011 DFFHQX1
Xdat_bcd_o_reg[13] dat_bcd_o[13] clk_i n246 1012 1013 DFFHQX1
Xdat_bcd_o_reg[12] dat_bcd_o[12] clk_i n245 1014 1015 DFFHQX1
Xdat_bcd_o_reg[11] dat_bcd_o[11] clk_i n244 1016 1017 DFFHQX1
Xdat_bcd_o_reg[10] dat_bcd_o[10] clk_i n243 1018 1019 DFFHQX1
Xdat_bcd_o_reg[9] dat_bcd_o[9] clk_i n242 1020 1021 DFFHQX1
Xdat_bcd_o_reg[8] dat_bcd_o[8] clk_i n241 1022 1023 DFFHQX1
Xdat_bcd_o_reg[7] dat_bcd_o[7] clk_i n240 1024 1025 DFFHQX1
Xdat_bcd_o_reg[6] dat_bcd_o[6] clk_i n239 1026 1027 DFFHQX1
Xdat_bcd_o_reg[5] dat_bcd_o[5] clk_i n238 1028 1029 DFFHQX1
Xdat_bcd_o_reg[4] dat_bcd_o[4] clk_i n237 1030 1031 DFFHQX1
Xdat_bcd_o_reg[3] dat_bcd_o[3] clk_i n236 1032 1033 DFFHQX1
Xdat_bcd_o_reg[2] dat_bcd_o[2] clk_i n235 1034 1035 DFFHQX1
Xdat_bcd_o_reg[1] dat_bcd_o[1] clk_i n234 1036 1037 DFFHQX1
Xdat_bcd_o_reg[0] dat_bcd_o[0] clk_i n233 1038 1039 DFFHQX1
Xbcd_reg_reg[3] bcd_reg[3] clk_i n269 1040 1041 DFFHQX1
Xbcd_reg_reg[7] bcd_reg[7] clk_i n265 1042 1043 DFFHQX1
Xbcd_reg_reg[11] bcd_reg[11] clk_i n261 1044 1045 DFFHQX1
Xbcd_reg_reg[15] bcd_reg[15] clk_i n257 1046 1047 DFFHQX1
Xbcd_reg_reg[19] bcd_reg[19] clk_i n253 1048 1049 DFFHQX1
Xbin_reg_reg[0] bin_next[1] clk_i n288 1050 1051 DFFHQX1
Xbin_reg_reg[1] bin_next[2] clk_i n287 1052 1053 DFFHQX1
Xbin_reg_reg[2] bin_next[3] clk_i n286 1054 1055 DFFHQX1
Xbin_reg_reg[3] bin_next[4] clk_i n285 1056 1057 DFFHQX1
Xbin_reg_reg[4] bin_next[5] clk_i n284 1058 1059 DFFHQX1
Xbin_reg_reg[5] bin_next[6] clk_i n283 1060 1061 DFFHQX1
Xbin_reg_reg[6] bin_next[7] clk_i n282 1062 1063 DFFHQX1
Xbin_reg_reg[7] bin_next[8] clk_i n281 1064 1065 DFFHQX1
Xbin_reg_reg[8] bin_next[9] clk_i n280 1066 1067 DFFHQX1
Xbin_reg_reg[9] bin_next[10] clk_i n279 1068 1069 DFFHQX1
Xbin_reg_reg[10] bin_next[11] clk_i n278 1070 1071 DFFHQX1
Xbin_reg_reg[11] bin_next[12] clk_i n277 1072 1073 DFFHQX1
Xbin_reg_reg[12] bin_next[13] clk_i n276 1074 1075 DFFHQX1
Xbin_reg_reg[13] bin_next[14] clk_i n275 1076 1077 DFFHQX1
Xbin_reg_reg[14] bin_next[15] clk_i n274 1078 1079 DFFHQX1
Xbin_reg_reg[15] bcd_next[0] clk_i n273 1080 1081 DFFHQX1
Xbcd_reg_reg[4] bcd_reg[4] clk_i n268 1082 1083 DFFHQX1
Xbcd_reg_reg[8] bcd_reg[8] clk_i n264 1084 1085 DFFHQX1
Xbcd_reg_reg[12] bcd_reg[12] clk_i n260 1086 1087 DFFHQX1
```

Xbcd_reg_reg[16] bcd_reg[16] clk_i n256 1088 1089 DFFHQX1
Xbcd_reg_reg[0] bcd_reg[0] clk_i n272 1090 1091 DFFHQX1
Xbcd_reg_reg[5] bcd_reg[5] clk_i n267 1092 1093 DFFHQX1
Xbcd_reg_reg[9] bcd_reg[9] clk_i n263 1094 1095 DFFHQX1
Xbcd_reg_reg[13] bcd_reg[13] clk_i n259 1096 1097 DFFHQX1
Xbcd_reg_reg[17] bcd_reg[17] clk_i n255 1098 1099 DFFHQX1
Xbcd_reg_reg[1] bcd_reg[1] clk_i n271 1100 1101 DFFHQX1
Xbcd_reg_reg[6] bcd_reg[6] clk_i n266 1102 1103 DFFHQX1
Xbcd_reg_reg[10] bcd_reg[10] clk_i n262 1104 1105 DFFHQX1
Xbcd_reg_reg[14] bcd_reg[14] clk_i n258 1106 1107 DFFHQX1
Xbcd_reg_reg[18] bcd_reg[18] clk_i n254 1108 1109 DFFHQX1
Xbcd_reg_reg[2] bcd_reg[2] clk_i n270 1110 1111 DFFHQX1
Xbit_count_reg[3] bit_count[3] clk_i n230 1112 1113 DFFHQX1
Xbit_count_reg[2] bit_count[2] clk_i n231 1114 1115 DFFHQX1
Xbit_count_reg[1] bit_count[1] clk_i n232 1116 1117 DFFHQX1
Xbusy_bit_reg busy_bit clk_i n289 1118 1119 DFFHQX1
Xbit_count_reg[0] bit_count[0] clk_i n290 1120 1121 DFFHQX1
XU278 n291 n306 n305 1122 1123 NOR2X1
XU279 n229 n197 n198 1124 1125 NOR2X1
XU280 n295 n294 1126 1127 INVX2
XU281 n306 n307 1128 1129 INVX2
XU282 n307 n78 1130 1131 INVX2
XU283 n300 n297 1132 1133 INVX2
XU284 n301 n297 1134 1135 INVX2
XU285 n296 n294 1136 1137 INVX2
XU286 n298 n297 1138 1139 INVX2
XU287 n299 n297 1140 1141 INVX2
XU288 n97 bcd_next[16] 1142 1143 INVX2
XU289 n112 bcd_next[12] 1144 1145 INVX2
XU290 n127 bcd_next[8] 1146 1147 INVX2
XU291 n218 bcd_next[4] 1148 1149 INVX2
XU292 n81 N26 1150 1151 INVX2
XU293 N103 n333 n334 1152 1153 XOR2X1
XU294 n197 bit_count[3] bit_count[2] bit_count[1] bit_count[0] 1154 1155
+ NAND4X1
XU295 n78 n197 n198 done_o rst_i 1156 1157 NOR4BX1
XU296 n304 n36 1158 1159 BUFX2
XU297 n36 n295 rst_i 1160 1161 NOR2X1
XU298 n305 n152 1162 1163 BUFX2
XU299 n152 start_i busy_bit rst_i 1164 1165 NOR3BX1
XU300 n198 ce_i 1166 1167 INVX2
XU301 done_o busy_bit 1168 1169 INVX2
XU302 n294 n292 1170 1171 INVX2
XU303 n292 n229 rst_i start_i done_o 1172 1173 NOR4BX1
XU304 n255 n87 n88 1174 1175 NAND2X1
XU305 n88 n78 n69 1176 1177 NAND2X1
XU306 n87 n291 bcd_reg[17] 1178 1179 NAND2X1
XU307 n259 n102 n103 1180 1181 NAND2X1
XU308 n103 n78 n58 1182 1183 NAND2X1
XU309 n102 n291 bcd_reg[13] 1184 1185 NAND2X1
XU310 n263 n117 n118 1186 1187 NAND2X1
XU311 n118 n306 n47 1188 1189 NAND2X1
XU312 n117 n291 bcd_reg[9] 1190 1191 NAND2X1
XU313 n267 n132 n133 1192 1193 NAND2X1

XU314 n133 n306 n35 1194 1195 NAND2X1
XU315 n132 n291 bcd_reg[5] 1196 1197 NAND2X1
XU316 n273 n149 n150 n151 1198 1199 NAND3X1
XU317 n150 dat_binary_i[15] n305 1200 1201 NAND2X1
XU318 n149 bin_next[15] n306 1202 1203 NAND2X1
XU319 n151 bcd_next[0] n291 1204 1205 NAND2X1
XU320 n274 n153 n154 n155 1206 1207 NAND3X1
XU321 n154 dat_binary_i[14] n305 1208 1209 NAND2X1
XU322 n153 bin_next[14] n78 1210 1211 NAND2X1
XU323 n155 bin_next[15] n291 1212 1213 NAND2X1
XU324 n275 n156 n157 n158 1214 1215 NAND3X1
XU325 n157 dat_binary_i[13] n305 1216 1217 NAND2X1
XU326 n156 bin_next[13] n306 1218 1219 NAND2X1
XU327 n158 bin_next[14] n291 1220 1221 NAND2X1
XU328 n276 n159 n160 n161 1222 1223 NAND3X1
XU329 n160 dat_binary_i[12] n305 1224 1225 NAND2X1
XU330 n159 bin_next[12] n78 1226 1227 NAND2X1
XU331 n161 bin_next[13] n291 1228 1229 NAND2X1
XU332 n277 n162 n163 n164 1230 1231 NAND3X1
XU333 n163 dat_binary_i[11] n305 1232 1233 NAND2X1
XU334 n162 bin_next[11] n78 1234 1235 NAND2X1
XU335 n164 bin_next[12] n291 1236 1237 NAND2X1
XU336 n278 n165 n166 n167 1238 1239 NAND3X1
XU337 n166 dat_binary_i[10] n305 1240 1241 NAND2X1
XU338 n165 bin_next[10] n78 1242 1243 NAND2X1
XU339 n167 bin_next[11] n291 1244 1245 NAND2X1
XU340 n279 n168 n169 n170 1246 1247 NAND3X1
XU341 n169 dat_binary_i[9] n305 1248 1249 NAND2X1
XU342 n168 bin_next[9] n78 1250 1251 NAND2X1
XU343 n170 bin_next[10] n291 1252 1253 NAND2X1
XU344 n280 n171 n172 n173 1254 1255 NAND3X1
XU345 n172 dat_binary_i[8] n305 1256 1257 NAND2X1
XU346 n171 bin_next[8] n78 1258 1259 NAND2X1
XU347 n173 bin_next[9] n291 1260 1261 NAND2X1
XU348 n281 n174 n175 n176 1262 1263 NAND3X1
XU349 n175 dat_binary_i[7] n305 1264 1265 NAND2X1
XU350 n174 bin_next[7] n78 1266 1267 NAND2X1
XU351 n176 bin_next[8] n291 1268 1269 NAND2X1
XU352 n282 n177 n178 n179 1270 1271 NAND3X1
XU353 n178 dat_binary_i[6] n305 1272 1273 NAND2X1
XU354 n177 bin_next[6] n306 1274 1275 NAND2X1
XU355 n179 bin_next[7] n291 1276 1277 NAND2X1
XU356 n283 n180 n181 n182 1278 1279 NAND3X1
XU357 n181 dat_binary_i[5] n305 1280 1281 NAND2X1
XU358 n180 bin_next[5] n306 1282 1283 NAND2X1
XU359 n182 bin_next[6] n291 1284 1285 NAND2X1
XU360 n284 n183 n184 n185 1286 1287 NAND3X1
XU361 n184 dat_binary_i[4] n305 1288 1289 NAND2X1
XU362 n183 bin_next[4] n306 1290 1291 NAND2X1
XU363 n185 bin_next[5] n291 1292 1293 NAND2X1
XU364 n285 n186 n187 n188 1294 1295 NAND3X1
XU365 n187 dat_binary_i[3] n305 1296 1297 NAND2X1
XU366 n186 bin_next[3] n306 1298 1299 NAND2X1
XU367 n188 bin_next[4] n291 1300 1301 NAND2X1

XU368 n286 n189 n190 n191 1302 1303 NAND3X1
XU369 n190 dat_binary_i[2] n305 1304 1305 NAND2X1
XU370 n189 bin_next[2] n306 1306 1307 NAND2X1
XU371 n191 bin_next[3] n291 1308 1309 NAND2X1
XU372 n287 n192 n193 n194 1310 1311 NAND3X1
XU373 n193 dat_binary_i[1] n305 1312 1313 NAND2X1
XU374 n192 bin_next[1] n78 1314 1315 NAND2X1
XU375 n194 bin_next[2] n291 1316 1317 NAND2X1
XU376 n289 n305 n199 1318 1319 NAND2BX1
XU377 n199 n304 busy_bit 1320 1321 NAND2X1
XU378 n237 n227 n228 1322 1323 NAND2X1
XU379 n228 bcd_next[4] n303 1324 1325 NAND2X1
XU380 n227 dat_bcd_o[4] n304 1326 1327 NAND2X1
XU381 n241 n43 n44 1328 1329 NAND2X1
XU382 n44 bcd_next[8] n303 1330 1331 NAND2X1
XU383 n43 dat_bcd_o[8] n304 1332 1333 NAND2X1
XU384 n245 n54 n55 1334 1335 NAND2X1
XU385 n55 bcd_next[12] n303 1336 1337 NAND2X1
XU386 n54 dat_bcd_o[12] n304 1338 1339 NAND2X1
XU387 n303 n302 1340 1341 INVX2
XU388 n249 n65 n66 1342 1343 NAND2X1
XU389 n66 bcd_next[16] n303 1344 1345 NAND2X1
XU390 n65 dat_bcd_o[16] n304 1346 1347 NAND2X1
XU391 n268 n136 n137 1348 1349 NAND2X1
XU392 n137 bcd_next[4] n78 1350 1351 NAND2X1
XU393 n136 n291 bcd_reg[4] 1352 1353 NAND2X1
XU394 n256 n91 n92 1354 1355 NAND2X1
XU395 n92 n78 bcd_next[16] 1356 1357 NAND2X1
XU396 n91 n291 bcd_reg[16] 1358 1359 NAND2X1
XU397 n260 n106 n107 1360 1361 NAND2X1
XU398 n107 n306 bcd_next[12] 1362 1363 NAND2X1
XU399 n106 n291 bcd_reg[12] 1364 1365 NAND2X1
XU400 n264 n121 n122 1366 1367 NAND2X1
XU401 n122 n306 bcd_next[8] 1368 1369 NAND2X1
XU402 n121 n291 bcd_reg[8] 1370 1371 NAND2X1
XU403 n233 n212 n213 1372 1373 NAND2X1
XU404 n213 bcd_next[0] n303 1374 1375 NAND2X1
XU405 n212 dat_bcd_o[0] n304 1376 1377 NAND2X1
XU406 n272 n147 n148 1378 1379 NAND2X1
XU407 n148 bcd_next[0] n78 1380 1381 NAND2X1
XU408 n147 bcd_reg[0] n291 1382 1383 NAND2X1
XU409 n288 n195 n196 1384 1385 NAND2X1
XU410 n196 dat_binary_i[0] n305 1386 1387 NAND2X1
XU411 n195 bin_next[1] n291 1388 1389 NAND2X1
XU412 n247 n59 n60 1390 1391 NAND2X1
XU413 n60 n295 n61 1392 1393 NAND2X1
XU414 n59 dat_bcd_o[14] n304 1394 1395 NAND2X1
XU415 n253 n76 n77 1396 1397 NAND2X1
XU416 n77 n306 n75 1398 1399 NAND2X1
XU417 n76 bcd_reg[19] n291 1400 1401 NAND2X1
XU418 n254 n83 n84 1402 1403 NAND2X1
XU419 n84 n306 n72 1404 1405 NAND2X1
XU420 n83 n291 bcd_reg[18] 1406 1407 NAND2X1
XU421 n261 n108 n109 1408 1409 NAND2X1

XU422 n109 n306 n53 1410 1411 NAND2X1
XU423 n108 bcd_reg[11] n291 1412 1413 NAND2X1
XU424 n262 n113 n114 1414 1415 NAND2X1
XU425 n114 n306 n50 1416 1417 NAND2X1
XU426 n113 n291 bcd_reg[10] 1418 1419 NAND2X1
XU427 n265 n123 n124 1420 1421 NAND2X1
XU428 n124 n306 n42 1422 1423 NAND2X1
XU429 n123 bcd_reg[7] n291 1424 1425 NAND2X1
XU430 n266 n128 n129 1426 1427 NAND2X1
XU431 n129 n306 n39 1428 1429 NAND2X1
XU432 n128 n291 bcd_reg[6] 1430 1431 NAND2X1
XU433 n269 n138 n139 1432 1433 NAND2X1
XU434 n139 n306 n140 1434 1435 NAND2X1
XU435 n138 bcd_reg[3] n291 1436 1437 NAND2X1
XU436 n270 n141 n142 1438 1439 NAND2X1
XU437 n142 n306 n143 1440 1441 NAND2X1
XU438 n141 bcd_reg[2] n291 1442 1443 NAND2X1
XU439 n271 n144 n145 1444 1445 NAND2X1
XU440 n145 n306 n146 1446 1447 NAND2X1
XU441 n144 bcd_reg[1] n291 1448 1449 NAND2X1
XU442 n257 n93 n94 1450 1451 NAND2X1
XU443 n94 n78 n64 1452 1453 NAND2X1
XU444 n93 bcd_reg[15] n291 1454 1455 NAND2X1
XU445 n258 n98 n99 1456 1457 NAND2X1
XU446 n99 n78 n61 1458 1459 NAND2X1
XU447 n98 n291 bcd_reg[14] 1460 1461 NAND2X1
XU448 n234 n214 n215 1462 1463 NAND2X1
XU449 n215 n299 n146 1464 1465 NAND2X1
XU450 n214 dat_bcd_o[1] n304 1466 1467 NAND2X1
XU451 n235 n219 n220 1468 1469 NAND2X1
XU452 n220 n300 n143 1470 1471 NAND2X1
XU453 n219 dat_bcd_o[2] n304 1472 1473 NAND2X1
XU454 n236 n223 n224 1474 1475 NAND2X1
XU455 n224 n301 n140 1476 1477 NAND2X1
XU456 n223 dat_bcd_o[3] n304 1478 1479 NAND2X1
XU457 n238 n32 n33 1480 1481 NAND2X1
XU458 n33 n296 n35 1482 1483 NAND2X1
XU459 n32 dat_bcd_o[5] n304 1484 1485 NAND2X1
XU460 n239 n37 n38 1486 1487 NAND2X1
XU461 n38 n296 n39 1488 1489 NAND2X1
XU462 n37 dat_bcd_o[6] n304 1490 1491 NAND2X1
XU463 n240 n40 n41 1492 1493 NAND2X1
XU464 n41 n298 n42 1494 1495 NAND2X1
XU465 n40 dat_bcd_o[7] n304 1496 1497 NAND2X1
XU466 n243 n48 n49 1498 1499 NAND2X1
XU467 n49 n300 n50 1500 1501 NAND2X1
XU468 n48 dat_bcd_o[10] n304 1502 1503 NAND2X1
XU469 n244 n51 n52 1504 1505 NAND2X1
XU470 n52 n301 n53 1506 1507 NAND2X1
XU471 n51 dat_bcd_o[11] n304 1508 1509 NAND2X1
XU472 n248 n62 n63 1510 1511 NAND2X1
XU473 n63 n296 n64 1512 1513 NAND2X1
XU474 n62 dat_bcd_o[15] n304 1514 1515 NAND2X1
XU475 n250 n67 n68 1516 1517 NAND2X1

XU476 n68 n296 n69 1518 1519 NAND2X1
XU477 n67 dat_bcd_o[17] n304 1520 1521 NAND2X1
XU478 n251 n70 n71 1522 1523 NAND2X1
XU479 n71 n298 n72 1524 1525 NAND2X1
XU480 n70 dat_bcd_o[18] n304 1526 1527 NAND2X1
XU481 n252 n73 n74 1528 1529 NAND2X1
XU482 n74 n299 n75 1530 1531 NAND2X1
XU483 n73 dat_bcd_o[19] n304 1532 1533 NAND2X1
XU484 n242 n45 n46 1534 1535 NAND2X1
XU485 n46 n296 n47 1536 1537 NAND2X1
XU486 n45 dat_bcd_o[9] n304 1538 1539 NAND2X1
XU487 n246 n56 n57 1540 1541 NAND2X1
XU488 n57 n303 n58 1542 1543 NAND2X1
XU489 n56 dat_bcd_o[13] n304 1544 1545 NAND2X1
XU490 n334 bit_count[1] bit_count[0] 1546 1547 NAND2X1
XU491 N24 N23 bcd_reg[17] 1548 1549 XOR2X1
XU492 N20 N19 bcd_reg[13] 1550 1551 XOR2X1
XU493 N16 N15 bcd_reg[9] 1552 1553 XOR2X1
XU494 N12 N11 bcd_reg[5] 1554 1555 XOR2X1
XU495 N8 N7 bcd_reg[1] 1556 1557 XOR2X1
XU496 n302 n34 1558 1559 INVX2
XU497 n34 n229 rst_i start_i done_o 1560 1561 NOR4BX1
XU498 n58 n104 n105 1562 1563 NAND2X1
XU499 n105 bcd_next[16] N19 1564 1565 NAND2X1
XU500 n104 bcd_reg[12] n97 1566 1567 NAND2X1
XU501 n47 n119 n120 1568 1569 NAND2X1
XU502 n120 bcd_next[12] N15 1570 1571 NAND2X1
XU503 n119 bcd_reg[8] n112 1572 1573 NAND2X1
XU504 n35 n134 n135 1574 1575 NAND2X1
XU505 n135 bcd_next[8] N11 1576 1577 NAND2X1
XU506 n134 bcd_reg[4] n127 1578 1579 NAND2X1
XU507 n64 n95 n96 1580 1581 NAND2X1
XU508 n95 bcd_reg[14] n97 1582 1583 NAND2X1
XU509 n96 N21 bcd_next[16] 1584 1585 NAND2X1
XU510 N21 n325 n324 1586 1587 NAND2X1
XU511 n61 n100 n101 1588 1589 NAND2X1
XU512 n101 N20 bcd_next[16] 1590 1591 NAND2X1
XU513 n100 bcd_reg[13] n97 1592 1593 NAND2X1
XU514 n53 n110 n111 1594 1595 NAND2X1
XU515 n110 bcd_reg[10] n112 1596 1597 NAND2X1
XU516 n111 N17 bcd_next[12] 1598 1599 NAND2X1
XU517 N17 n320 n319 1600 1601 NAND2X1
XU518 n50 n115 n116 1602 1603 NAND2X1
XU519 n116 N16 bcd_next[12] 1604 1605 NAND2X1
XU520 n115 bcd_reg[9] n112 1606 1607 NAND2X1
XU521 n42 n125 n126 1608 1609 NAND2X1
XU522 n125 bcd_reg[6] n127 1610 1611 NAND2X1
XU523 n126 N13 bcd_next[8] 1612 1613 NAND2X1
XU524 N13 n315 n314 1614 1615 NAND2X1
XU525 n39 n130 n131 1616 1617 NAND2X1
XU526 n131 N12 bcd_next[8] 1618 1619 NAND2X1
XU527 n130 bcd_reg[5] n127 1620 1621 NAND2X1
XU528 n140 n225 n226 1622 1623 NAND2X1
XU529 n225 bcd_reg[2] n218 1624 1625 NAND2X1

XU530 n226 N9 bcd_next[4] 1626 1627 NAND2X1
XU531 N9 n310 n309 1628 1629 NAND2X1
XU532 n69 n89 n90 1630 1631 NAND2X1
XU533 n90 N26 N23 1632 1633 NAND2X1
XU534 n89 bcd_reg[16] n81 1634 1635 NAND2X1
XU535 n75 n79 n80 1636 1637 NAND2X1
XU536 n79 bcd_reg[18] n81 1638 1639 NAND2X1
XU537 n80 N25 N26 1640 1641 NAND2X1
XU538 N25 n330 n329 1642 1643 NAND2X1
XU539 n72 n85 n86 1644 1645 NAND2X1
XU540 n86 N24 N26 1646 1647 NAND2X1
XU541 n85 bcd_reg[17] n81 1648 1649 NAND2X1
XU542 n297 n293 1650 1651 INVX2
XU543 n293 n229 rst_i start_i done_o 1652 1653 NOR4BX1
XU544 n310 N8 N7 bcd_reg[2] 1654 1655 NAND3X1
XU545 n330 N24 N23 bcd_reg[18] 1656 1657 NAND3X1
XU546 n325 N20 N19 bcd_reg[14] 1658 1659 NAND3X1
XU547 n320 N16 N15 bcd_reg[10] 1660 1661 NAND3X1
XU548 n315 N12 N11 bcd_reg[6] 1662 1663 NAND3X1
XU549 N7 bcd_reg[0] 1664 1665 INVX2
XU550 N23 bcd_reg[16] 1666 1667 INVX2
XU551 N19 bcd_reg[12] 1668 1669 INVX2
XU552 N15 bcd_reg[8] 1670 1671 INVX2
XU553 N11 bcd_reg[4] 1672 1673 INVX2
XU554 n329 bcd_reg[18] n328 1674 1675 NAND2BX1
XU555 n328 N24 N23 1676 1677 NAND2X1
XU556 n324 bcd_reg[14] n323 1678 1679 NAND2BX1
XU557 n323 N20 N19 1680 1681 NAND2X1
XU558 n319 bcd_reg[10] n318 1682 1683 NAND2BX1
XU559 n318 N16 N15 1684 1685 NAND2X1
XU560 n314 bcd_reg[6] n313 1686 1687 NAND2BX1
XU561 n313 N12 N11 1688 1689 NAND2X1
XU562 n143 n221 n222 1690 1691 NAND2X1
XU563 n222 N8 bcd_next[4] 1692 1693 NAND2X1
XU564 n221 bcd_reg[1] n218 1694 1695 NAND2X1
XU565 n146 n216 n217 1696 1697 NAND2X1
XU566 n217 N7 bcd_next[4] 1698 1699 NAND2X1
XU567 n216 bcd_reg[0] n218 1700 1701 NAND2X1
XU568 n309 bcd_reg[2] n308 1702 1703 NAND2BX1
XU569 n308 N8 N7 1704 1705 NAND2X1
XU570 N26 bcd_reg[19] n332 1706 1707 NAND2BX1
XU571 n332 bcd_reg[18] n331 1708 1709 NAND2X1
XU572 n331 bcd_reg[17] bcd_reg[16] 1710 1711 OR2X1
XU573 bcd_next[4] bcd_reg[3] n312 1712 1713 NAND2BX1
XU574 n312 bcd_reg[2] n311 1714 1715 NAND2X1
XU575 n311 bcd_reg[1] bcd_reg[0] 1716 1717 OR2X1
XU576 bcd_next[16] bcd_reg[15] n327 1718 1719 NAND2BX1
XU577 n327 bcd_reg[14] n326 1720 1721 NAND2X1
XU578 n326 bcd_reg[13] bcd_reg[12] 1722 1723 OR2X1
XU579 bcd_next[12] bcd_reg[11] n322 1724 1725 NAND2BX1
XU580 n322 bcd_reg[10] n321 1726 1727 NAND2X1
XU581 n321 bcd_reg[9] bcd_reg[8] 1728 1729 OR2X1
XU582 bcd_next[8] bcd_reg[7] n317 1730 1731 NAND2BX1
XU583 n317 bcd_reg[6] n316 1732 1733 NAND2X1

```
XU584 n316 bcd_reg[5] bcd_reg[4] 1734 1735 OR2X1
XU585 N102 bit_count[1] bit_count[0] 1736 1737 XOR2X1
XU586 N104 bit_count[3] n335 1738 1739 XOR2X1
XU587 n335 n334 n333 1740 1741 NOR2X1
XU588 n333 bit_count[2] 1742 1743 INVX2
XU589 N101 bit_count[0] 1744 1745 INVX2
XU590 n230 n203 busy_bit 1746 1747 AND2X1
XU591 n203 n204 n197 n205 1748 1749 NAND3X1
XU592 n205 bit_count[3] n198 1750 1751 NAND2X1
XU593 n204 N104 ce_i 1752 1753 NAND2X1
XU594 n231 n206 busy_bit 1754 1755 AND2X1
XU595 n206 n207 n197 n208 1756 1757 NAND3X1
XU596 n208 bit_count[2] n198 1758 1759 NAND2X1
XU597 n207 N103 ce_i 1760 1761 NAND2X1
XU598 n232 n209 busy_bit 1762 1763 AND2X1
XU599 n209 n210 n197 n211 1764 1765 NAND3X1
XU600 n211 bit_count[1] n198 1766 1767 NAND2X1
XU601 n210 N102 ce_i 1768 1769 NAND2X1
XU602 n290 n200 busy_bit 1770 1771 AND2X1
XU603 n200 n201 n197 n202 1772 1773 NAND3X1
XU604 n202 bit_count[0] n198 1774 1775 NAND2X1
XU605 n201 N101 ce_i 1776 1777 NAND2X1
.ENDS
```

Τελικό αρχείο SPICE προς εκτέλεση επαυξημένης μεθόδου Λογικού Φόρτου

```
v1 dat_binary_i[0] 0 pulse(0 5 15n)
v2 dat_binary_i[1] 0 pulse(0 5 15n)
v3 dat_binary_i[2] 0 pulse(0 5 15n)
v4 dat_binary_i[3] 0 pulse(0 5 15n)
v5 dat_binary_i[4] 0 pulse(0 5 15n)
v6 dat_binary_i[5] 0 pulse(0 5 15n)
v7 dat_binary_i[6] 0 pulse(0 5 15n)
v8 dat_binary_i[7] 0 pulse(0 5 15n)
v9 dat_binary_i[8] 0 pulse(0 5 15n)
v10 dat_binary_i[9] 0 pulse(0 5 15n)
v11 dat_binary_i[10] 0 pulse(0 5 15n)
v12 dat_binary_i[11] 0 pulse(0 5 15n)
v13 dat_binary_i[12] 0 pulse(0 5 15n)
v14 dat_binary_i[13] 0 pulse(0 5 15n)
v15 dat_binary_i[14] 0 pulse(0 5 15n)
v16 dat_binary_i[15] 0 pulse(0 5 15n)
v17 ce_i 0 pulse(0 5 15n)
v18 rst_i 0 pulse(0 5 15n)
v19 start_i 0 pulse(0 5 15n)
v20 clk_i 0 pulse(0 5 15n)


$ Ορισμοί βιβλιοθήκης υλοποίησης
```

```
$ Spice netlist generated by v2lvs
$ v2007.3_18.11   Wed Aug 22 13:24:38 PDT 2007

.SUBCKT binary_to_bcd clk_i ce_i rst_i start_i dat_binary_i[15]
+ dat_binary_i[14] dat_binary_i[13] dat_binary_i[12] dat_binary_i[11]
+ dat_binary_i[10] dat_binary_i[9] dat_binary_i[8] dat_binary_i[7]
+ dat_binary_i[6] dat_binary_i[5] dat_binary_i[4] dat_binary_i[3]
+ dat_binary_i[2] dat_binary_i[1] dat_binary_i[0] dat_bcd_o[19] dat_bcd_o[18]
+ dat_bcd_o[17] dat_bcd_o[16] dat_bcd_o[15] dat_bcd_o[14] dat_bcd_o[13]
+ dat_bcd_o[12] dat_bcd_o[11] dat_bcd_o[10] dat_bcd_o[9] dat_bcd_o[8]
+ dat_bcd_o[7] dat_bcd_o[6] dat_bcd_o[5] dat_bcd_o[4] dat_bcd_o[3] dat_bcd_o[2]
+ dat_bcd_o[1] dat_bcd_o[0] done_o
Xdat_bcd_o_reg[19] dat_bcd_o[19] clk_i n252 1000 1001 DFFHQX1
Xdat_bcd_o_reg[18] dat_bcd_o[18] clk_i n251 1002 1003 DFFHQX1
Xdat_bcd_o_reg[17] dat_bcd_o[17] clk_i n250 1004 1005 DFFHQX1
Xdat_bcd_o_reg[16] dat_bcd_o[16] clk_i n249 1006 1007 DFFHQX1
Xdat_bcd_o_reg[15] dat_bcd_o[15] clk_i n248 1008 1009 DFFHQX1
Xdat_bcd_o_reg[14] dat_bcd_o[14] clk_i n247 1010 1011 DFFHQX1
Xdat_bcd_o_reg[13] dat_bcd_o[13] clk_i n246 1012 1013 DFFHQX1
Xdat_bcd_o_reg[12] dat_bcd_o[12] clk_i n245 1014 1015 DFFHQX1
Xdat_bcd_o_reg[11] dat_bcd_o[11] clk_i n244 1016 1017 DFFHQX1
Xdat_bcd_o_reg[10] dat_bcd_o[10] clk_i n243 1018 1019 DFFHQX1
Xdat_bcd_o_reg[9] dat_bcd_o[9] clk_i n242 1020 1021 DFFHQX1
Xdat_bcd_o_reg[8] dat_bcd_o[8] clk_i n241 1022 1023 DFFHQX1
Xdat_bcd_o_reg[7] dat_bcd_o[7] clk_i n240 1024 1025 DFFHQX1
Xdat_bcd_o_reg[6] dat_bcd_o[6] clk_i n239 1026 1027 DFFHQX1
Xdat_bcd_o_reg[5] dat_bcd_o[5] clk_i n238 1028 1029 DFFHQX1
Xdat_bcd_o_reg[4] dat_bcd_o[4] clk_i n237 1030 1031 DFFHQX1
Xdat_bcd_o_reg[3] dat_bcd_o[3] clk_i n236 1032 1033 DFFHQX1
Xdat_bcd_o_reg[2] dat_bcd_o[2] clk_i n235 1034 1035 DFFHQX1
Xdat_bcd_o_reg[1] dat_bcd_o[1] clk_i n234 1036 1037 DFFHQX1
Xdat_bcd_o_reg[0] dat_bcd_o[0] clk_i n233 1038 1039 DFFHQX1
Xbcd_reg_reg[3] bcd_reg[3] clk_i n269 1040 1041 DFFHQX1
Xbcd_reg_reg[7] bcd_reg[7] clk_i n265 1042 1043 DFFHQX1
Xbcd_reg_reg[11] bcd_reg[11] clk_i n261 1044 1045 DFFHQX1
Xbcd_reg_reg[15] bcd_reg[15] clk_i n257 1046 1047 DFFHQX1
Xbcd_reg_reg[19] bcd_reg[19] clk_i n253 1048 1049 DFFHQX1
Xbin_reg_reg[0] bin_next[1] clk_i n288 1050 1051 DFFHQX1
Xbin_reg_reg[1] bin_next[2] clk_i n287 1052 1053 DFFHQX1
Xbin_reg_reg[2] bin_next[3] clk_i n286 1054 1055 DFFHQX1
Xbin_reg_reg[3] bin_next[4] clk_i n285 1056 1057 DFFHQX1
Xbin_reg_reg[4] bin_next[5] clk_i n284 1058 1059 DFFHQX1
Xbin_reg_reg[5] bin_next[6] clk_i n283 1060 1061 DFFHQX1
Xbin_reg_reg[6] bin_next[7] clk_i n282 1062 1063 DFFHQX1
Xbin_reg_reg[7] bin_next[8] clk_i n281 1064 1065 DFFHQX1
Xbin_reg_reg[8] bin_next[9] clk_i n280 1066 1067 DFFHQX1
Xbin_reg_reg[9] bin_next[10] clk_i n279 1068 1069 DFFHQX1
Xbin_reg_reg[10] bin_next[11] clk_i n278 1070 1071 DFFHQX1
Xbin_reg_reg[11] bin_next[12] clk_i n277 1072 1073 DFFHQX1
Xbin_reg_reg[12] bin_next[13] clk_i n276 1074 1075 DFFHQX1
Xbin_reg_reg[13] bin_next[14] clk_i n275 1076 1077 DFFHQX1
Xbin_reg_reg[14] bin_next[15] clk_i n274 1078 1079 DFFHQX1
```

Xbin_reg_reg[15] bcd_next[0] clk_i n273 1080 1081 DFFHQX1
Xbcd_reg_reg[4] bcd_reg[4] clk_i n268 1082 1083 DFFHQX1
Xbcd_reg_reg[8] bcd_reg[8] clk_i n264 1084 1085 DFFHQX1
Xbcd_reg_reg[12] bcd_reg[12] clk_i n260 1086 1087 DFFHQX1
Xbcd_reg_reg[16] bcd_reg[16] clk_i n256 1088 1089 DFFHQX1
Xbcd_reg_reg[0] bcd_reg[0] clk_i n272 1090 1091 DFFHQX1
Xbcd_reg_reg[5] bcd_reg[5] clk_i n267 1092 1093 DFFHQX1
Xbcd_reg_reg[9] bcd_reg[9] clk_i n263 1094 1095 DFFHQX1
Xbcd_reg_reg[13] bcd_reg[13] clk_i n259 1096 1097 DFFHQX1
Xbcd_reg_reg[17] bcd_reg[17] clk_i n255 1098 1099 DFFHQX1
Xbcd_reg_reg[1] bcd_reg[1] clk_i n271 1100 1101 DFFHQX1
Xbcd_reg_reg[6] bcd_reg[6] clk_i n266 1102 1103 DFFHQX1
Xbcd_reg_reg[10] bcd_reg[10] clk_i n262 1104 1105 DFFHQX1
Xbcd_reg_reg[14] bcd_reg[14] clk_i n258 1106 1107 DFFHQX1
Xbcd_reg_reg[18] bcd_reg[18] clk_i n254 1108 1109 DFFHQX1
Xbcd_reg_reg[2] bcd_reg[2] clk_i n270 1110 1111 DFFHQX1
Xbit_count_reg[3] bit_count[3] clk_i n230 1112 1113 DFFHQX1
Xbit_count_reg[2] bit_count[2] clk_i n231 1114 1115 DFFHQX1
Xbit_count_reg[1] bit_count[1] clk_i n232 1116 1117 DFFHQX1
Xbusy_bit_reg busy_bit clk_i n289 1118 1119 DFFHQX1
Xbit_count_reg[0] bit_count[0] clk_i n290 1120 1121 DFFHQX1
XU278 n291 n306 n305 1122 1123 NOR2X1
XU279 n229 n197 n198 1124 1125 NOR2X1
XU280 n295 n294 1126 1127 INVX2
XU281 n306 n307 1128 1129 INVX2
XU282 n307 n78 1130 1131 INVX2
XU283 n300 n297 1132 1133 INVX2
XU284 n301 n297 1134 1135 INVX2
XU285 n296 n294 1136 1137 INVX2
XU286 n298 n297 1138 1139 INVX2
XU287 n299 n297 1140 1141 INVX2
XU288 n97 bcd_next[16] 1142 1143 INVX2
XU289 n112 bcd_next[12] 1144 1145 INVX2
XU290 n127 bcd_next[8] 1146 1147 INVX2
XU291 n218 bcd_next[4] 1148 1149 INVX2
XU292 n81 N26 1150 1151 INVX2
XU293 N103 n333 n334 1152 1153 XOR2X1
XU294 n197 bit_count[3] bit_count[2] bit_count[1] bit_count[0] 1154 1155
+ NAND4X1
XU295 n78 n197 n198 done_o rst_i 1156 1157 NOR4BX1
XU296 n304 n36 1158 1159 BUFX2
XU297 n36 n295 rst_i 1160 1161 NOR2X1
XU298 n305 n152 1162 1163 BUFX2
XU299 n152 start_i busy_bit rst_i 1164 1165 NOR3BX1
XU300 n198 ce_i 1166 1167 INVX2
XU301 done_o busy_bit 1168 1169 INVX2
XU302 n294 n292 1170 1171 INVX2
XU303 n292 n229 rst_i start_i done_o 1172 1173 NOR4BX1
XU304 n255 n87 n88 1174 1175 NAND2X1
XU305 n88 n78 n69 1176 1177 NAND2X1
XU306 n87 n291 bcd_reg[17] 1178 1179 NAND2X1
XU307 n259 n102 n103 1180 1181 NAND2X1
XU308 n103 n78 n58 1182 1183 NAND2X1
XU309 n102 n291 bcd_reg[13] 1184 1185 NAND2X1

XU310 n263 n117 n118 1186 1187 NAND2X1
XU311 n118 n306 n47 1188 1189 NAND2X1
XU312 n117 n291 bcd_reg[9] 1190 1191 NAND2X1
XU313 n267 n132 n133 1192 1193 NAND2X1
XU314 n133 n306 n35 1194 1195 NAND2X1
XU315 n132 n291 bcd_reg[5] 1196 1197 NAND2X1
XU316 n273 n149 n150 n151 1198 1199 NAND3X1
XU317 n150 dat_binary_i[15] n305 1200 1201 NAND2X1
XU318 n149 bin_next[15] n306 1202 1203 NAND2X1
XU319 n151 bcd_next[0] n291 1204 1205 NAND2X1
XU320 n274 n153 n154 n155 1206 1207 NAND3X1
XU321 n154 dat_binary_i[14] n305 1208 1209 NAND2X1
XU322 n153 bin_next[14] n78 1210 1211 NAND2X1
XU323 n155 bin_next[15] n291 1212 1213 NAND2X1
XU324 n275 n156 n157 n158 1214 1215 NAND3X1
XU325 n157 dat_binary_i[13] n305 1216 1217 NAND2X1
XU326 n156 bin_next[13] n306 1218 1219 NAND2X1
XU327 n158 bin_next[14] n291 1220 1221 NAND2X1
XU328 n276 n159 n160 n161 1222 1223 NAND3X1
XU329 n160 dat_binary_i[12] n305 1224 1225 NAND2X1
XU330 n159 bin_next[12] n78 1226 1227 NAND2X1
XU331 n161 bin_next[13] n291 1228 1229 NAND2X1
XU332 n277 n162 n163 n164 1230 1231 NAND3X1
XU333 n163 dat_binary_i[11] n305 1232 1233 NAND2X1
XU334 n162 bin_next[11] n78 1234 1235 NAND2X1
XU335 n164 bin_next[12] n291 1236 1237 NAND2X1
XU336 n278 n165 n166 n167 1238 1239 NAND3X1
XU337 n166 dat_binary_i[10] n305 1240 1241 NAND2X1
XU338 n165 bin_next[10] n78 1242 1243 NAND2X1
XU339 n167 bin_next[11] n291 1244 1245 NAND2X1
XU340 n279 n168 n169 n170 1246 1247 NAND3X1
XU341 n169 dat_binary_i[9] n305 1248 1249 NAND2X1
XU342 n168 bin_next[9] n78 1250 1251 NAND2X1
XU343 n170 bin_next[10] n291 1252 1253 NAND2X1
XU344 n280 n171 n172 n173 1254 1255 NAND3X1
XU345 n172 dat_binary_i[8] n305 1256 1257 NAND2X1
XU346 n171 bin_next[8] n78 1258 1259 NAND2X1
XU347 n173 bin_next[9] n291 1260 1261 NAND2X1
XU348 n281 n174 n175 n176 1262 1263 NAND3X1
XU349 n175 dat_binary_i[7] n305 1264 1265 NAND2X1
XU350 n174 bin_next[7] n78 1266 1267 NAND2X1
XU351 n176 bin_next[8] n291 1268 1269 NAND2X1
XU352 n282 n177 n178 n179 1270 1271 NAND3X1
XU353 n178 dat_binary_i[6] n305 1272 1273 NAND2X1
XU354 n177 bin_next[6] n306 1274 1275 NAND2X1
XU355 n179 bin_next[7] n291 1276 1277 NAND2X1
XU356 n283 n180 n181 n182 1278 1279 NAND3X1
XU357 n181 dat_binary_i[5] n305 1280 1281 NAND2X1
XU358 n180 bin_next[5] n306 1282 1283 NAND2X1
XU359 n182 bin_next[6] n291 1284 1285 NAND2X1
XU360 n284 n183 n184 n185 1286 1287 NAND3X1
XU361 n184 dat_binary_i[4] n305 1288 1289 NAND2X1
XU362 n183 bin_next[4] n306 1290 1291 NAND2X1
XU363 n185 bin_next[5] n291 1292 1293 NAND2X1

XU364 n285 n186 n187 n188 1294 1295 NAND3X1
XU365 n187 dat_binary_i[3] n305 1296 1297 NAND2X1
XU366 n186 bin_next[3] n306 1298 1299 NAND2X1
XU367 n188 bin_next[4] n291 1300 1301 NAND2X1
XU368 n286 n189 n190 n191 1302 1303 NAND3X1
XU369 n190 dat_binary_i[2] n305 1304 1305 NAND2X1
XU370 n189 bin_next[2] n306 1306 1307 NAND2X1
XU371 n191 bin_next[3] n291 1308 1309 NAND2X1
XU372 n287 n192 n193 n194 1310 1311 NAND3X1
XU373 n193 dat_binary_i[1] n305 1312 1313 NAND2X1
XU374 n192 bin_next[1] n78 1314 1315 NAND2X1
XU375 n194 bin_next[2] n291 1316 1317 NAND2X1
XU376 n289 n305 n199 1318 1319 NAND2BX1
XU377 n199 n304 busy_bit 1320 1321 NAND2X1
XU378 n237 n227 n228 1322 1323 NAND2X1
XU379 n228 bcd_next[4] n303 1324 1325 NAND2X1
XU380 n227 dat_bcd_o[4] n304 1326 1327 NAND2X1
XU381 n241 n43 n44 1328 1329 NAND2X1
XU382 n44 bcd_next[8] n303 1330 1331 NAND2X1
XU383 n43 dat_bcd_o[8] n304 1332 1333 NAND2X1
XU384 n245 n54 n55 1334 1335 NAND2X1
XU385 n55 bcd_next[12] n303 1336 1337 NAND2X1
XU386 n54 dat_bcd_o[12] n304 1338 1339 NAND2X1
XU387 n303 n302 1340 1341 INVX2
XU388 n249 n65 n66 1342 1343 NAND2X1
XU389 n66 bcd_next[16] n303 1344 1345 NAND2X1
XU390 n65 dat_bcd_o[16] n304 1346 1347 NAND2X1
XU391 n268 n136 n137 1348 1349 NAND2X1
XU392 n137 bcd_next[4] n78 1350 1351 NAND2X1
XU393 n136 n291 bcd_reg[4] 1352 1353 NAND2X1
XU394 n256 n91 n92 1354 1355 NAND2X1
XU395 n92 n78 bcd_next[16] 1356 1357 NAND2X1
XU396 n91 n291 bcd_reg[16] 1358 1359 NAND2X1
XU397 n260 n106 n107 1360 1361 NAND2X1
XU398 n107 n306 bcd_next[12] 1362 1363 NAND2X1
XU399 n106 n291 bcd_reg[12] 1364 1365 NAND2X1
XU400 n264 n121 n122 1366 1367 NAND2X1
XU401 n122 n306 bcd_next[8] 1368 1369 NAND2X1
XU402 n121 n291 bcd_reg[8] 1370 1371 NAND2X1
XU403 n233 n212 n213 1372 1373 NAND2X1
XU404 n213 bcd_next[0] n303 1374 1375 NAND2X1
XU405 n212 dat_bcd_o[0] n304 1376 1377 NAND2X1
XU406 n272 n147 n148 1378 1379 NAND2X1
XU407 n148 bcd_next[0] n78 1380 1381 NAND2X1
XU408 n147 bcd_reg[0] n291 1382 1383 NAND2X1
XU409 n288 n195 n196 1384 1385 NAND2X1
XU410 n196 dat_binary_i[0] n305 1386 1387 NAND2X1
XU411 n195 bin_next[1] n291 1388 1389 NAND2X1
XU412 n247 n59 n60 1390 1391 NAND2X1
XU413 n60 n295 n61 1392 1393 NAND2X1
XU414 n59 dat_bcd_o[14] n304 1394 1395 NAND2X1
XU415 n253 n76 n77 1396 1397 NAND2X1
XU416 n77 n306 n75 1398 1399 NAND2X1
XU417 n76 bcd_reg[19] n291 1400 1401 NAND2X1

XU418 n254 n83 n84 1402 1403 NAND2X1
XU419 n84 n306 n72 1404 1405 NAND2X1
XU420 n83 n291 bcd_reg[18] 1406 1407 NAND2X1
XU421 n261 n108 n109 1408 1409 NAND2X1
XU422 n109 n306 n53 1410 1411 NAND2X1
XU423 n108 bcd_reg[11] n291 1412 1413 NAND2X1
XU424 n262 n113 n114 1414 1415 NAND2X1
XU425 n114 n306 n50 1416 1417 NAND2X1
XU426 n113 n291 bcd_reg[10] 1418 1419 NAND2X1
XU427 n265 n123 n124 1420 1421 NAND2X1
XU428 n124 n306 n42 1422 1423 NAND2X1
XU429 n123 bcd_reg[7] n291 1424 1425 NAND2X1
XU430 n266 n128 n129 1426 1427 NAND2X1
XU431 n129 n306 n39 1428 1429 NAND2X1
XU432 n128 n291 bcd_reg[6] 1430 1431 NAND2X1
XU433 n269 n138 n139 1432 1433 NAND2X1
XU434 n139 n306 n140 1434 1435 NAND2X1
XU435 n138 bcd_reg[3] n291 1436 1437 NAND2X1
XU436 n270 n141 n142 1438 1439 NAND2X1
XU437 n142 n306 n143 1440 1441 NAND2X1
XU438 n141 bcd_reg[2] n291 1442 1443 NAND2X1
XU439 n271 n144 n145 1444 1445 NAND2X1
XU440 n145 n306 n146 1446 1447 NAND2X1
XU441 n144 bcd_reg[1] n291 1448 1449 NAND2X1
XU442 n257 n93 n94 1450 1451 NAND2X1
XU443 n94 n78 n64 1452 1453 NAND2X1
XU444 n93 bcd_reg[15] n291 1454 1455 NAND2X1
XU445 n258 n98 n99 1456 1457 NAND2X1
XU446 n99 n78 n61 1458 1459 NAND2X1
XU447 n98 n291 bcd_reg[14] 1460 1461 NAND2X1
XU448 n234 n214 n215 1462 1463 NAND2X1
XU449 n215 n299 n146 1464 1465 NAND2X1
XU450 n214 dat_bcd_o[1] n304 1466 1467 NAND2X1
XU451 n235 n219 n220 1468 1469 NAND2X1
XU452 n220 n300 n143 1470 1471 NAND2X1
XU453 n219 dat_bcd_o[2] n304 1472 1473 NAND2X1
XU454 n236 n223 n224 1474 1475 NAND2X1
XU455 n224 n301 n140 1476 1477 NAND2X1
XU456 n223 dat_bcd_o[3] n304 1478 1479 NAND2X1
XU457 n238 n32 n33 1480 1481 NAND2X1
XU458 n33 n296 n35 1482 1483 NAND2X1
XU459 n32 dat_bcd_o[5] n304 1484 1485 NAND2X1
XU460 n239 n37 n38 1486 1487 NAND2X1
XU461 n38 n296 n39 1488 1489 NAND2X1
XU462 n37 dat_bcd_o[6] n304 1490 1491 NAND2X1
XU463 n240 n40 n41 1492 1493 NAND2X1
XU464 n41 n298 n42 1494 1495 NAND2X1
XU465 n40 dat_bcd_o[7] n304 1496 1497 NAND2X1
XU466 n243 n48 n49 1498 1499 NAND2X1
XU467 n49 n300 n50 1500 1501 NAND2X1
XU468 n48 dat_bcd_o[10] n304 1502 1503 NAND2X1
XU469 n244 n51 n52 1504 1505 NAND2X1
XU470 n52 n301 n53 1506 1507 NAND2X1
XU471 n51 dat_bcd_o[11] n304 1508 1509 NAND2X1

XU472 n248 n62 n63 1510 1511 NAND2X1
XU473 n63 n296 n64 1512 1513 NAND2X1
XU474 n62 dat_bcd_o[15] n304 1514 1515 NAND2X1
XU475 n250 n67 n68 1516 1517 NAND2X1
XU476 n68 n296 n69 1518 1519 NAND2X1
XU477 n67 dat_bcd_o[17] n304 1520 1521 NAND2X1
XU478 n251 n70 n71 1522 1523 NAND2X1
XU479 n71 n298 n72 1524 1525 NAND2X1
XU480 n70 dat_bcd_o[18] n304 1526 1527 NAND2X1
XU481 n252 n73 n74 1528 1529 NAND2X1
XU482 n74 n299 n75 1530 1531 NAND2X1
XU483 n73 dat_bcd_o[19] n304 1532 1533 NAND2X1
XU484 n242 n45 n46 1534 1535 NAND2X1
XU485 n46 n296 n47 1536 1537 NAND2X1
XU486 n45 dat_bcd_o[9] n304 1538 1539 NAND2X1
XU487 n246 n56 n57 1540 1541 NAND2X1
XU488 n57 n303 n58 1542 1543 NAND2X1
XU489 n56 dat_bcd_o[13] n304 1544 1545 NAND2X1
XU490 n334 bit_count[1] bit_count[0] 1546 1547 NAND2X1
XU491 N24 N23 bcd_reg[17] 1548 1549 XOR2X1
XU492 N20 N19 bcd_reg[13] 1550 1551 XOR2X1
XU493 N16 N15 bcd_reg[9] 1552 1553 XOR2X1
XU494 N12 N11 bcd_reg[5] 1554 1555 XOR2X1
XU495 N8 N7 bcd_reg[1] 1556 1557 XOR2X1
XU496 n302 n34 1558 1559 INVX2
XU497 n34 n229 rst_i start_i done_o 1560 1561 NOR4BX1
XU498 n58 n104 n105 1562 1563 NAND2X1
XU499 n105 bcd_next[16] N19 1564 1565 NAND2X1
XU500 n104 bcd_reg[12] n97 1566 1567 NAND2X1
XU501 n47 n119 n120 1568 1569 NAND2X1
XU502 n120 bcd_next[12] N15 1570 1571 NAND2X1
XU503 n119 bcd_reg[8] n112 1572 1573 NAND2X1
XU504 n35 n134 n135 1574 1575 NAND2X1
XU505 n135 bcd_next[8] N11 1576 1577 NAND2X1
XU506 n134 bcd_reg[4] n127 1578 1579 NAND2X1
XU507 n64 n95 n96 1580 1581 NAND2X1
XU508 n95 bcd_reg[14] n97 1582 1583 NAND2X1
XU509 n96 N21 bcd_next[16] 1584 1585 NAND2X1
XU510 N21 n325 n324 1586 1587 NAND2X1
XU511 n61 n100 n101 1588 1589 NAND2X1
XU512 n101 N20 bcd_next[16] 1590 1591 NAND2X1
XU513 n100 bcd_reg[13] n97 1592 1593 NAND2X1
XU514 n53 n110 n111 1594 1595 NAND2X1
XU515 n110 bcd_reg[10] n112 1596 1597 NAND2X1
XU516 n111 N17 bcd_next[12] 1598 1599 NAND2X1
XU517 N17 n320 n319 1600 1601 NAND2X1
XU518 n50 n115 n116 1602 1603 NAND2X1
XU519 n116 N16 bcd_next[12] 1604 1605 NAND2X1
XU520 n115 bcd_reg[9] n112 1606 1607 NAND2X1
XU521 n42 n125 n126 1608 1609 NAND2X1
XU522 n125 bcd_reg[6] n127 1610 1611 NAND2X1
XU523 n126 N13 bcd_next[8] 1612 1613 NAND2X1
XU524 N13 n315 n314 1614 1615 NAND2X1
XU525 n39 n130 n131 1616 1617 NAND2X1

XU526 n131 N12 bcd_next[8] 1618 1619 NAND2X1
XU527 n130 bcd_reg[5] n127 1620 1621 NAND2X1
XU528 n140 n225 n226 1622 1623 NAND2X1
XU529 n225 bcd_reg[2] n218 1624 1625 NAND2X1
XU530 n226 N9 bcd_next[4] 1626 1627 NAND2X1
XU531 N9 n310 n309 1628 1629 NAND2X1
XU532 n69 n89 n90 1630 1631 NAND2X1
XU533 n90 N26 N23 1632 1633 NAND2X1
XU534 n89 bcd_reg[16] n81 1634 1635 NAND2X1
XU535 n75 n79 n80 1636 1637 NAND2X1
XU536 n79 bcd_reg[18] n81 1638 1639 NAND2X1
XU537 n80 N25 N26 1640 1641 NAND2X1
XU538 N25 n330 n329 1642 1643 NAND2X1
XU539 n72 n85 n86 1644 1645 NAND2X1
XU540 n86 N24 N26 1646 1647 NAND2X1
XU541 n85 bcd_reg[17] n81 1648 1649 NAND2X1
XU542 n297 n293 1650 1651 INVX2
XU543 n293 n229 rst_i start_i done_o 1652 1653 NOR4BX1
XU544 n310 N8 N7 bcd_reg[2] 1654 1655 NAND3X1
XU545 n330 N24 N23 bcd_reg[18] 1656 1657 NAND3X1
XU546 n325 N20 N19 bcd_reg[14] 1658 1659 NAND3X1
XU547 n320 N16 N15 bcd_reg[10] 1660 1661 NAND3X1
XU548 n315 N12 N11 bcd_reg[6] 1662 1663 NAND3X1
XU549 N7 bcd_reg[0] 1664 1665 INVX2
XU550 N23 bcd_reg[16] 1666 1667 INVX2
XU551 N19 bcd_reg[12] 1668 1669 INVX2
XU552 N15 bcd_reg[8] 1670 1671 INVX2
XU553 N11 bcd_reg[4] 1672 1673 INVX2
XU554 n329 bcd_reg[18] n328 1674 1675 NAND2BX1
XU555 n328 N24 N23 1676 1677 NAND2X1
XU556 n324 bcd_reg[14] n323 1678 1679 NAND2BX1
XU557 n323 N20 N19 1680 1681 NAND2X1
XU558 n319 bcd_reg[10] n318 1682 1683 NAND2BX1
XU559 n318 N16 N15 1684 1685 NAND2X1
XU560 n314 bcd_reg[6] n313 1686 1687 NAND2BX1
XU561 n313 N12 N11 1688 1689 NAND2X1
XU562 n143 n221 n222 1690 1691 NAND2X1
XU563 n222 N8 bcd_next[4] 1692 1693 NAND2X1
XU564 n221 bcd_reg[1] n218 1694 1695 NAND2X1
XU565 n146 n216 n217 1696 1697 NAND2X1
XU566 n217 N7 bcd_next[4] 1698 1699 NAND2X1
XU567 n216 bcd_reg[0] n218 1700 1701 NAND2X1
XU568 n309 bcd_reg[2] n308 1702 1703 NAND2BX1
XU569 n308 N8 N7 1704 1705 NAND2X1
XU570 N26 bcd_reg[19] n332 1706 1707 NAND2BX1
XU571 n332 bcd_reg[18] n331 1708 1709 NAND2X1
XU572 n331 bcd_reg[17] bcd_reg[16] 1710 1711 OR2X1
XU573 bcd_next[4] bcd_reg[3] n312 1712 1713 NAND2BX1
XU574 n312 bcd_reg[2] n311 1714 1715 NAND2X1
XU575 n311 bcd_reg[1] bcd_reg[0] 1716 1717 OR2X1
XU576 bcd_next[16] bcd_reg[15] n327 1718 1719 NAND2BX1
XU577 n327 bcd_reg[14] n326 1720 1721 NAND2X1
XU578 n326 bcd_reg[13] bcd_reg[12] 1722 1723 OR2X1
XU579 bcd_next[12] bcd_reg[11] n322 1724 1725 NAND2BX1

XU580 n322 bcd_reg[10] n321 1726 1727 NAND2X1
XU581 n321 bcd_reg[9] bcd_reg[8] 1728 1729 OR2X1
XU582 bcd_next[8] bcd_reg[7] n317 1730 1731 NAND2BX1
XU583 n317 bcd_reg[6] n316 1732 1733 NAND2X1
XU584 n316 bcd_reg[5] bcd_reg[4] 1734 1735 OR2X1
XU585 N102 bit_count[1] bit_count[0] 1736 1737 XOR2X1
XU586 N104 bit_count[3] n335 1738 1739 XOR2X1
XU587 n335 n334 n333 1740 1741 NOR2X1
XU588 n333 bit_count[2] 1742 1743 INVX2
XU589 N101 bit_count[0] 1744 1745 INVX2
XU590 n230 n203 busy_bit 1746 1747 AND2X1
XU591 n203 n204 n197 n205 1748 1749 NAND3X1
XU592 n205 bit_count[3] n198 1750 1751 NAND2X1
XU593 n204 N104 ce_i 1752 1753 NAND2X1
XU594 n231 n206 busy_bit 1754 1755 AND2X1
XU595 n206 n207 n197 n208 1756 1757 NAND3X1
XU596 n208 bit_count[2] n198 1758 1759 NAND2X1
XU597 n207 N103 ce_i 1760 1761 NAND2X1
XU598 n232 n209 busy_bit 1762 1763 AND2X1
XU599 n209 n210 n197 n211 1764 1765 NAND3X1
XU600 n211 bit_count[1] n198 1766 1767 NAND2X1
XU601 n210 N102 ce_i 1768 1769 NAND2X1
XU602 n290 n200 busy_bit 1770 1771 AND2X1
XU603 n200 n201 n197 n202 1772 1773 NAND3X1
XU604 n202 bit_count[0] n198 1774 1775 NAND2X1
XU605 n201 N101 ce_i 1776 1777 NAND2X1

Inserting wire Capacities

c0 0 dat_binary_i[15] 0.007342pf
c1 0 dat_binary_i[14] 0.006662pf
c2 0 dat_binary_i[13] 0.007227pf
c3 0 dat_binary_i[12] 0.006577pf
c4 0 dat_binary_i[11] 0.007342pf
c5 0 dat_binary_i[10] 0.006698pf
c6 0 dat_binary_i[9] 0.008630pf
c7 0 dat_binary_i[8] 0.006698pf
c8 0 dat_binary_i[7] 0.006698pf
c9 0 dat_binary_i[6] 0.006698pf
c10 0 dat_binary_i[5] 0.006662pf
c11 0 dat_binary_i[4] 0.006782pf
c12 0 dat_binary_i[3] 0.006698pf
c13 0 dat_binary_i[2] 0.007084pf
c14 0 dat_binary_i[1] 0.006974pf
c15 0 dat_binary_i[0] 0.006720pf
c16 0 dat_bcd_o[19] 0.008398pf
c17 0 dat_bcd_o[18] 0.007596pf
c18 0 dat_bcd_o[17] 0.009549pf
c19 0 dat_bcd_o[16] 0.008144pf
c20 0 dat_bcd_o[15] 0.009371pf
c21 0 dat_bcd_o[14] 0.008611pf
c22 0 dat_bcd_o[13] 0.009000pf
c23 0 dat_bcd_o[12] 0.010627pf
c24 0 dat_bcd_o[11] 0.009508pf

c25 0 dat_bcd_o[10] 0.007036pf
c26 0 dat_bcd_o[9] 0.008777pf
c27 0 dat_bcd_o[8] 0.009124pf
c28 0 dat_bcd_o[7] 0.008788pf
c29 0 dat_bcd_o[6] 0.008796pf
c30 0 dat_bcd_o[5] 0.008895pf
c31 0 dat_bcd_o[4] 0.011926pf
c32 0 dat_bcd_o[3] 0.008148pf
c33 0 dat_bcd_o[2] 0.008545pf
c34 0 dat_bcd_o[1] 0.010063pf
c35 0 dat_bcd_o[0] 0.008147pf
c36 0 clk_i 0.079851pf
c37 0 ce_i 0.009788pf
c38 0 rst_i 0.009976pf
c39 0 start_i 0.008585pf
c40 0 done_o 0.009522pf
c41 0 bin_next[15] 0.001061pf
c42 0 bin_next[14] 0.001192pf
c43 0 bin_next[13] 0.001620pf
c44 0 bin_next[12] 0.001536pf
c45 0 bin_next[11] 0.000936pf
c46 0 bin_next[10] 0.001327pf
c47 0 bin_next[9] 0.002095pf
c48 0 bin_next[8] 0.001621pf
c49 0 bin_next[7] 0.001457pf
c50 0 bin_next[6] 0.002500pf
c51 0 bin_next[5] 0.002251pf
c52 0 bin_next[4] 0.001791pf
c53 0 bin_next[3] 0.001747pf
c54 0 bin_next[2] 0.002823pf
c55 0 bin_next[1] 0.002036pf
c56 0 N7 0.002202pf
c57 0 N8 0.004462pf
c58 0 N9 0.000312pf
c59 0 N11 0.007188pf
c60 0 N12 0.003288pf
c61 0 N13 0.000988pf
c62 0 N15 0.004579pf
c63 0 N16 0.001630pf
c64 0 N17 0.000312pf
c65 0 N19 0.001923pf
c66 0 N20 0.002201pf
c67 0 N21 0.002024pf
c68 0 N23 0.002877pf
c69 0 N24 0.002313pf
c70 0 N25 0.000583pf
c71 0 N26 0.002379pf
c72 0 busy_bit 0.006650pf
c73 0 N101 0.001217pf
c74 0 N102 0.001941pf
c75 0 N103 0.000828pf
c76 0 N104 0.001393pf
c77 0 n32 0.000585pf
c78 0 n33 0.002802pf

c79 0 n34 0.013034pf
c80 0 n35 0.005681pf
c81 0 n36 0.026424pf
c82 0 n37 0.002277pf
c83 0 n38 0.007042pf
c84 0 n39 0.002613pf
c85 0 n40 0.000312pf
c86 0 n41 0.007998pf
c87 0 n42 0.003192pf
c88 0 n43 0.001837pf
c89 0 n44 0.002628pf
c90 0 n45 0.000752pf
c91 0 n46 0.001195pf
c92 0 n47 0.000764pf
c93 0 n48 0.000312pf
c94 0 n49 0.001470pf
c95 0 n50 0.004187pf
c96 0 n51 0.000312pf
c97 0 n52 0.003311pf
c98 0 n53 0.001580pf
c99 0 n54 0.000787pf
c100 0 n55 0.001593pf
c101 0 n56 0.000311pf
c102 0 n57 0.002640pf
c103 0 n58 0.003614pf
c104 0 n59 0.000312pf
c105 0 n60 0.008350pf
c106 0 n61 0.003456pf
c107 0 n62 0.001856pf
c108 0 n63 0.003369pf
c109 0 n64 0.005909pf
c110 0 n65 0.000585pf
c111 0 n66 0.008226pf
c112 0 n67 0.000904pf
c113 0 n68 0.007432pf
c114 0 n69 0.002033pf
c115 0 n70 0.000735pf
c116 0 n71 0.006158pf
c117 0 n72 0.005104pf
c118 0 n73 0.001040pf
c119 0 n74 0.007902pf
c120 0 n75 0.002217pf
c121 0 n76 0.000311pf
c122 0 n77 0.001388pf
c123 0 n78 0.048778pf
c124 0 n79 0.000583pf
c125 0 n80 0.001024pf
c126 0 n81 0.002024pf
c127 0 n83 0.000651pf
c128 0 n84 0.001618pf
c129 0 n85 0.000785pf
c130 0 n86 0.000156pf
c131 0 n87 0.000481pf
c132 0 n88 0.000512pf

c133 0 n89 0.000311pf
c134 0 n90 0.000596pf
c135 0 n91 0.000499pf
c136 0 n92 0.001297pf
c137 0 n93 0.000565pf
c138 0 n94 0.000155pf
c139 0 n95 0.000785pf
c140 0 n96 0.001111pf
c141 0 n97 0.001260pf
c142 0 n98 0.000565pf
c143 0 n99 0.000917pf
c144 0 n100 0.000565pf
c145 0 n101 0.000155pf
c146 0 n102 0.001177pf
c147 0 n103 0.000682pf
c148 0 n104 0.000954pf
c149 0 n105 0.000325pf
c150 0 n106 0.000735pf
c151 0 n107 0.003940pf
c152 0 n108 0.000311pf
c153 0 n109 0.002984pf
c154 0 n110 0.000312pf
c155 0 n111 0.001356pf
c156 0 n112 0.002045pf
c157 0 n113 0.000584pf
c158 0 n114 0.001274pf
c159 0 n115 0.000499pf
c160 0 n116 0.000325pf
c161 0 n117 0.000785pf
c162 0 n118 0.000155pf
c163 0 n119 0.000396pf
c164 0 n120 0.000410pf
c165 0 n121 0.001092pf
c166 0 n122 0.000512pf
c167 0 n123 0.000311pf
c168 0 n124 0.000512pf
c169 0 n125 0.000668pf
c170 0 n126 0.000155pf
c171 0 n127 0.002043pf
c172 0 n128 0.002701pf
c173 0 n129 0.001024pf
c174 0 n130 0.001326pf
c175 0 n131 0.000410pf
c176 0 n132 0.001037pf
c177 0 n133 0.000851pf
c178 0 n134 0.000650pf
c179 0 n135 0.000155pf
c180 0 n136 0.000311pf
c181 0 n137 0.000851pf
c182 0 n138 0.000499pf
c183 0 n139 0.000940pf
c184 0 n140 0.004344pf
c185 0 n141 0.001037pf
c186 0 n142 0.000680pf

c187 0 n143 0.004672pf
c188 0 n144 0.000311pf
c189 0 n145 0.000156pf
c190 0 n146 0.003290pf
c191 0 n147 0.000836pf
c192 0 n148 0.000579pf
c193 0 n149 0.000508pf
c194 0 n150 0.007223pf
c195 0 n151 0.000429pf
c196 0 n152 0.019982pf
c197 0 n153 0.001099pf
c198 0 n154 0.008259pf
c199 0 n155 0.000853pf
c200 0 n156 0.000254pf
c201 0 n157 0.007548pf
c202 0 n158 0.000410pf
c203 0 n159 0.002794pf
c204 0 n160 0.004347pf
c205 0 n161 0.001257pf
c206 0 n162 0.001015pf
c207 0 n163 0.007347pf
c208 0 n164 0.000326pf
c209 0 n165 0.000253pf
c210 0 n166 0.009922pf
c211 0 n167 0.000682pf
c212 0 n168 0.000442pf
c213 0 n169 0.007483pf
c214 0 n170 0.000156pf
c215 0 n171 0.000254pf
c216 0 n172 0.012346pf
c217 0 n173 0.000156pf
c218 0 n174 0.000253pf
c219 0 n175 0.011632pf
c220 0 n176 0.000495pf
c221 0 n177 0.001328pf
c222 0 n178 0.010949pf
c223 0 n179 0.000579pf
c224 0 n180 0.000357pf
c225 0 n181 0.014758pf
c226 0 n182 0.000155pf
c227 0 n183 0.000252pf
c228 0 n184 0.010512pf
c229 0 n185 0.000410pf
c230 0 n186 0.000253pf
c231 0 n187 0.008303pf
c232 0 n188 0.000664pf
c233 0 n189 0.000423pf
c234 0 n190 0.007897pf
c235 0 n191 0.000325pf
c236 0 n192 0.000507pf
c237 0 n193 0.004768pf
c238 0 n194 0.000156pf
c239 0 n195 0.000585pf
c240 0 n196 0.006817pf

c241 0 n197 0.005962pf
c242 0 n198 0.004618pf
c243 0 n199 0.001061pf
c244 0 n200 0.001745pf
c245 0 n201 0.000253pf
c246 0 n202 0.000156pf
c247 0 n203 0.001497pf
c248 0 n204 0.001183pf
c249 0 n205 0.001028pf
c250 0 n206 0.001406pf
c251 0 n207 0.000254pf
c252 0 n208 0.000514pf
c253 0 n209 0.000481pf
c254 0 n210 0.000254pf
c255 0 n211 0.002036pf
c256 0 n212 0.000670pf
c257 0 n213 0.004201pf
c258 0 n214 0.000755pf
c259 0 n215 0.008733pf
c260 0 n216 0.000311pf
c261 0 n217 0.000155pf
c262 0 n218 0.001800pf
c263 0 n219 0.000312pf
c264 0 n220 0.009169pf
c265 0 n221 0.001177pf
c266 0 n222 0.001024pf
c267 0 n223 0.000585pf
c268 0 n224 0.009761pf
c269 0 n225 0.000395pf
c270 0 n226 0.000512pf
c271 0 n227 0.000787pf
c272 0 n228 0.005192pf
c273 0 n229 0.001927pf
c274 0 n230 0.001888pf
c275 0 n231 0.001329pf
c276 0 n232 0.000397pf
c277 0 n233 0.000358pf
c278 0 n234 0.002810pf
c279 0 n235 0.000762pf
c280 0 n236 0.000612pf
c281 0 n237 0.000254pf
c282 0 n238 0.000781pf
c283 0 n239 0.000254pf
c284 0 n240 0.001015pf
c285 0 n241 0.001289pf
c286 0 n242 0.000254pf
c287 0 n243 0.001522pf
c288 0 n244 0.000930pf
c289 0 n245 0.001270pf
c290 0 n246 0.001437pf
c291 0 n247 0.001691pf
c292 0 n248 0.001373pf
c293 0 n249 0.000611pf
c294 0 n250 0.000930pf

c295 0 n251 0.001097pf
c296 0 n252 0.001438pf
c297 0 n253 0.001440pf
c298 0 n254 0.001368pf
c299 0 n255 0.001185pf
c300 0 n256 0.001355pf
c301 0 n257 0.000761pf
c302 0 n258 0.000441pf
c303 0 n259 0.000252pf
c304 0 n260 0.000357pf
c305 0 n261 0.000254pf
c306 0 n262 0.000252pf
c307 0 n263 0.001184pf
c308 0 n264 0.000252pf
c309 0 n265 0.001184pf
c310 0 n266 0.000423pf
c311 0 n267 0.000423pf
c312 0 n268 0.001185pf
c313 0 n269 0.001185pf
c314 0 n270 0.000423pf
c315 0 n271 0.001101pf
c316 0 n272 0.000357pf
c317 0 n273 0.004888pf
c318 0 n274 0.001334pf
c319 0 n275 0.000996pf
c320 0 n276 0.000800pf
c321 0 n277 0.000969pf
c322 0 n278 0.001485pf
c323 0 n279 0.001927pf
c324 0 n280 0.001498pf
c325 0 n281 0.000573pf
c326 0 n282 0.000907pf
c327 0 n283 0.000466pf
c328 0 n284 0.000825pf
c329 0 n285 0.000657pf
c330 0 n286 0.000910pf
c331 0 n287 0.001229pf
c332 0 n288 0.000254pf
c333 0 n289 0.000442pf
c334 0 n290 0.000397pf
c335 0 n291 0.048491pf
c336 0 n292 0.014308pf
c337 0 n293 0.018821pf
c338 0 n308 0.000823pf
c339 0 n309 0.000918pf
c340 0 n310 0.000410pf
c341 0 n311 0.000940pf
c342 0 n312 0.002002pf
c343 0 n313 0.000466pf
c344 0 n314 0.000856pf
c345 0 n315 0.000240pf
c346 0 n316 0.001024pf
c347 0 n317 0.000552pf
c348 0 n318 0.000466pf

```
c349 0 n319 0.000917pf
c350 0 n320 0.000240pf
c351 0 n321 0.001363pf
c352 0 n322 0.001501pf
c353 0 n323 0.000571pf
c354 0 n324 0.000408pf
c355 0 n325 0.000324pf
c356 0 n326 0.000596pf
c357 0 n327 0.000636pf
c358 0 n328 0.000722pf
c359 0 n329 0.001003pf
c360 0 n330 0.000410pf
c361 0 n331 0.000155pf
c362 0 n332 0.001248pf
c363 0 n333 0.000736pf
c364 0 n334 0.001536pf
c365 0 n335 0.002267pf
c366 0 bcd_reg[19] 0.000895pf
c367 0 bcd_reg[18] 0.004698pf
c368 0 bcd_reg[17] 0.002830pf
c369 0 bcd_reg[16] 0.005116pf
c370 0 bcd_reg[15] 0.002209pf
c371 0 bcd_reg[14] 0.004728pf
c372 0 bcd_reg[13] 0.005040pf
c373 0 bcd_reg[12] 0.003383pf
c374 0 bcd_reg[11] 0.002586pf
c375 0 bcd_reg[10] 0.006153pf
c376 0 bcd_reg[9] 0.004449pf
c377 0 bcd_reg[8] 0.004272pf
c378 0 bcd_reg[7] 0.001826pf
c379 0 bcd_reg[6] 0.003617pf
c380 0 bcd_reg[5] 0.006737pf
c381 0 bcd_reg[4] 0.006166pf
c382 0 bcd_reg[3] 0.001402pf
c383 0 bcd_reg[2] 0.005164pf
c384 0 bcd_reg[1] 0.003056pf
c385 0 bcd_reg[0] 0.005005pf
c386 0 bcd_next[16] 0.005589pf
c387 0 bcd_next[12] 0.008410pf
c388 0 bcd_next[8] 0.011162pf
c389 0 bcd_next[4] 0.008136pf
c390 0 bcd_next[0] 0.007689pf
c391 0 bit_count[3] 0.003282pf
c392 0 bit_count[2] 0.005456pf
c393 0 bit_count[1] 0.002922pf
c394 0 bit_count[0] 0.007370pf
.ENDS
```

Ανανεωμένα μεγέθη τρανζίστορ παραδείγματος βελτιστοποίησης

Παρακάτω παρουσίαζονται τα ανανεωμένα μεγέθη τρανζίστορ του παραδείγματος βελτιστοποίησης. Η φόρμα με την οποία παρουσίαζονται είναι η παρακάτω:

<Updated Gate Code> <Fixed Size> <n-size> <p-size>

Και τα αντίστοιχα ανανεωμένα μεγέθη είναι τα παρακάτω:

done_o	Yes	0.00000060	0.00000090
n400	Yes	0.00000032	0.00000048
n298	No	0.00000060	0.00000090
n404	Yes	0.00000032	0.00000048
n291	Yes	0.00000060	0.00000120
n481	Yes	0.00000078	0.00000090
n285	No	0.00000060	0.00000120
n227	No	0.00000060	0.00000120
n402	Yes	0.00000078	0.00000090
n409	Yes	0.00000078	0.00000090
n413	Yes	0.00000078	0.00000090
n417	Yes	0.00000078	0.00000090
n419	Yes	0.00000078	0.00000090
n425	Yes	0.00000078	0.00000090
n429	Yes	0.00000078	0.00000090
n433	Yes	0.00000078	0.00000090
n435	Yes	0.00000078	0.00000090
n441	Yes	0.00000078	0.00000090
n445	Yes	0.00000078	0.00000090
n449	Yes	0.00000078	0.00000090
n451	Yes	0.00000078	0.00000090
n457	Yes	0.00000078	0.00000090
n461	Yes	0.00000078	0.00000090
n465	Yes	0.00000078	0.00000090
n467	Yes	0.00000078	0.00000090
n473	Yes	0.00000078	0.00000090
n477	Yes	0.00000078	0.00000090
n482	Yes	0.00000078	0.00000090
n290	Yes	0.00000078	0.00000090
n414	No	0.00000078	0.00000090
n225	No	0.00000060	0.00000120
n418	No	0.00000078	0.00000090
n434	No	0.00000078	0.00000090
n450	No	0.00000078	0.00000090
n466	No	0.00000078	0.00000090
n226	No	0.00000060	0.00000120
n284	No	0.00000071	0.00000100
n430	No	0.00000078	0.00000090
n446	No	0.00000078	0.00000090
n462	No	0.00000078	0.00000090
n478	No	0.00000078	0.00000090
n403	No	0.00000078	0.00000090
n420	No	0.00000078	0.00000090
n436	No	0.00000078	0.00000090
n452	No	0.00000078	0.00000090
n468	No	0.00000078	0.00000090
n236	No	0.00000079	0.00000092
n240	No	0.00000079	0.00000092
n410	No	0.00000078	0.00000090
n426	No	0.00000078	0.00000090

n442	No	0.00000078	0.00000090
n458	No	0.00000078	0.00000090
n474	No	0.00000078	0.00000090
n229	No	0.00000079	0.00000092
n231	No	0.00000079	0.00000092
n243	No	0.00000079	0.00000092
n247	No	0.00000079	0.00000092
n238	No	0.00000079	0.00000092
n242	No	0.00000079	0.00000092
n246	No	0.00000079	0.00000092