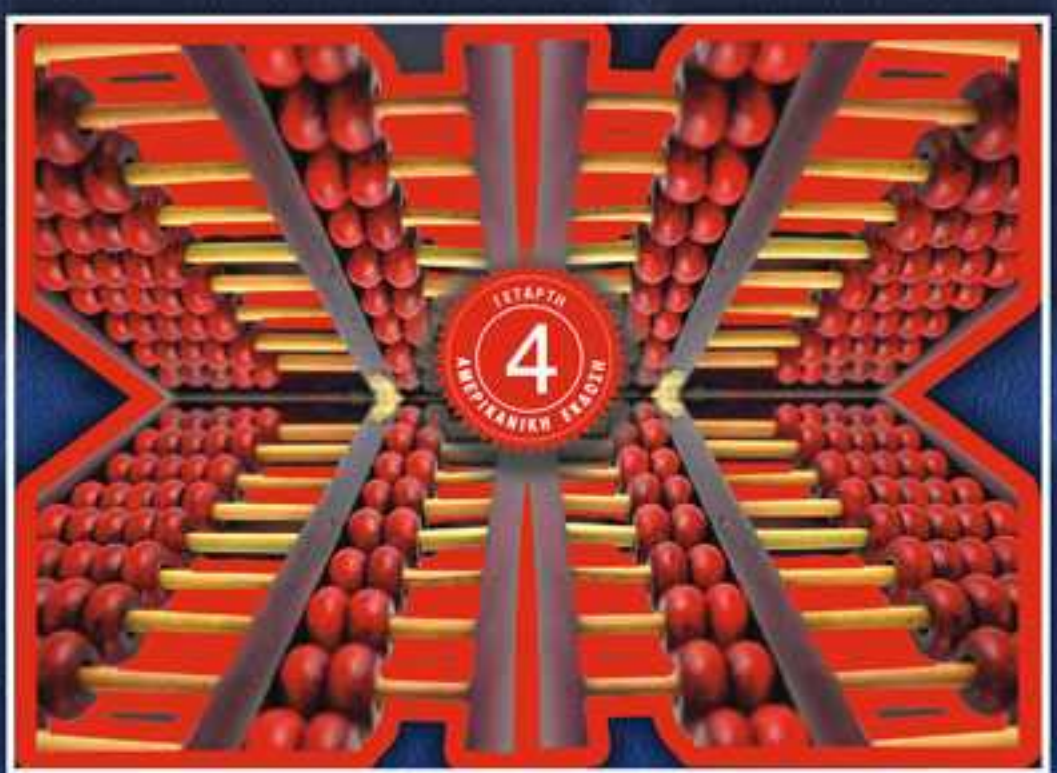
ΕΚΔΟΣΕΙΣ ΚΛΕΙΔΑΡΙΘΜΟΣ

ΤΟΜΟΣ Β΄

ΟΡΓΑΝΩΣΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΥΠΟΛΟΓΙΣΤΩΝ

Η ΔΙΑΣΥΝΔΕΣΗ ΥΛΙΚΟΥ ΚΑΙ ΛΟΓΙΣΜΙΚΟΥ



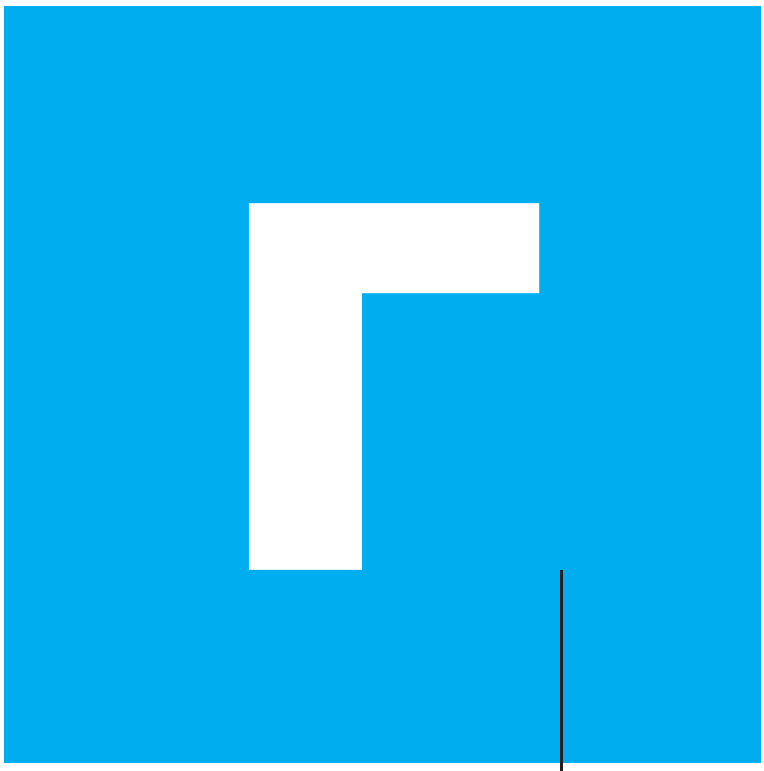
DAVID A. PATTERSON
JOHN L. HENNESSY

ΜΕΤΑΦΡΑΣΗ & ΕΠΙΣΤΗΜΟΝΙΚΗ ΕΠΙΜΕΛΕΙΑ
ΔΗΜΗΤΡΗΣ ΓΚΙΖΟΠΟΥΛΟΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΚΛΕΙΔΑΡΙΘΜΟΣ

Περιεχόμενα

ΠΑΡΑΡΤΗΜΑΤΑ



ΠΑΡΑΡΤΗΜΑ

Τα βασικά της λογικής σχεδίασης

Πάντα αγαπούσα αυτή τη λέξη, Boolean.

Claude Shannon
IEEE Spectrum, Απρίλιος 1992
(Η μεταπτυχιακή διατριβή τού Shannon έδειξε ότι η άλγεβρα που εφευρέθηκε από τον George Boole το 19ο αιώνα μπορούσε να αναπαραστήσει τους μηχανισμούς των ηλεκτρικών διακοπτών.)

ΠΕΡΙΕΧΟΜΕΝΑ

Γ.1 Εισαγωγή

Αυτό το παράρτημα παρέχει μια σύντομη εξέταση των αρχών της λογικής σχεδίασης. Δεν μπορεί να υποκαταστήσει το μάθημα της λογικής σχεδίασης ούτε θα σας επιτρέψει να σχεδιάσετε μεγάλα λογικά συστήματα που να λειτουργούν. Αν όμως έχετε λίγες ή καθόλου γνώσεις σχετικά με τη λογική σχεδίαση, αυτό το παράρτημα θα σας δώσει ένα σημαντικό υπόβαθρο για την κατανόηση όλου του υλικού αυτού του βιβλίου. Επιπλέον, αν θέλετε να καταλάβετε μερικά από τα κίνητρα πίσω από τον τρόπο με τον οποίο υλοποιούνται οι υπολογιστές, αυτό το υλικό θα εξυπηρετήσει ως μια χρήσιμη εισαγωγή. Αν η περιέργειά σας εξαφθεί αλλά δεν ικανοποιηθεί από αυτό το παράρτημα, η βιβλιογραφία στο τέλος παρέχει διάφορες επιπλέον πηγές πληροφοριών.

Η Ενότητα Γ.2 εισάγει τα βασικά δομικά στοιχεία τής λογικής, δηλαδή τις *πύλες* (gates). Η Ενότητα Γ.3 χρησιμοποιεί αυτά τα δομικά στοιχεία για την κατασκευή απλών *συνδυαστικών* (combinational) λογικών συστημάτων, τα οποία δεν περιέχουν μνήμη. Αν έχετε ήδη ασχοληθεί με λογικά ή ψηφιακά συστήματα, πιθανόν έχετε κάποια οικειότητα με τις πληροφορίες που περιέχονται σε αυτές τις δύο πρώτες ενότητες. Η Ενότητα Γ.5 δείχνει πώς να χρησιμοποιήσετε τις έννοιες των Ενοτήτων Γ.2 και Γ.3 στη σχεδίαση μιας ALU για τον επεξεργαστή MIPS. Η Ενότητα Γ.6 δείχνει πώς θα κατασκευάσετε ένα γρήγορο αθροιστή, και μπορείτε να την παραλείψετε χωρίς συνέπειες αν δεν ενδιαφέρεστε γι' αυτό το αντικείμενο. Η Ενότητα Γ.7 είναι μια σύντομη εισαγωγή στο θέμα του χρονισμού (clocking), ο οποίος είναι απαραίτητος για την εξέταση του τρόπου με τον οποίο δουλεύουν τα στοιχεία μνήμης. Η Ενότητα Γ.8 παρουσιάζει τα στοιχεία τής μνήμης, ενώ η Ενότητα Γ.9 επεκτείνει το θέμα εστιάζοντας στις μνήμες τυχαίας προσπέλασης· περιγράφει τόσο τα χαρακτηριστικά που είναι σημαντικά για την κατανόηση του τρόπου με τον οποίο χρησιμοποιούνται στο Κεφάλαιο 4, όσο και το υπόβαθρο πολλών από τις πλευρές τής σχεδίασης ιεραρχιών μνήμης στο Κεφάλαιο 5. Η Ενότητα Γ.10 περιγράφει τη σχεδίαση και τη χρήση των μηχανών πεπερασμένης κατάστασης, οι οποίες είναι μπλοκ ακολουθιακής λογικής. Αν έχετε την πρόθεση να διαβάσετε το Παράρτημα Δ, θα πρέπει να έχετε κατανοήσει σε βάθος το υλικό των Ενοτήτων Γ.2 έως Γ.10. Αν όμως σκοπεύετε να διαβάσετε μόνο το υλικό για τον έλεγχο στο Κεφάλαιο 4, μπορείτε να περάσετε γρήγορα τα παραρτήματα, αλλά πρέπει να έχετε κάποια οικειότητα με όλο το υλικό εκτός της Ενότητας Γ.11. Η Ενότητα Γ.11 είναι γι' αυτούς που θέλουν μια βαθύτερη κατανόηση των μεθοδολογιών χρονισμού. Εξηγεί τα βασικά του τρόπου που δουλεύει ο ακμοπυροδοτούμενος χρονισμός (edge-triggered clocking), εισάγει μια άλλη μέθοδο χρονισμού, και περιγράφει σύντομα το πρόβλημα του συγχρονισμού των ασύγχρονων εισόδων.

Σε ολόκληρο αυτό το παράρτημα, όπου κρίναμε κατάλληλο, συμπεριλάβαμε τμήματα κώδικα Verilog για να δείξουμε πώς μπορεί να αναπαρασταθεί η λογική στη Verilog, την οποία παρουσιάζουμε στην Ενότητα Γ.4. Ένα πιο εκτεταμένο και πλήρες εκπαιδευτικό βοήθημα της Verilog (στα αγγλικά) υπάρχει στην ιστοσελίδα του βιβλίου (www.klidarithmos.gr/cod4e-gr).

Γ.2 Πύλες, πίνακες αληθείας, και λογικές εξισώσεις

Τα ηλεκτρονικά στο εσωτερικό ενός σύγχρονου υπολογιστή είναι *ψηφιακά*. Τα ψηφιακά ηλεκτρονικά λειτουργούν με δύο μόνον επίπεδα τάσης: μια υψηλή τάση και μια χαμηλή τάση. Όλες οι άλλες τιμές τάσης είναι προσωρινές και παρουσιάζονται κατά τη μετάβαση μεταξύ των δύο αυτών τιμών. (Όπως θα πούμε αργότερα σε αυτή την ενότητα, μια πιθανή παγίδα στην ψηφιακή σχεδίαση είναι η δειγματοληψία ενός σήματος όταν δεν είναι ούτε υψηλό ούτε χαμηλό.) Το γεγονός ότι οι υπολογιστές είναι ψηφιακοί είναι επίσης ένας βασικός λόγος για τον οποίο χρησιμοποιούν δυαδικούς αριθμούς, επειδή ένα δυαδικό σύστημα ταιριάζει με τη θεμελιώδη αφαίρεση (abstraction) που είναι εγγενής στα ηλεκτρονικά. Στις διάφορες οικογένειες λογικής, οι τιμές και οι σχέσεις ανάμεσα στα δύο επίπεδα τάσης διαφέρουν. Έτσι, αντί να αναφερόμαστε σε επίπεδα τάσης, μιλούμε για σήματα που είναι (λογικά) αληθή, ή 1, ή είναι **ενεργοποιημένα** (asserted)· ή σήματα που είναι (λογικά) ψευδή, ή 0, ή **απενεργοποιημένα** (deasserted). Οι τιμές 0 και 1 ονομάζονται *συμπληρώματα* ή *αντίστροφες* η μία της άλλης.

ενεργοποιημένο σήμα (asserted signal) Ένα σήμα που είναι (λογικά) αληθές, ή 1.

απενεργοποιημένο σήμα (deasserted signal) Ένα σήμα που είναι (λογικά) ψευδές, ή 0.

Τα λογικά μπλοκ κατατάσσονται σε έναν από δύο τύπους, ανάλογα με το αν περιέχουν μνήμη. Τα μπλοκ χωρίς μνήμη ονομάζονται *συνδυαστικά* (combinational)· η έξοδος ενός συνδυαστικού μπλοκ εξαρτάται μόνον από την τρέχουσα είσοδο. Στα μπλοκ με μνήμη, οι έξοδοι μπορεί να εξαρτώνται τόσο από τις εισόδους όσο και από την τιμή που είναι αποθηκευμένη στη μνήμη, η οποία ονομάζεται *κατάσταση* (state) του λογικού μπλοκ. Σε αυτή και στην επόμενη ενότητα, θα επικεντρωθούμε μόνο στη **συνδυαστική λογική** (combinational logic). Μετά την εισαγωγή διαφορετικών στοιχείων μνήμης στην Ενότητα Γ.8, θα περιγράψουμε πώς σχεδιάζεται η **ακολουθιακή λογική** (sequential logic), η οποία είναι λογική που περιλαμβάνει κατάσταση.

συνδυαστική λογική (combinational logic) Ένα σύστημα λογικής του οποίου τα μπλοκ δεν περιέχουν μνήμη και συνεπώς υπολογίζουν την ίδια έξοδο για την ίδια είσοδο.

ακολουθιακή λογική (sequential logic) Μια ομάδα από λογικά στοιχεία που περιέχουν μνήμη και, συνεπώς, των οποίων η τιμή εξαρτάται από τις εισόδους καθώς επίσης και από τα τρέχοντα περιεχόμενα της μνήμης.

Πίνακες αληθείας

Επειδή ένα μπλοκ συνδυαστικής λογικής δεν περιέχει μνήμη, μπορεί να καθοριστεί πλήρως με τον ορισμό των τιμών των εξόδων για κάθε πιθανό σύνολο τιμών των εισόδων. Μια τέτοια περιγραφή φυσιολογικά δίνεται με τη μορφή ενός *πίνακα αληθείας* (truth table). Για ένα λογικό μπλοκ με n εισόδους, υπάρχουν 2^n καταχωρίσεις στον πίνακα αληθείας, επειδή τόσοι είναι οι πιθανοί συνδυασμοί των τιμών εισόδου. Κάθε καταχώριση προσδιορίζει την τιμή όλων των εξόδων για το συγκεκριμένο συνδυασμό εισόδων.

ΠΑΡΑΔΕΙΓΜΑ

Πίνακες αληθείας

Θεωρήστε μια λογική συνάρτηση με τρεις εισόδους, A, B, και C, και τρεις εξόδους, D, E, και F. Η συνάρτηση ορίζεται ως εξής: η D είναι αληθής αν τουλάχιστον μία είσοδος είναι αληθής, η E είναι αληθής αν ακριβώς δύο είσοδοι είναι αληθείς, και η F είναι αληθής μόνον αν και οι τρεις είσοδοι είναι αληθείς. Κατασκευάστε τον πίνακα αληθείας αυτής της συνάρτησης.

ΑΠΑΝΤΗΣΗ

Ο πίνακας αληθείας θα περιέχει $2^3 = 8$ καταχωρίσεις. Είναι ο εξής:

Είσοδοι			Έξοδοι		
A	B	C	D	E	F
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	0	1

Οι πίνακες αληθείας μπορούν να περιγράψουν πλήρως οποιαδήποτε συνδυαστική λογική συνάρτηση· ωστόσο, αυξάνονται σε μέγεθος γρήγορα και μπορεί να μην είναι εύκολο να κατανοηθούν. Μερικές φορές θέλουμε να κατασκευάσουμε μια λογική συνάρτηση που θα είναι 0 για πολλούς συνδυασμούς εισόδων, οπότε χρησιμοποιούμε μια συντομογραφία που καθορίζει μόνο τις καταχωρίσεις του πίνακα αληθείας για τις μη μηδενικές εξόδους. Αυτή η προσέγγιση χρησιμοποιείται στο Κεφάλαιο 4 και στο Παράρτημα Δ.

Άλγεβρα Boole

Μια άλλη προσέγγιση είναι να εκφράσουμε τη λογική συνάρτηση με λογικές εξισώσεις. Αυτό γίνεται με τη χρήση της *άλγεβρας Boole* (Boolean algebra, που πήρε το όνομά της από τον George Boole, ένα μαθηματικό του 19ου αιώνα). Στην άλγεβρα Boole, όλες οι μεταβλητές έχουν τιμές 0 και 1 και, στην τυπική σημειογραφία, υπάρχουν τρεις τελεστές:

- Ο τελεστής OR γράφεται +, όπως στην παράσταση $A + B$. Το αποτέλεσμα ενός τελεστή OR είναι 1 αν οποιαδήποτε από τις μεταβλητές είναι 1. Η πράξη (λειτουργία) OR ονομάζεται επίσης *λογικό άθροισμα*, αφού το αποτέλεσμά της είναι 1 αν οποιοσδήποτε τελεστέος είναι 1.
- Ο τελεστής AND γράφεται ·, όπως στην παράσταση $A \cdot B$. Το αποτέλεσμα ενός τελεστή AND είναι 1 μόνον αν και οι δύο είσοδοι είναι 1.

Ο τελεστής AND ονομάζεται επίσης και *λογικό γινόμενο*, αφού το αποτέλεσμά του είναι 1 μόνον όταν και οι δύο τελεστέοι είναι 1.

- Ο μονομελής τελεστής NOT γράφεται $\bar{A}$. Το αποτέλεσμα ενός τελεστή NOT είναι 1 μόνον αν η είσοδος είναι 0. Η εφαρμογή του τελεστή NOT σε μια λογική τιμή έχει ως αποτέλεσμα την αντιστροφή ή άρνηση της τιμής (π.χ., αν η είσοδος είναι 0 η έξοδος είναι 1, και το αντίστροφο).

Υπάρχουν διάφοροι νόμοι της άλγεβρας Boole που είναι χρήσιμοι στο χειρισμό λογικών εξισώσεων.

- Ο νόμος του ουδέτερου στοιχείου (identity law): $A + 0 = A$ και $A \cdot 1 = A$.
- Ο νόμος του απορροφητικού στοιχείου (zero and one laws): $A \cdot 0 = 0$ και $A + 1 = 1$.
- Οι νόμοι του αντιστρόφου (inverse laws): $A + \bar{A} = 1$ και $A \cdot \bar{A} = 0$.
- Οι αντιμεταθετικοί νόμοι (commutative laws): $A + B = B + A$ και $A \cdot B = B \cdot A$.
- Οι προσεταιριστικοί νόμοι (associative laws): $A + (B + C) = (A + B) + C$ και $A \cdot (B \cdot C) = (A \cdot B) \cdot C$
- Οι επιμεριστικοί νόμοι (distributive laws): $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ και $A + (B \cdot C) = (A + B) \cdot (A + C)$

Επιπλέον, υπάρχουν ακόμα δύο χρήσιμα θεωρήματα, που ονομάζονται νόμοι του DeMorgan και εξετάζονται πιο αναλυτικά στις ασκήσεις.

Κάθε σύνολο λογικών συναρτήσεων μπορεί να γραφεί ως μια σειρά εξισώσεων με μια έξοδο στο αριστερό μέρος κάθε εξίσωσης, και έναν τύπο που αποτελείται από μεταβλητές και τους τρεις παραπάνω τελεστές στο δεξιό μέρος.

Λογικές εξισώσεις

ΠΑΡΑΔΕΙΓΜΑ

Δείξτε τις λογικές εξισώσεις για τις λογικές συναρτήσεις, *D*, *E*, και *F*, που περιγράφονται στο προηγούμενο παράδειγμα.

ΑΠΑΝΤΗΣΗ

Η εξίσωση για την *D* είναι:

$$D = A + B + C$$

Η *F* είναι εξίσου απλή:

$$F = A \cdot B \cdot C$$

Η *E* είναι λίγο πιο σύνθετη. Θεωρήστε ότι αποτελείται από δύο μέρη: τι πρέπει να είναι αληθές για να είναι η *E* αληθής (δύο από τις τρεις εισόδους πρέπει να είναι αληθείς), και τι δεν μπορεί να είναι αληθές (και οι τρεις δεν μπορούν να είναι αληθείς). Έτσι μπορούμε να γράψουμε την *E* ως εξής:

$$E = ((A \cdot B) + (A \cdot C) + (B \cdot C)) \cdot (\overline{A \cdot B \cdot C})$$

Μπορούμε επίσης να καταλήξουμε στην E διαπιστώνοντας ότι η E είναι αληθής μόνον αν ακριβώς δύο από τις εισόδους της είναι αληθείς. Τότε μπορούμε να γράψουμε την E ως ένα OR των τριών δυνατών όρων που έχουν δύο αληθείς εισόδους και μία ψευδή:

$$E = (A \cdot B \cdot \overline{C}) + (A \cdot C \cdot \overline{B}) + (B \cdot C \cdot \overline{A})$$

Η απόδειξη ότι αυτές οι δύο παραστάσεις είναι ισοδύναμες διερευνάται στις ασκήσεις.

Στη Verilog, όποτε είναι δυνατόν, περιγράφουμε τη συνδυαστική λογική με την εντολή ανάθεσης τιμής `assign`, η οποία περιγράφεται με αρχή τη σελίδα 34. Μπορούμε να γράψουμε έναν ορισμό για την E χρησιμοποιώντας τον τελεστή αποκλειστικού OR της Verilog ως `assign E = ~(A ^ B ^ C) & (A + B + C)` που είναι ένας ακόμη τρόπος να αναπαρασταθεί αυτή η συνάρτηση· οι D και F έχουν ακόμη απλούστερες αναπαραστάσεις, οι οποίες είναι ακριβώς όπως ο αντίστοιχος κώδικας τής γλώσσας C: `D = A | B | C` και `F = A & B & C`.

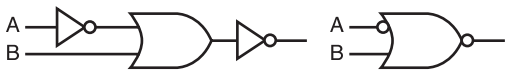
Πύλες

πύλη (gate) Μια διάταξη που υλοποιεί βασικές λογικές συναρτήσεις όπως οι AND και OR.

Τα μπλοκ λογικής κατασκευάζονται από **πύλες** (gates) που υλοποιούν τις βασικές λογικές συναρτήσεις. Για παράδειγμα, μια πύλη AND (AND gate) υλοποιεί τη συνάρτηση AND, και μια πύλη OR (OR gate) υλοποιεί τη συνάρτηση OR. Αφού τόσο η AND όσο και η OR είναι αντιμεταθετικές και προσεταιριστικές, μια πύλη AND ή μια πύλη OR μπορούν να έχουν πολλές εισόδους, με την έξοδο να είναι ίση με το AND ή το OR όλων των εισόδων. Η λογική συνάρτηση NOT υλοποιείται με έναν αντιστροφέα (inverter) που έχει πάντα μία μόνον είσοδο. Η τυπική αναπαράσταση αυτών των τριών λογικών δομικών στοιχείων φαίνεται στην Εικόνα Γ.2.1.

Αντί να σχεδιάζουμε τους αντιστροφείς απευθείας, μια συνηθισμένη πρακτική είναι να προσθέτουμε «φυσαλίδες» (bubbles) στις εισόδους ή την έξοδο μιας πύλης ώστε να προκαλέσουμε την αντιστροφή της λογικής τιμής αυτής της γραμμής εισόδου ή εξόδου. Για παράδειγμα, η Εικόνα Γ.2.2 δείχνει το λογικό διάγραμμα για τη συνάρτηση $\overline{A} + B$, χρησιμοποιώντας ρητούς αντιστροφείς στα αριστερά, και εισόδους και εξόδους με φυσαλίδες στα δεξιά.

ΕΙΚΟΝΑ Γ.2.1 Πρότυπο σχέδιο μιας πύλης AND, μιας πύλης OR, και ενός αντιστροφέα, από αριστερά προς τα δεξιά. Τα σήματα στα αριστερά κάθε συμβόλου είναι οι είσοδοι, ενώ η έξοδος εμφανίζεται δεξιά. Τόσο η πύλη AND όσο και η πύλη OR έχουν δύο εισόδους. Οι αντιστροφείς έχουν μία μόνον είσοδο.



ΕΙΚΟΝΑ Γ.2.2 Υλοποίηση της $\overline{A}+B$ με λογικές πύλες, ρητούς αντιστροφείς στα αριστερά, και εισόδους και έξοδο με φυσαλίδες στα δεξιά. Αυτή η λογική συνάρτηση μπορεί να απλοποιηθεί στην $A \cdot \overline{B}$ ή `A & ~ B` σε Verilog.

Οποιαδήποτε λογική συνάρτηση μπορεί να κατασκευαστεί με πύλες AND, πύλες OR, και αντιστροφείς· αρκετές από τις ασκήσεις σάς δίνουν την ευκαιρία να δοκιμάσετε την υλοποίηση μερικών συνήθων λογικών συναρτήσεων με πύλες. Στην επόμενη ενότητα, θα δούμε πώς μπορεί να κατασκευαστεί μια υλοποίηση οποιασδήποτε λογικής συνάρτησης με βάση αυτές τις γνώσεις.

Στην πραγματικότητα, όλες οι λογικές συναρτήσεις μπορούν να κατασκευαστούν μόνο με έναν τύπο λογικής πύλης, αν αυτή η πύλη αντιστραφεί. Οι δύο συνηθισμένες πύλες αντιστροφής ονομάζονται **NOR** και **NAND** και αντιστοιχούν σε ανεστραμμένες πύλες OR και AND, αντίστοιχα. Οι πύλες NOR και NAND ονομάζονται *οικουμενικές* (universal), αφού οποιαδήποτε λογική συνάρτηση μπορεί να κατασκευαστεί με έναν μόνο από αυτούς τους τύπους πύλης. Οι ασκήσεις διερευνούν αυτή την έννοια περαιτέρω.

πύλη NOR (NOR gate)
Μια ανεστραμμένη πύλη OR.

πύλη NAND (NAND gate)
Μια ανεστραμμένη πύλη AND.

Αυτοεξέταση

Είναι ισοδύναμες οι δύο λογικές παραστάσεις που ακολουθούν; Αν όχι, βρείτε ένα σύνολο τιμών των μεταβλητών για να δείξετε ότι δεν είναι:

- $(A \cdot B \cdot \overline{C}) + (A \cdot C \cdot \overline{B}) + (B \cdot C \cdot \overline{A})$
- $B \cdot (A \cdot \overline{C} + C \cdot \overline{A})$

Γ.3 Συνδυαστική λογική

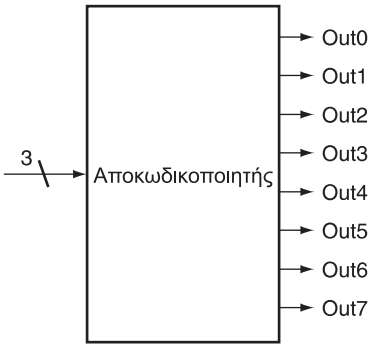
Σε αυτή την ενότητα, θα δούμε μερικά μεγαλύτερα δομικά στοιχεία λογικής που χρησιμοποιούμε σε μεγάλη έκταση, και θα εξετάσουμε τη σχεδίαση δομημένης λογικής που μπορεί να υλοποιηθεί αυτόματα από μια λογική εξίσωση ή πίνακα αληθείας με τη βοήθεια ενός προγράμματος μετάφρασης. Τέλος, θα εξερευνήσουμε την έννοια μιας διάταξης (array) από λογικά μπλοκ.

Αποκωδικοποιητές

Ένα λογικό μπλοκ που θα χρησιμοποιήσουμε στην κατασκευή μεγαλύτερων συστατικών είναι ένας **αποκωδικοποιητής** (decoder). Ο πιο συνηθισμένος τύπος αποκωδικοποιητή έχει μία είσοδο των n bit και 2^n εξόδους, όπου μόνο μία έξοδος είναι ενεργοποιημένη για κάθε συνδυασμό εισόδων. Αυτός ο αποκωδικοποιητής μεταφράζει την είσοδο των n bit σε ένα σήμα που αντιστοιχεί στη δυαδική τιμή της εισόδου των n bit. Έτσι, οι έξοδοι είναι συνήθως αριθμημένες, ας πούμε, Out0, Out1, …, Out2^n–1. Αν η τιμή της εισόδου

αποκωδικοποιητής (decoder)
Ένα λογικό μπλοκ που έχει μια είσοδο των n bit, και 2^n εξόδους όπου μόνο μία έξοδος είναι ενεργοποιημένη για κάθε συνδυασμό εισόδων.

είναι *i*, τότε η Out*i* θα είναι αληθής και όλες οι άλλες έξοδοι θα είναι ψευδείς. Η Εικόνα Γ.3.1 δείχνει έναν αποκωδικοποιητή των 3 bit και τον πίνακα αληθείας. Αυτός ο αποκωδικοποιητής ονομάζεται *αποκωδικοποιητής 3 σε 8* (3-to-8 decoder) επειδή υπάρχουν 3 είσοδοι και 8 (2^3) έξοδοι. Υπάρχει επίσης ένα λογικό στοιχείο που ονομάζεται *κωδικοποιητής* (encoder) και εκτελεί την αντίστροφη λειτουργία ενός αποκωδικοποιητή, παίρνοντας 2^n εισόδους και παράγοντας μία έξοδο των *n* bit.



α. Ένας αποκωδικοποιητής των 3 bit

Είσοδοι			Έξοδοι							
In2	In1	In0	Out7	Out6	Out5	Out4	Out3	Out2	Out1	Out0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

β. Ο πίνακας αληθείας του αποκωδικοποιητή των 3 bit

ΕΙΚΟΝΑ Γ.3.1 Ένας αποκωδικοποιητής των 3 bit έχει 3 εισόδους, που ονομάζονται In2, In1, και In0, και $2^3 = 8$ εξόδους, που ονομάζονται Out0 έως Out7. Μόνον η έξοδος που αντιστοιχεί στη δυαδική τιμή της εισόδου είναι αληθής, όπως φαίνεται στον πίνακα αληθείας. Η ετικέτα 3 στην είσοδο του αποκωδικοποιητή λέει ότι το σήμα εισόδου έχει εύρος 3 bit.

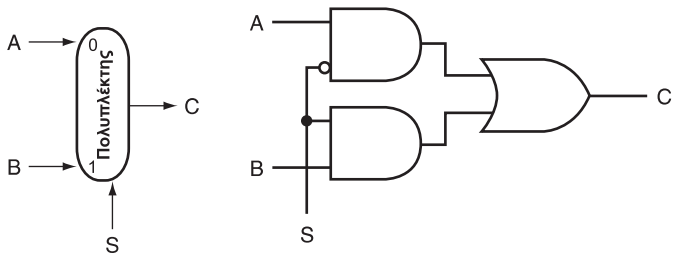
Πολυπλέκτες

Μια βασική λογική συνάρτηση που χρησιμοποιούμε πολύ συχνά στο Κεφάλαιο 4 είναι ο *πολυπλέκτης* (multiplexor). Ο πολυπλέκτης θα ήταν πιο κατάλληλο να ονομάζεται *επιλογέας* (selector), αφού η έξοδός του είναι μία από τις εισόδους που επιλέγει ένα σήμα ελέγχου. Ας πάρουμε τον πολυπλέκτη δύο εισόδων. Το αριστερό μέρος της Εικόνας Γ.3.2. δείχνει ότι αυτός ο πολυπλέκτης έχει τρεις εισόδους: δύο τιμές δεδομένων και μία **τιμή επιλογέα** (selector value) ή **τιμή ελέγχου** (control value). Η τιμή του επιλογέα καθορίζει ποια από τις εισόδους γίνεται έξοδος. Μπορούμε να αναπαραστήσουμε τη λογική συνάρτηση την οποία υπολογίζει ένας πολυπλέκτης δύο εισόδων, που φαίνεται με τη μορφή πυλών στο δεξιό μέρος της Εικόνας Γ.3.2, ως $C = (A \cdot \overline{S}) + (B \cdot S)$.

τιμή επιλογέα (selector value) Ονομάζεται επίσης **τιμή ελέγχου** (control value). Το σήμα ελέγχου που χρησιμοποιείται για να επιλέξει μία από τις τιμές εισόδου ενός πολυπλέκτη ως έξοδο του πολυπλέκτη.

Μπορούν να δημιουργηθούν πολυπλέκτες με οποιονδήποτε αριθμό εισόδων δεδομένων. Όταν υπάρχουν μόνο δύο είσοδοι, ο επιλογέας είναι ένα μόνο σήμα που επιλέγει μια από τις εισόδους αν είναι αληθής (1) και την άλλη αν είναι ψευδής (0). Αν υπάρχουν *n* είσοδοι δεδομένων, θα χρειαστούν $\lceil \log_2 n \rceil$ είσοδοι επιλογέα. Στην περίπτωση αυτή, ο πολυπλέκτης αποτελείται βασικά από τρία μέρη:

1. Έναν αποκωδικοποιητή που παράγει *n* σήματα, καθένα από τα οποία σηματοδοτεί μια διαφορετική τιμή εισόδου



ΕΙΚΟΝΑ Γ.3.2 Ένας πολυπλέκτης δύο εισόδων, στα αριστερά, και η υλοποίησή του με πύλες, στα δεξιά. Ο πολυπλέκτης έχει δύο εισόδους δεδομένων (*A* και *B*), που επισημαίνονται με *0* και *1*, και μια είσοδο επιλογέα (*S*), καθώς επίσης και μια έξοδο *C*. Η υλοποίηση των πολυπλεκτών στη Verilog απαιτεί λίγη περισσότερη δουλειά, ειδικά όταν έχουν περισσότερες από δύο εισόδους. Θα δείξουμε πώς γίνεται αυτό από τη σελίδα 34 και μετά.

2. Μια διάταξη από *n* πύλες AND, κάθε μία από τις οποίες συνδυάζει μία από τις εισόδους με ένα σήμα από τον αποκωδικοποιητή
3. Μια μεγάλη πύλη OR που συνδυάζει τις εξόδους από τις πύλες AND

Για τη συσχέτιση των εισόδων με τις τιμές τού επιλογέα, συνήθως δίνουμε αριθμητικές ετικέτες στις εισόδους δεδομένων (π.χ. 0, 1, 2, 3, ..., $n-1$) και ερμηνεύουμε τις εισόδους του επιλογέα δεδομένων ως δυαδικό αριθμό. Μερικές φορές, χρησιμοποιούμε έναν πολυπλέκτη με μη κωδικοποιημένα σήματα επιλογέα.

Οι πολυπλέκτες στη Verilog αναπαρίστανται εύκολα ως συνδυαστικά κυκλώματα με τη χρήση παραστάσεων *if*. Για μεγαλύτερους πολυπλέκτες, οι εντολές *case* είναι πιο βολικές αλλά πρέπει να προσέχετε ώστε να πραγματοποιείται σύνθεση συνδυαστικής λογικής.

Διεπίπεδη λογική και PLA

Όπως δείξαμε στην προηγούμενη ενότητα, κάθε λογική συνάρτηση μπορεί να υλοποιηθεί αποκλειστικά με συναρτήσεις AND, OR, και NOT. Για την ακρίβεια, ισχύει ένα ακόμα ισχυρότερο γεγονός. Κάθε λογική συνάρτηση μπορεί να γραφεί σε κανονική μορφή (canonical form), όπου κάθε είσοδος είναι είτε μια αληθής είτε μια συμπληρωματική μεταβλητή και υπάρχουν μόνο δύο επίπεδα πυλών — ένα που είναι το AND και ένα άλλο που είναι το OR — με μια πιθανή αντιστροφή στο τελικό άθροισμα. Μια τέτοια αναπαράσταση ονομάζεται *διεπίπεδη αναπαράσταση* (two-level representation) και υπάρχουν δύο μορφές, που ονομάζονται **άθροισμα γινομένων** (sum of products) και *γινόμενο αθροισμάτων* (product of sums). Μια αναπαράσταση αθροίσματος γινομένων είναι ένα λογικό άθροισμα (OR) από γινόμενα (όρους που χρησιμοποιούν τον τελεστή AND)· ένα γινόμενο αθροισμάτων είναι απλώς το αντίθετο. Στο προηγούμενο παράδειγμά μας, είχαμε δύο εξισώσεις για την έξοδο *E*:

άθροισμα γινομένων (sum of products) Μια μορφή λογικής αναπαράστασης που περιλαμβάνει ένα λογικό άθροισμα (OR) από γινόμενα (όρους που σχηματίζονται με έναν τελεστή AND).

$$E = ((A \cdot B) + (A \cdot C) + (B \cdot C)) \cdot (\overline{A \cdot B \cdot C})$$

και

$$E = (A \cdot B \cdot \overline{C}) + (A \cdot C \cdot \overline{B}) + (B \cdot C \cdot \overline{A})$$

Η δεύτερη αυτή εξίσωση είναι σε μια μορφή αθροίσματος γινομένων (sum-of-products): έχει δύο επίπεδα λογικής και οι μόνες αντιστροφές εφαρμόζονται σε μεμονωμένες μεταβλητές. Η πρώτη εξίσωση έχει τρία επίπεδα λογικής.

Επιπλέον ανάπτυξη: Μπορούμε επίσης να γράψουμε την *E* ως γινόμενο αθροισμάτων:

$$E = \overline{(\overline{A} + \overline{B} + C) \cdot (\overline{A} + \overline{C} + B) \cdot (\overline{B} + \overline{C} + A)}$$

Για να παραχθεί αυτή η μορφή, χρειάζεται να χρησιμοποιήσετε τα *θεωρήματα του DeMorgan*, που εξετάζονται στις ασκήσεις.

Σε αυτό το βιβλίο χρησιμοποιούμε τη μορφή αθροίσματος γινομένων. Είναι εύκολο να δούμε ότι κάθε λογική συνάρτηση μπορεί να αναπαρασταθεί ως ένα άθροισμα γινομένων, κατασκευάζοντας μια τέτοια αναπαράσταση από τον πίνακα αληθείας της συνάρτησης. Κάθε καταχώριση του πίνακα αληθείας για την οποία η συνάρτηση είναι αληθής αντιστοιχεί σε έναν όρο γινομένου. Ο όρος γινομένου αποτελείται από ένα λογικό γινόμενο όλων των εισόδων ή των συμπληρωμάτων των εισόδων, ανάλογα με το αν στην καταχώριση της συγκεκριμένης μεταβλητής στον πίνακα αληθείας αντιστοιχεί 0 ή 1. Η λογική συνάρτηση είναι το λογικό άθροισμα των όρων γινομένου όπου η συνάρτηση είναι αληθής. Αυτό φαίνεται πιο εύκολα με ένα παράδειγμα.

ΠΑΡΑΔΕΙΓΜΑ

Άθροισμα γινομένων

Δώστε την αναπαράσταση αθροίσματος γινομένων για τον παρακάτω πίνακα αληθείας της *D*.

Είσοδοι			Έξοδος
A	B	C	D
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

ΑΠΑΝΤΗΣΗ

Υπάρχουν τέσσερις όροι γινομένου, αφού η συνάρτηση είναι αληθής (1) για τέσσερις διαφορετικούς συνδυασμούς εισόδων. Αυτοί είναι:

$$\overline{A} \cdot \overline{B} \cdot C$$
$$\overline{A} \cdot B \cdot \overline{C}$$
$$A \cdot \overline{B} \cdot \overline{C}$$
$$A \cdot B \cdot C$$

Έτσι, μπορούμε να γράψουμε τη συνάρτηση D ως άθροισμα αυτών των όρων:

$$D = (\overline{A} \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot \overline{C}) + (A \cdot \overline{B} \cdot \overline{C}) + (A \cdot B \cdot C)$$

Σημειώστε ότι όρους στην εξίσωση δημιουργούν μόνον εκείνες οι καταχωρίσεις του πίνακα αληθείας για τις οποίες η συνάρτηση είναι αληθής.

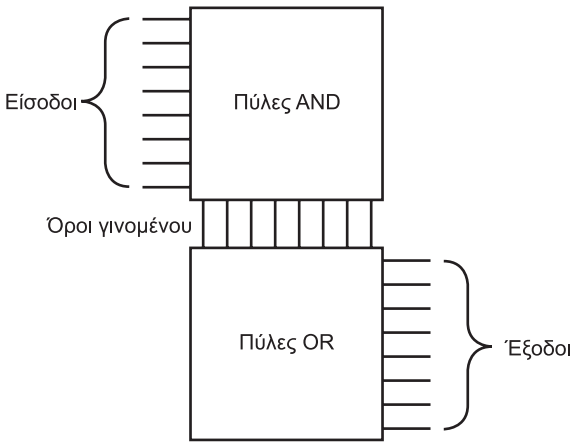
Μπορούμε να χρησιμοποιήσουμε αυτή τη σχέση μεταξύ του πίνακα αληθείας και μιας διεπίπεδης υλοποίησης για να δημιουργήσουμε μια υλοποίηση οποιουδήποτε συνόλου λογικών συναρτήσεων σε επίπεδο πυλών. Ένα σύνολο λογικών συναρτήσεων αντιστοιχεί σε έναν πίνακα αληθείας με πολλές στήλες εξόδων, όπως είδαμε στο παράδειγμα της σελίδας 16. Κάθε στήλη εξόδου αντιπροσωπεύει μια διαφορετική λογική συνάρτηση, η οποία μπορεί να κατασκευαστεί απευθείας από τον πίνακα αληθείας.

Η αναπαράσταση αθροίσματος γινομένων αντιστοιχεί σε μια συνηθισμένη υλοποίηση δομημένης λογικής που ονομάζεται **προγραμματίσιμος λογικός πίνακας** (programmable logic array — PLA). Ένα PLA διαθέτει ένα σύνολο εισόδων και αντίστοιχων συμπληρωματικών εισόδων (που μπορεί να υλοποιηθεί με ένα σύνολο από αντιστροφείς), και δύο στάδια λογικής. Το πρώτο στάδιο είναι μια διάταξη από πύλες AND οι οποίες σχηματίζουν ένα σύνολο από **όρους γινομένου** (που μερικές φορές ονομάζονται **ελαχιστόροι** — minterms)· κάθε όρος γινομένου μπορεί να αποτελείται από οποιεσδήποτε από τις εισόδους ή τα συμπληρώματά τους. Το δεύτερο στάδιο είναι μια διάταξη πυλών OR, κάθε μία από τις οποίες σχηματίζει ένα λογικό άθροισμα οποιουδήποτε αριθμού όρων γινομένου. Η Εικόνα Γ.3.3 δείχνει τη βασική μορφή ενός PLA.

προγραμματίσιμος λογικός πίνακας (programmable logic array — PLA) Στοιχείο δομημένης λογικής που αποτελείται από ένα σύνολο εισόδων και αντίστοιχων συμπληρωματικών εισόδων και δύο στάδια λογικής: το πρώτο παράγει όρους γινομένου των εισόδων και των συμπληρωμάτων των εισόδων και το δεύτερο παράγει όρους αθροίσματος των όρων γινομένων. Έτσι, τα PLA υλοποιούν λογικές συναρτήσεις ως αθροίσματα γινομένων.

ελαχιστόροι (minterms) Ονομάζονται επίσης **όροι γινομένου**. Ένα σύνολο από λογικές εισόδους που συνδέονται με σύζευξη (λειτουργίες AND)· οι όροι γινομένου σχηματίζουν το πρώτο στάδιο λογικής του προγραμματίσιμου λογικού πίνακα (programmable logic array — PLA).

Ένα PLA μπορεί να υλοποιήσει απευθείας τον πίνακα αληθείας ενός συνόλου λογικών συναρτήσεων με πολλές εισόδους και εξόδους. Αφού κάθε καταχώριση στην οποία ο πίνακας αληθείας είναι αληθής απαιτεί έναν όρο γινομένου, θα υπάρχει μια αντίστοιχη γραμμή στο PLA. Κάθε έξοδος αντιστοιχεί σε μια πιθανή γραμμή από πύλες OR στο δεύτερο στάδιο. Ο αριθμός των πυλών OR αντιστοιχεί στον αριθμό των καταχωρίσεων του πίνακα αληθείας για τις οποίες η έξοδος είναι αληθής. Το συνολικό μέγεθος ενός PLA, σαν αυτό που βλέπετε στην Εικόνα Γ.3.3, είναι ίσο με το άθροισμα του μεγέθους της διάταξης των πυλών AND (που ονομάζεται *επίπεδο AND* — AND plane) και του μεγέθους της διάταξης των πυλών OR (που ονομάζεται *επίπεδο OR* — OR plane). Στην Εικόνα Γ.3.3, μπορούμε να δούμε ότι το μέγεθος της διάταξης των πυλών AND είναι ίσο με τον αριθμό των εισόδων επί τον αριθμό των διαφορετικών όρων γινομένου, και το μέγεθος της διάταξης των πυλών OR είναι ο αριθμός των εξόδων επί τον αριθμό των όρων γινομένου.



ΕΙΚΟΝΑ Γ.3.3 Η βασική μορφή ενός PLA αποτελείται από έναν πίνακα πυλών AND ακολουθούμενο από έναν πίνακα πυλών OR. Κάθε καταχώριση στον πίνακα των πυλών AND είναι ένας όρος γινομένου που αποτελείται από οποιονδήποτε αριθμό εισόδων ή συμπληρωματικών εισόδων. Κάθε καταχώριση στον πίνακα των πυλών OR είναι ένας όρος αθροίσματος που αποτελείται από οποιονδήποτε αριθμό αυτών των όρων γινομένου.

Ένα PLA έχει δύο χαρακτηριστικά που του δίνουν τη δυνατότητα να αποτελεί έναν αποδοτικό τρόπο υλοποίησης ενός συνόλου συναρτήσεων. Πρώτον, μόνον οι καταχωρίσεις του πίνακα αληθείας που παράγουν μια αληθή τιμή για τουλάχιστον μία έξοδο συσχετίζονται με λογικές πύλες. Δεύτερον, κάθε διαφορετικός όρος γινομένου θα έχει μόνο μία καταχώριση στο PLA, ακόμη και αν ο όρος γινομένου χρησιμοποιείται σε πολλές εξόδους. Ας δούμε ένα παράδειγμα.

ΠΑΡΑΔΕΙΓΜΑ

ΑΠΑΝΤΗΣΗ

PLA

Θεωρήστε το σύνολο των λογικών συναρτήσεων που ορίζονται στο παράδειγμα της σελίδας 16. Δώστε μια υλοποίηση με PLA αυτού του παραδείγματος για τις D, E, και F.

Ο πίνακας αληθείας που κατασκευάσαμε προηγουμένως είναι:

Είσοδοι			Έξοδοι		
A	*B*	*C*	*D*	*E*	*F*
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	0	1

Αφού υπάρχουν επτά μοναδικοί όροι γινομένου με τουλάχιστον μία αληθή τιμή στο τμήμα της εξόδου, θα υπάρχουν επτά στήλες στο επίπεδο AND. Ο αριθμός των γραμμών στο επίπεδο AND είναι τρεις (αφού υπάρχουν τρεις είσοδοι), και επίσης υπάρχουν τρεις γραμμές στο επίπεδο OR (αφού υπάρχουν τρεις έξοδοι). Η Εικόνα Γ.3.4 παρουσιάζει το PLA που προκύπτει, με τους όρους γινομένου οι οποίοι αντιστοιχούν στις καταχωρίσεις του πίνακα αληθείας από επάνω προς τα κάτω.

Αντί να απεικονίσουν όλες τις πύλες, όπως κάναμε στην Εικόνα Γ.3.4, οι σχεδιαστές συχνά δείχνουν μόνο τη θέση των πυλών AND και των πυλών OR. Χρησιμοποιούνται κουκκίδες στην τομή της γραμμής σήματος ενός όρου γινομένου και μιας γραμμής εισόδου ή μιας γραμμής εξόδου όταν απαιτείται μια αντίστοιχη πύλη AND ή πύλη OR. Η Εικόνα Γ.3.5 δείχνει πώς γίνεται το PLA της Εικόνας Γ.3.4 όταν σχεδιάζεται με αυτόν τον τρόπο. Τα περιεχόμενα του PLA είναι σταθερά κατά τη δημιουργία του, παρόλο που υπάρχουν επίσης πολλές μορφές δομών οι οποίες μοιάζουν με PLA και ονομάζονται *PAL*, που μπορούν να προγραμματιστούν ηλεκτρονικά όταν ένας σχεδιαστής είναι έτοιμος να τις χρησιμοποιήσει.

Μνήμες ROM

Μια άλλη μορφή δομημένης λογικής που μπορεί να χρησιμοποιηθεί για την υλοποίηση ενός συνόλου λογικών συναρτήσεων είναι η **μνήμη μόνο για ανάγνωση** (read-only memory — ROM). Μια ROM ονομάζεται μνήμη επειδή έχει ένα σύνολο από θέσεις που μπορούν να αναγνωσθούν· ωστόσο, τα περιεχόμενα αυτών των θέσεων είναι σταθερά, συνήθως από το χρόνο κατασκευής τής ROM. Υπάρχουν επίσης **προγραμματίσιμες ROM** (programmable ROM — PROM) που μπορούν να προγραμματιστούν ηλεκτρονικά, όταν ένας σχεδιαστής γνωρίζει τα περιεχόμενά τους. Υπάρχουν επίσης απαλείψιμες PROM (erasable PROM)· αυτές οι συσκευές απαιτούν μια αργή διεργασία διαγραφής που χρησιμοποιεί υπεριώδες φως, και συνεπώς χρησιμοποιούνται ως μνήμες μόνο για ανάγνωση, εκτός από τα στάδια σχεδίασης και αποσφαλμάτωσης.

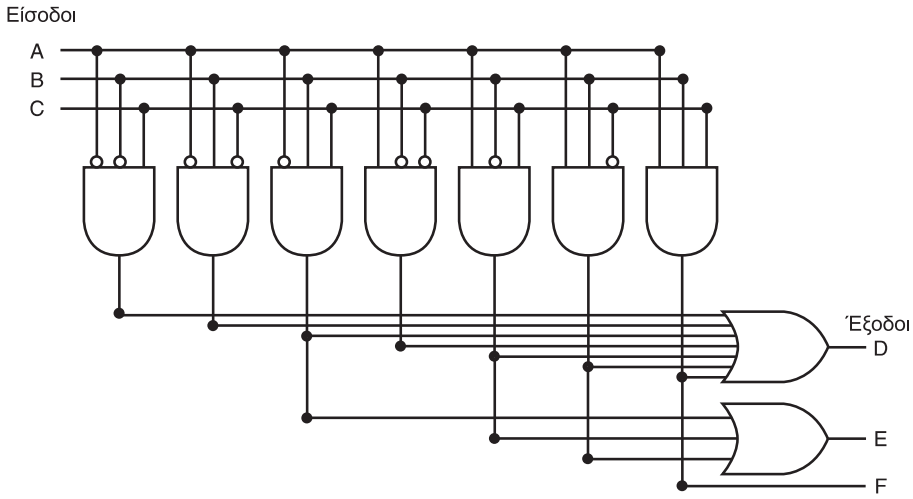
μνήμη μόνο για ανάγνωση (read-only memory — ROM) Μια μνήμη της οποίας τα περιεχόμενα καθορίζονται κατά την κατασκευή της, και μετά μπορούν μόνο να αναγνωσθούν. Η ROM χρησιμοποιείται ως δομημένη λογική για την υλοποίηση ενός συνόλου λογικών συναρτήσεων με τη χρήση των όρων που περιέχονται στις λογικές συναρτήσεις ως εισόδων διευθύνσεων και των εξόδων ως bit, σε κάθε λέξη της μνήμης.

προγραμματίσιμη ROM (programmable ROM — PROM) Μορφή μνήμης μόνο για ανάγνωση που μπορεί να προγραμματιστεί όταν ο σχεδιαστής γνωρίζει τα περιεχόμενά της.

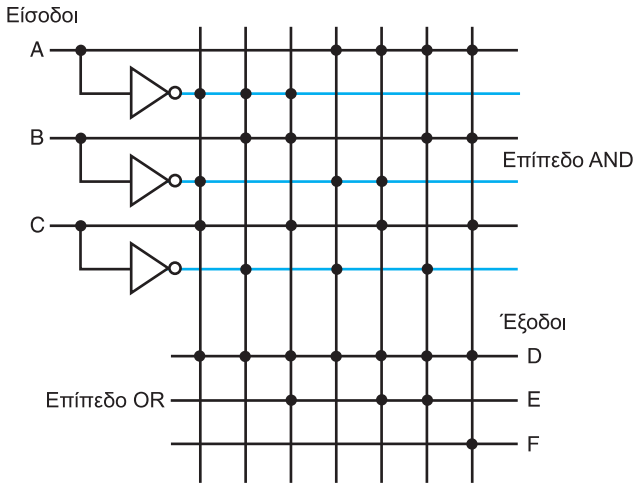
Μια ROM διαθέτει ένα σύνολο γραμμών εισόδου διευθύνσεων και ένα σύνολο εξόδων. Ο αριθμός των προσπελάσιμων καταχωρίσεων στη ROM καθορίζει τον αριθμό των γραμμών διευθύνσεων: αν η ROM περιέχει 2^m προσπελάσιμες καταχωρίσεις, που ονομάζεται *ύψος* (height), τότε υπάρχουν m γραμμές εισόδου. Ο αριθμός των bit σε κάθε προσπελάσιμη καταχώριση είναι ίσος με τον αριθμό των bit εξόδου και μερικές φορές ονομάζεται *πλάτος* (width) της ROM. Ο συνολικός αριθμός bit στη ROM είναι ίσος με το ύψος επί το πλάτος. Το ύψος και το πλάτος μερικές φορές αναφέρονται συνολικά ως *σχήμα* (shape) της ROM.

Μια ROM μπορεί να κωδικοποιήσει μια συλλογή λογικών συναρτήσεων απευθείας από τον πίνακα αληθείας. Για παράδειγμα, αν υπάρχουν n συναρτήσεις με m εισόδους, χρειαζόμαστε μια ROM με m γραμμές διευθύνσεων (και 2^m καταχωρίσεις), όπου κάθε καταχώριση έχει πλάτος n bit. Οι κατα-

χωρίσεις στο τμήμα εισόδου του πίνακα αληθείας αντιπροσωπεύουν τις διευθύνσεις των καταχωρίσεων της ROM, ενώ τα περιεχόμενα του τμήματος εξόδου του πίνακα αληθείας συνιστούν τα περιεχόμενα της ROM. Αν ο πίνακας αληθείας είναι οργανωμένος έτσι ώστε η ακολουθία των καταχωρίσεων στο τμήμα εισόδου να συνιστά μια ακολουθία δυαδικών αριθμών (όπως συμβαίνει με όλους τους πίνακες αληθείας που έχουμε δει μέχρι τώρα), τότε το τμήμα εξόδου δίνει τα περιεχόμενα της ROM επίσης με τη σειρά.



ΕΙΚΟΝΑ Γ.3.4 Το PLA για την υλοποίηση της λογικής συνάρτησης που περιγράψαμε στο παράδειγμα.



ΕΙΚΟΝΑ Γ.3.5 Ένα PLA σχεδιασμένο με τη χρήση κουκκίδων για να επισημανθούν τα συστατικά των όρων γινομένου και των όρων αθροίσματος στον πίνακα. Αντί να χρησιμοποιούνται αντιστροφείς στις πύλες, συνήθως όλες οι είσοδοι διατρέχουν όλο το πλάτος του επιπέδου AND, τόσο στην κανονική όσο και στη συμπληρωματική μορφή. Μια κουκκίδα στο επίπεδο AND σημαίνει ότι η είσοδος ή η αντίστροφή της εμφανίζεται στον όρο γινομένου. Μια κουκκίδα στο επίπεδο OR σημαίνει ότι ο αντίστοιχος όρος γινομένου εμφανίζεται στην αντίστοιχη έξοδο.

Στο προηγούμενο παράδειγμα, που ξεκινά στη σελίδα 24, υπήρχαν τρεις είσοδοι και τρεις έξοδοι. Αυτό οδηγεί σε μια ROM με $2^3 = 8$ καταχωρίσεις, κάθε μία με πλάτος 3 bit. Τα περιεχόμενα αυτών των καταχωρίσεων σε αύξουσα σειρά διευθύνσεων δίνονται απευθείας από το τμήμα εξόδου του πίνακα αληθείας που φαίνεται στη σελίδα 24.

Οι ROM και τα PLA έχουν στενή σχέση μεταξύ τους. Μια ROM είναι πλήρως αποκωδικοποιημένη: περιέχει μια πλήρη λέξη εξόδου για κάθε πιθανό συνδυασμό εισόδων. Ένα PLA είναι μόνο μερικώς αποκωδικοποιημένο. Αυτό σημαίνει ότι μια ROM περιέχει πάντα περισσότερες καταχωρίσεις. Για τον προηγούμενο πίνακα αληθείας της σελίδας 24, η ROM περιέχει καταχωρίσεις και για τις οκτώ πιθανές εισόδους, ενώ το PLA περιέχει μόνο τους επτά ενεργούς όρους γινομένου. Καθώς ο αριθμός των εισόδων αυξάνεται, ο αριθμός των καταχωρίσεων στη ROM αυξάνεται εκθετικά. Αντίθετα, για τις περισσότερες πραγματικές λογικές συναρτήσεις, ο αριθμός των όρων γινομένου αυξάνεται πολύ πιο αργά (δείτε τα παραδείγματα στο Παράρτημα Δ). Αυτή η διαφορά κάνει τα PLA γενικά πιο αποδοτικά για την υλοποίηση συνδυαστικών λογικών συναρτήσεων. Οι ROM έχουν το πλεονέκτημα ότι είναι ικανές να υλοποιήσουν οποιαδήποτε λογική συνάρτηση με τον κατάλληλο αριθμό εισόδων και εξόδων. Αυτό το πλεονέκτημα διευκολύνει την τροποποίηση των περιεχομένων τής ROM αν αλλάξει η λογική συνάρτηση, αφού το μέγεθος της ROM δε χρειάζεται να αλλάξει.

Εκτός από τις ROM και τα PLA τα σύγχρονα συστήματα λογικής σύνθεσης μετατρέπουν επίσης μικρά μπλοκ συνδυαστικής λογικής σε μια συλλογή από πύλες που μπορούν να τοποθετηθούν και να συνδεθούν αυτόματα. Παρόλο που μερικές μικρές συλλογές πυλών δεν είναι συνήθως αποδοτικές από πλευράς επιφάνειας, για μικρές λογικές συναρτήσεις έχουν μικρότερη επιβάρυνση από την αυστηρή δομή μιας ROM και ενός PLA και, συνεπώς, προτιμώνται.

Για τη σχεδίαση λογικής έξω από ένα προσαρμοσμένο (custom) ή ημιπροσαρμοσμένο (semi-custom) ολοκληρωμένο κύκλωμα, μια συνηθισμένη επιλογή είναι κάποια επιτόπου προγραμματίσιμη συσκευή (field programmable device)· περιγράφουμε αυτές τις συσκευές στην Ενότητα Γ.12.

Αδιάφοροι όροι

Συχνά, στην υλοποίηση μιας συνδυαστικής λογικής υπάρχουν περιπτώσεις όπου αδιαφορούμε για την τιμή κάποιας εξόδου, είτε επειδή μια άλλη έξοδος είναι αληθής είτε επειδή ένα υποσύνολο των συνδυασμών εισόδων καθορίζει τις τιμές των εξόδων. Τέτοιες περιπτώσεις αναφέρονται ως *αδιάφοροι όροι* (don't cares). Οι αδιάφοροι όροι είναι σημαντικοί επειδή διευκολύνουν τη βελτιστοποίηση της υλοποίησης μιας λογικής συνάρτησης.

Υπάρχουν δύο τύποι αδιάφορων όρων: αδιάφοροι όροι εξόδου και αδιάφοροι όροι εισόδου, που και οι δύο μπορούν να αναπαρασταθούν σε έναν πίνακα αληθείας. Οι *αδιάφοροι όροι εξόδου* εμφανίζονται όταν δεν ενδιαφερόμαστε για την τιμή μιας εξόδου για κάποιο συνδυασμό εισόδων. Εμφανίζονται ως X στο τμήμα εξόδου του πίνακα αληθείας. Όταν μια έξοδος είναι αδιάφορος όρος για κάποιο συνδυασμό εισόδων, ο σχεδιαστής ή το πρό-

γραμμα λογικής βελτιστοποίησης είναι ελεύθεροι να κάνουν την έξοδο αληθή ή ψευδή γι' αυτόν το συνδυασμό εισόδων. Οι *αδιάφοροι όροι εισόδου* εμφανίζονται όταν μια έξοδος εξαρτάται μόνον από μερικές εισόδους, και εμφανίζονται πάλι ως X, αλλά στο τμήμα εισόδου τού πίνακα αληθείας.

ΠΑΡΑΔΕΙΓΜΑ

Αδιάφοροι όροι

Θεωρήστε μια λογική συνάρτηση με εισόδους *A*, *B*, και *C* που ορίζεται ως εξής:

- Αν το *A* ή το *C* είναι αληθές, τότε η έξοδος *D* είναι αληθής ανεξάρτητα από την τιμή του *B*.
- Αν το *A* ή το *B* είναι αληθές, τότε η έξοδος *E* είναι αληθής ανεξάρτητα από την τιμή του *C*.
- Η έξοδος *F* είναι αληθής όταν ακριβώς μία από τις εισόδους είναι αληθής, παρόλο που δεν ενδιαφερόμαστε για την τιμή της *F* όταν η *D* και η *E* είναι και οι δύο αληθείς.

Δώστε τον πλήρη πίνακα αληθείας αυτής της συνάρτησης και τον πίνακα αληθείας με τη χρήση αδιάφορων όρων. Πόσοι όροι γινομένου απαιτούνται σε ένα PLA για κάθε μία από αυτές;

ΑΠΑΝΤΗΣΗ

Ο πλήρης πίνακας αληθείας χωρίς αδιάφορους όρους είναι:

Είσοδοι			Έξοδοι		
A	*B*	*C*	*D*	*E*	*F*
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	0	1	1
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	1	0

Αυτός απαιτεί επτά όρους γινομένου χωρίς βελτιστοποίηση. Ο πίνακας αληθείας γραμμένος με αδιάφορους όρους εξόδου είναι ως εξής:

Είσοδοι			Έξοδοι		
A	*B*	*C*	*D*	*E*	*F*
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	0	1	1
0	1	1	1	1	X
1	0	0	1	1	X
1	0	1	1	1	X
1	1	0	1	1	X
1	1	1	1	1	X

Αν χρησιμοποιήσουμε επίσης τους αδιάφορους όρους εισόδου, αυτός ο πίνακας αληθείας μπορεί να απλοποιηθεί περισσότερο και να οδηγήσει στον εξής:

Είσοδοι			Έξοδοι		
A	*B*	*C*	*D*	*E*	*F*
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	0	1	1
X	1	1	1	1	X
1	X	X	1	1	X

Αυτός ο απλοποιημένος πίνακας αληθείας απαιτεί ένα PLA με τέσσερις ελαχιστόρους, ή μπορεί να υλοποιηθεί με διακριτές πύλες· μία πύλη AND των δύο εισόδων και τρεις πύλες OR (δύο με τρεις εισόδους και μία με δύο εισόδους). Αυτό είναι συγκρίσιμο με τον αρχικό πίνακα αληθείας που είχε επτά ελαχιστόρους και θα απαιτούσε τέσσερις πύλες AND.

Η λογική ελαχιστοποίηση (logic minimization) είναι κρίσιμη για την επίτευξη αποδοτικών υλοποιήσεων. Ένα χρήσιμο εργαλείο για την ελαχιστοποίηση της τυχαίας λογικής «με το χέρι» είναι οι *χάρτες Karnaugh* (Karnaugh maps). Οι χάρτες Karnaugh αναπαριστούν γραφικά τον πίνακα αληθείας έτσι, ώστε οι όροι γινομένου που μπορούν να συνδυαστούν να φαίνονται εύκολα. Παρόλα αυτά, η βελτιστοποίηση σημαντικών λογικών συναρτήσεων με το χέρι, με τη χρήση χαρτών Karnaugh δεν είναι πρακτική, τόσο λόγω του μεγέθους των χαρτών όσο και της πολυπλοκότητάς τους. Ευτυχώς, η διαδικασία της λογικής ελαχιστοποίησης είναι σε μεγάλο βαθμό μηχανική και μπορεί να εκτελεστεί από εργαλεία σχεδιασμού. Στη διαδικασία της ελαχιστοποίησης, τα εργαλεία εκμεταλλεύονται τους αδιάφορους όρους και, συνεπώς, ο καθορισμός τους είναι σημαντικός. Στις βιβλιογραφικές αναφορές στο τέλος αυτού του παραρτήματος, θα βρείτε περαιτέρω ανάλυση της λογικής ελαχιστοποίησης, των χαρτών Karnaugh, και της θεωρίας που βρίσκεται πίσω από τέτοιους αλγορίθμους ελαχιστοποίησης.

Διατάξεις λογικών στοιχείων

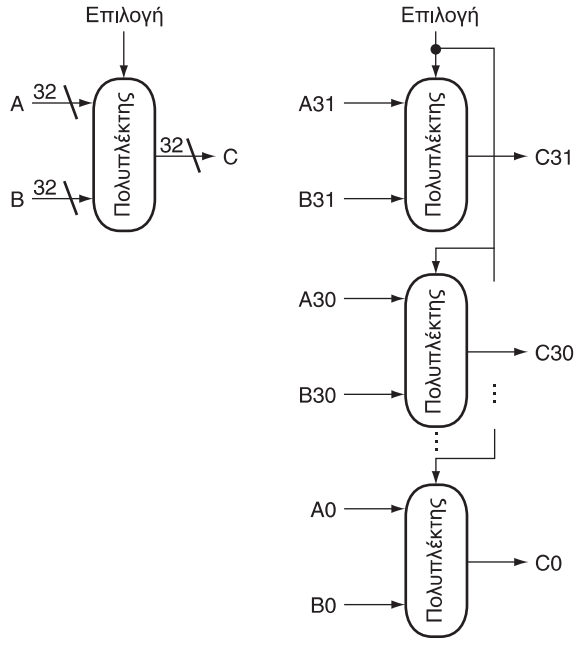
Πολλές από τις συνδυαστικές λειτουργίες που εκτελούνται σε δεδομένα πρέπει να εφαρμοστούν σε μια ολόκληρη λέξη δεδομένων (32 bit). Έτσι, συχνά χρειάζεται να κατασκευάσουμε μια διάταξη λογικών στοιχείων, την οποία μπορούμε να αναπαραστήσουμε απλώς δείχνοντας ότι μια δεδομένη λειτουργία θα εφαρμοστεί σε μια ολόκληρη συλλογή εισόδων. Για παράδειγμα, είδαμε στη σελίδα 20 πώς μοιάζει ένας πολυπλέκτης του 1 bit αλλά, μέσα σε μια μηχανή, τον περισσότερο χρόνο χρειάζεται να επιλέγουμε μεταξύ ενός ζεύγους *διαύλων*. Ένας **δίαυλος** (bus) είναι μια συλλογή γραμμών δεδομένων που αντιμετωπίζονται μαζί ως ένα μοναδικό λογικό σήμα. (Ο όρος *δίαυλος* χρησιμοποιείται επίσης για να δείξει μια κοινόχρηστη συλλο-

δίαυλος (bus) Στη λογική σχεδίαση, μια συλλογή γραμμών δεδομένων που αντιμετωπίζονται μαζί σαν ένα μοναδικό λογικό σήμα· επίσης, μια κοινόχρηστη συλλογή γραμμών με πολλές προελεύσεις και χρήσεις.

γή γραμμών με πολλές προελεύσεις και χρήσεις, ειδικά στο Κεφάλαιο 6 στο οποίο εξετάστηκαν οι δίαυλοι εισόδου/εξόδου.)

Για παράδειγμα, στο σύνολο εντολών του MIPS το αποτέλεσμα μιας εντολής που γράφεται σε έναν καταχωρητή μπορεί να προέρχεται από μία από δύο προελεύσεις. Ένας πολυπλέκτης χρησιμοποιείται για να επιλέξει ποιος από τους δύο διαύλους (καθένας με πλάτος 32 bit) θα γραφεί στον καταχωρητή του αποτελέσματος. Ο πολυπλέκτης του 1 bit, που δείξαμε προηγουμένως, θα χρειαστεί να επαναληφθεί 32 φορές.

Επισημαίνουμε ότι ένα σήμα είναι ένας δίαυλος και όχι μία μοναδική γραμμή του 1 bit, δείχνοντάς το σε ένα σχήμα με μια πιο παχιά γραμμή. Οι περισσότεροι δίαυλοι έχουν πλάτος 32 bit· αυτοί που δεν έχουν ετικέτες με το πλάτος τους. Όταν δείχνουμε μια λογική μονάδα της οποίας οι είσοδοι και οι έξοδοι είναι δίαυλοι, αυτό σημαίνει ότι η μονάδα πρέπει να αναπαραχθεί αρκετές φορές ώστε να καλύψει το πλάτος τής εισόδου. Η Εικόνα Γ.3.6 δείχνει πώς σχεδιάζουμε έναν πολυπλέκτη που επιλέγει ανάμεσα σε ένα ζεύγος διαύλων των 32 bit και πώς αυτό επεκτείνεται σε πολυπλέκτες του 1 bit. Μερικές φορές, χρειάζεται να κατασκευάσουμε μια διάταξη λογικών στοιχείων στην οποία οι είσοδοι μερικών στοιχείων στη διάταξη είναι έξοδοι από προηγούμενα στοιχεία. Για παράδειγμα, έτσι κατασκευάζεται μια ALU με πλάτος πολλών bit. Σε τέτοιες περιπτώσεις, πρέπει να δείξουμε με



α. Ένας πολυπλέκτης 2 σε 1 με πλάτος 32 bit

β. Ο πολυπλέκτης πλάτους 32 bit είναι στην πραγματικότητα μια διάταξη από 32 πολυπλέκτες του 1 bit.

ΕΙΚΟΝΑ Γ.3.6 Ένας πολυπλέκτης παρατάσσεται 32 φορές για να εκτελέσει μια επιλογή μεταξύ δύο εισόδων των 32 bit. Παρατηρήστε ότι εξακολουθεί να υπάρχει μόνον ένα σήμα επιλογής δεδομένων που χρησιμοποιείται σε όλους τους 32 πολυπλέκτες του 1 bit.

σαφήνεια πώς να κατασκευαστούν διατάξεις μεγαλύτερου πλάτους, αφού τα μεμονωμένα στοιχεία της διάταξης δεν είναι πλέον ανεξάρτητα, όπως στην περίπτωση ενός πολυπλέκτη με πλάτος 32 bit.

Αυτοεξέταση

Η ισοτιμία (parity) είναι μια συνάρτηση στην οποία η έξοδος εξαρτάται από τον αριθμό των 1 στην είσοδο. Για μια συνάρτηση άρτιας ισοτιμίας, η έξοδος είναι 1 αν η είσοδος έχει άρτιο αριθμό μονάδων (1). Υποθέστε ότι χρησιμοποιείται μια ROM για την υλοποίηση μιας συνάρτησης άρτιας ισοτιμίας με είσοδο των 4 bit. Ποιο από τα A, B, C, και D αντιπροσωπεύει τα περιεχόμενα της ROM;

Διεύθυνση	A	B	C	D
0	0	1	0	1
1	0	1	1	0
2	0	1	1	1
3	0	1	0	0
4	0	1	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	1	0	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	0	1
13	1	0	1	0
14	1	0	1	1
15	1	0	0	0

Γ.4 Χρήση μιας γλώσσας περιγραφής υλικού

γλώσσα περιγραφής υλικού (hardware description language) Μια γλώσσα προγραμματισμού για την περιγραφή υλικού, που χρησιμοποιείται στη δημιουργία προσομοιώσεων μιας σχεδίασης υλικού και ως είσοδος σε εργαλεία σύνθεσης τα οποία μπορούν να παραγάγουν πραγματικό υλικό.

Σήμερα, το μεγαλύτερο μέρος τής ψηφιακής σχεδίασης των επεξεργαστών και του σχετικού συστήματος υλικού γίνεται με τη βοήθεια μιας **γλώσσας περιγραφής υλικού** (hardware description language). Μια τέτοια γλώσσα εξυπηρετεί δύο σκοπούς. Πρώτον, παρέχει μια αφηρημένη περιγραφή του υλικού για την προσομοίωση και την αποσφαλμάτωση της σχεδίασης. Δεύτερον, με τη χρήση εργαλείων λογικής σύνθεσης και μετάφρασης σε υλικό, αυτή η περιγραφή μπορεί να μεταφραστεί στην υλοποίηση του υλικού.

Σε αυτή την ενότητα, εισάγουμε τη γλώσσα περιγραφής υλικού Verilog και δείχνουμε πώς μπορεί να χρησιμοποιηθεί στη συνδυαστική σχεδίαση. Στο υπόλοιπο του παραρτήματος, επεκτείνουμε τη χρήση της Verilog για να συμπεριλάβουμε τη σχεδίαση ακολουθιακής λογικής. Στις προαιρετικές ενότητες του Κεφαλαίου 4, οι οποίες υπάρχουν στο Παράρτημα Z, χρησιμοποιούμε τη Verilog για να περιγράψουμε υλοποιήσεις επεξεργαστών. Στην προαιρετική ενότητα του Κεφαλαίου 5, η οποία υπάρχει στο Παράρτημα Z,

ΤΟΜΟΣ Β΄

ΟΡΓΑΝΩΣΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΥΠΟΛΟΓΙΣΤΩΝ

Η διασύνδεση υλικού και λογισμικού

DAVID A. PATTERSON και JOHN L. HENNESSY

Με ειδικό πρόλογο του D.A. Patterson για την ελληνική έκδοση

ΜΕΤΑΦΡΑΣΗ & ΕΠΙΣΤΗΜΟΝΙΚΗ ΕΠΙΜΕΛΕΙΑ
ΔΗΜΗΤΡΗΣ ΓΚΙΖΟΠΟΥΛΟΣ, *ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ*

"Οι Patterson και Hennessy βελτίωσαν σε μεγάλο βαθμό αυτό που ήταν ήδη το χρυσό πρότυπο των βιβλίων. Στο ταχέως εξελισσόμενο πεδίο της αρχιτεκτονικής υπολογιστών, έχουν συγκεντρώσει έναν εντυπωσιακό αριθμό πρόσφατων μελετών περιπτώσεων και συγχρόνων θεμάτων σε ένα πλαίσιο θεμελιωδών γνώσεων των οποίων η αξία έχει αποδειχθεί στο χρόνο."

—Fred Chong, *University of California at Santa Barbara*

"Η νέα κάλυψη των πολυεπεξεργαστών και της παραλληλίας είναι εφαμίλλη του υψηλού επιπέδου αυτού του καλογραμμένου κλασικού βιβλίου. Παρέχει καλά τεκμηριωμένες, σταδιακές εισαγωγές στα νέα θέματα, καθώς και πολλές λεπτομέρειες και παραδείγματα που πηγαζουν από το τρέχον υλικό των υπολογιστών."

—John Greiner, *Rice University*

Το βιβλίο οργάνωσης υπολογιστών με τις μεγαλύτερες πωλήσεις σε όλο τον κόσμο έχει ενημερωθεί πλήρως ώστε να εστιάζει πλέον στην επαναστατική αλλαγή που συμβαίνει σήμερα στη βιομηχανία: τη μετάβαση από τους μονοεπεξεργαστές στους πολυπύρηνους μικροεπεξεργαστές. Αυτή η νέα έμφαση στην παραλληλία υποστηρίζεται από το ενημερωμένο περιεχόμενο του βιβλίου, το οποίο αντανακλά τις νεότερες τεχνολογίες με παραδείγματα που υπογραμμίζουν τις πρόσφατες σχεδιάσεις επεξεργαστών και τα πρότυπα μετροπρογραμμάτων. Όπως και στις προηγούμενες εκδόσεις του βιβλίου, ένας επεξεργαστής MIPS είναι ο πυρήνας που χρησιμοποιείται για την παρουσίαση των θεμελιωδών αρχών των τεχνολογιών του υλικού, της συμβολικής γλώσσας, της αριθμητικής υπολογιστών, της διοχέτευσης, των ιεραρχιών μνήμης, και της εισόδου/εξόδου. Συμπεριλαμβάνονται επίσης ενότητες για τις αρχιτεκτονικές ARM και x86.

Στην ιστοσελίδα της ελληνικής έκδοσης του βιβλίου παρέχεται μια εργαλειοθήκη με προσομοιωτές και μεταγλωττιστές, καθώς και ηλεκτρονικά εκπαιδευτικά εγχειρίδια (στα Αγγλικά) για τη χρήση τους.

ΚΥΡΙΟΤΕΡΕΣ ΠΡΟΣΘΗΚΕΣ ΤΗΣ ΤΕΤΑΡΤΗΣ ΑΜΕΡΙΚΑΝΙΚΗΣ ΕΚΔΟΣΗ

- Η επαναστατική αλλαγή από την ακολουθιακή στην παράλληλη υπολογιστική, με ένα νέο κεφάλαιο σχετικά με την παραλληλία και ενότητες σε κάθε κεφάλαιο που υπογραμμίζουν θέματα του παράλληλου υλικού και του παράλληλου λογισμικού.
- Ένα νέο παράρτημα γραμμένο από τον Διευθυντή της Επιστημονικής Ομάδας και τον Διευθυντή Αρχιτεκτονικής της NVIDIA, το οποίο καλύπτει την έλευση και τη μεγάλη σημασία των σύγχρονων GPU, και περιγράφει για πρώτη φορά αναλυτικά τον πολυεπεξεργαστή με υψηλή παραλληλία και υψηλή πολυνηματόωση που είναι βελτιστοποιημένος για οπτική υπολογιστική (visual computing).
- Μια καινοτόμος προσέγγιση για τη μέτρηση της απόδοσης των πολυπύρηνων επεξεργαστών — το μοντέλο «γραμμής στέγης» ("roofline") — με μετροπρογράμματα και ανάλυση της εκτέλεσής τους για τους AMD Opteron X4, Intel Xeon 5000, Sun UltraSPARC T2, και IBM Cell.
- Νέα ύλη σχετικά με τη μνήμη φλας και τις ακόνιστες μηχανές.
- Ένα μεγάλο σύνολο νέων ασκήσεων που καταλαμβάνουν περισσότερες από 200 σελίδες.
- Παρουσίαση του AMD Opteron X4 και του Intel Nehalem ως παραδειγμάτων του πραγματικού κόσμου σε όλη την έκταση του βιβλίου.
- Ενημέρωση όλων των παραδειγμάτων απόδοσης επεξεργαστών με τη χρήση του συνόλου μετροπρογραμμάτων SPEC CPU2006.

ELSEVIER

εκδόσεις ΚΛΕΙΔΑΡΙΘΜΟΣ
Δαμάσκου 4, Στοά της Λαμπής, 10440 ΑΘΗΝΑ, Τηλ. 210-5237835

Επισκεφθείτε μας στο Internet
www.kleidarithmos.gr

ISBN 978-960-461-351-9 (set)
ISBN 978-960-461-353-3 (Β΄ τόμος)
9 789604 613533