"My Binary Logic"
Ένας προσομοιωτής λογικών πυλών στο Scratch

Καραγιάννη Ελένη[1], Καραγιαννάκη Μαρία-Ελένη[2], Βασιλειάδης Αθανάσιος[3], Κωστουλίδης Αναστάσιος-Συμεών[4], Μουτεβελίδης Ιωάννης-Παναγιώτης[5], Βασιλειάδου Ευαγγελία[6], Βογιατζή Νίκη[7], Γοργία Φωτεινή[8], Ευγενιάδου Πηνελόπη[9], Τσολακίδου Παναγιώτα[10]

1..10 Μαθητές Γ Γυμνασίου, Ελληνικό Κολλέγιο Θεσσαλονίκης

Κωνσταντίνος Παρασκευόπουλος
Καθηγητής Πληροφορικής (ΠΕ19)
Ελληνικό Κολλέγιο Θεσσαλονίκης
diparaske@gmail.com

Περίληψη

Τα ψηφιακά κυκλώματα είναι σχεδιασμένα έτσι ώστε να λαμβάνουν αποφάσεις. Συνήθως δέχονται στην είσοδο δύο ή περισσότερες τιμές και παράγουν μία ή περισσότερες τιμές στην έξοδο. Η διαδικασία αυτή γίνεται με τη βοήθεια των λογικών πυλών (logic gates). Στην παρακάτω εργασία, χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Scratch για να οπτικοποιήσουμε και να προσομοιώσουμε τη λειτουργία μερικών από τις πλέον βασικές λογικές πύλες. Συγκεκριμένα σχεδιάσαμε και προγραμματίσαμε τη λειτουργία των πυλών OR, AND, NOT, NAND, NOR, XOR και XNOR. Στη συνέχεια χρησιμοποιήσαμε τις παραπάνω πύλες για να δημιουργήσουμε και να προσομοιώσουμε τη λειτουργία πιο σύνθετων λογικών κυκλωμάτων όπως είναι ο Half-Adder και ο Full-Adder.

Λέξεις κλειδιά: *Scratch, λογικές πύλες, προσομοίωση.*

1. Εισαγωγή

Η σημασία των λογικών κυκλωμάτων στην επιστήμη των ηλεκτρονικών υπολογιστών είναι προφανής, καθώς αποτελούν τις δομικές μονάδες κάθε ολοκληρωμένου κυκλώματος. Για το λόγο αυτό, δημιουργήσαμε ένα πρωτότυπο λογισμικό προσομοίωσης λογικών πυλών και λογικών κυκλωμάτων ώστε να βοηθήσουμε τους συμμαθητές μας στην καλύτερη κατανόηση της λειτουργίας τους. Για τον προγραμματισμό του προσομοιωτή, χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Scratch. Η γλώσσα Scratch είναι μια σχετικά καινούργια γλώσσα προγραμματισμού με γραφικό περιβάλλον με την οποία μπορούμε να κατασκευάσουμε εφαρμογές εύκολα, γρήγορα και διασκεδαστικά. Προτιμήθηκε αφού είναι μια ιδανική γλώσσα για αρχάριους προγραμματιστές σχεδιασμένη ειδικά για την εκπαίδευση.

2. Μελέτη της Άλγεβρας Boole

Ο υπολογιστής, εκτός από αριθμητικές πράξεις, έχει τη δυνατότητα να εκτελεί και συγκρίσεις, δηλαδή να επεξεργάζεται λογικά δεδομένα. Τα δεδομένα αυτά μπορεί να είναι απλές ή σύνθετες λογικές προτάσεις οι οποίες μπορούν να παρουσιαστούν με λογικά σύμβολα χρησιμοποιώντας την Άλγεβρα Boole. Πριν ξεκινήσουμε επομένως την υλοποίηση του προσομοιωτή και τον προγραμματισμό του, μελετήσαμε διεξοδικά την Άλγεβρα Boole αλλά και τον τρόπο με τον οποίο εφαρμόζεται στη σχεδίαση των ηλεκτρονικών κυκλωμάτων στους υπολογιστές.

2.1. Λογικές προτάσεις και άλγεβρα Boole

Μια πρόταση λέγεται λογική, όταν μπορεί να χαρακτηριστεί αληθής ή ψευδής. Ο πρώτος που διατύπωσε τους βασικούς κανόνες με τους οποίους οι λογικές προτάσεις μπορούν να παρουσιαστούν με μαθηματικά σύμβολα, ήταν ο Άγγλος μαθηματικός George Boole το 1847. Οι κανόνες που διατύπωσε χρησιμοποιούνται για να επιλύσουμε λογικά προβλήματα και αποτελούν τμήμα της Άλγεβρας Boole. Στην Άλγεβρα Boole υπάρχουν μόνο δύο είδη προτάσεων καθώς και δύο είδη μεταβλητών, αυτές που είναι αληθείς και αυτές που είναι ψευδείς. Οι κανόνες που καθορίζουν τις βασικές λογικές πράξεις ορίζονται από Πίνακες Αλήθειας στους οποίους περιλαμβάνονται όλοι οι δυνατοί συνδυασμοί των τιμών που μπορούν να πάρουν οι λογικές μεταβλητές καθώς και τα αποτελέσματα που προκύπτουν από αυτούς τους συνδυασμούς. Στον Πίνακα 1 μπορείτε να δείτε τους Πίνακες Αλήθειας που αντιστοιχούν στις πράξεις της άλγεβρας Boole που υποστηρίζονται από το λογισμικό προσομοίωσης που δημιουργήσαμε

Πίνακας 1*: Πίνακας Αλήθειας για τις πράξεις της άλγεβρας Boole που υποστηρίζονται από τον προσομοιωτή.*

Πίνακες Αλήθειας							
A	**B**	**A AND B**	**A OR B**	**A XOR B**	**A NAND B**	**A NOR B**	**A XNOR B**
Αληθής	Αληθής	Αληθής	Αληθής	Ψευδής	Ψευδής	Ψευδής	Αληθής
Αληθής	Ψευδής	Ψευδής	Αληθής	Αληθής	Αληθής	Ψευδής	Ψευδής
Ψευδής	Αληθής	Ψευδής	Αληθής	Αληθής	Αληθής	Ψευδής	Ψευδής
Ψευδής	Ψευδής	Ψευδής	Ψευδής	Ψευδής	Αληθής	Αληθής	Αληθής

Πίνακας Αλήθειας	
A	**NOT A**
Αληθής	Ψευδής
Ψευδής	Αληθής

3. Λογικές πύλες

Όπως είδαμε, στις πράξεις της Άλγεβρας Boole χρησιμοποιούνται μεταβλητές που παίρνουν δύο μόνο τιμές και δίνουν ως αποτέλεσμα πάλι δίτιμες μεταβλητές. Το 1938 ο C. Shannon έδειξε ότι η Άλγεβρα Boole μπορούσε να εφαρμοστεί στην απλοποίηση και στη σχεδίαση των ηλεκτρονικών κυκλωμάτων, όπως, για παράδειγμα στα τηλεφωνικά κυκλώματα. Αργότερα χρησιμοποιήθηκε για τη σχεδίαση των κυκλωμάτων των υπολογιστών.

Τα ηλεκτρονικά κυκλώματα που εκτελούν τις βασικές πράξεις της Άλγεβρας Boole καλούνται λογικές πύλες. Κάθε τέτοια πύλη δέχεται στην είσοδό της σήματα με τη μορφή υψηλής ή χαμηλής ηλεκτρικής τάσης και δίνει στην έξοδό της ένα μοναδικό λογικό αποτέλεσμα με τη μορφή υψηλής ή χαμηλής ηλεκτρικής τάσης. Συνδυάζοντας κατάλληλα λογικές πύλες δημιουργούνται πιο σύνθετα κυκλώματα που μπορούν να εκτελέσουν λογικές πράξεις.

Αρχικά για την υλοποίηση των λογικών πυλών χρησιμοποιήθηκαν διακριτά ηλεκτρονικά στοιχεία. Σήμερα στα ολοκληρωμένα κυκλώματα υπάρχει ένα μεγάλο πλήθος επιμέρους κυκλωμάτων, τα οποία με τη σειρά τους αποτελούνται από πλήθος λογικών πυλών.

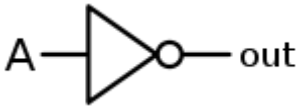
3.1. Οι υποστηριζόμενες λογικές πύλες

Οι λογικές πύλες που υποστηρίζονται από τον προσομοιωτή είναι οι NOT, AND, OR, NAND, NOR, XOR, και η XNOR. Παρακάτω περιγράφεται η λειτουργίας τους και για κάθε μία από αυτές δίνεται ο πίνακας αλήθειας καθώς και το σχήμα με το οποίο συμβολίζονται.

Η λογική πύλη NOT

Υλοποιεί τη λογική πράξη «Άρνηση». Έχει μια είσοδο και μια έξοδο. Όταν η είσοδος είναι 1 η έξοδος είναι 0, ενώ όταν η είσοδος είναι 0 η έξοδος είναι 1, δηλαδή αντιστρέφει την αξία της εισόδου. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

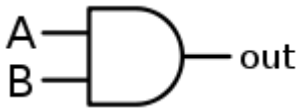
Είσοδοι	Έξοδος
A	**NOT A**
0	1
1	0

Κύκλωμα


Η λογική πύλη AND

Υλοποιεί τη λογική πράξη «Σύζευξη». Έχει δύο εισόδους και μια έξοδο. Η έξοδος είναι 1, όταν όλες οι είσοδοι είναι 1. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

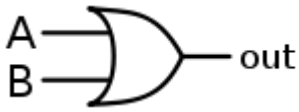
Είσοδοι		Έξοδος
A	**B**	**A AND B**
0	0	0
0	1	0
1	0	0
1	1	1

Κύκλωμα


Η λογική πύλη OR

Υλοποιεί τη λογική πράξη «Διάζευξη». Έχει δύο εισόδους και μια έξοδο. Η έξοδος είναι 1, όταν τουλάχιστον μια είσοδος είναι 1. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

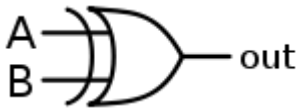
Είσοδοι		Έξοδος
A	**B**	**A OR B**
0	0	0
0	1	1
1	0	1
1	1	1

Κύκλωμα


Η λογική πύλη XOR

Υλοποιεί τη λογική πράξη «Αποκλειστική διάζευξη». Έχει δύο εισόδους και μια έξοδο. Η έξοδος είναι 1, όταν οι τιμές στις εισόδους είναι διαφορετικές μεταξύ τους. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

Είσοδοι		Έξοδος
A	**B**	**A XOR B**
0	0	0
0	1	1
1	0	1
1	1	0

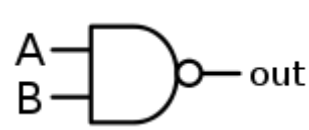
Κύκλωμα


Μπορούμε να δημιουργήσουμε πιο πολύπλοκες λογικές πύλες είτε προσθέτοντας περισσότερες εισόδους είτε συνδυάζοντας τις απλές με την πύλη ΟΧΙ (NOT).

Η λογική πύλη NAND

Αποτελείται από μια πύλη AND και μια πύλη NOT. Η πύλη NAND δίνει την αντίθετη έξοδο από την AND, δηλαδή δίνει λογικό 1 όταν υπάρχει τουλάχιστο ένα λογικό 0 στις εισόδους. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

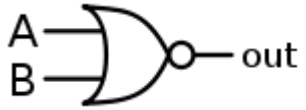
Είσοδοι		Έξοδος
A	**B**	**A NAND B**
0	0	1
0	1	1
1	0	1
1	1	0

Κύκλωμα


Η λογική πύλη NOR

Αποτελείται από μια πύλη OR και μια πύλη NOT. Η πύλη δίνει την αντίθετη έξοδο από την OR, δηλαδή δίνει λογικό 1 όταν και οι δύο είσοδοι είναι 0. Ο πίνακας αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

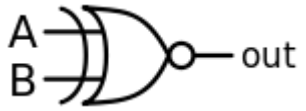
Είσοδοι		**Έξοδος**
A	**B**	**A NOR B**
0	0	1
0	1	0
1	0	0
1	1	0

Κύκλωμα


Η λογική 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 αλήθειας καθώς και το κυκλωματικό σχήμα της πύλης φαίνονται παρακάτω:

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

Κύκλωμα


3. Συνδυαστικά Λογικά Κυκλώματα

Οι παραπάνω λογικές πύλες μπορούν να συνδυαστούν και να δημιουργήσουν ένα συνδυαστικό λογικό κύκλωμα. Το κύκλωμα του Ημιαθροιστή (Half Adder) και το κύκλωμα του Αθροιστή (Full Adder) είναι δύο από τα περισσότερο γνωστά συνδυαστικά λογικά κυκλώματα και για αυτό επιλέξαμε να τα υλοποιήσουμε στον προσομοιωτή που κατασκευάσαμε. Η λειτουργίας τους περιγράφεται παρακάτω.

Το κύκλωμα του Ημιαθροιστή (Half Adder)

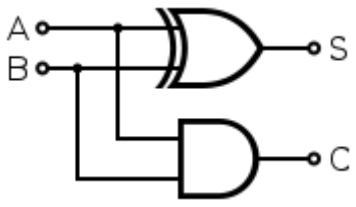
Ημιαθροιστής (Half Adder), λέγεται το συνδυαστικό κύκλωμα το οποίο εκτελεί την πρόσθεση δύο δυαδικών ψηφίων σύμφωνα με τον παρακάτω πίνακα:

+	**0**	**1**
0	0	1
1	1	10

Η πρόσθεση 1 + 1 δίνει άθροισμα 0 και κρατούμενο (carry) 1, το οποίο μεταφέρεται στην επόμενη πιο σημαντική τάξη ψηφίων. Αν A, B είναι δυαδικά ψηφία, τότε το άθροισμα τους S (Sum) και το κρατούμενο της πρόσθεσής τους c φαίνονται στον παρακάτω πίνακα αλήθειας.

Πίνακας Αλήθειας			
Είσοδοι		**Έξοδος**	
A	**B**	**S**	**C**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Το λογικό κύκλωμα του ημιαθροιστή βρίσκεται εύκολα, γιατί οι έξοδοι S και C είναι ίδιες με τις εξόδους των πυλών XOR και AND αντίστοιχα. Το κυκλωματικό σχήμα του Ημιαθροιστή φαίνεται παρακάτω:

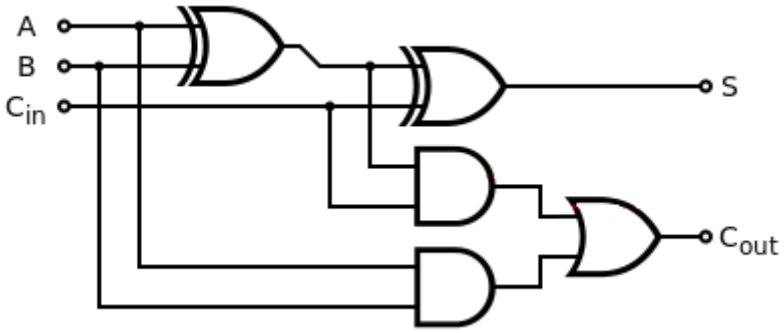


Το κύκλωμα του Αθροιστή (Full Adder)

Ο Αθροιστής είναι το συνδυαστικό κύκλωμα που εκτελεί την πρόσθεση τριών δυαδικών αριθμών και, συγκεκριμένα, δύο σημαντικών και ενός κρατουμένου. Το κρατούμενο ενδέχεται να έχει παραχθεί από προηγούμενη άθροιση. Το κύκλωμα του Αθροιστή, έχει τρεις εισόδους A, B, Cin, που αποτελούν τους δυο προσθετέους και το προηγούμενο κρατούμενο. Οι δυο έξοδοι S, Cout συμβολίζουν το άθροισμα και το νέο κρατούμενο. Ο πίνακας αλήθειας του αθροιστή έχει ως εξής:

Input			Output	
A	**B**	**Cin**	**Cout**	**S**
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

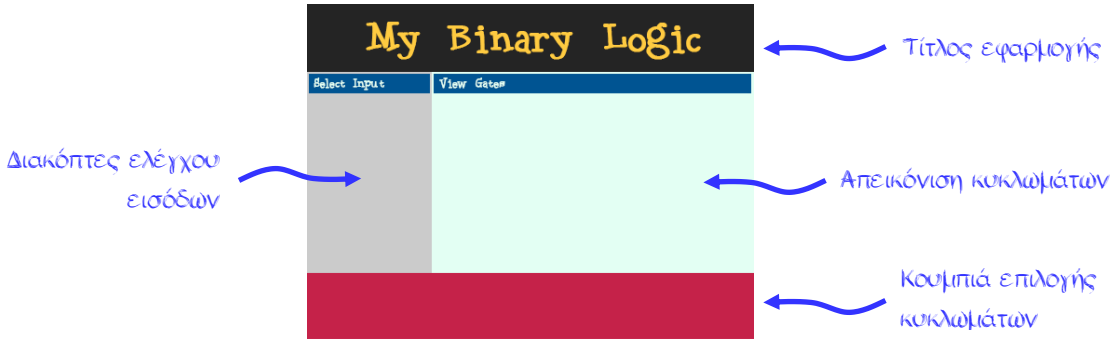
Το κυκλωματικό σχήμα του Αθροιστή φαίνεται παρακάτω:



4. Προγραμματισμός του προσομοιωτή

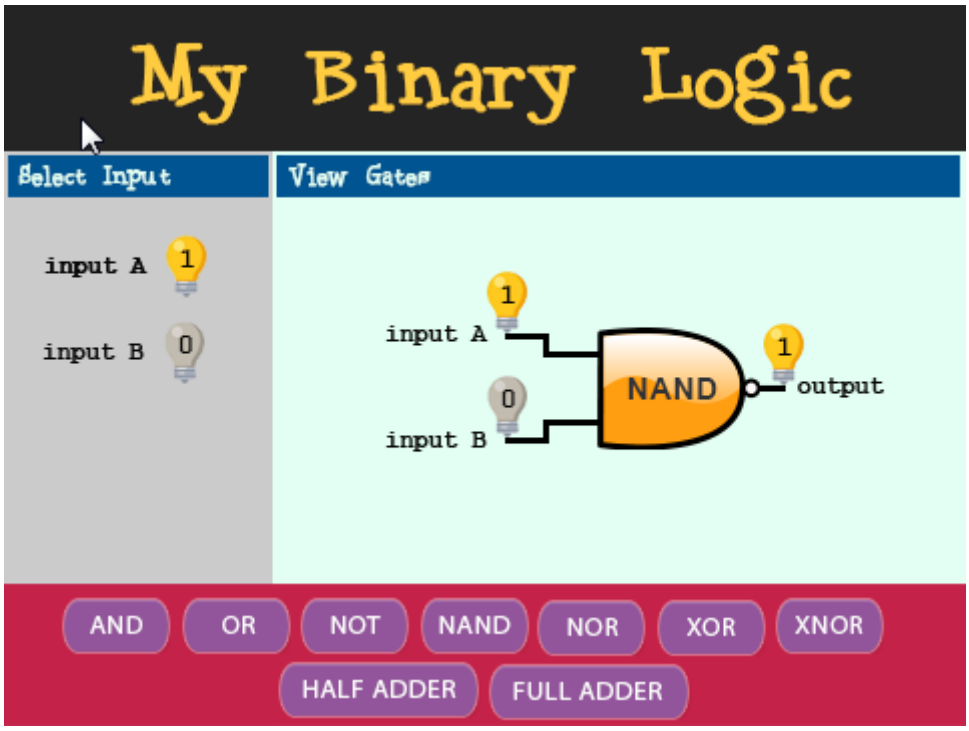
Αφού κατανοήσαμε πλήρως τη λειτουργία των λογικών πυλών και τον τρόπο με τον οποίο συνδυάζονται για να δημιουργήσουν τα κύκλωμα του Ημιαθροιστή και του Αθροιστή, ήρθε η ώρα να προγραμματίσουμε το λογισμικό του προσομοιωτή στο Scratch.

Ξεκινήσαμε από τη δημιουργία της διεπαφής χρήστη. Σχεδιάσαμε το υπόβαθρο της διεπαφής στο σχεδιαστικό εργαλείο του Scratch, φροντίζοντας να χωρίσουμε τη διαθέσιμη περιοχή σε 4 τμήματα. Όπως φαίνεται στην ***Εικόνα 1***, στο επάνω τμήμα τοποθετήθηκε ο τίτλος της εφαρμογής, στο αριστερό τμήμα δημιουργήθηκε χώρος για να μπουν αργότερα διακόπτες για τον έλεγχο των εισόδων των λογικών κυκλωμάτων, στο δεξί τμήμα δεσμεύτηκε χώρος για την απεικόνιση των κυκλωμάτων ενώ στο κάτω τμήμα της διαθέσιμης περιοχής δημιουργήθηκε χώρος για να τοποθετηθούν αργότερα τα κουμπιά επιλογής των λογικών κυκλωμάτων.

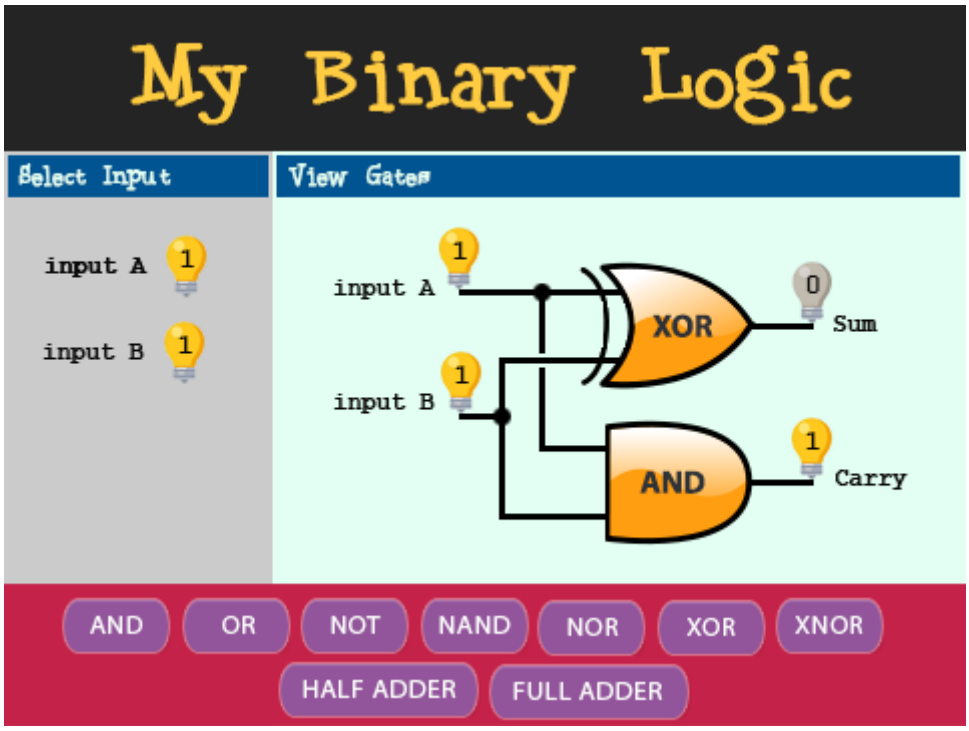


Εικόνα 1*: Τα τμήματα του υποβάθρου της εφαρμογής*

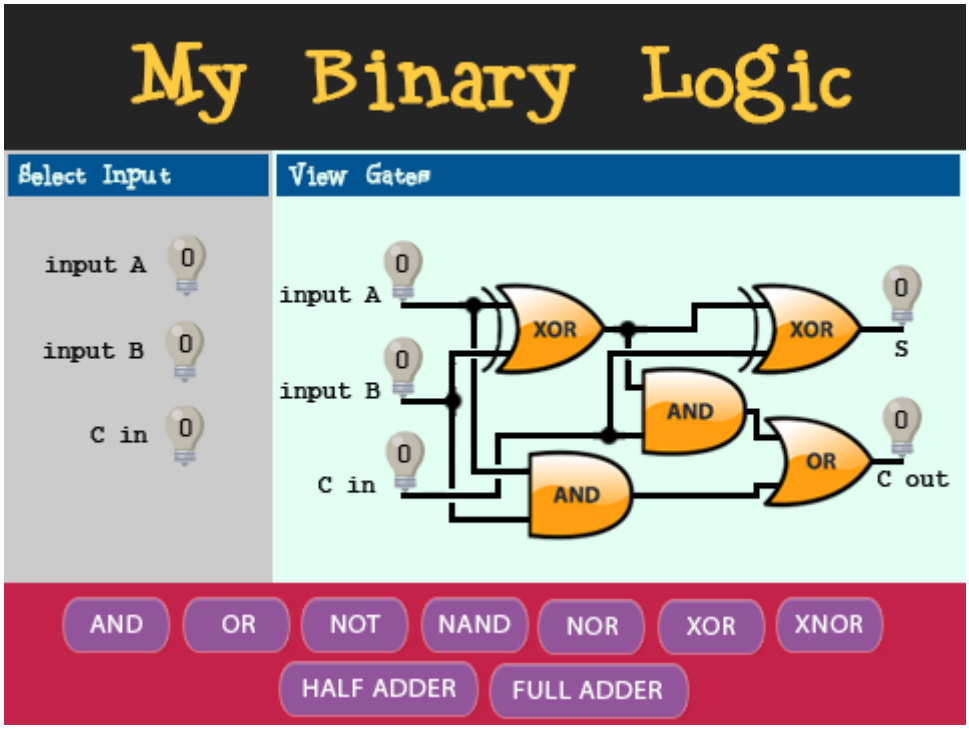
Στη συνέχεια σχεδιάστηκαν, τοποθετήθηκαν και προγραμματίστηκαν τα κουμπιά επιλογής λογικών κυκλωμάτων καθώς και οι διακόπτες εισόδου. Τέλος σχεδιάστηκαν, τοποθετήθηκαν και προγραμματίστηκαν οι λογικές πύλες και τα λογικά κυκλώματα. Στις ***Εικόνες 2, 3 και 4*** φαίνεται το γραφικό περιβάλλον της εφαρμογής.



Εικόνα 2*: Το γραφικό περιβάλλον της εφαρμογής με μία πύλη NAND*



Εικόνα 3*: Προσομοίωση της λειτουργίας ενός Ημιαθροιστή*



***Εικόνα** 4: Προσομοίωση της λειτουργίας ενός Αθροιστή*

Ο περιορισμένος χώρος της παρούσας εργασίας δεν μας επιτρέπει να παρουσιάσουμε τον κώδικα της εφαρμογής η οποία όμως διατίθεται για κατέβασμα στο διαδικτυακό τόπο του Ελληνικού Κολλεγίου Θεσσαλονίκης www.hellenic-college.gr στην ενότητα Εκπαιδευτικό Λογισμικό.

Ευχαριστίες

Θα θέλαμε να εκφράσουμε τις ευχαριστίες μας στον καθηγητή πληροφορικής του Ελληνικού Κολλεγίου Θεσσαλονίκης, κ. Κωνσταντίνο Παρασκευόπουλο για τις υποδείξεις και την καθοδήγηση που προσέφερε σε όλη τη διάρκεια της εργασίας

Βιβλιογραφία

1. Εφαρμογές Πληροφορικής Υπολογιστών (Α, Β, Γ Γενικού Λυκείου - Επιλογής) - Βιβλίο Μαθητή
2. https://scratch.mit.edu/
3. Παρασκευόπουλος Κ. (2014). Εισαγωγή στον προγραμματισμό Η/Υ - Εργαστηριακές Σημειώσεις, Θεσσαλονίκη, Ελληνικό Κολλέγιο Θεσσαλονίκης.