ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΔΙΑΤΜΗΜΑΤΙΚΟ ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΠΛΗΡΟΦΟΡΙΚΗ ΚΑΙ ΥΠΟΛΟΓΙΣΤΙΚΗ ΒΙΟΙΑΤΡΙΚΗ

«ΠΛΗΡΟΦΟΡΙΚΗ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗΝ ΑΣΦΑΛΕΙΑ, ΔΙΑΧΕΙΡΙΣΗ ΜΕΓΑΛΟΥ ΟΓΚΟΥ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ»

Εισαγωγή στην VHDL: Ένα διαδραστικό εκπαιδευτικό λογισμικό

Λιάλιος Α. Ιωάννης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Υπεύθυνος

Δαδαλιάρης Αντώνιος

Λαμία, Απρίλιος 2016

«Υπεύθυνη Δήλωση μη λογοκλοπής και ανάληψης προσωπικής ευθύνης»

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, και γνωρίζοντας τις συνέπειες της λογοκλοπής, δηλώνω υπεύθυνα και ενυπογράφως ότι η παρούσα εργασία με τίτλο **Εισαγωγή στην VHDL: Ένα διαδραστικό εκπαιδευτικό λογισμικό** αποτελεί προϊόν αυστηρά προσωπικής εργασίας και όλες οι πηγές από τις οποίες χρησιμοποίησα δεδομένα, ιδέες, φράσεις, προτάσεις ή λέξεις, είτε επακριβώς (όπως υπάρχουν στο πρωτότυπο ή μεταφρασμένες) είτε με παράφραση, έχουν δηλωθεί κατάλληλα και ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο/Η ΔΗΛΩΝ/-ΟΥΣΑ

Ημερομηνία

Υπογραφή

Εισαγωγή στην VHDL: Ένα διαδραστικό εκπαιδευτικό λογισμικό

Λιάλιος Α. Ιωάννης

Τριμελής Επιτροπή:

Δαδαλιάρης Αντώνιος (επιβλέπων/σα)

Κοζύρη Μαρία

Σταμούλης Γεώργιος

UNIVERSITY OF THESSALY

FACULTY OF SCIENCES

MASTER

«Informatics and Computational Biomedicine»

« *Direction of computer science with adjustments to safety, management large volume of data and simulation* »

Master Thesis

Introduction to VHDL: An interactive educational software

Lialios A. Ioannis

Supervisor : Dadaliaris Antonios

Lamia, April 2016

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή αυτής της διπλωματικής εργασίας κ. Αντώνη Δαδαλιάρη, για την συνεργασία που είχαμε τον τελευταίο ενάμιση χρόνο τόσο στα μαθήματα του μεταπτυχιακού προγράμματος όσο και κατά την διάρκεια εκπόνησης αυτής της εργασίας.

Πάνω από όλα όμως ευχαριστώ στους γονείς μου, για την ολόψυχη αγάπη και υποστήριξή τους όλα αυτά τα χρόνια τα χρόνια των σπουδών μου, καθώς ήταν δίπλα μου στις εύκολες και δύσκολες στιγμές όλης αυτής της διαδρομής. Αφιερώνω αυτή την εργασία στην μητέρα μου και στον πατέρα μου.

ΠΕΡΙΕΧΟΜΕΝΑ

Περίληψη

Το αντικείμενο ενασχόλησης αυτής της διπλωματικής εργασίας ήταν η κατασκευή μιας εκπαιδευτικής εφαρμογής, σχετικά με την γλώσσα περιγραφής υλικού VHDL. Η εφαρμογή υλοποιήθηκε σε γλώσσα Visual C Sharp (Visual C#), η οποία είναι μια αντικειμενοστραφής γλώσσα της ευρύτερης κατηγορίας των γλωσσών .NET και το εργαλείο που χρησιμοποιήθηκε για να χτιστούν και να τρέξουν τα προγράμματα, είναι το Visual Studio 2015 της Microsoft.

Η εφαρμογή υλοποιήθηκε με σκοπό να προσφέρει σε νέους χρήστες της γλώσσας VHDL, τις κατάλληλες γνώσεις για τις βασικές δομές της γλώσσας σε πρακτικό και θεωρητικό επίπεδο έτσι ώστε με την κατάλληλη εξάσκηση, να μπορούν να ανταπεξέλθουν στην πολυπλοκότητα της γλώσσας, σε περίπτωση που τους ζητηθεί να περιγράψουν ένα μεγάλο και πολύπλοκο ηλεκτρονικό κύκλωμα.

Συνεπώς, στα επόμενα κεφάλαια θα γίνει αναλυτική παρουσίαση του σχεδιασμού της εφαρμογής, των βασικών εργαλείων και των γλωσσών προγραμματισμού που χρησιμοποιήθηκαν, καθώς και των κύριων τεχνικών που οδήγησαν στον τελικό αποτέλεσμα.

Κεφάλαιο 1ο
Εργαλεία και Γλώσσες Προγραμματισμού

1.1 Εισαγωγή

Στο κεφάλαιο αυτό θα παρουσιασθούν τα εργαλεία και οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν για την κατασκευή της εφαρμογής. Συνοπτικά τα εργαλεία αυτά είναι το Visual Studio 2015 με το οποίο γίνονταν συγγραφή των προγραμμάτων σε C Sharp και η μεταγλώττιση τους, το XilinxISE όπου γίνονταν η συγγραφή και μεταγλώττιση προγραμμάτων σε VHDL και το Adobe Photoshop με το οποίο έγινε η επεξεργασία των εικόνων της εφαρμογής. Ο βασικός κορμός της εφαρμογής έχει υλοποιηθεί σε γλώσσα Visual C Sharp (Visual C#) και τα προγράμματα των παραδειγμάτων καθώς και τα προγράμματα των ασκήσεων είναι γραμμένα σε γλώσσα VHDL. Συνεπώς, οι γλώσσες προγραμματισμού που χρειάστηκαν για την εν λόγω εφαρμογή είναι δύο, η Visual C Sharp και η VHDL.

1.2 Περιβάλλον XilinxISE

Το Xilinx ISE, είναι ένα εργαλείο λογισμικού που παράγεται από την Xilinx για τη σύνθεση και την ανάλυση προγραμμάτων σε γλώσσες HDL. Επιτρέποντας στους προγραμματιστές να συνθέσουν και να μεταγλωττίσουν τα προγράμματα τους, να εκτελέσουν ανάλυση χρονισμού, να εξετάσουν τα παραγόμενα RTL διαγράμματα και να προσομοιώσουν την αντίδραση του κυκλώματος τους σε διαφορετικές λογικές καταστάσεις. Το Xilinx ISE χρησιμοποιείται κυρίως για τη σύνθεση του κυκλώματος και το σχεδιασμό, ενώ το ISim χρησιμοποιείται για τη δοκιμή σε επίπεδο συστήματος.

Η κύρια διεπαφή του Xilinx ISE είναι ο Project Navigator, η οποία περιλαμβάνει τον ιεραρχικό σχεδιασμό, ένα πρόγραμμα επεξεργασίας πηγαίου

κώδικα (Workplace) και την κονσόλα εξόδου (Transcript). Ο ιεραρχικός σχεδιασμός αποτελείται από τα αρχεία σχεδιασμού (modules), των οποίων οι εξαρτήσεις έχουν ερμηνευθεί από το Xilinx ISE και εμφανίζονται σαν δομή δέντρου. Όπως για παράδειγμα στο σχεδιασμό απλών chips μπορεί να υπάρξει μία κύρια μονάδα, με άλλες μονάδες να περιλαμβάνονται σε αυτή, παρόμοια με την main () υπορουτίνα σε προγράμματα C ++. Η κονσόλα εξόδου δείχνει την κατάσταση των τρεχόντων εργασιών, ενημερώνοντας τους μηχανικούς για θέματα σχεδιασμού, εμφανίζοντας προειδοποιήσεις, σφάλματα ή και τα δύο.

Ο έλεγχος σε επίπεδο συστήματος, για τα προγράμματα που είναι γραμμένα σε γλώσσα VHDL, μπορεί να γίνει με τον προσομοιωτή ISim. Η διαδικασία αυτή περιλαμβάνει εισαγωγή σημάτων και παραγωγή κυματομορφών με την απόκριση του συστήματος.

Το ISim μπορεί να χρησιμοποιηθεί για να εκτελεσθούν τα ακόλουθα είδη προσομοιώσεων:

- Logical verification, για να επαληθευτεί αν ο κώδικας που αναπτύχθηκε παράγει τα αναμενόμενα αποτελέσματα
- Behavioral verification, για να επαληθευτούν θέματα λογικής και χρόνου.

Τέλος, οι κατοχυρωμένοι με δίπλωμα ευρεσιτεχνίας αλγόριθμοι σύνθεσης της Xilinx, επιτρέπουν στις σχεδιάσεις σε γλώσσες HDL, να τρέχουν μέχρι και 30% γρηγορότερα από ανταγωνιστικά προγράμματα σχεδιασμού, και να επιτρέπουν μεγαλύτερη λογική πυκνότητα, η οποία μειώνει το κόστος του έργου.

1.3 Περιβάλλον VISUAL STUDIO 2015

Το Microsoft Visual Studio 2015 είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης από τη Microsoft, που χρησιμοποιείται για την ανάπτυξη προγραμμάτων για ηλεκτρονικούς υπολογιστές για το λειτουργικό σύστημα των Microsoft Windows, ιστοσελίδες, διαδικτυακές εφαρμογές καθώς και υπηρεσίες ιστού (web). Το Visual Studio 2015 χρησιμοποιεί πλατφόρμες ανάπτυξης λογισμικού της Microsoft, όπως τα Windows API, τα Windows Forms, το Windows Presentation Foundation, το Windows Store και το Microsoft Silverlight. Μπορεί να παράγει τόσο εγγενή κώδικα όσο και να διαχειριστεί κώδικα.

Το Visual Studio 2015 περιλαμβάνει έναν συντάκτη κώδικα (code editor) που γίνετε η συγγραφή του πηγαίου κώδικα, ένα ολοκληρωμένο πρόγραμμα εντοπισμού σφαλμάτων (debugger) που λειτουργεί τόσο ως πρόγραμμα εντοπισμού σφαλμάτων στο επίπεδο παραγωγής κώδικα όσο και ως πρόγραμμα εντοπισμού σφαλμάτων στο επίπεδο μηχανής κατά την διαδικασία της μεταγλώττισης. Άλλα ενσωματωμένα εργαλεία που περιλαμβάνει το Visual Studio 2015, είναι ο σχεδιαστής φορμών για τη δημιουργία εφαρμογών GUI, ο σχεδιαστής για web εφαρμογές, ο σχεδιαστής κλάσεων και ο σχεδιαστής για την δημιουργία βάσεων δεδομένων. Επίσης δέχεται Επεκτάσεις που βελτιώνουν τη λειτουργικότητα σε κάθε επίπεδο, όπως συντάκτες κώδικα και οπτικούς σχεδιαστές που χρησιμοποιούνται σε συγκεκριμένες γλώσσες όπως η C Sharp, η Visual Basic κα.

Το Visual Studio 2015 παρέχει περιβάλλον ανάπτυξης για διάφορες γλώσσες προγραμματισμού, ενώ άλλες υποστηρίζονται μέσω κάποιων επεκτάσεων των ήδη υφιστάμενων λειτουργιών που εγκαθίστανται στο αρχικό περιβάλλον. Οι γλώσσες για τις οποίες παρέχεται εξ αρχής περιβάλλον ανάπτυξης είναι C, C++, VB.NET (Visual Basic), C# (Visual C Sharp) και F#. Ενώ για τις γλώσσες Python, Ruby, Node.js, M, XML/XSLT, HTML/XHTML, JavaScript και CSS παρέχεται υποστήριξη μετά από εγκατάσταση προσθέτων πακέτων λειτουργιών (VSPackages). Γλώσσα που έχει υποστηριχθεί στο παρελθόν από το Visual Studio και δεν υποστηρίζεται πλέον είναι η JAVA.

Όλες οι εμπορικές εκδόσεις του Visual Studio μέχρι το 2014 που δέχονταν επεκτάσεις λειτουργιών, παρέχονταν δωρεάν από την Microsoft μόνο σε φοιτητές μέσω του προγράμματος DreamSpark, ενώ φέτος με την έκδοση Visual Studio 2015 Community όλοι οι χρήστες μπορούν να χρησιμοποιήσουν τις πλήρεις δυνατότητες του λογισμικού χωρίς κανένα κόστος, μόνο με μία απλή εγγραφή στην εταιρεία μέσω email. Η εφαρμογή που αναπτύχθηκε στα πλαίσια αυτής της διπλωματικής εργασίας, έγινε με την χρήση της Community έκδοσης του Visual Studio 2015 που παρέχεται δωρεάν στους χρήστες.

1.4 Γλώσσα VHDL

Η VHDL είναι μια γλώσσα περιγραφής υλικού για την ανάπτυξη ψηφιακών ηλεκτρονικών συστημάτων. Ως λέξη αποτελεί συντόμευση των λέξεων: VHSIC Hardware Description Language. Τα δε αρχικά VHSIC είναι με τη σειρά τους συντόμευση για τις λέξεις Very High-Speed Integrated Circuit (Ολοκληρωμένα Κυκλώματα Υψηλής Ταχύτητας). Η VHDL ως γλώσσα προγραμματισμού μπορεί να χρησιμοποιηθεί για την περιγραφή της συμπεριφοράς, της δομής αλλά και της εφαρμογής ψηφιακών συστημάτων. Με βάση αυτά τα χαρακτηριστικά η VHDL χαρακτηρίζεται σαν ένα εργαλείο ECAD (Electronic Computer Aided Design). Γενικά, σήμερα η χρήση εργαλείων CAD έχει επεκταθεί, καθώς η τεράστια ανάπτυξη της τεχνολογίας ημιαγωγών στην κατασκευή ολοκληρωμένων κυκλωμάτων έχει μετατοπίσει το κέντρο βάρους των μηχανικών, από την λεπτομερειακή υλοποίηση κυκλωμάτων, στην διαχείριση της αυξανόμενης πολυπλοκότητας. Πιο συγκεκριμένα, στη σημερινή εποχή ο μηχανικός-σχεδιαστής, περιορίζεται περισσότερο από την δυνατότητά του να ανταπεξέλθει την πολυπλοκότητα της σχεδίασης του παρά από την ικανότητα της τεχνολογίας να την υποστηρίξει. Αυτό το χάσμα έρχεται να γεφυρώσει η VHDL επιτρέποντας μία υψηλού επιπέδου περιγραφή της σχεδίασης και κατόπιν με την χρήση εργαλείων σύνθεσης (Logic Synthesis Tools) την αυτόματη αποτύπωση αυτής της σχεδίασης σε ολοκληρωμένη μορφή η οποία πρέπει να είναι εντός των προδιαγραφών που θέτει ο μηχανικός.

Η VHDL ως γλώσσα περιγραφής υλικού μπορεί να χρησιμοποιηθεί για την περιγραφή ενός ψηφιακού κυκλώματος όπως προαναφέραμε. Αυτό το κύκλωμα μπορεί να είναι από μία απλή λογική πύλη έως ένα ολοκληρωμένο ψηφιακό σύστημα. Στη VHDL το ψηφιακό κύκλωμα που σχεδιάζεται, αναφέρεται ως entity (οντότητα). Όμως ένα entity X όταν εμπεριέχεται μέσα σ' ένα entity Y τότε αυτό ονομάζεται component (στοιχείο).

Η VHDL έχει πολλά πλεονεκτήματα γιατί υποστηρίζει διάφορες μεθόδους ψηφιακής εξέλιξης όπως top-down ή bottom-up σχεδιασμό ή και συνδυασμό και των δύο. Αρκετά από τα σημερινά ηλεκτρονικά προϊόντα έχουν χρόνο ζωής

πάνω από 10 χρόνια, και συνεπώς πρέπει να επανασχεδιάζονται αρκετές φορές έτσι ώστε να εκμεταλλευθούν την νέα τεχνολογία. Ο απλούστερος τρόπος για να γίνει αυτό είναι να χρησιμοποιηθεί μια τεχνολογικά ανεξάρτητη σχεδίαση σε VHDL, που επιτρέπει να αλλάξει η τεχνολογία χρησιμοποιώντας αυτόματα εργαλεία. Η VHDL υποστηρίζει την μετατρεψιμότητα (modifiability) γιατί ως γλώσσα είναι εύκολο να διαβαστεί, είναι ιεραρχική και επίσης είναι δομημένη. Επίσης η γλώσσα υποστηρίζει επαναχρησιμοποιούμενα συστατικά, διαχείριση λαθών και επιβεβαίωση σωστής σύνταξης των εντολών. Οι ιεραρχίες περιγράφονται χρησιμοποιώντας δομική VHDL, διαδικασίες και συναρτήσεις. Η δομική VHDL μπορεί να συγκριθεί με ένα block διάγραμμα. Πολλά συστήματα υποστηρίζουν γραφική είσοδο που μπορεί να μεταφραστεί αυτόματα σε δομική VHDL. Η γλώσσα τέλος υποστηρίζει ταυτόχρονες και ακολουθιακές δομές.

Τα μειονεκτήματα της VHDL είναι λιγότερα, από τα πλεονεκτήματα που προσφέρει σαν γλώσσα αλλά σε αυτό το σημείο είναι σκόπιμο να αναφερθούν. Η VHDL είναι μια φλύαρη γλώσσα με την έννοια ότι για μια απλή λειτουργία μπορεί να χρειαστεί να γραφούν πολλές εντολές, ειδικά στην δομική (Structural) εκδοχή της. Επίσης έχει αυξημένη πολυπλοκότητα σαν γλώσσα και μπορεί να επέλθει μια πρόσκαιρη σύγχυση στον προγραμματιστή που ασχολείται πρώτη φορά με αυτήν, λόγο τον πολλών τρόπων που μπορεί να γραφεί ένα πρόγραμμα (Structural, Dataflow, Behavioral). Άλλη μια ιδιομορφία της VHDL είναι κατασκευές που έχουν παρόμοια σύνταξη μπορεί να εκτελούν τελείως διαφορετικές εργασίες. Τέλος η VHDL είναι ακατάλληλη για την επαλήθευση της λειτουργίας των βασικών πυλών, γιατί τα στοιχεία αυτά είναι άμεσα διαθέσιμα από τις βιβλιοθήκες της γλώσσας σε αντίθεση με την VERILOG που μπορεί να σχεδιαστεί μια λογική πύλη από μηδενική βάση.

1.5 .NET Framework

Το .NET Framework είναι ένα πλαίσιο λογισμικού (software framework) που προορίζεται για την πλατφόρμα των Windows. Αποτελείται από μια μεγάλη

βιβλιοθήκη κλάσεων και υποστηρίζει μια πλειάδα γλωσσών προγραμματισμού με τη δυνατότητα η μια να μπορεί να χρησιμοποιηθεί από την άλλη. Τα προγράμματα που γράφονται για το .NET Framework εκτελούνται σε ένα περιβάλλον εκτέλεσης γνωστό ως Common Language Runtime (CLR), ενός ειδικού λογισμικού, σχεδιασμένου να υποστηρίζει την εκτέλεση προγραμμάτων και την απρόσκοπτη συνεργασία με το λειτουργικό σύστημα. Το CLR περιέχει μια εικονική μηχανή (virtual machine) που διαχειρίζεται την εκτέλεση ενός προγράμματος και παρέχει μια σειρά σημαντικών υπηρεσιών όπως ασφάλεια, διαχείριση μνήμης και διαχείριση εξαιρέσεων.

Τα προγράμματα που γράφονται για το .NET χρησιμοποιούν τη βιβλιοθήκη κλάσεων (class library) του .NET, η οποία δίνει πρόσβαση στο περιβάλλον εκτέλεσής του (runtime environment). Βασικές λειτουργίες όπως οι γραφικές διεπαφές χρηστών (Graphical User Interfaces – GUIs), η επικοινωνία με βάσεις δεδομένων, η κρυπτογραφία, η ανάπτυξη web εφαρμογών και οι δικτυακές επικοινωνίες παρέχονται μέσω του Application Programming Interface (API) του .NET και μπορούν να συνδυαστούν με κώδικα από τους προγραμματιστές για τη δημιουργία ολοκληρωμένων εφαρμογών. Όσο ένα πρόγραμμα περιορίζεται στη χρήση της βιβλιοθήκης κλάσεων του .NET, μπορεί να τρέχει οπουδήποτε υπάρχει εγκατεστημένο το περιβάλλον εκτέλεσης του .NET.

Η γλώσσα C# είναι μια από τις γλώσσες που υποστηρίζει το .NET. Ο μεταγλωττιστής της C# στοχεύει ειδικά στο .NET κάτι που σημαίνει ότι τα προγράμματα γραμμένα σε C# θα τρέχουν πάντα στο .NET Framework. Αυτό έχει δυο σημαντικές συνέπειες για τη γλώσσα:

- Η αρχιτεκτονική της και οι μεθοδολογίες της αντικατοπτρίζουν τη δομή του.NET.
- Σε πολλές περιπτώσεις, ειδικά χαρακτηριστικά της γλώσσας εξαρτώνται στην πραγματικότητα από τα χαρακτηριστικά του .NET ή τις βασικές κλάσεις του .NET.

1.6 Γλώσσα C Sharp

Η C# είναι μια σχετικά καινούργια γλώσσα, ενώ έχει γίνει μια από τις πιο διαδεδομένες αντικειμενοστραφείς γλώσσες προγραμματισμού. Δημιουργήθηκε από την Microsoft μέσα από την πλατφόρμα .NET και αργότερα αναγνωρίστηκε επισήμως από την Ecma (ECMA-334) και την ISO (ISO/IEC 2327:2006). Είναι μια από τις γλώσσες προγραμματισμού που δημιουργήθηκαν για την Common Language Infrastructure. Ο κύριος σκοπός της γλώσσας είναι, να είναι μια απλή αντικειμενοστραφής γλώσσα για γενική χρήση. Στις 15 Αυγούστου 2012 κυκλοφόρησε η έκδοση 5.0 η οποία είναι η πιο πρόσφατη μέχρι σήμερα.

Κατά την διάρκεια της δημιουργίας της πλατφόρμας .NET οι κλάσεις και οι βιβλιοθήκες γράφτηκαν χρησιμοποιώντας έναν compiler με το όνομα Simple Managed C (SMC). Τον Ιανουάριο του 1999 ο Anders Hejlsberg συγκρότησε μια ομάδα με σκοπό να φτιάξει μια καινούρια γλώσσα με όνομα Cool (C-like Object Oriented Language). Παρόλο που η Microsoft σκεφτόταν να κρατήσει το όνομα Cool σαν το τελικό όνομα της γλώσσας αυτό δεν έγινε ποτέ για λόγους πνευματικών δικαιωμάτων. Μέχρι τον Ιούλιο του 2000 όπου ανακοινώθηκε η πλατφόρμα .NET η γλώσσα είχε είδη μετονομαστεί σε C# στην οποία αργότερα εισήχθησαν οι βιβλιοθήκες της ASP.NET.

Ο James Gosling, (σχεδιαστής της Java) το 1994 μαζί με τον Bill Joy (συνιδρυτής της Sun Microsystems) αποκάλεσαν την C# μια απομίμηση της Java. Ο Gosling επίσης συμπλήρωσε ότι η C# είναι ίδια με την Java απλά χωρίς αξιοπιστία παραγωγικότητα και ασφάλεια. Οι συγγραφείς ενός βιβλίου της C# ισχυρίστηκαν ότι η Java και η C# είναι πανομοιότυπες επαναληπτικές και χωρίς καινοτομίες. Τον Ιούνιο του 2000 ο Anders Hejlsberg υποστήριξε ότι η C# δεν είναι κλώνος της Java αλλά ότι είναι πολύ πιο κοντά στην C++.

Από τότε που κυκλοφόρησε η δεύτερη έκδοση της C# το Νοέμβριο του 2005, η C# και η Java άρχισαν να απομακρύνονται η μία από την άλλη, με αποτέλεσμα όσο περνάει ο καιρός να μοιάζουν όλο και λιγότερο. Μια από τις πρώτες σημαντικές διαφορές ήταν στην υλοποίηση των generic object . Η C# παρέχει

"πρώτης-κλάσης" generic objects τα οποία μπορούν να χρησιμοποιηθούν σαν οποιαδήποτε άλλη κλάση με τον κώδικα να εκτελείται κατά τη διάρκεια της φόρτωσής της. Αντιθέτως τα generic object της Java παρέχονται από τη γλώσσα κατά τη διάρκεια της σύνταξής της και δεν επηρεάζουν τον ήδη υπάρχων κώδικα. Επιπλέον η C# πρόσθεσε κάποια ακόμα χαρακτηριστικά στην τρίτη έκδοσή της τα οποία επιτρέπουν στους προγραμματιστές να χρησιμοποιούν τεχνικές όπως τα closures.

1.7 Photoshop

Το Photoshop, χρησιμοποιήθηκε στην εργασία για την επεξεργασία των εικόνων της εφαρμογής και για την επεξεργασία εικόνων του βιβλίου της εργασίας. Για το Photoshop δεν θα γίνει αναλυτική παρουσίαση, γιατί το εν λόγο λογισμικό θα μπορούσε να καλύψει από μόνο του την θεματολογία μιας ολόκληρης πτυχιακής εργασίας. Θα γίνει όμως μια σύντομη αναφορά σε κάποιες γενικές πληροφορίες και μια ιστορική αναδρομή.

Το Adobe Photoshop ή απλά Photoshop, είναι ένα πρόγραμμα επεξεργασίας γραφικών που αναπτύχθηκε και κυκλοφόρησε από την Adobe Systems. Αυτή τη στιγμή αποτελεί τον ηγέτη της αγοράς των προγραμμάτων επεξεργασίας εικόνων, και είναι το προϊόν σήμα κατατεθέν της Adobe Systems.

Το 1987 ο Τόμας Κνόλ, ένας φοιτητής του Πανεπιστημίου του Μίσιγκαν, ανέπτυξε ένα πρόγραμμα που εμφάνιζε εικόνες σε αποχρώσεις του γκρίζου σε μονοχρωματικό περιβάλλον. Αυτό το πρόγραμμα, το οποίο ονόμασε Display, τράβηξε την προσοχή του αδερφού του Τζον Κνόλ, ο οποίος πρότεινε στον Τόμας να αναπτύξει ένα πλήρες πρόγραμμα επεξεργασίας εικόνας. Ο Τόμας σε συνεργασία με τον αδερφό του, ανέπτυξε ένα πρόγραμμα, το οποίο ονόμασαν ImagePro. Αργότερα το ίδιο έτος, ο Τόμας μετονόμασε το πρόγραμμα σε Photoshop. Ο Τζον Κνόλ παρουσίασε το πρόγραμμα σε μηχανικούς της Apple και της Adobe. Οι παρουσιάσεις ήταν επιτυχείς και η Adobe αποφάσισε να αγοράσει την άδεια και να το διανείμει τον Σεπτέμβριο του 1988.

1.8 ArgoUML

Το ArgoUML είναι μια εφαρμογή για δημιουργία UML διαγραμμάτων, γραμμένη σε Java και κυκλοφόρησε στο πλαίσιο του open source Eclipse Public License. Λόγω του ότι είναι μια εφαρμογή Java, είναι διαθέσιμη σε οποιαδήποτε πλατφόρμα υποστηρίζεται από την Java. Το ArgoUML είναι ένα ισχυρό αλλά εύκολο στη χρήση του διαδραστικό γραφικό περιβάλλον εργασίας, που υποστηρίζει το σχεδιασμό, την ανάπτυξη και την τεκμηρίωση αντικειμενοστραφών εφαρμογών. Το παρόν πρόγραμμα είναι κατάλληλο για σχεδιαστές εφαρμογών, προγραμματιστές, επιχειρηματικούς αναλυτές, αναλυτές συστημάτων και άλλους επαγγελματίες που εμπλέκονται με την ανάλυση, το σχεδιασμό και την ανάπτυξη εφαρμογών.

Κάποια σημαντικά ιστορικά στοιχεία για το ArgoUML είναι ότι, αναπτύχθηκε αρχικά στο UC Irvine από τον Jason E. Robbins, όπου ουσιαστικά ήταν η διδακτορική του διατριβή. Είναι πλέον ένα λογισμικό ανοικτού κώδικα που φιλοξενείται από την Tigris.org. Το έργο της ανάπτυξης του λογισμικού ArgoUML περιλαμβάνει σήμερα πάνω από 19.000 εγγεγραμμένους χρήστες και πάνω από 150 προγραμματιστές. Το 2003, το ArgoUML κέρδισε το ετήσιο βραβείο αναγνωστών του περιοδικού Software Development Magazine στην κατηγορία " Design and Analysis Tools ". Η ανάπτυξη ArgoUML παρόλο της δυσκολίες που έχει να αντιμετωπίσει, συνεχίζεται έστω και με ανασχετικούς παράγοντες παρέχοντας το λογισμικό στο ευρύ κοινό.

Κεφάλαιο 2ο

Σχεδίαση και Υλοποίηση της Εφαρμογής

2.1 Εισαγωγή

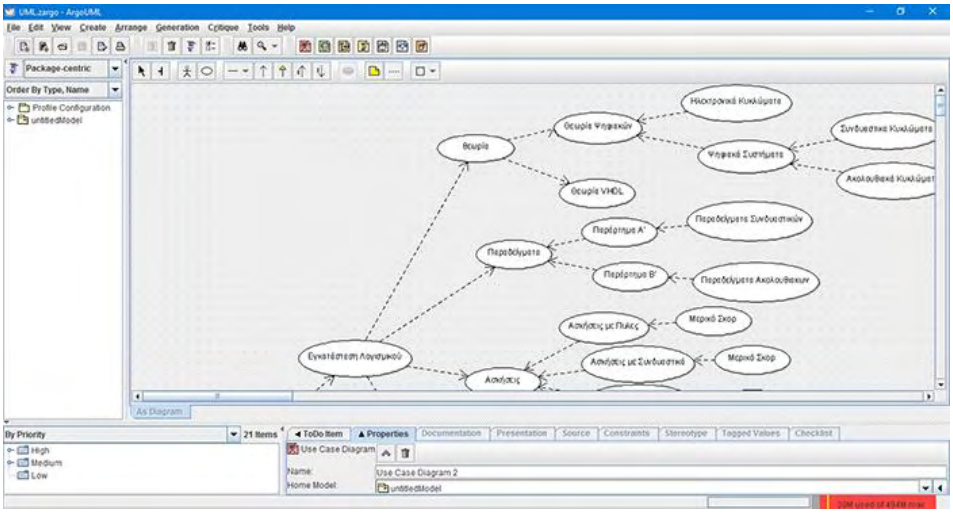
Στο κεφάλαιο αυτό θα παρουσιασθεί ο σχεδιασμός της εφαρμογής με χρήση διαγραμμάτων UML, όπου θα παρουσιασθούν αναλυτικά οι προσφερόμενες υπηρεσίες της εφαρμογής προς τον χρήστη και πως αυτές αργότερα υλοποιήθηκαν με την γλώσσα C# και προέκυψε το τελικό αποτέλεσμα . Επίσης στο κεφάλαιο αυτό θα παρουσιασθούν τα βασικότερα σημεία του κώδικα της εφαρμογής, έτσι ώστε ο αναγνώστης αυτής της διπλωματικής να αποκτήσει μια εικόνα τι συμβαίνει προγραμματιστικά στο υπόβαθρο της εφαρμογής και με ποιους μηχανισμούς επιτυγχάνεται η τελική εικόνα της εφαρμογής.

2.2 Προσφερόμενες λειτουργίες και σχεδιασμός τους

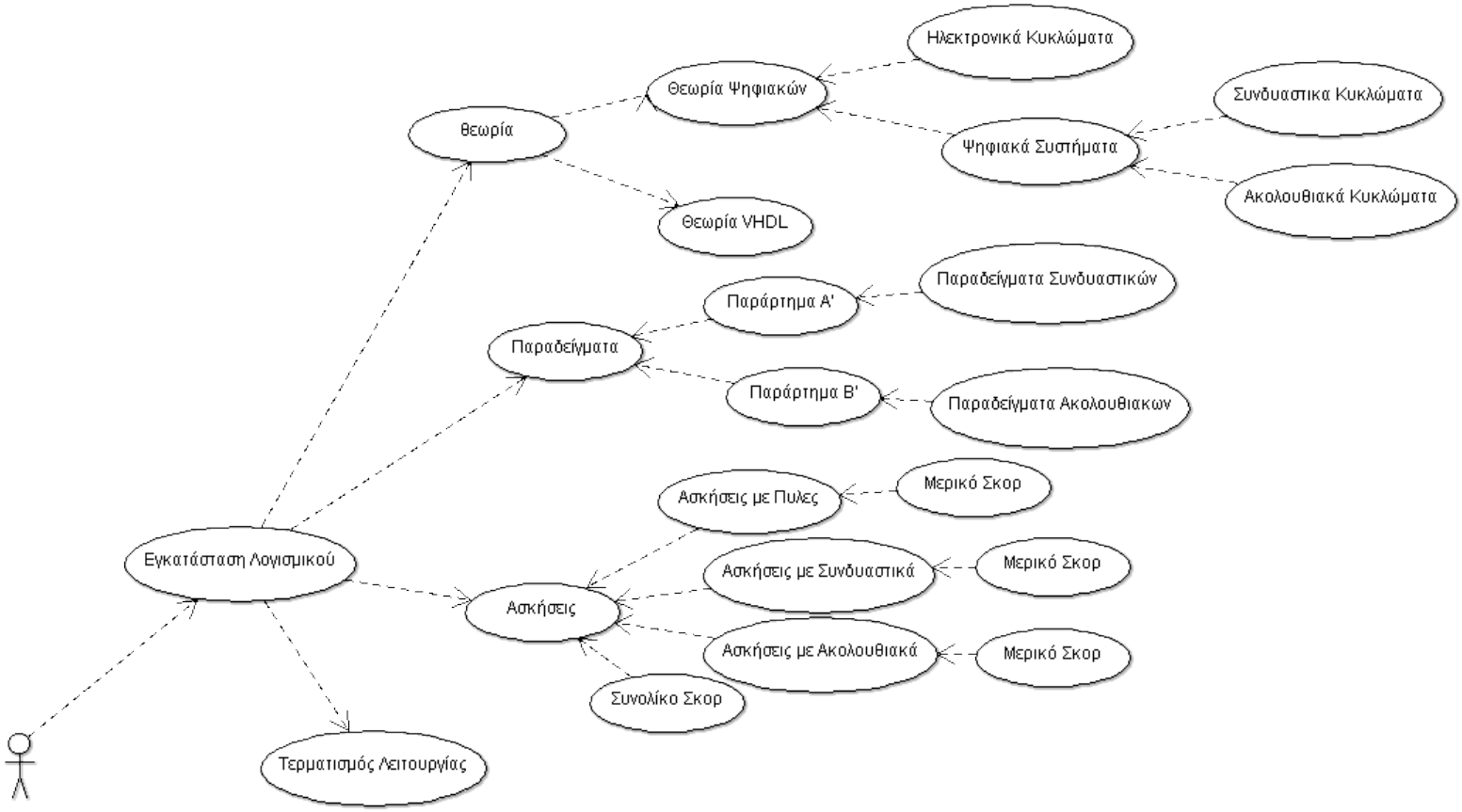
Οι λειτουργίες που εκτελούνται από την εφαρμογή που υλοποιήθηκε στα πλαίσια αυτής της διπλωματικής εργασίας, είναι εκπαιδευτικού περιεχομένου και αφορούν την γλώσσα περιγραφής υλικού VHDL. Αναλυτικότερα, εμπεριέχει αναφορές σχετικές με την θεωρία ηλεκτρονικών κυκλωμάτων, την γενικότερη θεωρία των ψηφιακών συστημάτων και ειδικές αναφορές στην θεωρία συνδυαστικών και ακολουθιακών κυκλωμάτων με τα οποία ασχολείται κατά κόρων η γλώσσα VHDL. Επίσης, στην εφαρμογή εκτός από την λειτουργία της θεωρίας εμπεριέχετε και η λειτουργία των παραδειγμάτων όπου περιλαμβάνει τα ποιο διαδεδομένα συνδυαστικά και ακολουθιακά κυκλώματα με τις λύσεις τους σε VHDL, έτσι ώστε ο χρήστης να πάρει μια πρώτη άποψη με το πώς είναι δομημένο ένα πρόγραμμα σε VHDL και πως γίνεται η αναπαράσταση του κυκλώματος στο προγραμματιστικά στην γλώσσα αυτή. Η σημαντικότερη λειτουργία της εφαρμογής είναι οι ασκήσεις, όπου ο χρήστης

μπορεί να εφαρμόσει την θεωρία στην πράξη ή να κάνει ένα πρόχειρο έλεγχο των γνώσεων του και να ανατρέξει ξανά στο θεωρητικό μέρος της εφαρμογής, μιας και το πρακτικό κομμάτι των ασκήσεων καλύπτει μια μεγάλη γκάμα ακολουθιακών και συνδυαστικών προβλημάτων που πρέπει να υλοποιηθούν προγραμματιστικά σε VHDL. Η λειτουργία των ασκήσεων αλλά και γενικότερα όλες οι λειτουργίες της εφαρμογής, αποτελούν ένα πολύ σημαντικό εργαλείο εξάσκησης, για τα άτομα που θέλουν να κάνουν τα πρώτα τους βήματα στο προγραμματισμό με VHDL.

Για να γίνουν καλύτερα αντιληπτές οι λειτουργίες της εφαρμογής στον αναγνώστη που δεν την έχει χρησιμοποιήσει, θεωρήθηκε σκόπιμο σε αυτό το σημείο να γίνει παρουσίαση του διαγράμματος περιπτώσεων χρήσης όπου στηρίχθηκε δομικά όλη η κατασκευή της εφαρμογής. Το διάγραμμα περιπτώσεων χρήσης είναι το ποιο σημαντικό κομμάτι του αρχικού σχεδιασμού της εφαρμογής καθώς παρουσιάζονται όλες οι λειτουργίες της εφαρμογής, καθώς και ποιες αλληλεπιδράσεις και συσχετίσεις υπάρχουν μεταξύ της εφαρμογής και του χρήστη. Με λίγα λόγια μας δίνει μια σαφή και συνεπή περιγραφή για το τι θα πρέπει να κάνει το σύστημα και ποιες είναι λειτουργικές απαιτήσεις του. Το διάγραμμα περιπτώσεων χρήσης που θα παρουσιασθεί σε αυτό το σημείο έχει δημιουργηθεί με το λογισμικό ArgoUML όπου στο προηγούμενο κεφάλαιο έγινε λεπτομερής αναφορά και θα παρουσιάσει το σύνολο των λειτουργιών της εφαρμογής.



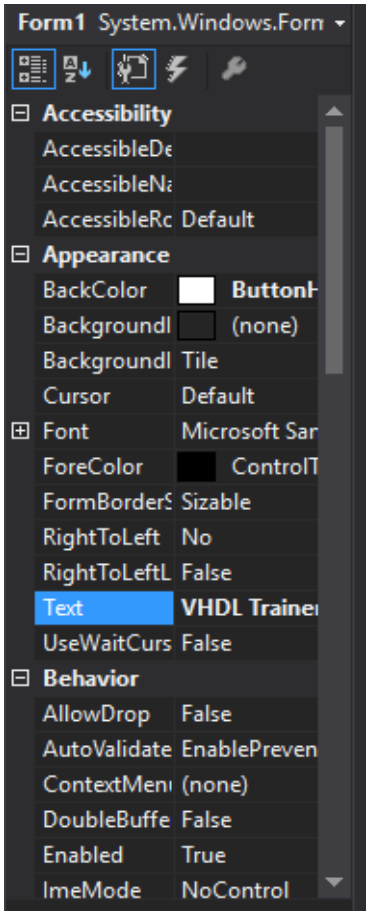
Εικόνα 1: Περιβάλλον ArgoUML



Εικόνα 2: Διάγραμμα Περιπτώσεων Χρήσης

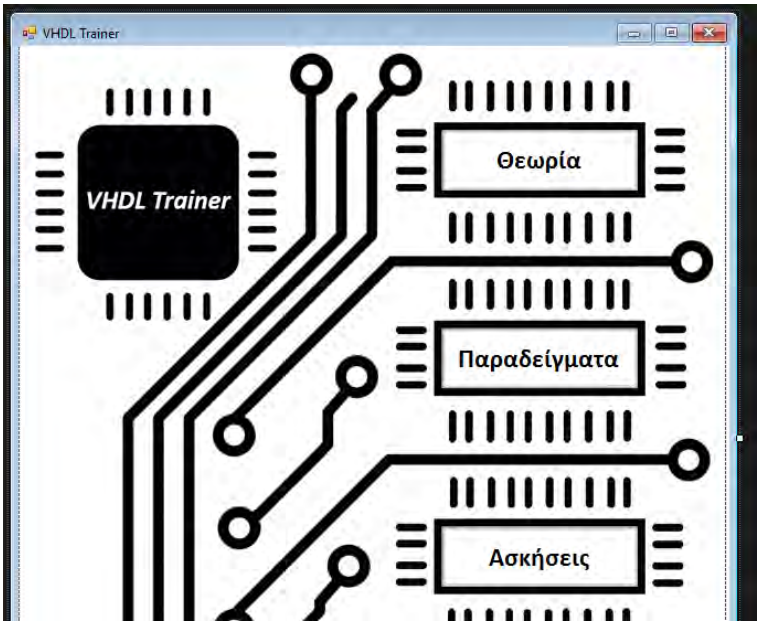
2.3 Βασικό μενού επιλογών

Σε αυτό το σημείο θα παρουσιασθεί η κατασκευή του παραθύρου του βασικού μενού επιλογών. Για να αναπτυχθεί η συγκεκριμένη λειτουργία έγινε χρήσης της γλώσσας C Sharp, έγιναν κάποιες παραμετροποιήσεις των γραφικών λειτουργιών του παραθύρου, μέσω του λογισμικού Visual Studio 2015 και επίσης έγινε χρήση εικόνων που επεξεργάστηκαν με το Photoshop. Το παράθυρο αυτό ουσιαστικά αποτελεί την ραχοκοκαλιά της εφαρμογής, αφού από εκεί γίνονται όλες οι επιλογές του χρήστη σχετικά με το ποια λειτουργία θέλει να ακολουθήσει. Η εικόνα που ακολουθεί είναι στιγμιότυπο από το λογισμικό Visual Studio, όπου σε αυτό γίνονται οι αρχικές ρυθμίσεις του παραθύρου σχετικά με το μέγεθος, χρώμα κτλ.



Εικόνα 3: Επεξεργασία φόρμας Visual Studio

Τώρα θα παρουσιασθεί ένα στιγμιότυπο της μορφής του παραθύρου του βασικού μενού.



Εικόνα 4: Βασικό μενού επιλογών

Εφόσον γίνει η τελική γραφική μορφοποίηση του παραθύρου δίνονται στις οντότητες του παραθύρου μέσω κώδικα λειτουργίες, η εικόνα που ακολουθεί είναι ο κώδικας C Sharp που αντιστοιχεί στο παράθυρο αυτό.

```
namespace VHDLTrainer
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)  //Θεωρία
        {
            Form2 f2 = new Form2();
            f2.ShowDialog();
        }

        private void button2_Click(object sender, EventArgs e)  //Παραδείγματα
        {
            Form4 f4 = new Form4();
            f4.ShowDialog();
        }

        private void button3_Click(object sender, EventArgs e)  //Ασκήσεις
        {
            Application.Exit();
        }

        private void button4_Click(object sender, EventArgs e)  //Τερματισμός
        {
            Form3 f3 = new Form3();
            f3.ShowDialog();
        }
    }
}
```

Εικόνα 5: Κώδικας C# βασικού παραθύρου επιλογών

2.4 Φόρμα προβολής θεωρίας

Η λειτουργία προβολής του θεωρητικού μέρους της γλώσσας VHDL καθώς και της βασικής θεωρίας των ψηφιακών συστημάτων, υλοποιήθηκε με την χρήση γλώσσας C Sharp, οι εικόνες που είναι το βασικότερο συστατικό στοιχείο αυτής της λειτουργίας επεξεργάστηκαν και μορφοποιήθηκαν σε συμβατές μορφές, με το λογισμικό της Adobe, το Photoshop. Η αρχική παραμετροποίηση του παραθύρου έγινε πάλι όπως και πριν μέσω του λογισμικού του Visual Studio, όποτε δεν κρίνεται σκόπιμο να παρουσιαστεί ξανά αυτή η ενέργεια. Οπότε σε αυτό το σημείο θα παρουσιασθεί ένα στιγμιότυπο της μορφής του παραθύρου της θεωρίας.

Εικόνα 7: Φόρμα προβολής Θεωρίας

Όταν τελειώσει η τελική γραφική μορφοποίηση του παραθύρου δίνονται στις οντότητες του παραθύρου μέσω κώδικα C Sharp λειτουργίες. Λόγω ότι ο κώδικας της λειτουργίας προβολής της θεωρίας είναι μακροσκελής, θα γίνει παρουσίαση του αρχικού τμήματος καθώς και του τελικού τμήματος για χάρη συντομίας μιας και είναι περίπου στις 1000 γραμμές.

```
namespace VHDLTrainer
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
        }

        //ΑΡΧΗ ΣΕΛΙΔΑΣ 0
        private void button1_Click(object sender, EventArgs e)  //>>
        {
            panel1.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)  //<<
        {
            button2.Enabled = false;
        }

        private void button3_Click(object sender, EventArgs e)  //Θεωρία Ψηφιακών
        {
            button3.Enabled = false;
        }
        private void button4_Click(object sender, EventArgs e)  //Θεωρία VHDL
        {
            panel1.Visible = true;
            panel2.Visible = true;
            panel3.Visible = true;
            panel4.Visible = true;
            panel5.Visible = true;
            panel6.Visible = true;
            panel7.Visible = true;
        }

        private void button5_Click(object sender, EventArgs e) //Τερματισμός
        {
            this.Close();
        }
        //ΤΕΛΟΣ ΣΕΛΙΔΑΣ 0
```

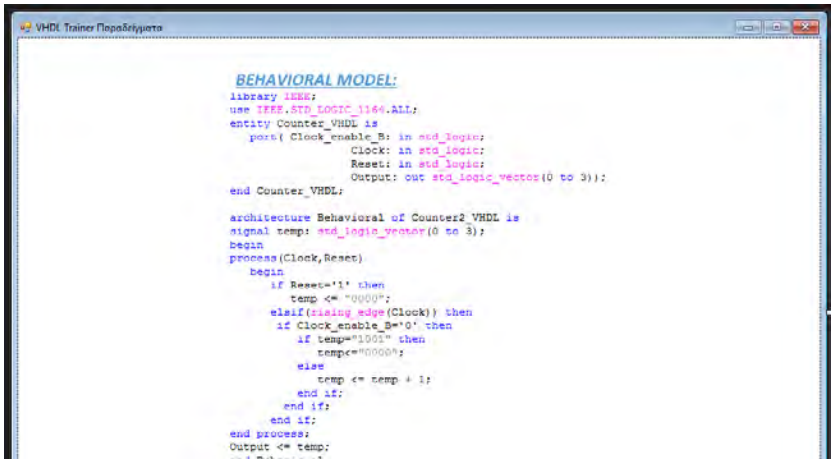
Εικόνα 8: Κώδικας C# παραθύρου προβολής θεωρίας μέρος Α'

```
        //ΑΡΧΗ ΣΕΛΙΔΑΣ 22
        private void button111_Click(object sender, EventArgs e)
        {
            button111.Enabled = false;
        }

        private void button112_Click(object sender, EventArgs e)
        {
            panel22.Visible = false;
        }

        private void button113_Click(object sender, EventArgs e)
        {
            panel1.Visible = false;
            panel2.Visible = false;
            panel3.Visible = false;
            panel4.Visible = false;
            panel5.Visible = false;
            panel6.Visible = false;
            panel7.Visible = false;
            panel8.Visible = false;
            panel9.Visible = false;
            panel10.Visible = false;
            panel11.Visible = false;
            panel12.Visible = false;
            panel13.Visible = false;
            panel14.Visible = false;
            panel15.Visible = false;
            panel16.Visible = false;
            panel17.Visible = false;
            panel18.Visible = false;
            panel19.Visible = false;
            panel20.Visible = false;
            panel21.Visible = false;
            panel22.Visible = false;

        }

        private void button114_Click(object sender, EventArgs e)
        {
            panel1.Visible = true;
            panel2.Visible = true;
            panel3.Visible = true;
            panel4.Visible = true;
            panel5.Visible = true;
            panel6.Visible = true;
            panel7.Visible = true;
            panel8.Visible = false;
            panel9.Visible = false;
            panel10.Visible = false;
            panel11.Visible = false;
            panel12.Visible = false;
            panel13.Visible = false;
            panel14.Visible = false;
            panel15.Visible = false;
            panel16.Visible = false;
            panel17.Visible = false;
            panel18.Visible = false;
            panel19.Visible = false;
            panel20.Visible = false;
            panel22.Visible = false;
        }

        private void button115_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        //ΤΕΛΟΣ ΣΕΛΙΔΑΣ 22
    }
}
```

Εικόνα 9: Κώδικας C# παραθύρου προβολής θεωρίας μέρος Β'

2.5 Φόρμα προβολής παραδειγμάτων

Στο σημείο αυτό είναι σκόπιμο να αναφερθούν κάποιες πληροφορίες σχετικά με τα συστατικά μέρη της λειτουργίας προβολής παραδειγμάτων. Το παράθυρο της λειτουργίας προβολής παραδειγμάτων είναι παρόμοιο με το παράθυρο που εξετάσαμε πριν, της λειτουργίας προβολής θεωρίας. Ουσιαστικά είναι αναπτυγμένο σε C# όπως και τα υπόλοιπα μέρη της εφαρμογής. Οι εικόνες όπως και στην προηγούμενη ενότητα αποτελούν το βασικό συστατικό μέρος και είναι κατασκευασμένες και μορφοποιημένες στο Photoshop. Οι αρχικές ιδιότητες του παραθύρου έχουν δοθεί, όπως και πριν μέσω του Visual Studio. Κρίνεται σκόπιμο λοιπόν να παρουσιασθεί ένα στιγμιότυπο της μορφής του παραθύρου των παραδειγμάτων, ώστε ο αναγνώστης να έχει εικόνα του πράγματος για το οποίο γίνεται λόγος.

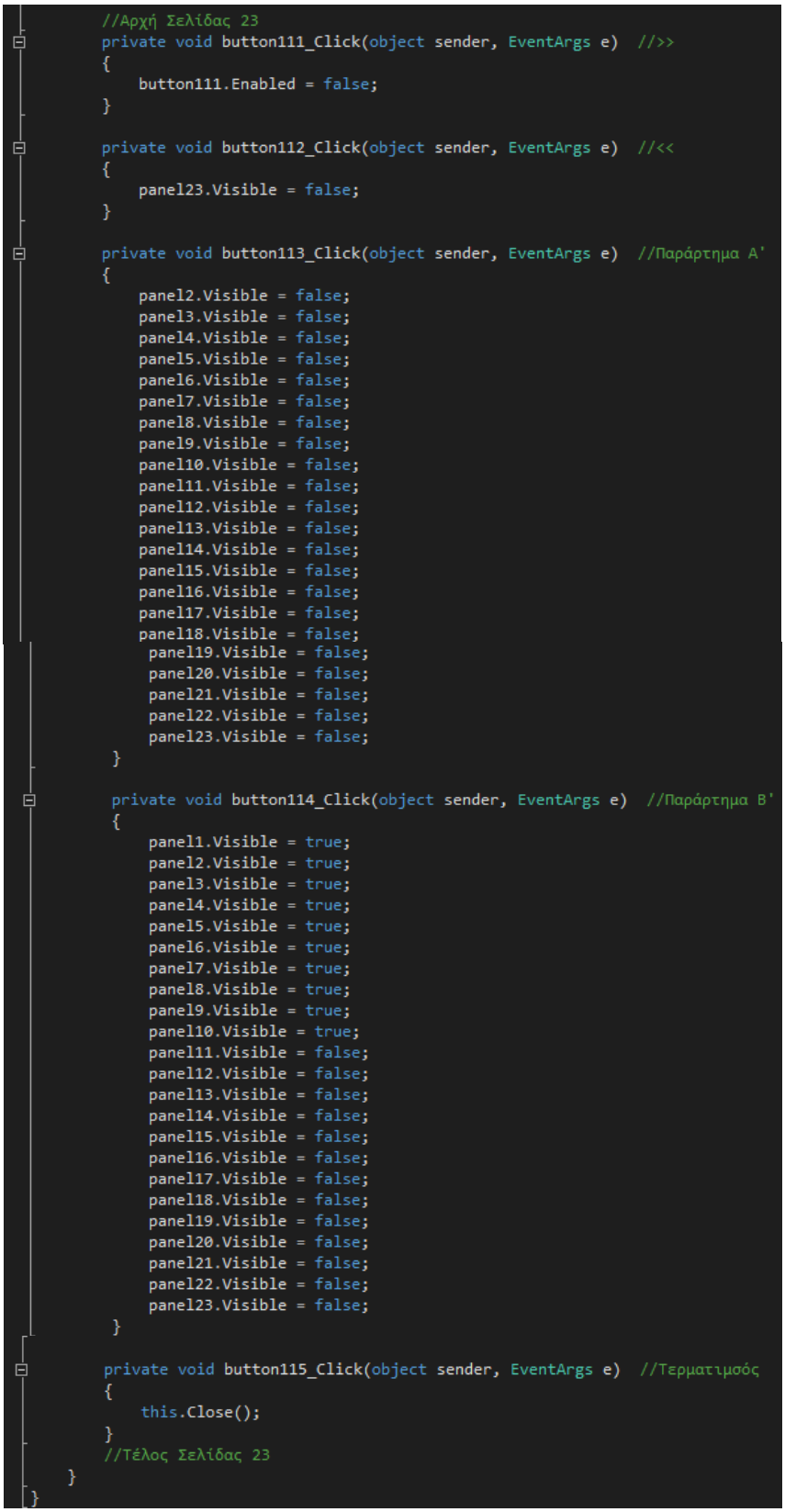


Εικόνα 11: Φόρμα προβολής παραδειγμάτων

Το τελευταίο στάδιο ανάπτυξης της λειτουργίας προβολής παραδειγμάτων, είναι η ανάπτυξη του κώδικα. Η ανάπτυξη του κώδικα διενεργείτε για κάθε οντότητα που υπάρχει στο αρχικό παράθυρο σε ξεχωριστές κλάσεις και παρουσιάζεται στις ακόλουθες εικόνες. Λόγω ότι ο κώδικας και σε αυτή την περίπτωση είναι μεγάλος θα γίνει αποσπασματική παρουσίαση του.

```
namespace VHDLTrainer
{
    public partial class Form3 : Form
    {
        public Form3()
        {
            InitializeComponent();
        }

        //Αρχή Σελίδας 1
        private void button1_Click(object sender, EventArgs e)  //>>
        {
            panel2.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)  //<<
        {
            button2.Enabled = false;
        }

        private void button3_Click(object sender, EventArgs e)  //Παράρτημα Α'
        {
            button3.Enabled = false;
        }

        private void button4_Click(object sender, EventArgs e)  //Παράρτημα Β'
        {
            panel1.Visible = true;
            panel2.Visible = true;
            panel3.Visible = true;
            panel4.Visible = true;
            panel5.Visible = true;
            panel6.Visible = true;
            panel7.Visible = true;
            panel8.Visible = true;
            panel9.Visible = true;
            panel10.Visible = true;
        }

        private void button5_Click(object sender, EventArgs e)  //Τερματισμός
        {
            this.Close();
        }
        //Τέλος Σελίδας 1
```

Εικόνα 12: Κώδικας C# παραθύρου προβολής παραδειγμάτων μέρος Α'

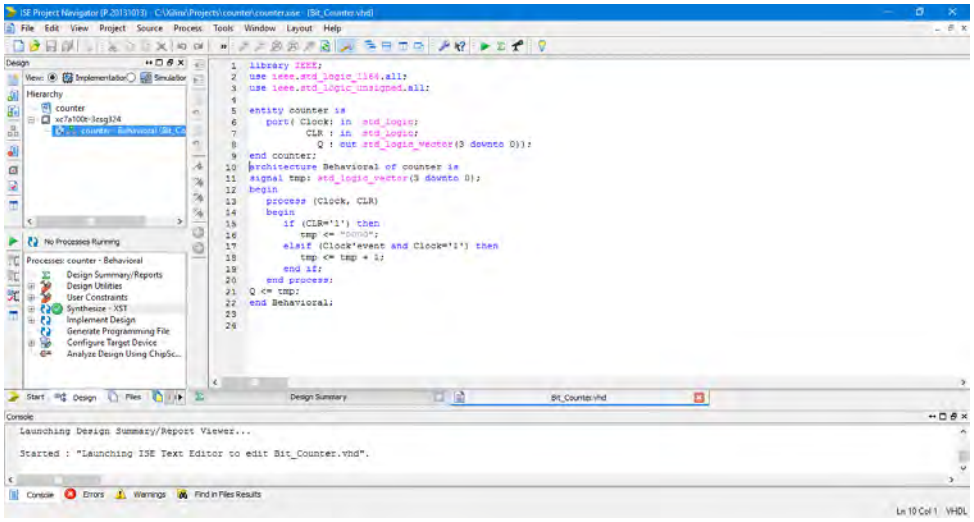


```
        //Αρχή Σελίδας 23
        private void button111_Click(object sender, EventArgs e)  //>>
        {
            button111.Enabled = false;
        }

        private void button112_Click(object sender, EventArgs e)  //<<
        {
            panel23.Visible = false;
        }

        private void button113_Click(object sender, EventArgs e)  //Παράρτημα Α'
        {
            panel2.Visible = false;
            panel3.Visible = false;
            panel4.Visible = false;
            panel5.Visible = false;
            panel6.Visible = false;
            panel7.Visible = false;
            panel8.Visible = false;
            panel9.Visible = false;
            panel10.Visible = false;
            panel11.Visible = false;
            panel12.Visible = false;
            panel13.Visible = false;
            panel14.Visible = false;
            panel15.Visible = false;
            panel16.Visible = false;
            panel17.Visible = false;
            panel18.Visible = false;
             panel19.Visible = false;
             panel20.Visible = false;
             panel21.Visible = false;
             panel22.Visible = false;
             panel23.Visible = false;
         }

         private void button114_Click(object sender, EventArgs e)  //Παράρτημα Β'
         {
             panel1.Visible = true;
             panel2.Visible = true;
             panel3.Visible = true;
             panel4.Visible = true;
             panel5.Visible = true;
             panel6.Visible = true;
             panel7.Visible = true;
             panel8.Visible = true;
             panel9.Visible = true;
             panel10.Visible = true;
             panel11.Visible = false;
             panel12.Visible = false;
             panel13.Visible = false;
             panel14.Visible = false;
             panel15.Visible = false;
             panel16.Visible = false;
             panel17.Visible = false;
             panel18.Visible = false;
             panel19.Visible = false;
             panel20.Visible = false;
             panel21.Visible = false;
             panel22.Visible = false;
             panel23.Visible = false;
         }

        private void button115_Click(object sender, EventArgs e)  //Τερματισμός
        {
            this.Close();
        }
        //Τέλος Σελίδας 23
    }
}
```

Εικόνα 13: Κώδικας C# παραθύρου προβολής παραδειγμάτων μέρος Β'

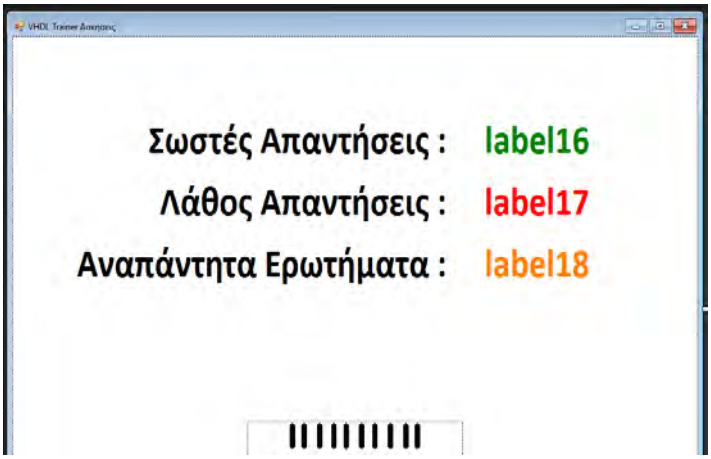
2.6 Φόρμα ασκήσεων

Η λειτουργία των ασκήσεων αποτελεί την ποιο σημαντική λειτουργία της εφαρμογής, αφού ο απώτερος σκοπός της είναι να διευρύνει της γνώσεις του χρήστη σχετικά με την γλώσσα VHDL, μέσω της εξάσκησης και της εφαρμογής της θεωρίας στην πράξη. Η δημιουργία της λειτουργίας των ασκήσεων ήταν πιο περίπλοκη από τις άλλες που έχουν αναφερθεί μέχρι στιγμής, γιατί εκτός από την χρήση της C Sharp και του Visual Studio, αναπτύχθηκαν και προγράμματα σε VHDL μέσω του λογισμικού XilinxISE. Για την δημιουργία και την επεξεργασία των εικόνων και σε αυτή την περίπτωση χρησιμοποιήθηκε το Adobe Photoshop. Επίσης στην λειτουργία των ασκήσεων όπως και στις δύο προηγούμενες, ο κώδικας σε C Sharp είναι μακροσκελέστατος οπότε θα παρουσιασθούν στιγμιότυπα από συγκεκριμένα τμήματα κώδικα που θεωρήθηκαν τα πιο σημαντικά. Πρώτα από όλα όμως θα παρουσιασθεί μια εικόνα που μέχρι στιγμής δεν έχει ξανά παρουσιασθεί στην διπλωματική και αυτή αφορά το λογισμικό XilinxISE.



Εικόνα 14: Interface λογισμικού XilinxISE

Στην διπλανή εικόνα παρουσιάζεται η μορφή που έχει το παράθυρο των ασκήσεων και συγκεκριμένα φαίνεται η τελική σελίδα, που εμφανίζεται το σκορ που συγκέντρωσε ο χρήστης.



Εικόνα 16: Φόρμα ασκήσεων

Στο σημείο αυτό θα γίνει παρουσίαση του κώδικα της λειτουργίας των ασκήσεων. Το πρώτο τμήμα κώδικα που παρουσιάζεται είναι από την πρώτη σελίδα της φόρμας με τα κουμπιά των τριών ενοτήτων των ασκήσεων, το κουμπί του συνολικού σκορ καθώς και του κουμπιού τερματισμού.

```
namespace VHDLTrainer
{
    public partial class Form4 : Form
    {
        public Form4()
        {
            InitializeComponent();
        }
        int score, neg_score, neutral_score, score_s, neg_score_s,
            neutral_score_s, score_a, neg_score_a, neutral_score_a;
        int score_all, neg_score_all, neutral_score_all;
        //Αρχή Σελίδας 0
        private void button1_Click(object sender, EventArgs e)  //Αριστερά
        {
            panel3.Visible = true;
        }

        private void button2_Click(object sender, EventArgs e)  //Μέση
        {
            panel3.Visible = true;
            panel4.Visible = true;
            panel5.Visible = true;
            panel6.Visible = true;
            panel7.Visible = true;
            panel8.Visible = true;
            panel9.Visible = true;
            panel10.Visible = true;
            panel11.Visible = true;
            panel12.Visible = true;
            panel13.Visible = true;
            panel14.Visible = true;
            panel15.Visible = true;
            panel16.Visible = true;
            panel17.Visible = true;
            panel18.Visible = true;
            panel19.Visible = true;
            button52.Enabled = false;
        }

        private void button3_Click(object sender, EventArgs e)  //Δεξιά
        {
            panel3.Visible = true;
            panel4.Visible = true;
            panel5.Visible = true;
            panel6.Visible = true;
            panel7.Visible = true;
            panel8.Visible = true;
            panel9.Visible = true;
            panel10.Visible = true;
            panel11.Visible = true;
            panel12.Visible = true;
            panel13.Visible = true;
            panel14.Visible = true;
            panel15.Visible = true;
            panel16.Visible = true;
            panel17.Visible = true;
            panel18.Visible = true;
            panel19.Visible = true;
            panel20.Visible = true;
            panel21.Visible = true;
            panel22.Visible = true;
            panel23.Visible = true;
            panel24.Visible = true;
            panel25.Visible = true;
            panel26.Visible = true;
            panel27.Visible = true;
```

Εικόνα 17: Κώδικας C# παραθύρου ασκήσεων τμήμα πρώτο μέρος Α'

```
            panel28.Visible = true;
            panel29.Visible = true;
            panel30.Visible = true;
            panel31.Visible = true;
            panel32.Visible = true;
            panel33.Visible = true;
            panel34.Visible = true;
            panel35.Visible = true;
            button99.Enabled = false;
        }

        private void button4_Click(object sender, EventArgs e)  //Συνολικό Σκορ
        {
            score_all = score + score_s + score_a;
            neg_score_all = neg_score + neg_score_s + neg_score_a;
            neutral_score_all = neutral_score + neutral_score_s + neutral_score_a;
            string score22 = score_all.ToString();
            string score23 = neg_score_all.ToString();
            string score24 = neutral_score_all.ToString();

            if (score_all > 20)
            {
                Form f7 = new Form7(score22, score23, score24);
                f7.ShowDialog();
            }
            else
            {
                Form f8 = new Form8(score22, score23, score24);
                f8.ShowDialog();
            }

        }
        private void button5_Click(object sender, EventArgs e)
        {

        }

        private void button6_Click(object sender, EventArgs e)  //Τερματισμός
        {
            this.Close();
        }
        //Τέλος Σελίδας 0
```

Εικόνα 18: Κώδικας C# παραθύρου ασκήσεων τμήμα πρώτο μέρος Β'

Το δεύτερο τμήμα κώδικα που παρουσιάζεται είναι από την τελευταία και την προτελευταία σελίδα της λειτουργίας των ασκήσεων. Στην προτελευταία σελίδα από άποψη κώδικα θα δούμε το κουμπί που οδηγεί στην προηγούμενη σελίδα, το κουμπί που οδηγεί στην επόμενη σελίδα, το κουμπί υποβολής που περιέχει έναν αλγόριθμο με μετρητές για την εξαγωγή των αποτελεσμάτων. Στην τελευταία σελίδα υπάρχουν κάποια lebels που πυροδοτούνται για να πάρουν τιμές μέσω του κουμπιού επόμενο της προτελευταίας σελίδας και ένα κουμπί τερματισμού που επιστρέφει τον χρήστη στην πρώτη σελίδα της λειτουργίας.

```
        //Αρχή Σελίδας 42
        private void radioButton118_CheckedChanged(object sender, EventArgs e) //A
        {

        }

        private void radioButton119_CheckedChanged(object sender, EventArgs e) //B
        {

        }

        private void radioButton120_CheckedChanged(object sender, EventArgs e) //Γ
        {

        }

        private void button126_Click(object sender, EventArgs e) //Προηγ.
        {
            panel44.Visible = false;
        }
        private void button127_Click(object sender, EventArgs e) //Υποβολη
        {
            if (radioButton119.Checked)
            {
                Form5 f5 = new Form5();
                f5.ShowDialog();
                score_a = score_a + 1;
                button127.Enabled = false;
            }
            else if (radioButton120.Checked)
            {
                Form6 f6 = new Form6();
                f6.ShowDialog();
                button127.Enabled = false;
                neg_score_a = neg_score_a + 1;
            }
            else if (radioButton118.Checked)
            {
                Form6 f7 = new Form6();
                f7.ShowDialog();
                button127.Enabled = false;
                neg_score_a = neg_score_a + 1;
            }
        }

        private void button128_Click(object sender, EventArgs e) //Επόμενο
        {
            panel45.Visible=true;

           string score7 = score_a.ToString();
           label16.Text = score7;
           string score8 = neg_score_a.ToString();
           label17.Text = score8;
```

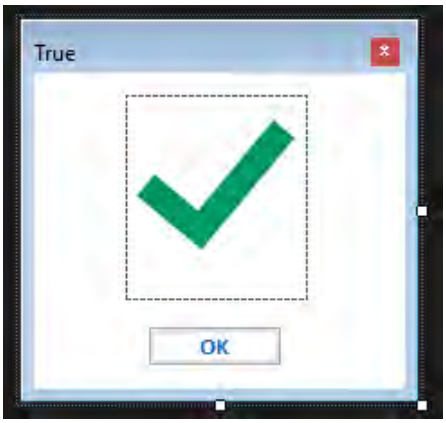
Εικόνα 19: Κώδικας C# παραθύρου ασκήσεων τμήμα δεύτερο μέρος Α'

```
            neutral_score_a = 10 - neg_score_a - score_a;
            string score9 = neutral_score_a.ToString();
            label18.Text = score9;
        }
        //Τέλος Σελίδας 42

        //Αρχή Σελίδας 43
        private void panel45_Paint(object sender, PaintEventArgs e)
        {

        }
        private void button129_Click(object sender, EventArgs e)
        {
            panel3.Visible = false;
            panel4.Visible = false;
            panel5.Visible = false;
            panel6.Visible = false;
            panel7.Visible = false;
            panel8.Visible = false;
            panel9.Visible = false;
            panel10.Visible = false;
            panel11.Visible = false;
            panel12.Visible = false;
            panel13.Visible = false;
            panel14.Visible = false;
            panel15.Visible = false;
            panel16.Visible = false;
            panel17.Visible = false;
            panel18.Visible = false;
            panel19.Visible = false;
            panel20.Visible = false;
            panel21.Visible = false;
            panel22.Visible = false;
            panel23.Visible = false;
            panel24.Visible = false;
            panel25.Visible = false;
            panel26.Visible = false;
            panel27.Visible = false;
            panel28.Visible = false;
            panel29.Visible = false;
            panel30.Visible = false;
            panel31.Visible = false;
            panel32.Visible = false;
            panel33.Visible = false;
            panel34.Visible = false;
            panel35.Visible = false;
            panel36.Visible = false;
            panel37.Visible = false;
            panel38.Visible = false;
            panel39.Visible = false;
            panel40.Visible = false;
            panel41.Visible = false;
            panel42.Visible = false;
            panel43.Visible = false;
            panel44.Visible = false;
        }
        //Τέλος Σελίδας 43
    }
}
```

Εικόνα 20: Κώδικας C# παραθύρου ασκήσεων τμήμα δεύτερο μέρος Β'

2.7 Φόρμα προβολής σωστής και λάθος απάντησης

Η λειτουργία προβολής της σωστής και της λάθος απάντησης εμφανίζεται μετά από κάθε σωστή ή λάθος απάντηση του χρήστη στις ερωτήσεις πολλαπλής επιλογής. Η λειτουργία αυτή υλοποιήθηκε με γλώσσα C Sharp, όπου καλείται η εν λόγω λειτουργία μετά τον κατάλληλο έλεγχο που διενεργείται στη λειτουργία των ασκήσεων. Παρακάτω παρουσιάζονται μερικές χαρακτηριστικές εικόνες της λειτουργίας αυτής.

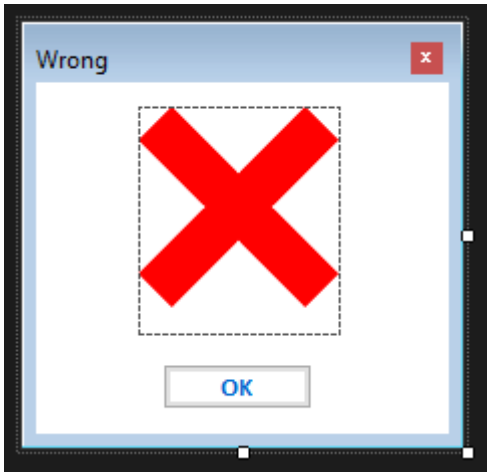


Εικόνα 22: Φόρμα σωστής απάντησης

Ο κώδικας C Sharp αυτής της λειτουργίας είναι πολύ απλός και ουσιαστικά καλείται μια κλάση για το κλείσιμο της φόρμας.

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    public partial class Form5 : Form
    {
        public Form5()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Εικόνα 23: Κώδικας C# λειτουργίας σωστής απάντησης



Εικόνα 25: Φόρμα λάθος απάντησης

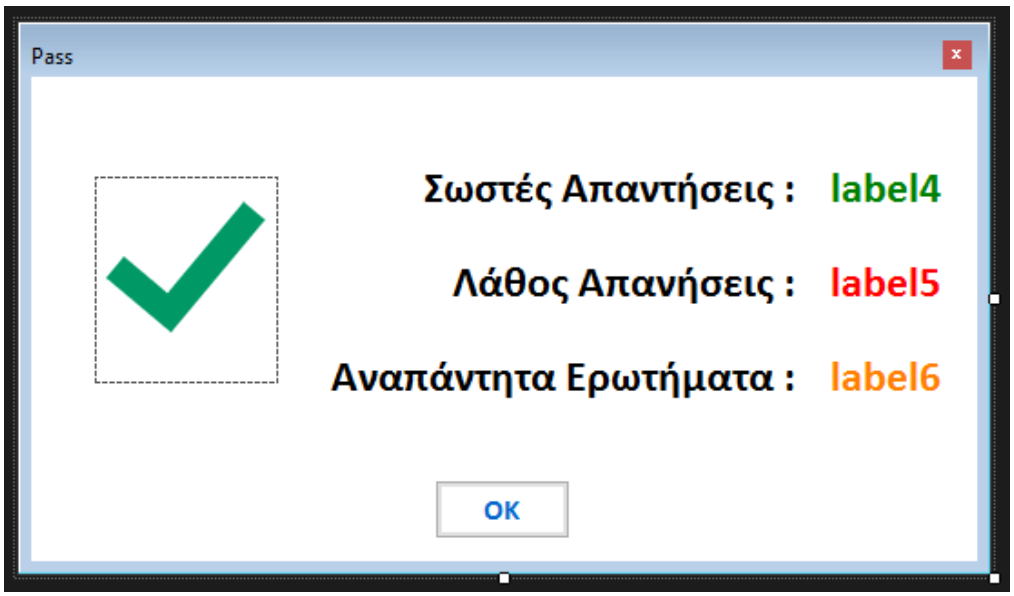
Ο κώδικας C Sharp αυτής της λειτουργίας όπως και της λειτουργίας προβολής της λάθος απάντησης, είναι πολύ απλώς και ουσιαστικά καλείται μια κλάση για το κλείσιμο της φόρμας.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    public partial class Form6 : Form
    {
        public Form6()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Εικόνα 26: Κώδικας C# λειτουργίας λάθος απάντησης

2.9 Φόρμα προβολής συγκεντρωτικών αποτελεσμάτων

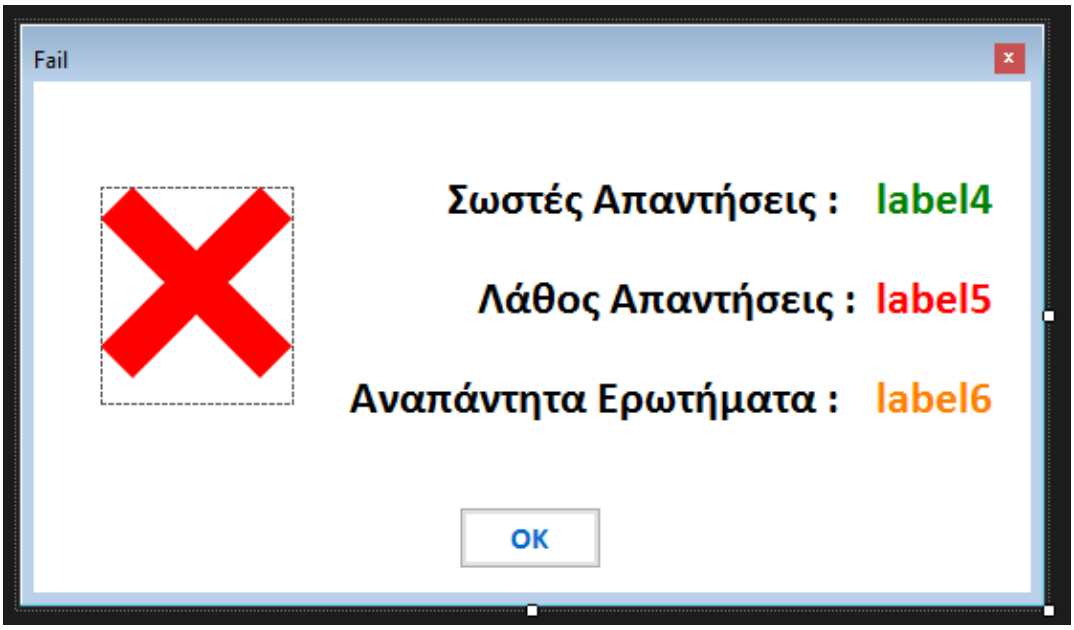
Η λειτουργία αυτή εμφανίζει τα συγκεντρωτικά αποτελέσματα με το σκορ που συγκέντρωσε ο χρήστης και στις τρείς ενότητες με τις ασκήσεις και ανάλογα αν το σκορ είναι πάνω από 20 εμφανίζει θετικό σύμβολο αλλιώς εμφανίζεται αρνητικό σύμβολο. Η λειτουργία αυτή αναπτύχθηκε με την γλώσσα C Sharp όπως και οι προηγούμενες και παρακάτω θα παρουσιασθούν μερικές χαρακτηριστικές εικόνες της λειτουργίας αυτής.



Εικόνα 28: Φόρμα με θετικά συγκεντρωτικά αποτελέσματα

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    public partial class Form7 : Form
    {
        public Form7(string mystring1, string mystring2, string mystring3)
        {
            InitializeComponent();
            label4.Text = mystring1;
            label5.Text = mystring2;
            label6.Text = mystring3;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Εικόνα 29: Κώδικας C# λειτουργίας για θετικά συγκεντρωτικά αποτελέσματα



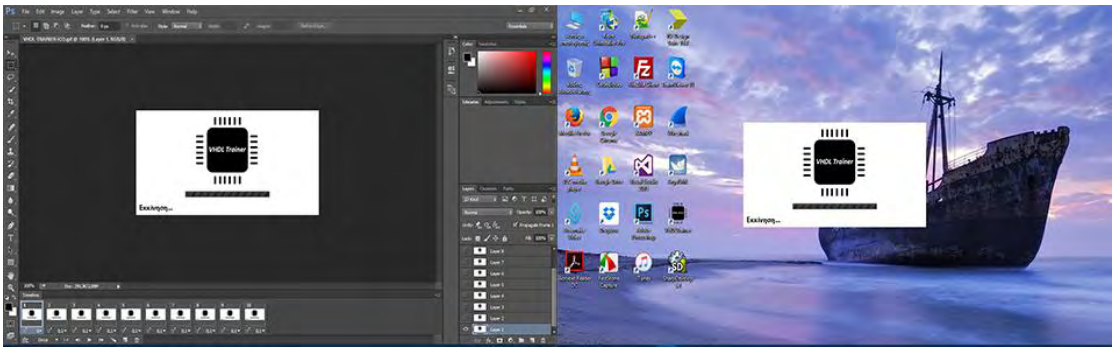
Εικόνα 31: Φόρμα με αρνητικά συγκεντρωτικά αποτελέσματα

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    public partial class Form8 : Form
    {
        public Form8(string mystring1, string mystring2, string mystring3)
        {
            InitializeComponent();
            label4.Text = mystring1;
            label5.Text = mystring2;
            label6.Text = mystring3;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Εικόνα 32: Κώδικας C# λειτουργίας για αρνητικά συγκεντρωτικά αποτελέσματα

2.9 Splash screen με χρήση animation

Κατά την εκκίνηση της εφαρμογής εμφανίζεται μία splash screen, που περιέχει μια εικόνα κινουμένων σχεδίων (animation) με το λογότυπο της εφαρμογής και μια γραμμή εκκίνησης. Ύστερα από το πέρας ενός συγκεκριμένου χρονικού ορίου ο χρήστης οδηγείτε στην κεντρική οθόνη της εφαρμογής. Η εικόνα των κινούμενων σχεδίων (animation) κατασκευάστηκε με την χρήση του Adobe Photoshop.



Εικόνα 33: Animation της splash screen

Ο κώδικας που αναπτύχθηκε για την λειτουργία αυτή, είναι σε C Sharp όπως και για τις άλλες λειτουργίες της εφαρμογής, με την μόνη διαφοροποίηση ότι εδώ γίνετε επέμβαση στον κώδικα του κυρίου σημείου εισόδου της εφαρμογής ώστε να δοθεί η κατεύθυνση με ποια λειτουργία θα ξεκινάει η εφαρμογή.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    public partial class Form9 : Form
    {
        public Form9()
        {
            InitializeComponent();
            timer1.Enabled = true;
            timer1.Interval = 3000;
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            timer1.Stop();
            this.DialogResult = DialogResult.OK;
            this.Close();

        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VHDLTrainer
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Form9 sp = new Form9();
            if (sp.ShowDialog() == DialogResult.OK)
            {
                Application.Run(new Form1());
            }
        }
    }
}
```

Εικόνα 34: Κώδικας C# της splash screen

Κεφάλαιο 3ο

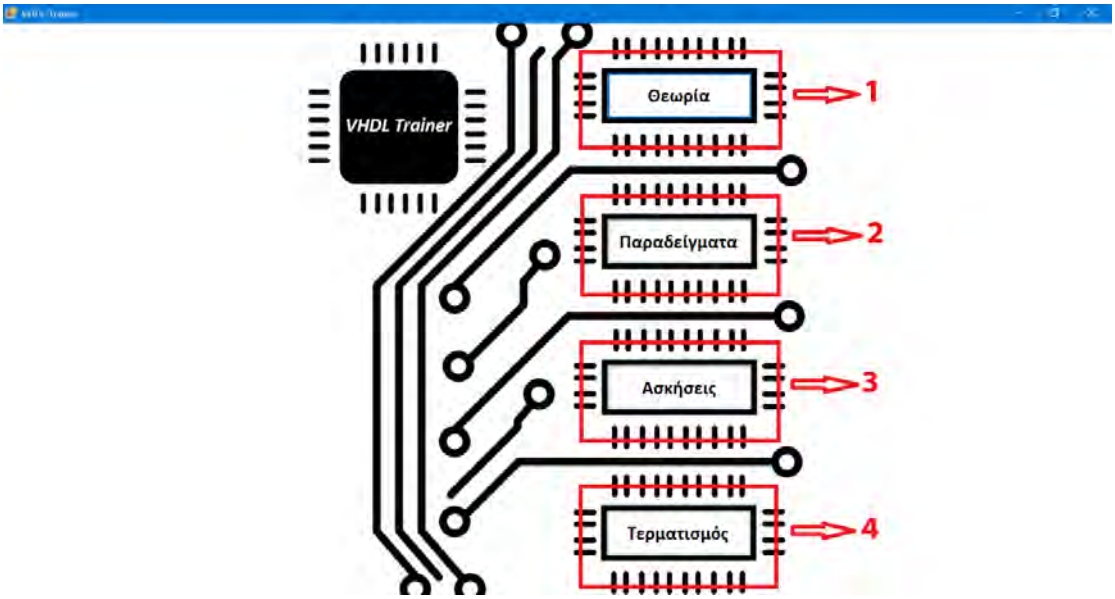
Λειτουργικότητα Εφαρμογής

3.1 Εισαγωγή

Στο κεφάλαιο αυτό θα γίνει παρουσίαση των δυνατοτήτων που προσφέρει η εφαρμογή στον χρήστη καθώς και ο τρόπος λειτουργίας της. Με άλλα λόγια το κεφάλαιο αυτό θα περιλαμβάνει αναλυτική περιγραφή της διεπαφής (interface) του χρήστη, με επισύναψη εικόνων, ώστε να γίνει ποιο κατανοητή η φιλοσοφία της εφαρμογής στον αναγνώστη αυτής της διπλωματικής εργασίας. Το κεφάλαιο αυτό επίσης θα μπορούσε να αποτελέσει από μία άλλη οπτική γωνία και οδηγό χρήσης της εφαρμογής (manual).

3.2 Βασική λειτουργία Εφαρμογής

Η βασική λειτουργία της εφαρμογής αποτελείται από την κύρια διεπαφή με την οποία έρχεται αντιμέτωπος ο χρήστης και καλείται να επιλέξει την δραστηριότητα με την οποία θέλει να ασχοληθεί.



Εικόνα 35: Εικόνα κύριας διεπαφής του χρήστη

Όπως βλέπουμε στην παραπάνω εικόνα ο χρήστης καλείται να επιλέξει μεταξύ τεσσάρων επιλογών, η πρώτη επιλογή είναι της θεωρίας, η δεύτερη

επιλογή είναι των παραδειγμάτων, η τρίτη είναι των ασκήσεων και τέλος η τέταρτη επιλογή είναι ο τερματισμός της εφαρμογής. Ουσιαστικά αυτή η λειτουργία παίζει τον ρόλο του επιλογέα μεταξύ των διαφόρων λειτουργιών της εφαρμογής.

3.3 Λειτουργία παρουσίασης θεωρίας

Εφόσον ο χρήστης επιλέξει την πρώτη επιλογή θα οδηγηθεί στην λειτουργία παρουσίασης της θεωρίας των ψηφιακών συστημάτων και της γλώσσας VHDL. Η γραφική αναπαράσταση της λειτουργίας αυτής παρουσιάζεται στην παρακάτω εικόνα.

Εικόνα 36: Εικόνα διεπαφής παρουσίασης θεωρίας

Οι ενέργειες που μπορεί να πραγματοποιήσει ο χρήστης σε αυτήν την λειτουργία είναι να προχωρήσει στην επόμενη σελίδα μέσω της επιλογής με την επισήμανση 1, να επιστρέψει στην προηγούμενη σελίδα μέσω της επιλογής με την επισήμανση 2, να οδηγηθεί απευθείας στο κεφάλαιο με την θεωρία ψηφιακών μέσω της επιλογής με την επισήμανση 3, επίσης παρόμοια ενέργεια πραγματοποιεί και η λειτουργία με την επισήμανση 4 που οδηγεί απευθείας στο κεφάλαιο με την θεωρία της VHDL και τέλος η επιλογή με την επισήμανση 5 οδηγεί στο παράθυρο της βασικής λειτουργίας. Η λειτουργία αυτή περιέχει όλες εκείνες τις επιλογές που θα κάνουν την ζωή του χρήστη ποιο εύκολη, σε ότι αφορά την περιήγηση του στις σελίδες τις θεωρίας. Η λειτουργία που ακολουθεί είναι παρόμοια με την λειτουργία που μόλις παρουσιάσθηκε και έχει να κάνει μεν την παρουσίαση παραδειγμάτων σε γλώσσα VHDL.

3.4 Λειτουργία παρουσίασης παραδειγμάτων

Αν ο χρήστης κάνει επιλογή της δεύτερης λειτουργίας από την κύρια διεπαφή της εφαρμογής θα ενεργοποιηθεί η λειτουργία παρουσίασης παραδειγμάτων. Το παράθυρο που θα εμφανιστεί στον χρήστη θα έχει την παρακάτω μορφή και θα μπορούν να γίνουν διαφορές ενέργειες που αφορούν την περιήγηση του χρήστη στην λειτουργία των παραδειγμάτων.

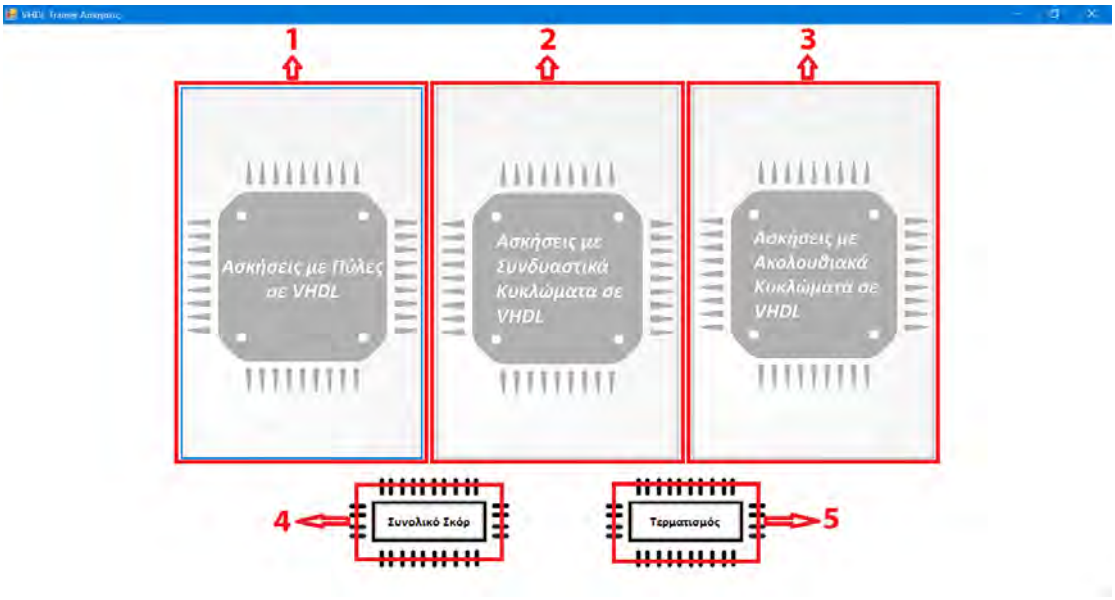
Εικόνα 37: Εικόνα διεπαφής παρουσίασης παραδειγμάτων

Οι ενέργειες που μπορεί να εκτελέσει ο χρήστης είναι η προώθηση στην επόμενη σελίδα (επιλογή 1), η επιστροφή στην προηγούμενη σελίδα (επιλογή 2), να οδηγηθεί απευθείας στις ασκήσεις του παραρτήματος Α’ (επιλογή 3), παρομοίως να οδηγηθεί απευθείας στις ασκήσεις του παραρτήματος Β’ (επιλογή 4), να εξέλθει της λειτουργίας και να επιστρέψει στο βασικό μενού (επιλογή 5). Ουσιαστικά η λειτουργία αυτή αποτελεί κλώνο της λειτουργίας προβολής της θεωρίας, γιατί πραγματεύονται σχεδόν παρόμοια αντικείμενα οπότε οι δύο λειτουργίες σχεδιάστηκαν και υλοποιήθηκαν με παρόμοιο τρόπο.

3.5 Λειτουργία ασκήσεων

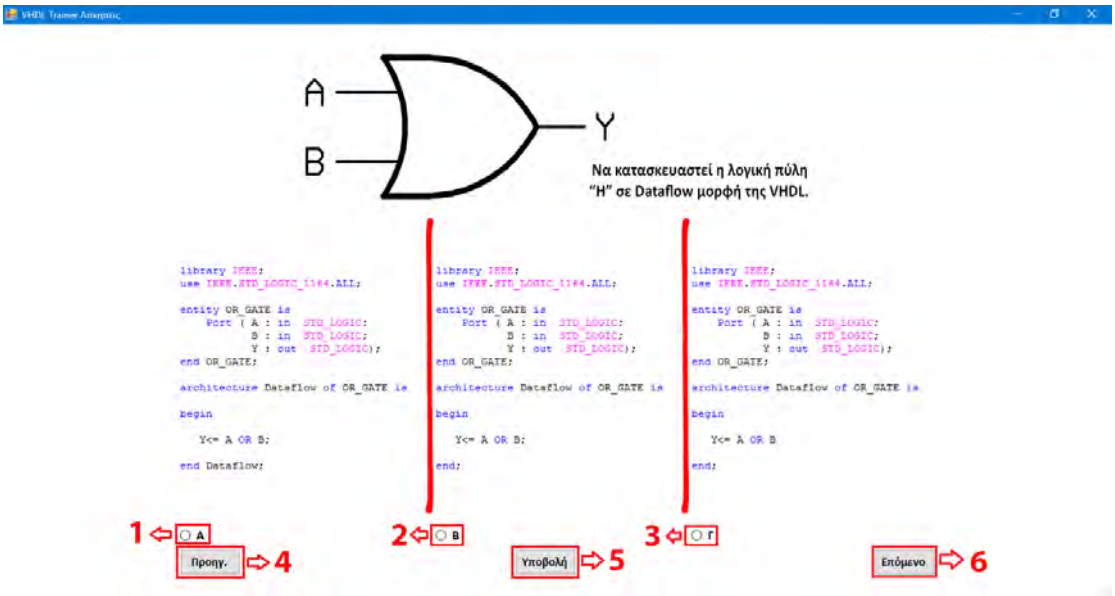
Εφόσον ο χρήστης κάνει την επιλογή της τρίτης λειτουργίας από το βασικό μενού επιλογών θα οδηγηθεί στην λειτουργία με τις δραστηριότητες των ασκήσεων. Η λειτουργία αυτή έχει αυξημένη πολυπλοκότητα σε σχέση με τις άλλες λειτουργίες καθώς περιέχει πολλαπλά επίπεδα λειτουργιών και ενεργοποιεί συγκεκριμένα παράθυρα προς τον χρήστη ανάλογα με το γεγονός

που λαμβάνει χώρα. Όταν ο χρήστης εισέρχεται στην λειτουργία των ασκήσεων ανοίγεται μπροστά του ένα παράθυρο που έχει την παρακάτω μορφή.



Εικόνα 38: Εικόνα λειτουργίας ασκήσεων 1

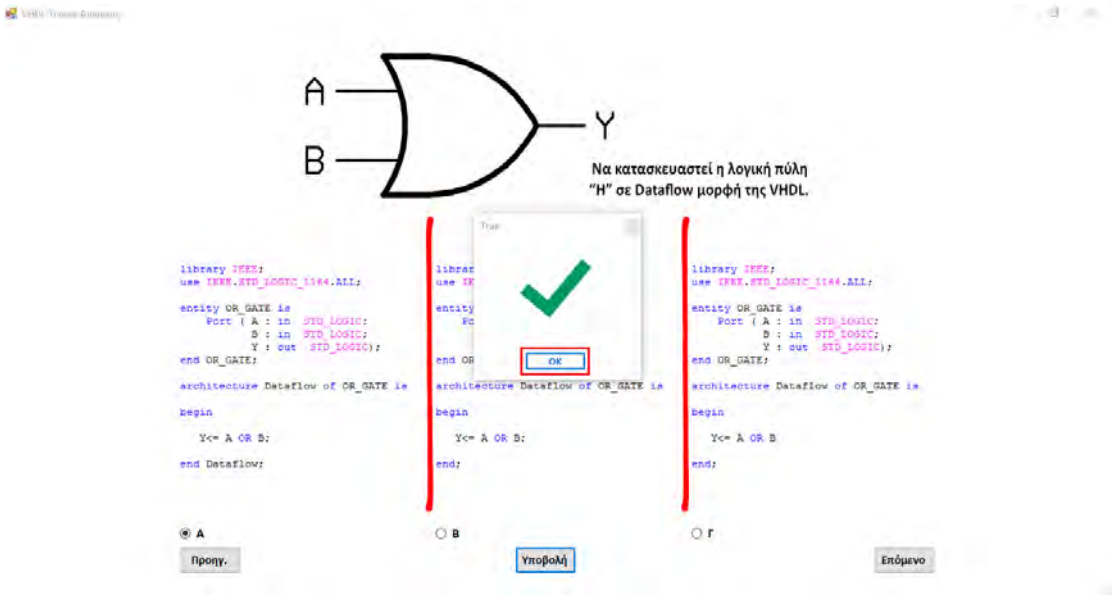
Αν ο χρήστης επιλέξει μια δραστηριότητα ασκήσεων μέσω των επιλογών 1,2,3 θα οδηγηθεί σε ένα παράθυρο της παρακάτω μορφής.



Εικόνα 39: Εικόνα λειτουργίας ασκήσεων 2

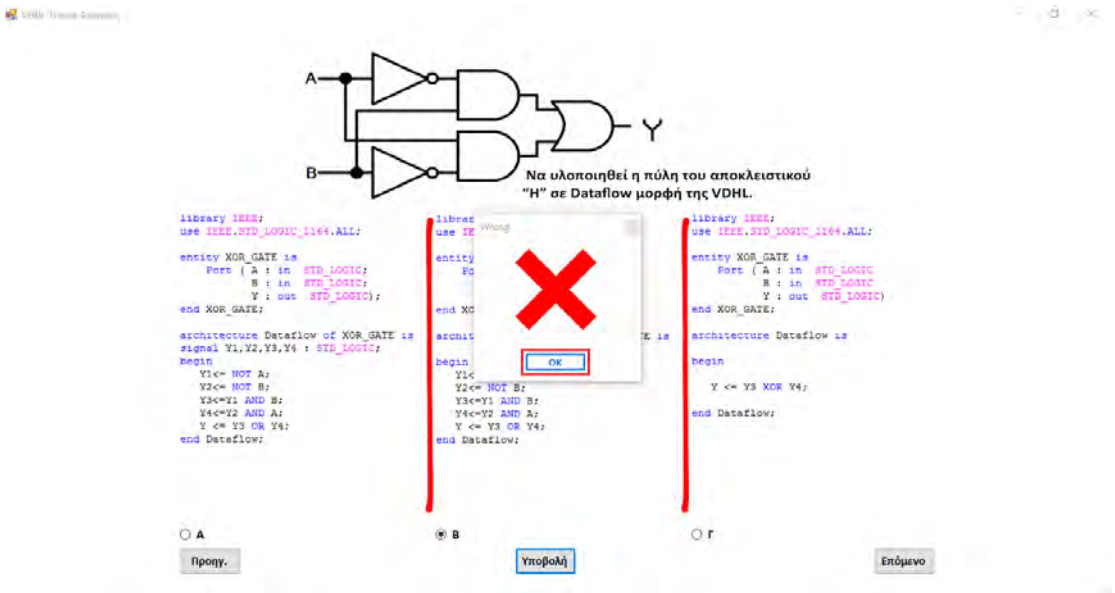
Στην δραστηριότητα των ασκήσεων ο χρήστης μπορεί να επιλέξει μια απάντηση μέσω των επιλογών 1,2 και 3, να γυρίσει στην προηγούμενη σελίδα

ερωτήσεων μέσω της επιλογής 4, να υποβάλει την απάντηση του και να ελέγξει αν είναι σωστή μέσω της επιλογής 5 και τέλος αν προχωρήσει στην επόμενη σελίδα ερωτήσεων. Αν η απάντηση του είναι σωστή τότε θα εμφανιστεί ένα παράθυρο της παρακάτω μορφής.



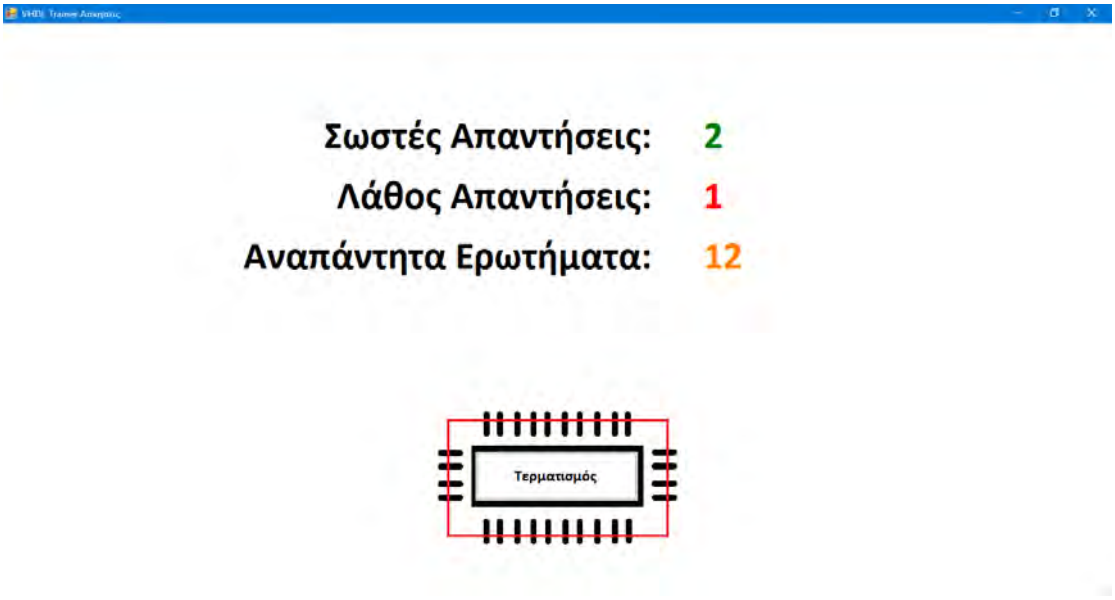
Εικόνα 40: Εικόνα λειτουργίας ασκήσεων σωστή απάντηση

Σε περίπτωση που η απάντηση είναι λανθασμένη εμφανίζεται ένα παράθυρο που έχει την παρακάτω μορφή.



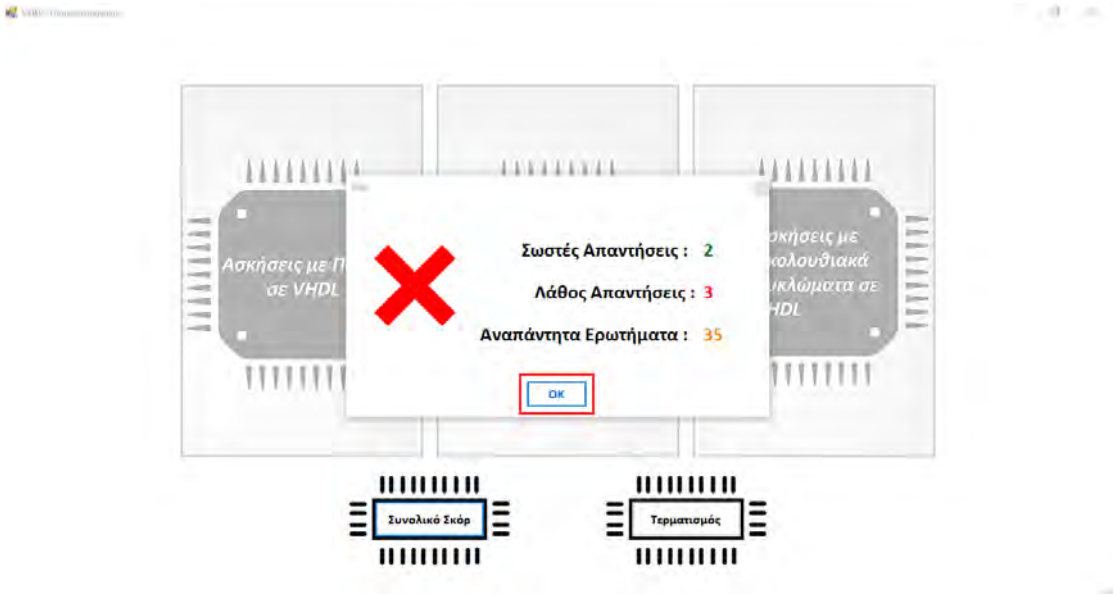
Εικόνα 41: Εικόνα λειτουργίας ασκήσεων λάθος απάντηση

Στο τέλος κάθε σετ ασκήσεων εμφανίζονται οι επιδόσεις του χρήστη με τις σωστές και λάθος απαντήσεις, η λειτουργία αυτή έχει την παρακάτω μορφή.



Εικόνα 42: Εικόνα λειτουργίας ασκήσεων επιδόσεις χρήστη

Αν ο χρήστης περατώσει όλες τις ασκήσεις και των τριών σετ τότε μπορεί να δει τα συγκεντρωτικά αποτελέσματα επιλέγοντας την τέταρτη επιλογή στο αρχικό παράθυρο των ασκήσεων.



Εικόνα 43: Εικόνα λειτουργίας ασκήσεων συνολικές επιδόσεις χρήστη

Παρουσιάζοντας το παράθυρο με τα συγκεντρωτικά αποτελέσματα κλείνει ο κύκλος της εκτενούς παρουσίασης της εφαρμογής καθώς και ο κύκλος αυτής της διπλωματικής εργασίας.

3.6 Επίλογος

Φτάνοντας στο τέλος της εν λόγω διπλωματικής εργασίας και μετά το τέλος της αναλυτικής περιγραφής της εφαρμογής, είναι δόκιμο να αναφερθεί ότι αυτή η εφαρμογή αποτελεί μια προσπάθεια να συγκροτηθεί ένα ολοκληρωμένο λογισμικό παρουσίασης της γλώσσας VHDL. Με σκοπό να βοηθήσει τα άτομα που θέλουν να ασχοληθούν πρώτη φορά με την συγκεκριμένη γλώσσα περιγραφής υλικού. Για αυτό το λόγω η ανάπτυξη της εφαρμογής έγινε με γνώμονα να είναι όσο το δυνατόν φιλικότερη ως προς τον χρήστη.

Η ανάπτυξη της εφαρμογής θα μπορούσε να συνεχιστεί προσθέτοντας διάφορους μηχανισμούς ή εμπλουτίζοντας την υπάρχουσα εφαρμογή με περισσότερη θεωρία και ασκήσεις σχετικές με την VHDL. Επίσης θα μπορούσε να προστεθεί μια βάση δεδομένων ώστε να υπάρχει λεπτομερή καταγραφή των σκορ που συγκέντρωσαν οι χρήστες στις διάφορες ασκήσεις. Η χρήση οπτικοακουστικού υλικό θα μπορούσε να αποτελέσει πολύ χρήσιμη προσθήκη, που θα βοηθούσε αποτελεσματικότερα τον χρήστη. Τέλος η θεματολογία της εφαρμογής θα μπορούσε να εμπλουτιστεί ακόμη περισσότερο με την προσθήκη θεωρίας και ασκήσεων και άλλων γλωσσών HDL όπως η VERILOG, οπότε η εφαρμογή θα μπορούσε να αποτελέσει ένα πλήρες εργαλείο εκμάθησης γλωσσών HDL.

Βιβλιογραφία

1. John Sharp, **Microsoft Visual C# 2008 Step by Step**, Εκδόσεις: Κλειδάριθμος (2008)

2. Παναγιώτης Μ. Παπάζογλου, **Αρχιτεκτονική & Προγραμματισμός Μικροεπεξεργαστών**, Εκδόσεις: Ίων (2010)

3. Ι. Χ. Παναγιωτόπουλος, **ΑΠΟ ΤΗ JAVA ΣΤΗ C#**, διαθέσιμο [online]: <http://studentguru.gr/m/books/134628>, Πειραιάς (2003)

4. John Sharp, **Microsoft Visual C# 2008 Step by Step**, Εκδόσεις: O'Reilly (2013)

5. Joseph Albahari, Ben Albahari, **C# 5.0 in a Nutshell, Fifth Edition**, Εκδόσεις: O'Reilly (2012)

6. Karli Watson, Christian Nagel, Jacob Hammer Pedersen, Jon D. Reid, Morgan Skinner, **Beginning Visual C# 2010**, Εκδόσεις: Wiley (2010)

7. Kenneth L. Short, **VHDL for engineers**, Εκδόσεις: Pearson Education (2009)

8. Διονύσης Ευσταθίου, **ΕΙΣΑΓΩΓΗ ΣΤΗ VHDL**, διαθέσιμο [online]: <http://www.mhl.tuc.gr/Controller?event=SHOW_HOME>,Χανιά (2001)

9. Σταύρος Σουραβλάς, **Βασικά Συνδυαστικά Κυκλώματα**, διαθέσιμο [online]: <http://opencourses.uom.gr/courses/efarmosmenhs-plhroforikhs/607-arxitektonikh-hlektronikon-ypologiston/enothtes>, Θεσσαλονίκη (2013)

10. Σταύρος Σουραβλάς, **Βασικά Ακολουθιακά Κυκλώματα**, διαθέσιμο [online]: <http://opencourses.uom.gr/courses/efarmosmenhs-plhroforikhs/607-arxitektonikh-hlektronikon-ypologiston/enothtes>, Θεσσαλονίκη (2013)

11. Νικόλαος Ασημάκης, Γιάννης Βουρβουλάκης, Θάνος Κακαρούντας, Νέλλη Λελίγκου, **e-βιβλίο ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ**, διαθέσιμο [online]: <http://elnsite.teilam.gr/ebooks/digital_design/>, Λαμία (2009)

12. Αντώνιος Δαδαλιάρης, **Σημειώσεις VHDL**, Λαμία (2015)

13. Wikipedia, <https://en.wikipedia.org/wiki/Main_Page>

14. Ιωάννης Καλόμοιρος, **Εισαγωγή στη γλώσσα VHDL**, διαθέσιμο [online]:<http://teachers.teicm.gr/kalomiros/Embedded_Systems.htm>, Σέρρες (2012)

15. Α. Ν. ΣΚΟΔΡΑΣ, **ΒΑΣΙΚΑ ΑΚΟΛΟΥΘΙΑΚΑ ΚΥΚΛΩΜΑΤΑ**, διαθέσιμο [online]: <http://dsmc.eap.gr/downloads/>, Πάτρα (2007)

16. Α. Ν. ΣΚΟΔΡΑΣ, **ΣΥΓΧΡΟΝΑ ΑΚΟΛΟΥΘΙΑΚΑ ΚΥΚΛΩΜΑΤΑ**, διαθέσιμο [online]: <http://dsmc.eap.gr/downloads/>, Πάτρα (2007)

17. Χρ. Καβουσιανός, **Συνδυαστικά Κυκλώματα σε HDL**, διαθέσιμο [online]: <http://www.cs.uoi.gr/~kabousia/DigitalDesign%CE%99GR.htm>, Ιωάννινα (2008)

18. Χρ. Καβουσιανός, **Ακολουθιακή Λογική σε HDL**, διαθέσιμο [online]: <http://www.cs.uoi.gr/~kabousia/DigitalDesign%CE%99GR.htm>, Ιωάννινα (2008)

19. Χρ. Καβουσιανός, **Ακολουθιακά Κυκλώματα**, διαθέσιμο [online]: <http://www.cs.uoi.gr/~kabousia/DigitalDesign%CE%99GR.htm>, Ιωάννινα (2008)