Υλοποίηση Βαθιών Νευρωνικών δικτύων σε FPGAs

<Παπαθανασίου Αλέξανρος>

Διπλωματική Εργασία

Επιβλέπων: Χρυσοβαλάντης Καβουσιανός

Ιωάννινα, Οκτώβριος,2024

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA

Ευχαριστίες

Θα ήθελα να εκφράσω τη βαθύτατη ευγνωμοσύνη μου σε όσους με στήριξαν καθ' όλη τη διάρκεια του ταξιδιού της ολοκλήρωσης αυτής της διατριβής. Πρώτα απ' όλα, θέλω να ευχαριστήσω τους γονείς μου για την αμέριστη αγάπη, την ενθάρρυνση και την πίστη τους σε μένα. Η συνεχής υποστήριξή σας υπήρξε η μεγαλύτερη πηγή δύναμής μου. Στους φίλους μου, σας ευχαριστώ που ήσασταν οι φωνητικοί μου σύμβουλοι, για την ατελείωτη ενθάρρυνσή σας και που ήσασταν πάντα εκεί όταν χρειαζόμουν ένα διάλειμμα. Η παρουσία σας έκανε τις προκλήσεις πιο εύκολα υποφερτές. Ένα ιδιαίτερο ευχαριστώ στον κ. Καβουσιανό Χρυσοβαλάντη, ο οποίος είχε την ιδέα και μου έδωσε την ευκαιρία να ασχολήθω με αυτή την εργασία. Η καθοδήγηση, η υπομονή και η διορατική ανατροφοδότηση ήταν ανεκτίμητη καθ' όλη τη διάρκεια αυτής της διαδικασίας. Τέλος, εκφράζω την ευγνωμοσύνη μου στους καθηγήτες της επιτροπής κ. Ευθυμίου και κ. Σφήκα. Εκτιμώ βαθύτατα το χρόνο και την προσοχή σας στην αξιολόγηση της εργασίας μου.

Σας ευχαριστώ όλους για τη συμβολή σας σε αυτό το επίτευγμα.

Περίληψη

Η παρούσα διατριβή διερευνά την υλοποίηση βαθιών νευρωνικών δικτύων (DNNs) σε μια διάταξη προγραμματιζόμενων πυλών πεδίου Cyclone II (FPGA), με κύρια εστίαση στην αξιολόγηση της ικανότητας της FPGA να διαχειρίζεται την αυξανόμενη πολυπλοκότητα των νευρωνικών δικτύων. Συγκεκριμένα, η έρευνα επικεντρώνεται σε ένα πολυεπίπεδο perceptron (MLP) που εκπαιδεύεται με το σύνολο δεδομένων 8x8 ψηφίων, με στόχο τον προσδιορισμό του μέγιστου αριθμού νευρώνων που μπορεί να φιλοξενηθεί αποτελεσματικά εντός των περιορισμών πόρων του Cyclone II. Η μελέτη παρέχει μια ολοκληρωμένη ανάλυση των σχεδιαστικών εκτιμήσεων, συμπεριλαμβανομένης της κατανομής των πόρων, της καθυστέρησης και της κατανάλωσης ενέργειας, για τη βελτιστοποίηση της ανάπτυξης των DNNs στην FPGA. Τα πειραματικά αποτελέσματα καταδεικνύουν τη δυνατότητα της FPGA να παρέχει αποτελεσματική εξαγωγή συμπερασμάτων νευρωνικών δικτύων, αποκαλύπτοντας τα πρακτικά όρια και τις συμβιβαστικές λύσεις απόδοσης που είναι εγγενείς στη χρήση ενός Cyclone II για τέτοιου είδους εργασίες. Τα ευρήματα υποδεικνύουν σημαντικές βελτιώσεις στην ταχύτητα και την ενεργειακή απόδοση σε σύγκριση με τις παραδοσιακές λύσεις που βασίζονται σε CPU και GPU, τονίζοντας τη σκοπιμότητα των FPGA για εφαρμογές μηχανικής μάθησης, ιδίως σε περιβάλλοντα με περιορισμένους πόρους. Επιπλέον, η εργασία αυτή αναδεικνύει τις στρατηγικές μεθοδολογίες σχεδιασμού που είναι απαραίτητες για τη μεγιστοποίηση της χρήσης των FPGA, συνεισφέροντας πολύτιμες γνώσεις στον τομέα της επιτάχυνσης υλικού για νευρωνικά δίκτυα. Αυτές οι γνώσεις προσφέρουν πρακτικές κατευθυντήριες γραμμές για τη μελλοντική έρευνα και ανάπτυξη στη βελτιστοποίηση της απόδοσης των DNN σε πλατφόρμες FPGA.

Λέξεις Κλειδιά: Νευρωνικά Δίκτυα, FPGA

Abstract

This thesis investigates the implementation of deep neural networks (DNNs) on a Cyclone II Field-Programmable Gate Array (FPGA), with a primary focus on evaluating the FPGA's capacity to manage increasing neural network complexity. Specifically, the research centers on a multilayer perceptron (MLP) trained with the 8x8 digits dataset, aiming to determine the maximum number of neurons that can be accommodated within the Cyclone II's resource constraints. The study provides a comprehensive analysis of design considerations, including resource allocation, latency, and power consumption, to optimize the deployment of DNNs on the FPGA. Experimental results demonstrate the FPGA's potential to deliver efficient neural network inference, revealing practical limits and performance trade-offs inherent in utilizing a Cyclone II for such tasks. The findings indicate significant improvements in speed and energy efficiency compared to traditional CPU and GPU-based solutions, emphasizing the feasibility of FPGAs for machine learning applications, particularly in resource-constrained environments. Additionally, this work highlights the strategic design methodologies necessary for maximizing FPGA utilization, contributing valuable insights into the field of hardware acceleration for neural networks. These insights offer practical guidelines for future research and development in optimizing DNN performance on FPGA platforms.

Keywords: <Neural Networks>, <FPGA>

Περιεχόμενα

Κεφάλαιο 1. <Εισαγωγή> **1**

1.1 < Ιστορικό και κίνητρο> 1

1.2 <Στόχοι της Έρευνας> 2

1.3 <Συμβολή της διατριβής> 4

Κεφάλαιο 2. <Θεωρητικό Υπόβαθρο> **6**

2.1 Εισαγωγή στα νευρωνικά δίκτυα 6

2.1.1 <Βασικές Έννοιες> *6*

2.1.2 <Τύποι Νευρωνικών Δικτύων> *7*

2.1.3 <Εκπαίδευση στα νευρωνικά δίκτυα> *10*

2.1.4 < Εφαρμογές νευρωνικών δικτύων> *11*

2.2 O Multilayer Perceptron (MLP) 11

2.2.1 <Δομή και λειτουργία> *12*

2.2.2 <Εκπαίδευση και backpropagation> *13*

2.2.3 <Εφαρμογές των MLPs> *14*

Κεφάλαιο 3. < Θεωρία FPGA και αρχιτεκτονική Cyclone II > **16**

3.1 Εισαγωγή στα FPGA 16

3.1.1 < Βασικές έννοιες και αρχιτεκτονική > *16*

3.1.2 < Δυνατότητα αναδιαμόρφωσης και ευελιξία > *17*

3.1.3 < Δυνατότητες παράλληλης επεξεργασίας > *18*

3.1.4 < Πλεονεκτήματα των FPGAs > *18*

3.1.5 < Εφαρμογές των FPGA > *19*

3.2 Το Cyclone II FPGA 20

3.3 Το DE2 Development Board 22

Κεφάλαιο 4. < Υλοποίηση του DNN στο FPGA> **24**

4.1 Εκτιμήσεις Σχεδιασμού 24

4.1.1 < Κατανομή πόρων > *25*

4.1.2 < Καθυστέρηση και απόδοση > *26*

4.2 Κατασκευή του DNN (MLP) για FPGA 27

4.2.1 < Προετοιμασία συνόλου δεδομένων (Digits Dataset)> *27*

4.2.2 < Σχεδιασμός αρχιτεκτονικής MLP > *29*

4.3 Τεχνικές βελτιστοποίησης 33

4.3.1 < Κβαντισμός > *34*

4.3.2 < Παράλληλη επεξεργασία > *35*

4.3.3 < Βελτιστοποίηση πόρων> *37*

4.3.4 < Συναρτήσεις ενεργοποίησης: Hardmax vs. Softmax> *39*

4.4 Εφαρμογή στο Cyclone II 40

4.4.1 < Ρύθμιση του περιβάλλοντος ανάπτυξης > *40*

4.4.2 < Κωδικοποίηση και σύνθεση HDL > *42*

Κεφάλαιο 5. < Πειραματικά αποτελέσματα > **47**

5.1 Φάση Προσομοίωσης 47

5.2 Αξιοποίηση πόρων και βελτιστοποίηση 49

5.3 Εξερεύνηση διαφορετικων διαρρυθμίσεων του DNN 56

5.4 Πειραματική Διαρρύθμιση 59

5.5 Πειραματικά Αποτελέσματα 61

5.6 Σύγκριση αποτελεσμάτων 64

5.7 Συζήτηση των αποτελεσμάτων 67

Κεφάλαιο 6. < Συμπεράσματα και μελλοντικές εργασίες> **70**

6.1 Συμπέρασμα 70

6.2 Μελλοντική εργασία 71

6.3 Τελικές σκέψεις 73

Κεφάλαιο 1. <Εισαγωγή>

1.1 < Ιστορικό και κίνητρο>

Η ταχεία πρόοδος της τεχνητής νοημοσύνης (AI) και της μηχανικής μάθησης (ML) έχει επηρεάσει σημαντικά πολλούς τομείς, όπως η αναγνώριση εικόνων, η επεξεργασία φυσικής γλώσσας και τα αυτόνομα συστήματα. Στο επίκεντρο αυτών των εξελίξεων βρίσκονται τα βαθιά νευρωνικά δίκτυα (DNN), τα οποία έχουν επιδείξει αξιοσημείωτες ικανότητες στη μάθηση και την εκτέλεση πολύπλοκων εργασιών. Ωστόσο, οι υπολογιστικές απαιτήσεις των DNN δημιουργούν σημαντικές προκλήσεις, ιδίως όσον αφορά την ταχύτητα επεξεργασίας και την ενεργειακή απόδοση. Παραδοσιακά χρησιμοποιούμε κεντρικές μονάδες επεξεργασίας (CPU) και μονάδες επεξεργασίας γραφικών (GPU) για την εκπαίδευση και την ανάπτυξη των DNNs. Ενώ οι GPU είναι ιδιαίτερα αποτελεσματικές για εργασίες παράλληλης επεξεργασίας, η υψηλή κατανάλωση ενέργειας και το κόστος τους μπορεί να είναι απαγορευτικά για ορισμένες εφαρμογές, ιδίως σε περιβάλλοντα με περιορισμένους πόρους, όπως ο υπολογισμός ακμών ή τα ενσωματωμένα συστήματα. Ως αποτέλεσμα, βλέπουμε ένα αυξανόμενο ενδιαφέρον για τη διερεύνηση εναλλακτικών λύσεων υλικού που παρέχουν την απαραίτητη υπολογιστική ισχύ, διατηρώντας παράλληλα την ενεργειακή αποδοτικότητα. Οι συστοιχίες προγραμματιζόμενων πυλών πεδίου (FPGA) έχουν αναδειχθεί ως μια πολλά υποσχόμενη λύση σε αυτό το πλαίσιο. Οι FPGAs προσφέρουν έναν μοναδικό συνδυασμό ευελιξίας, δυνατοτήτων παράλληλης επεξεργασίας και χαμηλής κατανάλωσης ενέργειας, καθιστώντας τις κατάλληλες για την υλοποίηση DNNs. Σε αντίθεση με τις σταθερές αρχιτεκτονικές υλικού, οι FPGA μπορούν να αναδιαμορφωθούν ώστε να προσαρμόσουν τους πόρους τους σε συγκεκριμένες υπολογιστικές εργασίες, παρέχοντας μια προσαρμόσιμη πλατφόρμα για ποικίλες εφαρμογές. Στην παρούσα διατριβή, εστιάζουμε στην υλοποίηση DNNs στο Cyclone II FPGA, μια ευρέως χρησιμοποιούμενη και οικονομικά αποδοτική πλατφόρμα FPGA.

Συγκεκριμένα, στοχεύουμε να διερευνήσουμε τα όρια του Cyclone II όσον αφορά τον μέγιστο αριθμό νευρώνων που μπορεί να φιλοξενήσει, χρησιμοποιώντας ως μελέτη περίπτωσης ένα πολυεπίπεδο perceptron (MLP) που εκπαιδεύεται στο σύνολο δεδομένων 8x8 ψηφίων. Με τη διερεύνηση των πρακτικών περιορισμών και των συμβιβασμών επιδόσεων, επιδιώκουμε να παράσχουμε πολύτιμες πληροφορίες σχετικά με τη βιωσιμότητα και τη βελτιστοποίηση υλοποιήσεων νευρωνικών δικτύων που βασίζονται σε FPGA.

Το κίνητρό μας είναι διττό. Πρώτον, στοχεύουμε να συνεισφέρουμε στον αυξανόμενο όγκο γνώσεων σχετικά με την επιτάχυνση υλικού για τη μηχανική μάθηση, παρέχοντας εμπειρικά δεδομένα και ανάλυση σχετικά με τη χρήση FPGA για DNN. Δεύτερον, επιδιώκουμε να αντιμετωπίσουμε τις πρακτικές προκλήσεις της ανάπτυξης DNNs σε περιβάλλοντα περιορισμένων πόρων, όπου οι παραδοσιακές λύσεις υλικού μπορεί να μην είναι εφικτές. Μέσω αυτής της έρευνας, στοχεύουμε να αποδείξουμε ότι οι FPGA μπορούν να προσφέρουν μια βιώσιμη και αποτελεσματική εναλλακτική λύση για την υλοποίηση των DNN, ανοίγοντας το δρόμο για την ευρύτερη υιοθέτησή τους σε διάφορες εφαρμογές

1.2 <Στόχοι της Έρευνας>

Στην παρούσα μελέτη, σκοπός μας είναι να διερευνήσουμε την υλοποίηση βαθιών νευρωνικών δικτύων (DNN) σε μια διάταξη προγραμματιζόμενων πυλών Cyclone II (FPGA). Ο πρωταρχικός μας στόχος είναι να διερευνήσουμε τις δυνατότητες και τους περιορισμούς της Cyclone II FPGA στον χειρισμό της πολυπλοκότητας των DNN, εστιάζοντας συγκεκριμένα σε ένα πολυεπίπεδο perceptron (MLP) που εκπαιδεύεται με το σύνολο δεδομένων 8x8 ψηφίων. Για να το επιτύχουμε αυτό, έχουμε περιγράψει διάφορους συγκεκριμένους στόχους:

1. **Σχεδιασμός και υλοποίηση του MLP στο Cyclone II**: Θα σχεδιάσουμε μια αρχιτεκτονική MLP κατάλληλη για το Cyclone II FPGA. Αυτό περιλαμβάνει τη διαμόρφωση της FPGA για να φιλοξενήσει το MLP και τη διασφάλιση ότι μπορεί να εκτελέσει αποτελεσματικά τους απαραίτητους υπολογισμούς.

2. **Ανάλυση της χρήσης των πόρων**: Στόχος μας είναι να αναλύσουμε τη χρήση των πόρων της Cyclone II FPGA κατά την υλοποίηση του MLP. Αυτό περιλαμβάνει τον προσδιορισμό του μέγιστου αριθμού νευρώνων και επιπέδων

που μπορούν να φιλοξενηθούν αποτελεσματικά εντός των περιορισμών πόρων της FPGA.

3. **Αξιολόγηση επιδόσεων**: Θα αξιολογήσουμε τις επιδόσεις του MLP που υλοποιήθηκε με FPGA όσον αφορά την ταχύτητα, την καθυστέρηση και την κατανάλωση ενέργειας. Αυτό θα περιλαμβάνει τη συγκριτική αξιολόγηση της υλοποίησης FPGA έναντι των παραδοσιακών υλοποιήσεων που βασίζονται σε CPU και GPU, ώστε να αναδειχθούν τα πλεονεκτήματα και οι πιθανές αντισταθμίσεις.

4. **Τεχνικές βελτιστοποίησης**: Θα εξερευνήσουμε διάφορες τεχνικές βελτιστοποίησης, όπως η κβάντιση, το κλάδεμα και η παράλληλη επεξεργασία, για να βελτιώσουμε την απόδοση και την αποδοτικότητα του MLP στην FPGA. Στόχος είναι η μεγιστοποίηση της χρήσης της FPGA και η βελτίωση της συνολικής απόδοσης του νευρωνικού δικτύου.

5. **Πειραματική επικύρωση**: Θα πραγματοποιήσουμε εκτεταμένα πειράματα για να επικυρώσουμε την αποτελεσματικότητα της υλοποίησης μας με βάση την FPGA. Αυτό περιλαμβάνει τη δοκιμή της ακρίβειας του MLP στο σύνολο δεδομένων 8x8 ψηφίων και τη σύγκρισή του με άλλες υλοποιήσεις υλικού.

6. **Κατευθυντήριες γραμμές για μελλοντική έρευνα**: Με βάση τα ευρήματά μας, θα παράσχουμε πρακτικές κατευθυντήριες γραμμές και συστάσεις για τη μελλοντική έρευνα και ανάπτυξη στον τομέα των υλοποιήσεων νευρωνικών δικτύων με βάση FPGA. Αυτό θα περιλαμβάνει πληροφορίες σχετικά με τις στρατηγικές σχεδιασμού, τις τεχνικές βελτιστοποίησης και τους πιθανούς τομείς για περαιτέρω διερεύνηση.

Με την επίτευξη αυτών των στόχων, επιδιώκουμε να καταδείξουμε τη σκοπιμότητα και τα πλεονεκτήματα της χρήσης FPGA για την υλοποίηση DNN, ιδίως σε περιβάλλοντα με περιορισμένους πόρους. Η μελέτη μας θα συμβάλει στον αυξανόμενο όγκο γνώσεων σχετικά με την επιτάχυνση υλικού για τη μηχανική μάθηση και θα παράσχει πολύτιμες πληροφορίες για τους ερευνητές και τους επαγγελματίες του τομέα.

1.3 <Συμβολή της διατριβής>

Η παρούσα διατριβή συμβάλλει σημαντικά στον τομέα των βαθιών νευρωνικών δικτύων (DNN) και της υλοποίησής τους σε συστοιχίες προγραμματιζόμενων πυλών πεδίου (FPGA), εστιάζοντας ειδικά στην FPGA Cyclone II. Οι συνεισφορές μας μπορούν να συνοψιστούν ως εξής:

1. **Πλαίσιο σχεδίασης και υλοποίησης**: Παρέχουμε ένα ολοκληρωμένο πλαίσιο για τη σχεδίαση και την υλοποίηση ενός πολυεπίπεδου perceptron (MLP) στο Cyclone II FPGA. Αυτό περιλαμβάνει λεπτομερείς μεθοδολογίες για τη διαμόρφωση της FPGA ώστε να ανταποκρίνεται στις υπολογιστικές απαιτήσεις του MLP, εξετάζοντας τόσο τις πτυχές υλικού όσο και λογισμικού.

2. **Πληροφορίες σχετικά με την αξιοποίηση των πόρων**: Παρουσιάζουμε μια εμπεριστατωμένη ανάλυση της χρήσης των πόρων της Cyclone II FPGA κατά την υλοποίηση του MLP. Η μελέτη μας προσδιορίζει τον μέγιστο αριθμό νευρώνων και επιπέδων που μπορεί να φιλοξενήσει η FPGA, προσφέροντας πολύτιμα δεδομένα για ερευνητές και προγραμματιστές που εργάζονται με παρόμοιους περιορισμούς.

3. **Συγκριτική αξιολόγηση επιδόσεων**: Πραγματοποιούμε εκτεταμένες αξιολογήσεις επιδόσεων, συγκρίνοντας την υλοποίηση του MLP με βάση FPGA με τις παραδοσιακές υλοποιήσεις CPU και GPU. Τα αποτελέσματα της συγκριτικής μας αξιολόγησης αναδεικνύουν τα πλεονεκτήματα της ταχύτητας, της καθυστέρησης και της κατανάλωσης ενέργειας από τη χρήση FPGA για εργασίες DNN, παρέχοντας εμπειρικές αποδείξεις για τα πλεονεκτήματά τους.

4. **Τεχνικές βελτιστοποίησης**: Για να βελτιώσουμε την απόδοση και την αποδοτικότητα του MLP στην FPGA, διερευνούμε και εφαρμόζουμε διάφορες τεχνικές βελτιστοποίησης, όπως κβαντοποίηση, κλάδεμα και παράλληλη επεξεργασία. Τα ευρήματά μας καταδεικνύουν τον πρακτικό αντίκτυπο αυτών των τεχνικών στη χρήση της FPGA και την απόδοση του DNN.

5. **Πειραματική επικύρωση**: Μέσω αυστηρού πειραματισμού, επικυρώνουμε την αποτελεσματικότητα της υλοποίησης του MLP που βασίζεται σε FPGA. Τα

πειράματά μας, που πραγματοποιήθηκαν με τη χρήση του συνόλου δεδομένων 8x8 ψηφίων, επιβεβαιώνουν την ακρίβεια και την αποτελεσματικότητα της λύσης FPGA, υποστηρίζοντας τη βιωσιμότητά της για πραγματικές εφαρμογές.

6. **Πρακτικές κατευθυντήριες γραμμές και συστάσεις**: Με βάση την έρευνά μας και τα ευρήματά μας, προσφέρουμε πρακτικές κατευθυντήριες γραμμές και συστάσεις για μελλοντικές εργασίες σε υλοποιήσεις νευρωνικών δικτύων με βάση FPGA. Αυτές οι κατευθυντήριες γραμμές καλύπτουν στρατηγικές σχεδιασμού, τεχνικές βελτιστοποίησης και πιθανές περιοχές για περαιτέρω διερεύνηση, με στόχο να βοηθήσουν τους ερευνητές και τους επαγγελματίες στη μεγιστοποίηση των δυνατοτήτων των FPGA για εργασίες DNN.

Αντιμετωπίζοντας αυτούς τους τομείς, η διατριβή μας συμβάλλει στην κατανόηση και την πρόοδο των υλοποιήσεων DNN που βασίζονται σε FPGA. Καταδεικνύουμε τη σκοπιμότητα και τα οφέλη της χρήσης των FPGAs, ιδίως του Cyclone II, για αποδοτική επεξεργασία νευρωνικών δικτύων, ανοίγοντας το δρόμο για την ευρύτερη υιοθέτησή τους σε διάφορες εφαρμογές, ιδίως εκείνες με περιορισμούς πόρων.

Κεφάλαιο 2. <Θεωρητικό Υπόβαθρο>

2.1 Εισαγωγή στα νευρωνικά δίκτυα

Τα νευρωνικά δίκτυα, εμπνευσμένα από τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου, είναι μια κατηγορία μοντέλων μηχανικής μάθησης που έχουν σχεδιαστεί για να αναγνωρίζουν πρότυπα και να εκτελούν σύνθετες εργασίες. Αποτελούνται από διασυνδεδεμένους κόμβους ή νευρώνες, οργανωμένους σε επίπεδα. Αυτά τα στρώματα συνεργάζονται για να μετατρέψουν τα δεδομένα εισόδου σε έξοδο με νόημα μέσω μιας διαδικασίας εκπαίδευσης από παραδείγματα. Σε αυτή την ενότητα, παρέχουμε μια θεμελιώδη επισκόπηση των νευρωνικών δικτύων, συζητώντας τις βασικές έννοιες, τους τύπους και τα βασικά χαρακτηριστικά τους.

2.1.1 <Βασικές Έννοιες>

Στον πυρήνα τους, τα νευρωνικά δίκτυα αποτελούνται από τρία είδη στρωμάτων: στρώματα εισόδου, κρυφά στρώματα και στρώματα εξόδου. Κάθε νευρώνας σε ένα στρώμα λαμβάνει σήματα εισόδου, τα επεξεργάζεται και μεταδίδει την έξοδο στους νευρώνες του επόμενου στρώματος. Αυτή η επεξεργασία περιλαμβάνει την εφαρμογή ενός σταθμισμένου αθροίσματος των εισόδων ακολουθούμενου από μια μη γραμμική συνάρτηση ενεργοποίησης. Τα κύρια στοιχεία ενός νευρωνικού δικτύου περιλαμβάνουν:

- **Νευρώνες**: Βασικές μονάδες που λαμβάνουν είσοδο, εφαρμόζουν έναν μετασχηματισμό και μεταβιβάζουν το αποτέλεσμα στο επόμενο στρώμα.

- **Βάρη**: Παράμετροι που ρυθμίζουν τα σήματα εισόδου. Η εκπαίδευση στα νευρωνικά δίκτυα περιλαμβάνει την ενημέρωση αυτών των βαρών για την ελαχιστοποίηση του σφάλματος μέσω αλγορίθμων όπως η οπισθοδιάδοση.

- **Συναρτήσεις ενεργοποίησης:** Μη γραμμικές συναρτήσεις που εφαρμόζονται στο σταθμισμένο άθροισμα των εισόδων, εισάγοντας μη γραμμικότητα στο μοντέλο. Οι συνήθεις συναρτήσεις ενεργοποίησης περιλαμβάνουν τις sigmoid, tanh και ReLU.

- **Biases**: Πρόσθετες παράμετροι που προσαρμόζουν την έξοδο ανεξάρτητα από τα σήματα εισόδου, βελτιώνοντας την ευελιξία του μοντέλου.

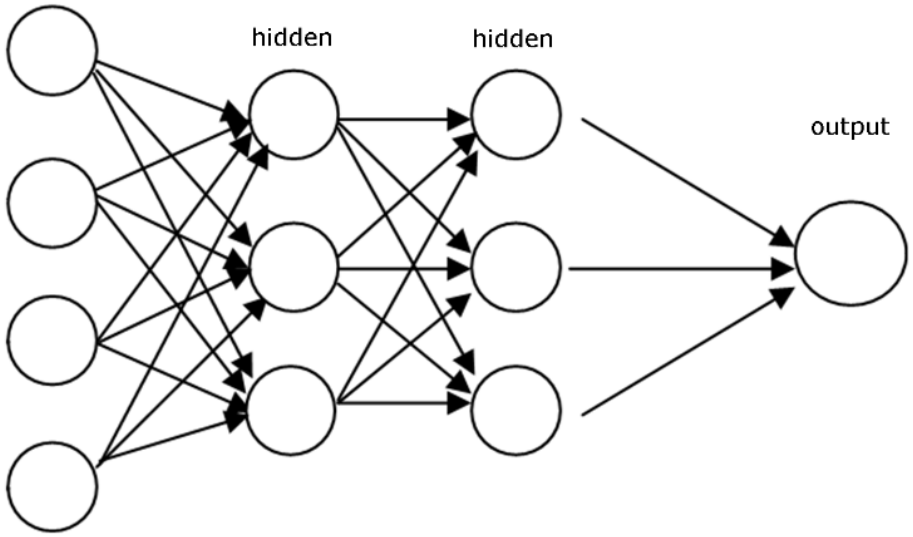
Αυτά τα βασικά στοιχεία επιτρέπουν στα νευρωνικά δίκτυα να μοντελοποιούν πολύπλοκα πρότυπα και σχέσεις στα δεδομένα, προσαρμόζοντας επαναληπτικά τις παραμέτρους με βάση τα παραδείγματα εκπαίδευσης [GBC16], [Hay99].

2.1.2 <Τύποι Νευρωνικών Δικτύων>

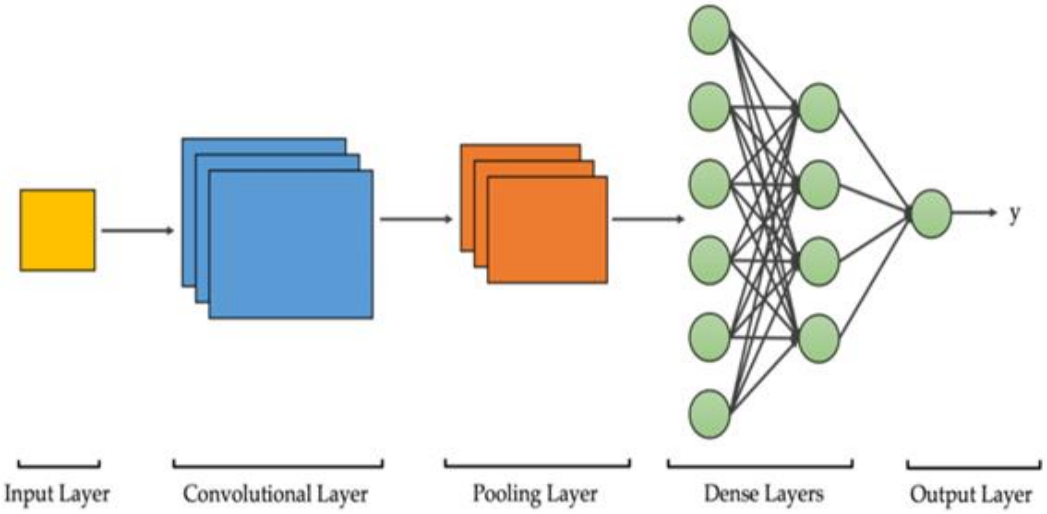
Υπάρχουν διάφοροι τύποι νευρωνικών δικτύων, καθένας από τους οποίους είναι κατάλληλος για διαφορετικούς τύπους εργασιών. Μερικά από τα πιο συνηθισμένα περιλαμβάνουν:

- **Feedforward Neural Networks** Αυτός είναι ο απλούστερος τύπος νευρωνικού δικτύου, όπου οι συνδέσεις μεταξύ των κόμβων δεν σχηματίζουν κύκλους. Η πληροφορία ρέει προς μία κατεύθυνση, από το επίπεδο εισόδου στο επίπεδο εξόδου. Κάθε νευρώνας επεξεργάζεται τις εισόδους του υπολογίζοντας ένα σταθμισμένο άθροισμα, εφαρμόζοντας μια μη γραμμική συνάρτηση ενεργοποίησης και στη συνέχεια μεταβιβάζοντας το αποτέλεσμα στο επόμενο στρώμα. Τα δίκτυα τροφοδότησης χρησιμοποιούνται συνήθως για εργασίες όπως η ταξινόμηση και η παλινδρόμηση, όπου το μοντέλο μαθαίνει να αντιστοιχίζει τις εισόδους στις εξόδους. Η δομή τους επιτρέπει την απλή εκπαίδευση μέσω αλγορίθμων όπως η κάθοδος κλίσης, καθιστώντας τα ιδανικά

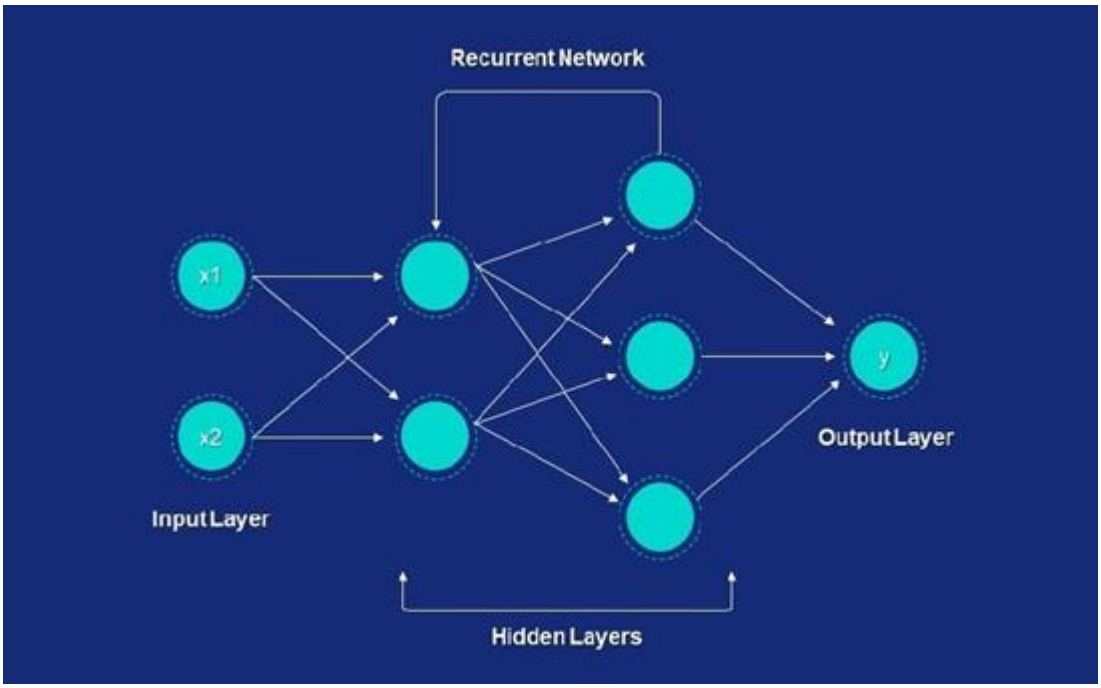
για εφαρμογές που απαιτούν γρήγορη επεξεργασία και ερμηνεία των δεδομένων [GBC16].



- **Convolutional Neural Networks (CNN)**: Τα CNN χρησιμοποιούνται κυρίως για εργασίες αναγνώρισης εικόνας και βίντεο και έχουν σχεδιαστεί για να μαθαίνουν αυτόματα και προσαρμοστικά χωρικές ιεραρχίες χαρακτηριστικών. Η αρχιτεκτονική τους αποτελείται από συνελικτικά στρώματα που εφαρμόζουν φίλτρα στα δεδομένα εισόδου, ανιχνεύοντας αποτελεσματικά ακμές, υφές και μοτίβα. Ακολουθούν στρώματα συγκέντρωσης που μειώνουν τις χωρικές διαστάσεις, επιτρέποντας μια πιο συμπαγή αναπαράσταση των χαρακτηριστικών, διατηρώντας παράλληλα τις βασικές πληροφορίες. Ο συνδυασμός αυτών των στρωμάτων επιτρέπει στα CNN να επιτυγχάνουν υψηλή ακρίβεια σε εργασίες όπως η ανίχνευση αντικειμένων και η αναγνώριση προσώπου, ξεπερνώντας τις παραδοσιακές προσεγγίσεις με την αποτελεσματική διαχείριση μεγάλων συνόλων οπτικών δεδομένων [Hay99].

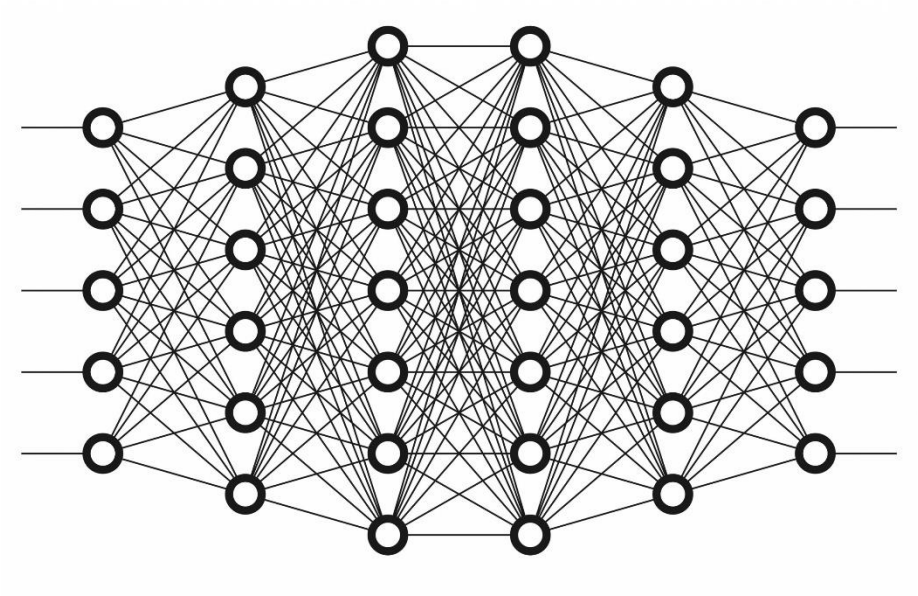


- **Recurrent Neural Networks (RNNs):** Τα RNN έχουν σχεδιαστεί για διαδοχικά δεδομένα, όπου η σειρά των εισόδων έχει σημασία. Διαθέτουν συνδέσεις που σχηματίζουν κατευθυνόμενους κύκλους, επιτρέποντάς τους να διατηρούν μνήμη προηγούμενων εισόδων. Αυτό το χαρακτηριστικό καθιστά τα RNN ιδιαίτερα κατάλληλα για εργασίες όπως η μοντελοποίηση γλώσσας, η αναγνώριση ομιλίας και η πρόβλεψη χρονοσειρών. Κάθε νευρώνας σε ένα RNN μπορεί να επηρεάσει τις επόμενες καταστάσεις του, επιτρέποντας στο δίκτυο να μαθαίνει χρονικά μοτίβα και εξαρτήσεις. Ωστόσο, τα RNN αντιμετωπίζουν συχνά προκλήσεις, όπως οι εξαφανιζόμενες και εκρηκτικές κλίσεις, οι οποίες μπορούν να περιπλέξουν τη διαδικασία εκπαίδευσης, ιδίως όταν πρόκειται για μεγάλες ακολουθίες. Έχουν αναπτυχθεί προηγμένες αρχιτεκτονικές, όπως τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (LSTM) και οι Gated Recurrent Units (GRUs), για να μετριάσουν αυτά τα προβλήματα και να βελτιώσουν την απόδοση σε διαδοχικές εργασίες [GBC16].



- **Multilayer Perceptrons (MLPs):** Τα MLP είναι ένας τύπος νευρωνικού δικτύου feedforward που αποτελείται από πολλαπλά στρώματα νευρώνων, συμπεριλαμβανομένων ενός ή περισσότερων κρυφών στρωμάτων. Κάθε νευρώνας σε ένα MLP εκτελεί ένα σταθμισμένο άθροισμα των εισόδων του, εφαρμόζει μια μη γραμμική συνάρτηση ενεργοποίησης και προωθεί την έξοδο στο επόμενο επίπεδο. Τα MLP είναι ισχυρά εργαλεία για τη μοντελοποίηση πολύπλοκων σχέσεων μεταξύ εισόδων και εξόδων, καθιστώντας τα κατάλληλα

για διάφορες εφαρμογές, όπως η πρόβλεψη, η ταξινόμηση δεδομένων και η αναγνώριση προτύπων. Η ικανότητά τους να μαθαίνουν μη γραμμικές απεικονίσεις επιτρέπει στα MLP να γενικεύουν καλά σε νέα δεδομένα, υπό την προϋπόθεση ότι εκπαιδεύονται σε ένα επαρκώς αντιπροσωπευτικό σύνολο δεδομένων. Αυτή η ευελιξία έχει οδηγήσει στην ευρεία χρήση τους σε πολλούς τομείς, από τη χρηματοδότηση έως την υγειονομική περίθαλψη [Hay99].



2.1.3 <Εκπαίδευση στα νευρωνικά δίκτυα>

Η εκπαίδευση στα νευρωνικά δίκτυα περιλαμβάνει συνήθως δύο βασικές διαδικασίες: την προς τα εμπρός διάδοση και την οπισθοδιάδοση.

- **Forward Propagation**: Σε αυτή τη φάση, τα δεδομένα εισόδου περνούν μέσα από το δίκτυο, στρώμα προς στρώμα, μέχρι να φτάσουν στο στρώμα εξόδου. Κάθε νευρώνας υπολογίζει ένα σταθμισμένο άθροισμα των εισόδων του, εφαρμόζει μια συνάρτηση ενεργοποίησης και μεταβιβάζει το αποτέλεσμα στο επόμενο επίπεδο [GBC16].
- **Backpropagation**: Πρόκειται για τη διαδικασία προσαρμογής των βαρών ώστε να ελαχιστοποιηθεί το σφάλμα μεταξύ της προβλεπόμενης εξόδου και του πραγματικού στόχου. Περιλαμβάνει τον υπολογισμό της κλίσης της συνάρτησης απώλειας σε σχέση με κάθε βάρος και την ενημέρωση των βαρών με τη χρήση ενός αλγορίθμου βελτιστοποίησης, όπως η κάθοδος κλίσης [RHW86].

2.1.4 < Εφαρμογές νευρωνικών δικτύων>

Τα νευρωνικά δίκτυα έχουν εφαρμοστεί με επιτυχία σε ένα ευρύ φάσμα τομέων λόγω της ικανότητάς τους να μοντελοποιούν πολύπλοκες σχέσεις. Ορισμένες αξιοσημείωτες εφαρμογές περιλαμβάνουν:

- **Αναγνώριση εικόνας και ομιλίας**: Τα νευρωνικά δίκτυα, ιδίως τα CNN, έχουν φέρει επανάσταση στην αναγνώριση εικόνας και ομιλίας, επιτυγχάνοντας υψηλή ακρίβεια σε εργασίες όπως η ανίχνευση αντικειμένων και η μετατροπή ομιλίας σε κείμενο.
- **Επεξεργασία φυσικής γλώσσας (NLP)**: Στην NLP, τα νευρωνικά δίκτυα χρησιμοποιούνται για εργασίες όπως η γλωσσική μετάφραση, η ανάλυση συναισθήματος και η παραγωγή κειμένου [GBC16].
- **Αυτόνομα συστήματα**: Τα νευρωνικά δίκτυα διαδραματίζουν καθοριστικό ρόλο στην ανάπτυξη αυτόνομων οχημάτων και ρομπότ, επιτρέποντάς τους να αντιλαμβάνονται και να ερμηνεύουν το περιβάλλον τους.
- **Υγειονομική περίθαλψη**: Στον ιατρικό τομέα, τα νευρωνικά δίκτυα βοηθούν στη διάγνωση ασθενειών, στην ανάλυση ιατρικών εικόνων και στην πρόβλεψη των αποτελεσμάτων των ασθενών[Hay99].

Τα νευρωνικά δίκτυα συνεχίζουν να εξελίσσονται, με τη συνεχή έρευνα να επικεντρώνεται στη βελτίωση της αποδοτικότητας, της ερμηνευσιμότητας και της δυνατότητας εφαρμογής τους σε νέα και δύσκολα προβλήματα. Στην επόμενη ενότητα, θα εμβαθύνουμε στις ιδιαιτερότητες του πολυεπίπεδου perceptron (MLP), ενός θεμελιώδους τύπου νευρωνικού δικτύου που αποτελεί τη βάση της εφαρμογής μας σε FPGA.

2.2 Ο Multilayer Perceptron (MLP)

Ο Multilayer Perceptron (MLP) είναι ένας τύπος νευρωνικού δικτύου πρόωσης που αποτελείται από πολλαπλά στρώματα νευρώνων, συμπεριλαμβανομένου τουλάχιστον ενός κρυφού στρώματος μεταξύ των στρωμάτων εισόδου και εξόδου. Τα MLP είναι κατάλληλα για εργασίες που απαιτούν την εκπαίδευση σύνθετων μοτίβων και σχέσεων από δεδομένα. Αυτή η ενότητα καλύπτει τη δομή και τη λειτουργία των MLP, τη

διαδικασία εκπαίδευσής τους με τη χρήση της οπισθοδιάδοσης και τις ποικίλες εφαρμογές τους.

2.2.1 <Δομή και λειτουργία>

Η αρχιτεκτονική του MLP περιλαμβάνει τρεις κύριους τύπους στρωμάτων:

Στρώμα εισόδου: Το πρώτο στρώμα, το οποίο λαμβάνει τα ακατέργαστα δεδομένα και τα μεταβιβάζει στα επόμενα στρώματα. Κάθε νευρώνας στο στρώμα εισόδου αντιπροσωπεύει ένα χαρακτηριστικό των δεδομένων εισόδου.

Κρυφά στρώματα: Ένα ή περισσότερα στρώματα μεταξύ των στρωμάτων εισόδου και εξόδου, όπου πραγματοποιείται το μεγαλύτερο μέρος του υπολογισμού και της εκπαίδευσης. Κάθε νευρώνας στα κρυφά στρώματα εφαρμόζει ένα σταθμισμένο άθροισμα των εισόδων του ακολουθούμενο από μια μη γραμμική συνάρτηση ενεργοποίησης. Η πολυπλοκότητα του MLP αυξάνεται με τον αριθμό των κρυφών στρωμάτων και των νευρώνων, επιτρέποντάς του να μοντελοποιεί πιο περίπλοκες σχέσεις στα δεδομένα.

Στρώμα εξόδου: Το τελικό στρώμα που παράγει τις προβλέψεις ή τις ταξινομήσεις του δικτύου. Η δομή του στρώματος εξόδου εξαρτάται από τη συγκεκριμένη εργασία- για παράδειγμα, μπορεί να αποτελείται από έναν μόνο νευρώνα για εργασίες παλινδρόμησης ή πολλαπλούς νευρώνες για εργασίες ταξινόμησης, με κάθε νευρώνα να αντιπροσωπεύει μια κλάση ή μια τιμή εξόδου.

Κάθε νευρώνας στο MLP συνδέεται με νευρώνες στο προηγούμενο και στα επόμενα στρώματα μέσω σταθμισμένων συνδέσεων. Τα βάρη καθορίζουν την ισχύ και την κατεύθυνση των συνδέσεων και προσαρμόζονται κατά τη διάρκεια της εκπαίδευσης ώστε να ελαχιστοποιείται το σφάλμα μεταξύ της προβλεπόμενης εξόδου και του πραγματικού στόχου [BM94].

2.2.2 <Εκπαίδευση και backpropagation>

Η εκπαίδευση ενός MLP περιλαμβάνει την προσαρμογή των βαρών του δικτύου ώστε να ελαχιστοποιηθεί το σφάλμα στις προβλέψεις του. Η διαδικασία εκτελείται συνήθως χρησιμοποιώντας τα ακόλουθα βήματα:

1. **Εμπρόσθια διάδοση**: Τα δεδομένα εισόδου τροφοδοτούνται μέσω του δικτύου από το στρώμα εισόδου στο στρώμα εξόδου. Κάθε νευρώνας υπολογίζει ένα σταθμισμένο άθροισμα των εισόδων του, εφαρμόζει μια συνάρτηση ενεργοποίησης και μεταβιβάζει το αποτέλεσμα στο επόμενο στρώμα. Η διαδικασία αυτή συνεχίζεται μέχρι το στρώμα εξόδου να παράγει την τελική πρόβλεψη.

2. **Υπολογισμός απωλειών**: Η διαφορά μεταξύ της προβλεπόμενης εξόδου και του πραγματικού στόχου υπολογίζεται με τη χρήση μιας συνάρτησης απώλειας, όπως το μέσο τετραγωνικό σφάλμα (MSE) για εργασίες παλινδρόμησης ή η απώλεια cross-Entropy για εργασίες ταξινόμησης. Η συνάρτηση απώλειας ποσοτικοποιεί το σφάλμα των προβλέψεων του δικτύου.

3. **Backpropagation**: Αυτός ο αλγόριθμος υπολογίζει την κλίση της συνάρτησης απώλειας σε σχέση με κάθε βάρος στο δίκτυο εφαρμόζοντας τον κανόνα της αλυσίδας του λογισμού. Οι κλίσεις χρησιμοποιούνται για την ενημέρωση των βαρών προς την κατεύθυνση που μειώνει το σφάλμα. Ο αλγόριθμος οπισθοδιάδοσης περιλαμβάνει:

4. **Διάδοση σφαλμάτων**: Το σφάλμα διαδίδεται προς τα πίσω μέσω του δικτύου από το στρώμα εξόδου στο στρώμα εισόδου, στρώμα προς στρώμα.\

5. **Υπολογισμός κλίσης**: Υπολογίζονται οι κλίσεις της συνάρτησης απώλειας ως προς κάθε βάρος.

6. **Ενημέρωση βαρών**: Τα βάρη ενημερώνονται χρησιμοποιώντας έναν αλγόριθμο βελτιστοποίησης, όπως ο αλγόριθμος Stochastic Gradient Descent (SGD), ο

οποίος προσαρμόζει τα βάρη με βάση τις υπολογισμένες κλίσεις και έναν ρυθμό μάθησης.

7. **Εποχές και σύγκλιση**: Η διαδικασία της προς τα εμπρός διάδοσης, του υπολογισμού της απώλειας και της οπισθοδιάδοσης επαναλαμβάνεται για πολλές εποχές ή επαναλήψεις σε ολόκληρο το σύνολο δεδομένων εκπαίδευσης, έως ότου η απόδοση του δικτύου συγκλίνει και η απώλεια σταθεροποιηθεί [Hay99].

2.2.3 <Εφαρμογές των MLPs>

Τα Multilayer Perceptrons είναι ευέλικτα και έχουν εφαρμοστεί με επιτυχία σε ένα ευρύ φάσμα προβλημάτων, όπως:

- **Ταξινόμηση**: Τα MLP χρησιμοποιούνται για εργασίες ταξινόμησης, όπου ο στόχος είναι να κατατάξουν τα δεδομένα εισόδου σε μία από διάφορες κλάσεις. Οι εφαρμογές περιλαμβάνουν την αναγνώριση εικόνων (π.χ. αναγνώριση αντικειμένων σε εικόνες), την αναγνώριση χειρόγραφων ψηφίων και την ιατρική διάγνωση (π.χ. ταξινόμηση ασθενειών με βάση δεδομένα ασθενών).

- **Παλινδρόμηση**: Τα MLP μπορούν να μοντελοποιήσουν συνεχείς σχέσεις μεταξύ μεταβλητών εισόδου και εξόδου, καθιστώντας τα χρήσιμα για εργασίες παλινδρόμησης. Παραδείγματα περιλαμβάνουν την πρόβλεψη των τιμών των μετοχών, την πρόβλεψη των πωλήσεων και την εκτίμηση της αξίας των ακινήτων.

- **Προσέγγιση συναρτήσεων**: Αυτό είναι χρήσιμο σε σενάρια όπου η ρητή μοντελοποίηση της σχέσης μεταξύ εισόδων και εξόδων είναι δύσκολη. Αυτό περιλαμβάνει εργασίες όπως η μοντελοποίηση και ο έλεγχος συστημάτων.

- **Πρόβλεψη χρονοσειρών**: Αν και τα RNN και LSTM χρησιμοποιούνται πιο συχνά για διαδοχικά δεδομένα, τα MLP μπορούν να εφαρμοστούν στην πρόβλεψη χρονοσειρών ενσωματώνοντας μεταβλητές με χρονική υστέρηση ως χαρακτηριστικά εισόδου. Οι εφαρμογές περιλαμβάνουν την πρόβλεψη οικονομικών δεικτών και την πρόβλεψη προτύπων ζήτησης.

- **Μάθηση χαρακτηριστικών**: Οι MLP μπορούν να χρησιμοποιηθούν για εργασίες μάθησης χωρίς επίβλεψη, όπως η εξαγωγή χαρακτηριστικών και η μείωση της διαστατικότητας. Μπορούν να ανακαλύψουν μοτίβα και αναπαραστάσεις στα δεδομένα που είναι χρήσιμα για άλλες εργασίες, όπως η ομαδοποίηση και η ανίχνευση ανωμαλιών.

Συνοψίζοντας, το Multilayer Perceptron είναι μια θεμελιώδης αρχιτεκτονική νευρωνικών δικτύων με ευρεία εφαρμογή σε διάφορους τομείς. Η ικανότητά του να μαθαίνει πολύπλοκα μοτίβα και σχέσεις από τα δεδομένα το καθιστά ένα ισχυρό εργαλείο τόσο για εργασίες ταξινόμησης όσο και παλινδρόμησης. Στην ενότητα που ακολουθεί, θα διερευνήσουμε τη συγκεκριμένη εφαρμογή των MLP στην έρευνά μας, εστιάζοντας στην υλοποίησή τους στο Cyclone II FPGA.

Κεφάλαιο 3. < Θεωρία FPGA και αρχιτεκτονική Cyclone II >

3.1 Εισαγωγή στα FPGA

Τα Field-Programmable Gate Arrays (FPGA) είναι ευέλικτες και επαναδιαμορφώσιμες διατάξεις ημιαγωγών που προσφέρουν ένα μοναδικό μείγμα αποδοτικότητας υλικού και ευελιξίας λογισμικού. Σε αντίθεση με το παραδοσιακό υλικό σταθερής λειτουργίας, όπως τα ολοκληρωμένα κυκλώματα ειδικής εφαρμογής (ASIC), τα FPGA μπορούν να επαναπρογραμματιστούν για να εκτελέσουν ένα ευρύ φάσμα υπολογιστικών εργασιών. Όπως περιγράφεται στις [AH18] και [LW15], στην παρούσα ενότητα παρουσιάζονται οι θεμελιώδεις έννοιες των FPGA, η αρχιτεκτονική τους και τα πλεονεκτήματά τους, θέτοντας τις βάσεις για την κατανόηση της καταλληλότητάς τους για την υλοποίηση βαθιών νευρωνικών δικτύων (DNN).

3.1.1 < Βασικές έννοιες και αρχιτεκτονική >

Ένα FPGA αποτελείται από μια σειρά προγραμματιζόμενων λογικών μπλοκ, διασυνδέσεων και μπλοκ εισόδου/εξόδου (I/O). Αυτά τα στοιχεία μπορούν να διαμορφωθούν ώστε να εκτελούν σύνθετους ψηφιακούς υπολογισμούς, καθιστώντας

τις FPGA εξαιρετικά προσαρμόσιμες σε διάφορες εφαρμογές. Τα κύρια στοιχεία ενός FPGA περιλαμβάνουν:

1. **Λογικά μπλοκ**: Πρόκειται για τις θεμελιώδεις δομικές μονάδες ενός FPGA, ικανές να υλοποιούν βασικές λογικές λειτουργίες όπως AND, OR, NOT, καθώς και πιο σύνθετη συνδυαστική και ακολουθιακή λογική. Κάθε λογικό μπλοκ περιέχει συνήθως έναν πίνακα αναζήτησης (LUT), flip-flops και πολυπλέκτες.

2. **Διασυνδέσεις**: Πρόκειται για προγραμματιζόμενα κανάλια δρομολόγησης που συνδέουν τα λογικά μπλοκ εντός του FPGA. Παρέχουν τα μονοπάτια μέσω των οποίων ταξιδεύουν τα σήματα δεδομένων, επιτρέποντας στα λογικά μπλοκ να επικοινωνούν και να συνεργάζονται για την εκτέλεση σύνθετων εργασιών.

3. **Μπλοκ εισόδου/εξόδου**: Αυτά διασυνδέουν την FPGA με εξωτερικές συσκευές και συστήματα, επιτρέποντας τη ροή δεδομένων μέσα και έξω από την FPGA. Υποστηρίζουν διάφορα πρότυπα και πρωτόκολλα σηματοδότησης, καθιστώντας τις FPGA κατάλληλες για ένα ευρύ φάσμα εφαρμογών.

4. **Μπλοκ διαμορφώσιμης λογικής (CLBs)**: Πρόκειται για ομάδες λογικών μπλοκ που μπορούν να διαμορφωθούν για την υλοποίηση συγκεκριμένων λειτουργιών. Τα CLBs συνδέονται μέσω του δικτύου διασύνδεσης, επιτρέποντας το συνδυασμό τους με διάφορους τρόπους για την επίτευξη της επιθυμητής λογικής λειτουργικότητας.

3.1.2 < Δυνατότητα αναδιαμόρφωσης και ευελιξία >

Ένα από τα βασικά πλεονεκτήματα των FPGA είναι η δυνατότητα αναδιαμόρφωσης. Οι χρήστες μπορούν να επαναπρογραμματίσουν το υλικό της FPGA για να προσαρμοστούν σε διαφορετικές εργασίες και εφαρμογές, κατεβάζοντας ένα νέο bitstream διαμόρφωσης. Αυτή η ευελιξία επιτρέπει στους προγραμματιστές να επαναλαμβάνουν

γρήγορα τα σχέδια υλικού και να βελτιστοποιούν τις επιδόσεις για συγκεκριμένες εφαρμογές χωρίς την ανάγκη κατασκευής νέου υλικού. Η δυνατότητα επαναδιαμόρφωσης σημαίνει επίσης ότι οι FPGA μπορούν να ενημερώνονται στο πεδίο, επιτρέποντας συνεχείς βελτιώσεις και προσαρμογές στις μεταβαλλόμενες απαιτήσεις. Αυτό καθιστά τις FPGA ιδιαίτερα χρήσιμες σε δυναμικά περιβάλλοντα όπου οι ανάγκες των εφαρμογών μπορεί να εξελίσσονται με την πάροδο του χρόνου [LW15].

3.1.3 < Δυνατότητες παράλληλης επεξεργασίας >

Οι FPGA είναι εγγενώς παράλληλες, πράγμα που σημαίνει ότι πολλαπλοί υπολογισμοί μπορούν να πραγματοποιούνται ταυτόχρονα. Αυτός ο παραλληλισμός προκύπτει από τη δυνατότητα διαμόρφωσης πολλαπλών λογικών μπλοκ για την ταυτόχρονη εκτέλεση διαφορετικών τμημάτων μιας εργασίας. Αυτό καθιστά τις FPGA κατάλληλες για εφαρμογές που απαιτούν επεξεργασία υψηλής απόδοσης και απόδοση σε πραγματικό χρόνο, όπως η ψηφιακή επεξεργασία σήματος (DSP), η επεξεργασία εικόνας και η μηχανική μάθηση [AH18].

3.1.4 < Πλεονεκτήματα των FPGAs >

Τα FPGAs προσφέρουν αρκετά πλεονεκτήματα έναντι των παραδοσιακών πλατφορμών υλικού, όπως οι CPUs και οι GPUs, ιδίως σε ορισμένες εφαρμογές:

I. **Απόδοση**: Τα FPGA μπορούν να επιτύχουν υψηλές επιδόσεις μέσω παραλληλισμού σε επίπεδο υλικού και προσαρμογής, καθιστώντας τις ικανές να ικανοποιούν τις απαιτήσεις επεξεργασίας σε πραγματικό χρόνο.

II. **Ενεργειακή απόδοση**: Τα FPGA συχνά καταναλώνουν λιγότερη ενέργεια από τις CPU και τις GPU για ισοδύναμες εργασίες, επειδή μπορούν να βελτιστοποιηθούν σε επίπεδο υλικού για συγκεκριμένους υπολογισμούς.

III. **Ευελιξία**: Η δυνατότητα αναδιαμόρφωσης των FPGA επιτρέπει την ταχεία δημιουργία πρωτοτύπων και ανάπτυξη, καθώς και ενημερώσεις και προσαρμογές στο πεδίο.

IV. **Αποδοτικότητα κόστους**: Για μέτριους όγκους παραγωγής, οι FPGA μπορεί να είναι πιο αποδοτικές από τον σχεδιασμό και την κατασκευή προσαρμοσμένων ASIC, οι οποίες συνεπάγονται σημαντικό αρχικό κόστος και μεγαλύτερους χρόνους ανάπτυξης.

3.1.5 < Εφαρμογές των FPGA >

Λόγω της ευελιξίας τους, οι FPGAs χρησιμοποιούνται σε ένα ευρύ φάσμα εφαρμογών σε διάφορους κλάδους, όπως:

- **Τηλεπικοινωνίες**: Υλοποίηση υψηλής ταχύτητας επεξεργασίας δεδομένων και πρωτοκόλλων επικοινωνίας.
- **Αυτοκινητοβιομηχανία**: Ενεργοποίηση προηγμένων συστημάτων υποβοήθησης οδηγού (ADAS) και χαρακτηριστικών αυτόνομης οδήγησης.
- **Αεροδιαστημική και άμυνα**: Παροχή στιβαρής και αξιόπιστης επεξεργασίας για συστήματα κρίσιμης σημασίας.
- **Υγειονομική περίθαλψη**: Επιτάχυνση υπολογισμών σε ιατρικές απεικονίσεις και διαγνωστικό εξοπλισμό.
- **Καταναλωτικά ηλεκτρονικά**: Ενίσχυση της απόδοσης σε συσκευές όπως smartphones, κονσόλες παιχνιδιών και έξυπνες τηλεοράσεις.
- **Κέντρα δεδομένων**: Εκφόρτωση και επιτάχυνση εργασιών έντασης υπολογιστών για τη βελτίωση της συνολικής απόδοσης και αποδοτικότητας του συστήματος.

Στο πλαίσιο των βαθιών νευρωνικών δικτύων (DNN), οι FPGA προσφέρουν μια πολλά υποσχόμενη πλατφόρμα για την υλοποίηση και επιτάχυνση αυτών των μοντέλων. Οι

δυνατότητες παράλληλης επεξεργασίας τους, η ενεργειακή αποδοτικότητα και η δυνατότητα αναδιαμόρφωσης τα καθιστούν κατάλληλα τόσο για εργασίες εκπαίδευσης όσο και για εργασίες εξαγωγής συμπερασμάτων, ιδίως σε περιβάλλοντα με περιορισμένους πόρους. Στην επόμενη ενότητα, θα εμβαθύνουμε στις ιδιαιτερότητες της FPGA Cyclone II, εξερευνώντας λεπτομερώς την αρχιτεκτονική και τα χαρακτηριστικά της.

3.2 Το Cyclone II FPGA

Η οικογένεια FPGA Cyclone II, που αναπτύχθηκε από την Altera (που τώρα ανήκει στην Intel), αναγνωρίζεται για την προσφορά βέλτιστης ισορροπίας μεταξύ επιδόσεων, αποδοτικότητας ισχύος και σχέσης κόστους-αποτελεσματικότητας, καθιστώντας την ιδιαίτερα κατάλληλη για ένα ευρύ φάσμα εφαρμογών, συμπεριλαμβανομένων των ενσωματωμένων συστημάτων, της ψηφιακής επεξεργασίας σήματος και, ολοένα και περισσότερο, των βαθιών νευρωνικών δικτύων (DNN). Όπως περιγράφεται λεπτομερώς στο εγχειρίδιο συσκευής Cyclone II της Intel, το FPGA Cyclone II EP2C35F672C6 παρέχει μια στιβαρή και ευέλικτη πλατφόρμα για την υλοποίηση και τη δοκιμή αρχιτεκτονικών DNN εντός ενός περιορισμένου περιβάλλοντος υλικού [IC17].

Το EP2C35F672C6 FPGA αποτελείται από περίπου 33.000 λογικά στοιχεία (LEs), καθιστώντας το ικανό να χειρίζεται σημαντικές υπολογιστικές εργασίες. Κάθε LE αποτελείται από έναν πίνακα αναζήτησης (LUT) και ένα flip-flop, τα οποία μαζί επιτρέπουν την υλοποίηση τόσο συνδυαστικών όσο και ακολουθιακών λογικών λειτουργιών. Αυτή η εκτεταμένη συστοιχία LEs επιτρέπει στο FPGA να φιλοξενήσει πολύπλοκες αρχιτεκτονικές DNN, οι οποίες απαιτούν εκτεταμένες δυνατότητες παράλληλης επεξεργασίας. Ένα άλλο κρίσιμο συστατικό του EP2C35F672C6 FPGA είναι οι 70 ενσωματωμένοι πολλαπλασιαστές του. Αυτοί οι αποκλειστικοί πολλαπλασιαστές είναι απαραίτητοι για την αποτελεσματική εκτέλεση αριθμητικών πράξεων, όπως τα εσωτερικά γινόμενα, τα οποία είναι κεντρικής σημασίας στα νευρωνικά δίκτυα καθώς όλοι οι παρτάμετροι στο σύστημα μας είναι σε διανυσματική μορφή. Με τη χρήση πολλαπλασιαστών που βασίζονται σε υλικό αντί για πολλαπλασιασμό που υλοποιείται με λογισμικό, η ταχύτητα και η αποτελεσματικότητα των λειτουργιών των DNN βελτιώνονται σημαντικά, επιτρέποντας την επεξεργασία δεδομένων σε πραγματικό

χρόνο. Κάθε νευρώνας χρησιμοποιεί αυτούς τους πολλαπλασιαστές για τον υπολογισμό των σταθμισμένων αθροισμάτων των εισόδων. Επιπλέον, μέσω χρονικού διαμοιρασμού της επεξεργασίας των επιπέδων, οι πολλαπλασιαστές μπορούν να επαναχρησιμοποιηθούν σε πολλαπλούς νευρώνες, επιτρέποντας την υποστήριξη περισσότερων νευρώνων από τον αριθμό των φυσικών πολλαπλασιαστών. Το FPGA διαθέτει επίσης διάφορα μπλοκ μνήμης στο τσιπ, συμπεριλαμβανομένης της μνήμης RAM, τα οποία είναι απαραίτητα για την αποθήκευση των ενδιάμεσων δεδομένων, των βαρών και των ενεργοποιήσεων που δημιουργούνται κατά την επεξεργασία των νευρωνικών δικτύων. Η αρχιτεκτονική της μνήμης του EP2C35F672C6 έχει σχεδιαστεί για να υποστηρίζει γρήγορα και αποδοτικά πρότυπα πρόσβασης, γεγονός που είναι ζωτικής σημασίας για τη διατήρηση υψηλής απόδοσης σε υλοποιήσεις DNN. Αυτή η δυνατότητα είναι ιδιαίτερα σημαντική όταν πρόκειται για μεγάλα σύνολα δεδομένων ή όταν το μοντέλο απαιτεί συχνή πρόσβαση στις αποθηκευμένες παραμέτρους κατά την εκπαίδευση και την εξαγωγή συμπερασμάτων. Οι βρόχοι κλειδώματος φάσης (PLL) στο Cyclone II EP2C35F672C6 χρησιμοποιούνται για τη διαχείριση του ρολογιού, διασφαλίζοντας ότι τα διάφορα τμήματα της FPGA λειτουργούν συγχρονισμένα. Η δυνατότητα δημιουργίας πολλαπλών συχνοτήτων ρολογιού επιτρέπει τη λεπτομερή ρύθμιση της απόδοσης διαφορετικών μονάδων εντός της FPGA, βελτιστοποιώντας το συγχρονισμό των λειτουργιών που είναι κρίσιμες για την απόδοση του DNN. Η FPGA είναι εξοπλισμένη με πολυάριθμα διαμορφώσιμα pins εισόδου/εξόδου, οι οποίες υποστηρίζουν διάφορα πρότυπα τάσης και πρωτόκολλα επικοινωνίας. Αυτή η ευελιξία επιτρέπει στην FPGA να διασυνδέεται απρόσκοπτα με εξωτερικές συσκευές, αισθητήρες και περιφερειακά εξαρτήματα, καθιστώντας την μια ευέλικτη πλατφόρμα για ένα ευρύ φάσμα εφαρμογών, συμπεριλαμβανομένης της ανάπτυξης DNN σε περιβάλλοντα υπολογιστών άκρων. Συνολικά, το FPGA Cyclone II EP2C35F672C6 στην πλακέτα DE2 προσφέρει ένα καλά ισορροπημένο μείγμα επεξεργαστικής ισχύος, πόρων μνήμης και δυνατοτήτων εισόδου/εξόδου, καθιστώντας το ιδανική επιλογή για τη διερεύνηση των ορίων της υλοποίησης του DNN σε τεχνολογία FPGA. Η αρχιτεκτονική και οι πόροι της είναι κατάλληλα προσαρμοσμένοι στον παραλληλισμό και την υπολογιστική ένταση που είναι εγγενείς στις λειτουργίες DNN, παρέχοντας μια πρακτική πλατφόρμα για την εκτέλεση πολύπλοκων μοντέλων νευρωνικών δικτύων σε πραγματικό χρόνο.

3.3 Το DE2 Development Board

Η αναπτυξιακή πλακέτα DE2, που παράγεται από την Terasic Technologies, χρησιμεύει ως η κύρια πλατφόρμα υλικού για την παρούσα μελέτη. Εξοπλισμένη με το Cyclone II EP2C35F672C6 FPGA, η πλακέτα DE2 παρέχει ένα ολοκληρωμένο περιβάλλον για τον σχεδιασμό, την υλοποίηση και τη δοκιμή σύνθετων ψηφιακών συστημάτων, συμπεριλαμβανομένων των βαθιών νευρωνικών δικτύων (DNN). Όπως αναφέρεται λεπτομερώς στο εγχειρίδιο της Altera, η πλακέτα DE2 προσφέρει διάφορες επιλογές μνήμης, όπως SRAM (512 KB), SDRAM (8 MB) και Flash (4 MB). Ενώ αυτοί οι πόροι είναι πολύτιμοι για την αποθήκευση μεγάλων συνόλων δεδομένων, πινάκων βαρών και ενδιάμεσων υπολογισμών που είναι απαραίτητοι για την επεξεργασία του DNN, επιδιώξαμε να ελαχιστοποιήσουμε τη χρήση τους. Η αξιοποίηση αυτών των μπλοκ μνήμης θα απαιτούσε τη δημιουργία νέων μονάδων για τη μεταφορά βαρών και άλλων δεδομένων, κάτι που θα κατανάλωνε πρόσθετα λογικά στοιχεία. Αντ' αυτού, δώσαμε προτεραιότητα στη διατήρηση αυτών των πόρων για τους ίδιους τους νευρώνες. Η πλακέτα DE2 διαθέτει επίσης μια σειρά περιφερειακών διεπαφών που διευκολύνουν την αλληλεπίδραση με την FPGA και τις εξωτερικές συσκευές. Αυτές περιλαμβάνουν pins εισόδου/εξόδου γενικού σκοπού (GPIO), διακόπτες, LED και οθόνες 7 τμημάτων, οι οποίες είναι ανεκτίμητες για την αποσφαλμάτωση, την αλληλεπίδραση με τον χρήστη και την οπτικοποίηση των λειτουργιών του DNN σε πραγματικό χρόνο. Οι θύρες επικοινωνίας της πλακέτας -συμπεριλαμβανομένων των USB, RS-232 και Ethernet- επιτρέπουν την αποτελεσματική μεταφορά δεδομένων και τη συνδεσιμότητα με εξωτερικές συσκευές και δίκτυα, κάτι που είναι ιδιαίτερα σημαντικό για εφαρμογές που περιλαμβάνουν επεξεργασία δεδομένων μεγάλης κλίμακας ή απαιτούν ενσωμάτωση με άλλα συστήματα. Επιπλέον, η πλακέτα DE2 είναι εξοπλισμένη με διεπαφές πολυμέσων, όπως υποδοχές VGA και ήχου, παρέχοντας τη δυνατότητα ανάπτυξης και δοκιμής εφαρμογών πολυμέσων. Αυτές οι διεπαφές επιτρέπουν την ανάπτυξη των DNN σε σενάρια που περιλαμβάνουν επεξεργασία εικόνας και ήχου, καθιστώντας την πλακέτα DE2 μια ευέλικτη πλατφόρμα για ένα ευρύ φάσμα εφαρμογών. Το περιβάλλον ανάπτυξης για την πλακέτα DE2 υποστηρίζεται από διάφορα εργαλεία λογισμικού, με κυριότερο το λογισμικό Quartus II, το οποίο είναι το κύριο εργαλείο για τη σχεδίαση, τη σύνθεση και την υλοποίηση FPGA. Το Quartus II υποστηρίζει την εισαγωγή σχεδιασμού σε γλώσσα περιγραφής υλικού (HDL), την προσομοίωση και την αποσφαλμάτωση υλικού, προσφέροντας μια ολοκληρωμένη σουίτα εργαλείων για την ανάπτυξη σύνθετων ψηφιακών συστημάτων. Για σχέδια που ενσωματώνουν soft processors, το

Nios II Embedded Design Suite (EDS) παρέχει εργαλεία για την ανάπτυξη και την αποσφαλμάτωση ενσωματωμένου λογισμικού, επεκτείνοντας περαιτέρω τις δυνατότητες της πλακέτας DE2. Το βοηθητικό πρόγραμμα Terasic System Builder απλοποιεί τη διαδικασία διαμόρφωσης της πλακέτας DE2, επιτρέποντας στους χρήστες να ρυθμίσουν γρήγορα το υλικό και τις περιφερειακές συνδέσεις τους. Αυτό το εργαλείο είναι ιδιαίτερα χρήσιμο για την ταχεία δημιουργία πρωτοτύπων, επιτρέποντας στους προγραμματιστές να επαναλαμβάνουν τα σχέδιά τους γρήγορα και αποτελεσματικά. Συνοψίζοντας, η αναπτυξιακή πλακέτα DE2, με το Cyclone II EP2C35F672C6 FPGA και τις εκτεταμένες περιφερειακές λειτουργίες, παρέχει μια ισχυρή και ευέλικτη πλατφόρμα για την υλοποίηση και τον πειραματισμό με βαθιά νευρωνικά δίκτυα. Το ολοκληρωμένο περιβάλλον ανάπτυξής του, σε συνδυασμό με την επεξεργαστική ισχύ και τους πόρους μνήμης της FPGA, διευκολύνει τον αποτελεσματικό σχεδιασμό, τη δοκιμή και τη βελτιστοποίηση των μοντέλων DNN. Αυτό καθιστά την πλακέτα DE2 μια ιδανική πλατφόρμα για τη διερεύνηση των πρακτικών ορίων της υλοποίησης DNN σε FPGA, παρέχοντας πολύτιμες γνώσεις σχετικά με τις προκλήσεις και τις ευκαιρίες της ανάπτυξης νευρωνικών δικτύων σε περιβάλλοντα με περιορισμένους πόρους [AC12].

Στο επόμενο κεφάλαιο, θα εμβαθύνουμε στις πρακτικές πτυχές της υλοποίησης ενός DNN στο Cyclone II EP2C35F672C6 FPGA χρησιμοποιώντας την πλακέτα DE2. Αυτό περιλαμβάνει λεπτομερείς συζητήσεις σχετικά με τις στρατηγικές σχεδίασης, τις τεχνικές βελτιστοποίησης και τη διαδικασία υλοποίησης βήμα προς βήμα.

Κεφάλαιο 4. < Υλοποίηση του DNN στο FPGA>

Αυτό το κεφάλαιο περιγράφει τη διαδικασία υλοποίησης ενός βαθύ νευρωνικού δικτύου (DNN), συγκεκριμένα ενός πολυεπίπεδου perceptron (MLP), στο Cyclone II EP2C35F672C6 FPGA. Εμβαθύνει στις εκτιμήσεις σχεδιασμού που είναι απαραίτητες για την επιτυχή υλοποίηση, στα βήματα που αφορούν την κατασκευή του DNN για την ανάπτυξη σε FPGA, στις τεχνικές βελτιστοποίησης που εφαρμόζονται για την ενίσχυση της απόδοσης και στη λεπτομερή διαδικασία υλοποίησης στο Cyclone II. Αυτό το κεφάλαιο είναι δομημένο έτσι ώστε να καθοδηγεί τον αναγνώστη σε κάθε φάση του έργου, από τον εννοιολογικό σχεδιασμό έως την πρακτική υλοποίηση στο υλικό FPGA.

4.1 Εκτιμήσεις Σχεδιασμού

Η υλοποίηση ενός DNN σε ένα FPGA απαιτεί προσεκτικό σχεδιασμό και εξέταση διαφόρων παραγόντων που επηρεάζουν την απόδοση και τη σκοπιμότητα της σχεδίασης. Σε αυτούς περιλαμβάνονται η κατανομή των πόρων, η καθυστέρηση, η απόδοση και η κατανάλωση ενέργειας. Καθένας από αυτούς τους παράγοντες πρέπει να εξεταστεί για να διασφαλιστεί ότι το DNN μπορεί να υλοποιηθεί αποτελεσματικά εντός

των περιορισμών της FPGA, ενώ παράλληλα θα πληροί τις επιθυμητές μετρήσεις επιδόσεων.

4.1.1 < Κατανομή πόρων >

Το Cyclone II EP2C35F672C6 FPGA παρέχει έναν πεπερασμένο αριθμό πόρων, συμπεριλαμβανομένων περίπου 36.000 λογικών στοιχείων (LEs), 70 ενσωματωμένων πολλαπλασιαστών και αρκετών μπλοκ μνήμης on-chip. Αυτοί οι πόροι είναι απαραίτητοι για την υλοποίηση των διαφόρων στοιχείων του DNN, όπως οι νευρώνες, οι συναρτήσεις ενεργοποίησης και η αποθήκευση βάρους.

Αρχικά, τα λογικά στοιχεία (LEs) είναι τα δομικά στοιχεία της προγραμματιζόμενης λογικής της FPGA, ικανά να υλοποιούν βασικές συνδυαστικές και ακολουθιακές λογικές λειτουργίες. Στο πλαίσιο ενός DNN, τα LE χρησιμοποιούνται για την υλοποίηση της λογικής για τις λειτουργίες των νευρώνων, όπως το σταθμισμένο άθροισμα των εισόδων και η εφαρμογή των συναρτήσεων ενεργοποίησης. Η πρόκληση έγκειται στη διασφάλιση ότι ο αριθμός των νευρώνων και των στρωμάτων στο MLP δεν υπερβαίνει τα διαθέσιμα LEs, γεγονός που απαιτεί έναν αποτελεσματικό σχεδιασμό που μεγιστοποιεί τη χρήση κάθε LE. Οι ενσωματωμένοι πολλαπλασιαστές είναι ζωτικής σημασίας για την εκτέλεση του μεγάλου αριθμού αριθμητικών πράξεων που απαιτούνται στους υπολογισμούς του DNN, ιδίως στο εμπρόσθιο και στο οπίσθιο πέρασμα του MLP. Κάθε νευρώνας απαιτεί πολλαπλούς πολλαπλασιασμούς, οι οποίοι στη συνέχεια αθροίζονται για την παραγωγή της εξόδου. Δεδομένου του περιορισμένου αριθμού των ενσωματωμένων πολλαπλασιαστών, είναι απαραίτητη η στρατηγική κατανομή αυτών των πόρων, ενδεχομένως μοιράζοντας τους πολλαπλασιαστές μεταξύ των νευρώνων ή χρησιμοποιώντας τεχνικές όπως η χρονική πολυπλεξία για να διασφαλιστεί ότι το DNN μπορεί να υλοποιηθεί εντός των διαθέσιμων πόρων. Τα μπλοκ μνήμης στην FPGA χρησιμοποιούνται για την αποθήκευση των βαρών, των bias και των ενδιάμεσων δεδομένων που παράγονται κατά τη διάρκεια του υπολογισμού. Η αποδοτική κατανομή μνήμης είναι ζωτικής σημασίας για να εξασφαλιστεί ότι το DNN λειτουργεί εντός των περιορισμών της FPGA χωρίς να εξαντλείται η μνήμη. Ο σχεδιασμός πρέπει να λαμβάνει υπόψη το μέγεθος και τον αριθμό των μπλοκ μνήμης, διασφαλίζοντας ότι τα βάρη και οι ενεργοποιήσεις μπορούν να αποθηκευτούν και να προσπελαστούν αποτελεσματικά. Αυτό μπορεί να περιλαμβάνει τη χρήση εξωτερικών πόρων μνήμης ή τη βελτιστοποίηση του εύρους των bit των δεδομένων για τη μείωση

της χρήσης της μνήμης. Η κατανομή των πόρων πρέπει να εξισορροπείται με την ανάγκη για επεκτασιμότητα. Καθώς αυξάνεται η πολυπλοκότητα του DNN (π.χ. περισσότερα επίπεδα, περισσότεροι νευρώνες ανά επίπεδο), αυξάνεται επίσης η ζήτηση για LEs, πολλαπλασιαστές και μνήμη. Ο σχεδιασμός θα πρέπει να είναι κλιμακούμενος, επιτρέποντας μελλοντικές βελτιώσεις ή την προσθήκη πιο σύνθετων αρχιτεκτονικών DNN χωρίς να απαιτείται πλήρης επανασχεδιασμός.

4.1.2 < Καθυστέρηση και απόδοση >

Η καθυστέρηση και η απόδοση είναι κρίσιμες μετρικές επιδόσεων για κάθε εφαρμογή DNN, ιδίως όταν το μοντέλο αναπτύσσεται σε εφαρμογές πραγματικού χρόνου ή υψηλών επιδόσεων.

Η καθυστέρηση αναφέρεται στο χρόνο που χρειάζεται μια είσοδος για να διαδοθεί μέσω ολόκληρου του δικτύου και να παράγει μια έξοδο. Στο πλαίσιο μιας υλοποίησης FPGA, η καθυστέρηση επηρεάζεται από παράγοντες όπως το βάθος του MLP (δηλαδή ο αριθμός των στρωμάτων), η ταχύτητα των αριθμητικών πράξεων και η αποτελεσματικότητα της μεταφοράς δεδομένων μεταξύ των στρωμάτων. Για την ελαχιστοποίηση της καθυστέρησης, ο σχεδιασμός πρέπει να διασφαλίζει ότι η κρίσιμη διαδρομή -η μεγαλύτερη διαδρομή που πρέπει να διανύσουν τα δεδομένα μέσω του δικτύου- είναι βελτιστοποιημένη. Τεχνικές όπως η διοχέτευση μπορούν να χρησιμοποιηθούν για τη διάσπαση του υπολογισμού σε μικρότερα, πιο διαχειρίσιμα στάδια, επιτρέποντας την ταυτόχρονη επεξεργασία πολλαπλών εισόδων σε διαφορετικά στάδια της διοχέτευσης. Αυτό όχι μόνο μειώνει την καθυστέρηση αλλά αυξάνει και τη συνολική απόδοση. Η απόδοση, από την άλλη πλευρά, μετρά τον αριθμό των εισόδων που μπορούν να υποβληθούν σε επεξεργασία ανά μονάδα χρόνου. Η υψηλή απόδοση είναι ιδιαίτερα σημαντική σε εφαρμογές όπου πρέπει να γίνεται γρήγορη επεξεργασία μεγάλου όγκου δεδομένων, όπως στην αναγνώριση εικόνων ή στην ανάλυση δεδομένων σε πραγματικό χρόνο. Η ικανότητα της FPGA να εκτελεί παράλληλα πολλαπλές λειτουργίες αποτελεί σημαντικό πλεονέκτημα από αυτή την άποψη. Παραλληλοποιώντας τον υπολογισμό των ενεργοποιήσεων των νευρώνων και αξιοποιώντας τους ενσωματωμένους πολλαπλασιαστές της FPGA, ο σχεδιασμός μπορεί να επιτύχει υψηλή απόδοση, επεξεργαζόμενος ταυτόχρονα πολλαπλές εισόδους. Ωστόσο, αυτό πρέπει να εξισορροπηθεί με τους περιορισμούς των πόρων, καθώς η

αύξηση του παραλληλισμού αυξάνει επίσης τη ζήτηση για LEs, πολλαπλασιαστές και μνήμη.

Ο συμβιβασμός μεταξύ λανθάνουσας κατάστασης και ρυθμού μετάδοσης πρέπει να τύχει προσεκτικής διαχείρισης. Ενώ η διοχέτευση και ο παραλληλισμός μπορούν να βελτιώσουν την απόδοση μετάδοσης, μπορεί επίσης να εισάγουν πρόσθετη καθυστέρηση εάν δεν υλοποιηθούν σωστά. Ο σχεδιασμός πρέπει να επιτύχει μια ισορροπία, βελτιστοποιώντας και τις δύο μετρήσεις για να διασφαλιστεί ότι το DNN μπορεί να ανταποκριθεί στις απαιτήσεις απόδοσης της εφαρμογής-στόχου.

4.2 Κατασκευή του DNN (MLP) για FPGA

Η κατασκευή του MLP για την ανάπτυξη σε FPGA περιλαμβάνει πολλαπλά βήματα, από την προετοιμασία του συνόλου δεδομένων έως τον αρχιτεκτονικό σχεδιασμό του δικτύου. Κάθε βήμα πρέπει να εκτελείται προσεκτικά για να διασφαλιστεί ότι το MLP είναι αποτελεσματικό και αποδοτικό όταν υλοποιείται στην FPGA.

4.2.1 < Προετοιμασία συνόλου δεδομένων (Digits Dataset)>

Σε αυτή την ενότητα, περιγράφουμε τη λεπτομερή διαδικασία προετοιμασίας του συνόλου δεδομένων και εκπαίδευσης του Multilayer Perceptron (MLP) με τη χρήση του TensorFlow. Αυτή η προσέγγιση όχι μόνο επιτρέπει την αποτελεσματική εκπαίδευση, αλλά διευκολύνει επίσης την εξαγωγή των βαρών και των bias, οι οποίες είναι ζωτικής σημασίας για την ανάπτυξη του εκπαιδευμένου μοντέλου στην FPGA χωρίς την ανάγκη για εκπαίδευση στο chip.

4.2.1.1 < Φόρτωση και προεπεξεργασία συνόλου δεδομένων>

Χρησιμοποιήσαμε τη συνάρτηση load_digits από την ενότητα sklearn.datasets για να φορτώσουμε το σύνολο δεδομένων 8x8 pixel digits. Αυτό το σύνολο δεδομένων περιέχει 1.797 δείγματα χειρόγραφων ψηφίων, καθένα από τα οποία αναπαρίσταται

ως ένα πλέγμα 8x8 με τιμές εικονοστοιχείων που κυμαίνονται από 0 έως 16. Το σύνολο δεδομένων είναι ιδανικό για υλοποίηση σε FPGA λόγω του εύχρηστου μεγέθους του και της απλότητας των εικόνων.

Μόλις φορτώθηκε το σύνολο δεδομένων, κανονικοποιήσαμε τις τιμές εικονοστοιχείων στο εύρος [0, 1] διαιρώντας κάθε τιμή εικονοστοιχείου με το 16. Αυτή η κανονικοποίηση είναι ζωτικής σημασίας για να διασφαλιστεί ότι τα δεδομένα εισόδου είναι σε μορφή κατάλληλη για την εκπαίδευση του MLP, καθώς βοηθά το νευρωνικό δίκτυο να συγκλίνει πιο αποτελεσματικά, διατηρώντας τις τιμές εισόδου σε ένα τυποποιημένο εύρος.

Στη συνέχεια, το σύνολο δεδομένων χωρίστηκε σε σύνολα εκπαίδευσης και επικύρωσης χρησιμοποιώντας αναλογία 70/30 μέσω της συνάρτησης train_test_split. Αυτός ο διαχωρισμός διασφαλίζει ότι το μοντέλο έχει επαρκή δεδομένα για να μάθει, ενώ παράλληλα παρέχει ένα ξεχωριστό σύνολο δεδομένων για την αξιολόγηση της απόδοσής του κατά τη διάρκεια της εκπαίδευσης.

4.2.1.2 < Σχεδιασμός και εκπαίδευση αρχιτεκτονικής MLP>

Το μοντέλο MLP κατασκευάστηκε χρησιμοποιώντας το API Keras του TensorFlow. Το δίκτυο αποτελείται από τρία πυκνά στρώματα με 45, 37 και 10 νευρώνες αντίστοιχα. Κάθε στρώμα χρησιμοποιεί τη σιγμοειδή συνάρτηση ενεργοποίησης, η οποία είναι κατάλληλη για υλοποίηση σε FPGA λόγω της απλότητας και των ιδιοτήτων ομαλής κλίσης.

- Το πρώτο στρώμα ισοπεδώνει την εικόνα εισόδου 8x8 σε ένα διάνυσμα 64 διαστάσεων.
- Το δεύτερο στρώμα είναι ένα πλήρως συνδεδεμένο στρώμα με 45 νευρώνες.
- Το τρίτο στρώμα έχει 37 νευρώνες και το τελικό στρώμα εξάγει 10 νευρώνες που αντιστοιχούν στις κατηγορίες ψηφίων (0-9).

Το μοντέλο καταρτίστηκε χρησιμοποιώντας τον βελτιστοποιητή Adam και τη συνάρτηση απώλειας Sparse Categorical Crossentropy, η οποία είναι κατάλληλη για

εργασίες ταξινόμησης πολλαπλών κατηγοριών. Για να ενισχύσουμε την αποτελεσματικότητα της εκπαίδευσης και να αποφύγουμε την υπερπροσαρμογή, ενσωματώσαμε την έγκαιρη διακοπή με βάση την ακρίβεια επικύρωσης. Αυτή η στρατηγική διακόπτει την εκπαίδευση εάν η απόδοση του μοντέλου στο σύνολο επικύρωσης δεν βελτιωθεί για πέντε διαδοχικές εποχές, αποκαθιστώντας τα καλύτερα βάρη.

4.2.1.3 < Εξαγωγή βαρών και bias>

Μετά την εκπαίδευση, το επόμενο κρίσιμο βήμα ήταν η εξαγωγή των βαρών και των bias από το εκπαιδευμένο MLP. Αυτό είναι απαραίτητο για την υλοποίηση του MLP στην FPGA, όπου η επανεκπαίδευση του μοντέλου θα ήταν υπολογιστικά απαγορευτική. Χρησιμοποιώντας τη μέθοδο get_weights() του TensorFlow, εξάγαμε τα βάρη και τα bias για κάθε πυκνό στρώμα. Στη συνέχεια, αυτές οι παράμετροι μεταφέρθηκαν και μετατράπηκαν σε μορφή συμβατή με JSON για αποθήκευση. Η απόφαση για τη μετατόπιση των βαρών ελήφθη για να ευθυγραμμιστεί με την αναμενόμενη μορφή εισόδου για την υλοποίηση FPGA, όπου οι πολλαπλασιασμοί πινάκων συχνά απαιτούν συγκεκριμένες διατάξεις δεδομένων για τη βελτιστοποίηση της χρήσης πόρων και της υπολογιστικής απόδοσης. Τα εξαγόμενα βάρη και bias αποθηκεύτηκαν σε ένα αρχείο κειμένου σε μορφή JSON. Αυτό το αρχείο χρησιμεύει ως διαμόρφωση που μπορεί να φορτωθεί στο FPGA κατά την αρχικοποίηση, επιτρέποντας στο υλικό να εκτελεί συμπερασμό απευθείας χρησιμοποιώντας τις εκπαιδευμένες παραμέτρους χωρίς την ανάγκη εκπαίδευσης στο chip.

Εκπαιδεύοντας το MLP στο TensorFlow και εξάγοντας τις παραμέτρους του μοντέλου, εξασφαλίσαμε ότι η FPGA χρειαζόταν μόνο για την εκτέλεση συμπερασμού. Αυτή η προσέγγιση μείωσε σημαντικά τον υπολογιστικό φόρτο στην FPGA, επιτρέποντάς της να λειτουργεί αποτελεσματικά εντός των περιορισμών των πόρων της.

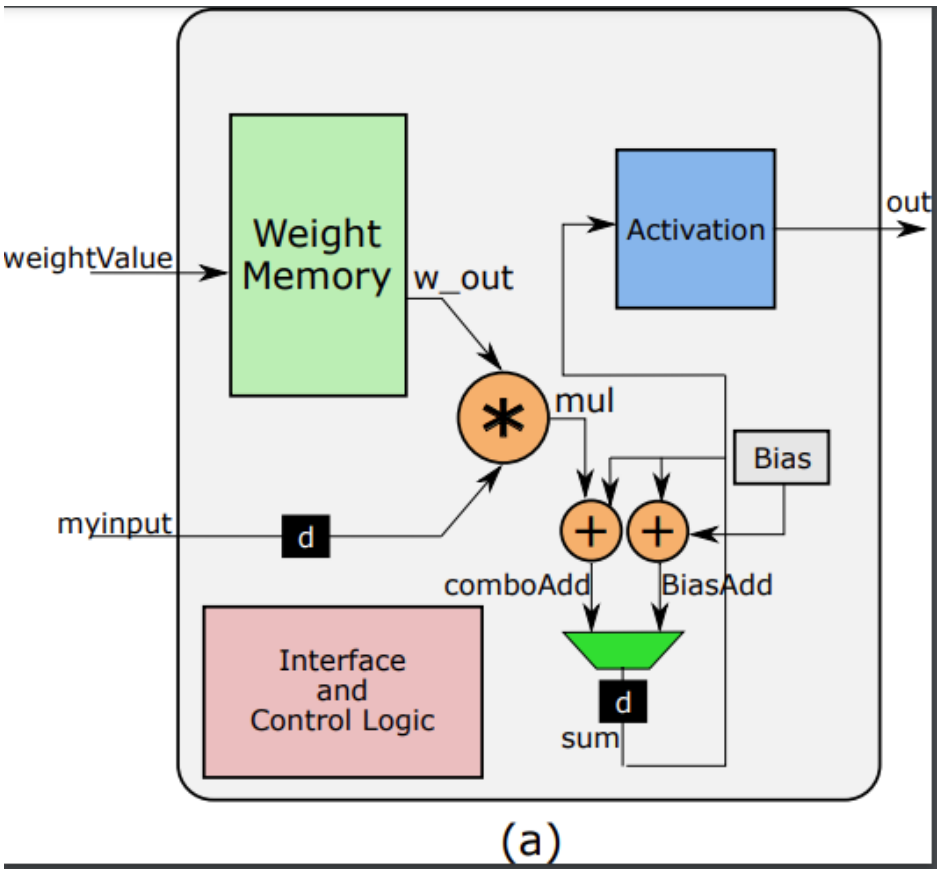
4.2.2 < Σχεδιασμός αρχιτεκτονικής MLP >

Σε αυτή την ενότητα, συζητάμε την αρχιτεκτονική του Multilayer Perceptron (MLP) που υλοποιείται στο FPGA. Η αρχιτεκτονική περιλαμβάνει διάφορα στάδια που επεξεργάζονται τα δεδομένα εισόδου, εφαρμόζοντας βάρη, bias και συναρτήσεις ενεργοποίησης σε πολλαπλά επίπεδα πριν από την παραγωγή της τελικής πρόβλεψης

εξόδου. Παρακάτω, παρέχουμε μια λεπτομερή ανάλυση της αρχιτεκτονικής MLP και των συγκεκριμένων λειτουργιών που λαμβάνουν χώρα σε κάθε στάδιο.

4.2.2.1 < Ο Perceptron>

Η βασική υπολογιστική μονάδα του MLP είναι το perceptron. Κάθε perceptron λαμβάνει εισόδους από το προηγούμενο επίπεδο (ή απευθείας από το επίπεδο εισόδου), τις πολλαπλασιάζει με τα αντίστοιχα βάρη, αθροίζει τα αποτελέσματα, προσθέτει το bias και στη συνέχεια περνά το άθροισμα μέσω μιας συνάρτησης ενεργοποίησης.



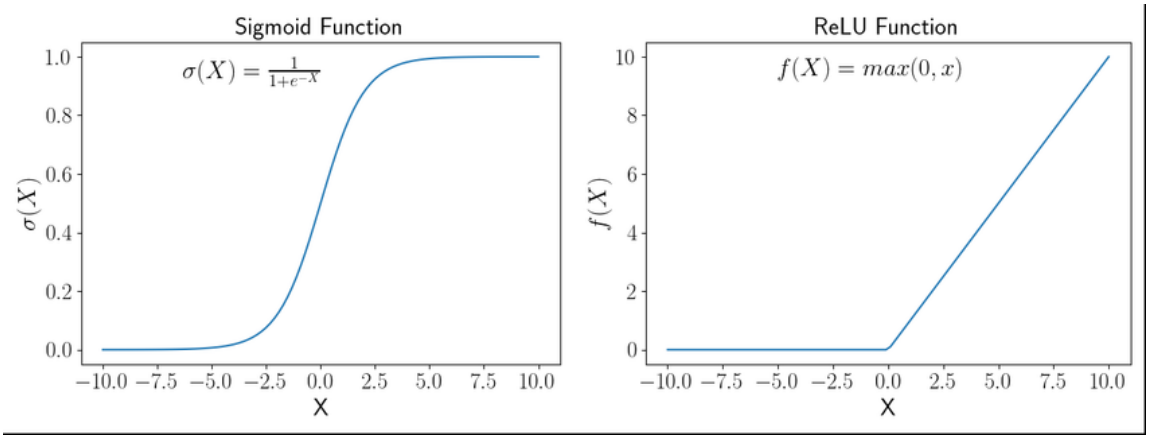
Εικόνα 4.2.1. Ο Perceptron

Μετατροπή εισόδου: Τα δεδομένα εισόδου αποτελούνται από εικόνες που μετατρέπονται πρώτα σε ένα διάνυσμα 64 τιμών, που αντιστοιχούν στις εντάσεις των εικονοστοιχείων. Οι τιμές αυτές αποτελούν το στρώμα εισόδου του MLP.

Αποθήκευση βαρών: Κάθε νευρώνας στο πρώτο στρώμα έχει 64 συσχετιζόμενα βάρη, αποθηκευμένα σε μια ROM στην FPGA. Αυτά τα βάρη έχουν προ-εκπαιδευτεί με τη χρήση του TensorFlow και εξάγονται κατά τη φάση της εκπαίδευσης για να αποφευχθεί το υπολογιστικό κόστος της εκπαίδευσης στο chip.

Υπολογισμός σταθμισμένου αθροίσματος: Κάθε τιμή εισόδου πολλαπλασιάζεται με ένα αντίστοιχο βάρος σε κάθε νευρώνα. Τα αποτελέσματα αθροίζονται και η προκατάληψη προστίθεται σε αυτό το άθροισμα.

Συνάρτηση ενεργοποίησης: Το σταθμισμένο άθροισμα περνά στη συνέχεια από μια συνάρτηση ενεργοποίησης. Πειραματιζόμαστε με δύο συναρτήσεις ενεργοποίησης: την Sigmoid και την ReLU (Rectified Linear Unit) και την σιγμοειδή συνάρτηση, που ορίζονται ως:



Εικόνα 4.2.2. Ορισμός sigmoid και RELU

Η υλοποίηση συναρτήσεων ενεργοποίησης όπως η σιγμοειδής σε υλικό μπορεί να είναι απαιτητική σε πόρους.Η sigmoid υλοποιείται με fixed-point representation σε HDL. Οι τιμές της συνάρτησης σιγμοειδούς υπολογίζονται εκ των προτέρων χρησιμοποιώντας ένα σενάριο Python και αποθηκεύονται σε μια ROM, επιτρέποντας την αποτελεσματική αναζήτηση κατά τη λειτουργία. Το τελικό άθροισμα από κάθε νευρώνα αντιστοιχίζεται στην αντίστοιχη τιμή ενεργοποίησης με βάση αυτόν τον προ-υπολογισμένο πίνακα. Αυτές οι μέθοδοι προσεγγίζουν τις συναρτήσεις ενεργοποίησης χρησιμοποιώντας λιγότερους πόρους από τον άμεσο υπολογισμό, απελευθερώνοντας LEs για άλλες λειτουργίες.

4.2.2.2 < Τα επίπεδα>

Η αρχιτεκτονική MLP αποτελείται από τρία κύρια επίπεδα: ένα επίπεδο εισόδου, δύο κρυφά επίπεδα και ένα επίπεδο εξόδου. Κάθε στρώμα εκτελεί τις ακόλουθες λειτουργίες:

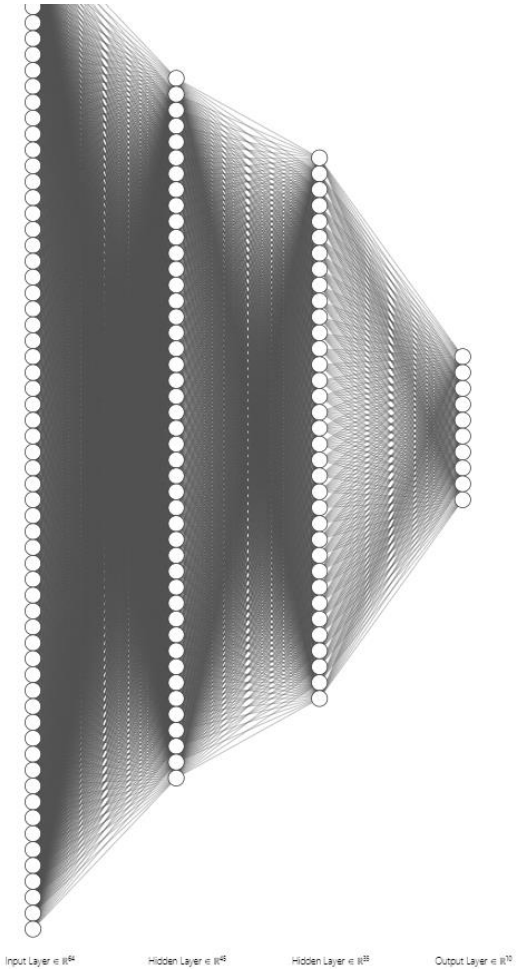
Πρώτο κρυφό στρώμα: Κάθε νευρώνας στο πρώτο κρυφό στρώμα επεξεργάζεται και τις 64 τιμές εισόδου, κάθε μία από τις οποίες σταθμίζεται από το συγκεκριμένο σύνολο

βαρών του νευρώνα. Το πρώτο κρυφό στρώμα αποτελείται από 45 νευρώνες, με αποτέλεσμα 45 εξόδους.

Δεύτερο κρυφό στρώμα: Η έξοδος από το πρώτο στρώμα χρησιμεύει ως είσοδος στο δεύτερο κρυφό στρώμα, το οποίο περιέχει 37 νευρώνες. Κάθε νευρώνας σε αυτό το στρώμα έχει 45 βάρη και παράγει μία τιμή εξόδου.

Στρώμα εξόδου: Το τελευταίο στρώμα αποτελείται από 10 νευρώνες, καθένας από τους οποίους παράγει μια έξοδο που αντιστοιχεί σε μία από τις 10 κλάσεις (ψηφία 0-9). Οι 37 έξοδοι από το δεύτερο κρυφό στρώμα τροφοδοτούνται σε καθέναν από τους 10 νευρώνες εξόδου.

Το αποτέλεσμα του τελικού στρώματος είναι ένα διάνυσμα 10 τιμών που αντιπροσωπεύουν την εμπιστοσύνη του δικτύου σε κάθε πιθανή κλάση για την εικόνα εισόδου.



Εικόνα 4.2.3. Ορισμός sigmoid και RELU

4.2.2.3 < Τελική Συνάρτηση Ενεργοποίησης>

Στο σχεδιασμό του MLP μας, η επιλογή των συναρτήσεων ενεργοποίησης παίζει καθοριστικό ρόλο στην ικανότητα του δικτύου να μαθαίνει και να κάνει προβλέψεις. Ενώ διερευνήσαμε διάφορες συναρτήσεις ενεργοποίησης, συμπεριλαμβανομένων των Sigmoid, ReLU και Softmax, τελικά επιλέξαμε να εφαρμόσουμε την Hardmax στο επίπεδο εξόδου λόγω των ειδικών περιορισμών και απαιτήσεων της υλοποίησης FPGA.

Συνάρτηση ενεργοποίησης Hardmax:

Επιλέξαμε να χρησιμοποιήσουμε τη συνάρτηση Hardmax στο στρώμα εξόδου του MLP μας για να ξεπεράσουμε τις προκλήσεις που θέτει η Softmax. Η Hardmax απλοποιεί τη διαδικασία λήψης αποφάσεων επιλέγοντας τον νευρώνα με την υψηλότερη τιμή ενεργοποίησης και αναθέτοντάς του πιθανότητα 1, ενώ σε όλους τους άλλους νευρώνες ανατίθεται πιθανότητα 0. Η συνάρτηση αυτή ορίζεται ως εξής:

$$\text{hard\_max}(x) = \begin{cases} 1, & \text{if } x \text{ is the maximum element of the input vector} \\ 0, & \text{otherwise} \end{cases}$$

Εικόνα 4.2.5. Ορισμός Hardmax

Η συνάρτηση Hardmax απαιτεί τη σύγκριση των εξόδων των νευρώνων στο στρώμα εξόδου, τον εντοπισμό της μέγιστης τιμής και τη ρύθμιση της αντίστοιχης εξόδου σε 1. Αυτή η διαδικασία είναι υπολογιστικά αποδοτική, καθώς απαιτεί μόνο βασικές πράξεις σύγκρισης και ελάχιστη χρήση πόρων. < Σύνοψη ροής δεδομένων>

Ο MLP επεξεργάζεται τις εικόνες εισόδου μετατρέποντάς τες σε διανύσματα, εφαρμόζοντας βάρη και bias σε κάθε στρώμα και, τέλος, παράγοντας μια πρόβλεψη μέσω της συνάρτησης Hardmax. Αυτή η διαδοχική ροή διασφαλίζει ότι η διαδικασία λήψης αποφάσεων του δικτύου είναι πλήρως διοχετευόμενη και βελτιστοποιημένη για υλοποίηση σε FPGA, εξισορροπώντας την υπολογιστική απόδοση με τους περιορισμούς πόρων.

4.3 Τεχνικές βελτιστοποίησης

Για να εξασφαλιστεί ότι το DNN αποδίδει αποτελεσματικά στην FPGA, εφαρμόζονται διάφορες τεχνικές βελτιστοποίησης. Οι τεχνικές αυτές αποσκοπούν στη μείωση των

απαιτήσεων σε πόρους, στη βελτίωση της υπολογιστικής απόδοσης και στη βελτίωση της συνολικής απόδοσης του MLP.

4.3.1 < Κβαντισμός >

Η κβάντιση είναι η διαδικασία μείωσης της ακρίβειας των βαρών και των bias στο νευρωνικό δίκτυο, η οποία με τη σειρά της μειώνει την υπολογιστική πολυπλοκότητα και τη χρήση μνήμης κατά την ανάπτυξη του μοντέλου σε ένα FPGA.

4.3.1.1 < Fixed-Point Representation>

Στο αρχικό μοντέλο TensorFlow, τα βάρη και τα bias αναπαρίστανται ως αριθμοί κινητής υποδιαστολής. Ενώ αυτό είναι κατάλληλο για υπολογισμούς υψηλής ακρίβειας σε CPU ή GPU, είναι αναποτελεσματικό για υλοποίηση σε FPGA λόγω των περιορισμένων πόρων που είναι διαθέσιμοι στο υλικό. Για να το αντιμετωπίσουμε αυτό, χρησιμοποιήσαμε μια αναπαράσταση σταθερής υποδιαστολής, όπου οι αριθμοί κινητής υποδιαστολής μετατρέπονται σε μορφή σταθερής υποδιαστολής με προκαθορισμένο ακέραιο και κλασματικό πλάτος bit.

Στην υλοποίησή μας, χρησιμοποιήσαμε ένα σενάριο Python για την εκτέλεση αυτής της μετατροπής. Το σενάριο, όπως φαίνεται στις προηγούμενες ενότητες, λαμβάνει τα εκπαιδευμένα βάρη και τα bias κινητής υποδιαστολής και τα μετατρέπει σε δυαδική μορφή σταθερής υποδιαστολής. Τα βασικά βήματα αυτής της μετατροπής περιλαμβάνουν:

1. **Κλιμάκωση**: Οι αριθμοί κινητής υποδιαστολής κλιμακώνονται με συντελεστή 2^fracWidth για τη μετατόπιση του δεκαδικού σημείου και τη μετατροπή του αριθμού σε ακέραιο.
2. **Σύσφιξη**: Η αναπαράσταση του ακέραιου αριθμού συσφίγγεται για να εξασφαλιστεί ότι χωράει στο καθορισμένο πλάτος bit, αποτρέποντας την υπερχείλιση ή την υποχείλιση.
3. **Μετατροπή του συμπληρώματος ως προς δύο**: Για αρνητικούς αριθμούς, το σενάριο τους μετατρέπει στη δυαδική μορφή του συμπληρώματος ως προς δύο.

Αυτή η μετατροπή σταθερού σημείου μειώνει σημαντικά το αποτύπωμα μνήμης του μοντέλου και επιτρέπει αποδοτικούς υπολογισμούς στο FPGA.

4.3.1.2 <Αποθήκευση βάρους και bias>

Μετά την κβάντιση, τα βάρη και τα bias αποθηκεύονται σε ξεχωριστά αρχεία, με κάθε αρχείο να αντιστοιχεί σε έναν συγκεκριμένο νευρώνα του MLP. Αυτά τα αρχεία περιλαμβάνονται στη συνέχεια στο σχέδιο FPGA, επιτρέποντας στην FPGA να έχει πρόσβαση στις κβαντισμένες παραμέτρους κατά τη διάρκεια της συμπερασματολογίας.

Η κβάντιση των βαρών και των bias σε χαμηλότερη ακρίβεια όχι μόνο μείωσε το αποτύπωμα μνήμης αλλά απλοποίησε και τις αριθμητικές πράξεις, καθιστώντας τις λιγότερο απαιτητικές από άποψη πόρων υλικού. Αυτό επέτρεψε να χωρέσουν περισσότεροι νευρώνες στους διαθέσιμους πόρους.

4.3.2 < Παράλληλη επεξεργασία >

Η παράλληλη επεξεργασία είναι μια κρίσιμη τεχνική βελτιστοποίησης για την επιτάχυνση της εξαγωγής συμπερασμάτων νευρωνικών δικτύων σε FPGA. Δεδομένου του εγγενούς παραλληλισμού στα νευρωνικά δίκτυα, όπου πολλαπλοί νευρώνες και επίπεδα μπορούν να υποβληθούν σε επεξεργασία ταυτόχρονα, οι FPGA είναι κατάλληλες για την υλοποίηση τέτοιων αρχιτεκτονικών.

4.3.2.1 < Παραλληλισμός στα επίπεδα MLP >

Στην υλοποίησή μας σε FPGA, εκμεταλλευτήκαμε τον παραλληλισμό σε διάφορα στρώματα και νευρώνες του MLP. Κάθε νευρώνας σε ένα στρώμα μπορεί να επεξεργαστεί ανεξάρτητα, καθώς οι λειτουργίες που εκτελούνται από τους νευρώνες - όπως ο πολλαπλασιασμός των εισόδων με τα βάρη, η άθροισή τους, η προσθήκη bias και η εφαρμογή συναρτήσεων ενεργοποίησης- είναι ανεξάρτητες μεταξύ τους. Με την παράλληλη υλοποίηση αυτών των λειτουργιών, μειώνουμε σημαντικά τον χρόνο που απαιτείται για την εξαγωγή συμπερασμάτων. Για παράδειγμα, στο πρώτο επίπεδο του MLP μας, κάθε νευρώνας λαμβάνει το ίδιο σύνολο εισόδων αλλά εφαρμόζει διαφορετικά βάρη. Παραλληλοποιώντας τις πράξεις πολλαπλασιασμού και συσσώρευσης σε όλους τους νευρώνες αυτού του επιπέδου, επιτυγχάνουμε σημαντική

μείωση της καθυστέρησης. Αυτή η παράλληλη επεξεργασία επεκτείνεται στα επόμενα στρώματα, βελτιώνοντας περαιτέρω τη συνολική απόδοση του συστήματος.

4.3.2.2 < Pipelining >

Η αγωγιμοποίηση υλοποιείται αποτελεσματικά στη σχεδίασή μας μέσω της διαχείρισης της κατάστασης κάθε στρώματος. Ενώ ένα στρώμα εξάγει τα δεδομένα του, το επόμενο στρώμα αρχίζει την επεξεργασία μόλις γίνει διαθέσιμη έγκυρη είσοδος. Αυτή η επικαλυπτόμενη εκτέλεση μειώνει τον χρόνο αδράνειας μεταξύ των στρωμάτων, επιτρέποντας τη συνεχή ροή και ενισχύοντας την απόδοση. Επιπλέον, η επεξεργασία εντός κάθε στρώματος χωρίζεται σε διακριτά στάδια - κάθε νευρώνας εκτελεί τους υπολογισμούς του ανεξάρτητα - επιτρέποντας την ταυτόχρονη εκτέλεση πολλαπλών λειτουργιών. Αυτός ο συνδυασμός διοχέτευσης μεταξύ των στρωμάτων μεγιστοποιεί την αποδοτικότητα της FPGA, η οποία επιτυγχάνεται μέσω του χειρισμού της μηχανής καταστάσεων μεταξύ των στρωμάτων, όπως φαίνεται παρακάτω:

```
always @(posedge s_axi_aclk)
begin
    if(reset)
    begin
        state_1 <= IDLE;
        count_1 <= 0;
        data_out_valid_1 <=0;
    end
    else
    begin
        case(state_1)
            IDLE: begin
                count_1 <= 0;
                data_out_valid_1 <=0;
                if (o1_valid[0] == 1'b1)
                begin
                    holdData_1 <= x1_out;
                    state_1 <= SEND;
                end
            end
            SEND: begin
                out_data_1 <= holdData_1[`dataWidth-1:0];
                holdData_1 <= holdData_1>>`dataWidth;
                count_1 <= count_1 +1;
                data_out_valid_1 <= 1;
                if (count_1 == `numNeuronLayer1)
                begin
                    state_1 <= IDLE;
                    data_out_valid_1 <= 0;
                end
            end
        endcase
    end
end
```

4.3.3 < Βελτιστοποίηση πόρων>

Η αποδοτική αξιοποίηση των πόρων είναι κρίσιμη κατά την υλοποίηση ενός MLP σε FPGA, καθώς οι FPGA διαθέτουν περιορισμένα λογικά μπλοκ, μνήμη και μονάδες DSP.

4.3.3.1 < Ελαχιστοποίηση του πλάτους bit >

Μία από τις πρωταρχικές τεχνικές βελτιστοποίησης πόρων είναι η ελαχιστοποίηση του εύρους bit της αναπαράστασης σταθερού σημείου. Επιλέγοντας προσεκτικά τα ακέραια και κλασματικά πλάτη bit με βάση το δυναμικό εύρος των βαρών και των bias, μειώνουμε τον αριθμό των bit που απαιτούνται για την αναπαράσταση κάθε παραμέτρου. Αυτή η μείωση όχι μόνο εξοικονομεί μνήμη αλλά μειώνει επίσης τη λογική που απαιτείται για τις αριθμητικές πράξεις, εξοικονομώντας έτσι πόρους FPGA.

4.3.3.2 < Διαχείριση μνήμης>

Η αποτελεσματική διαχείριση της μνήμης είναι ζωτικής σημασίας στις υλοποιήσεις νευρωνικών δικτύων που βασίζονται σε FPGA, ιδίως για το χειρισμό κβαντισμένων βαρών και προκαταλήψεων κατά τη διάρκεια της εξαγωγής συμπερασμάτων. Χρησιμοποιήσαμε δύο διαφορετικές προσεγγίσεις για τη διαχείριση των απαιτήσεων μνήμης του πολυεπίπεδου perceptron (MLP).

Προσέγγιση 1: Προσαρμοσμένη υλοποίηση ROM σε HDL

Σε αυτή την προσέγγιση, κωδικοποιήσαμε χειροκίνητα τη μνήμη μόνο για ανάγνωση (ROM) στο πλαίσιο της σχεδίασής μας σε HDL. Αυτή η μέθοδος περιελάμβανε τον ρητό ορισμό των περιεχομένων της ROM, όπου τα κβαντισμένα βάρη και οι προκαταλήψεις ήταν hardcoded. Έτσι κατά την διάρκεια της σύνθεσης το Quartus χρησιμοποιεί m;ono τα Memory Bits του FPGA που του έχουμε ορίσει. Με τον τρόπο αυτό, εξασφαλίσαμε ότι η χρήση της μνήμης βελτιστοποιήθηκε ειδικά για το μοντέλο MLP μας, αποφεύγοντας περιττές επιβαρύνσεις. Αυτό το επίπεδο ελέγχου επιτρέπει βελτιωμένες επιδόσεις όσον αφορά την καθυστέρηση και τη χρήση των πόρων.

Τα πλεονεκτήματα αυτής της μεθόδου περιλαμβάνουν:

- Προσαρμογή: Παρέχει ακριβή έλεγχο της διάταξης της μνήμης, βελτιστοποιώντας την απόδοση για το συγκεκριμένο μοντέλο νευρωνικού δικτύου.
- Αποδοτικότητα πόρων: Προσαρμόζοντας τη ROM ώστε να περιλαμβάνει μόνο τα απαραίτητα βάρη και τις προκαταλήψεις, ελαχιστοποιούμε τη χρήση των πόρων της FPGA.

Ωστόσο, αυτή η προσέγγιση μπορεί να γίνει πολύπλοκη και χρονοβόρα, ιδίως με μεγαλύτερα μοντέλα.

Προσέγγιση 2: Αξιοποίηση ενσωματωμένων μπλοκ μνήμης FPGA

Η δεύτερη προσέγγισή μας αξιοποίησε τα ενσωματωμένα μπλοκ μνήμης που παρέχονται από το υλικό του FPGA, όπως η Block RAM (BRAM). Αυτή η μέθοδος επιτρέπει τη φόρτωση κβαντισμένων παραμέτρων σε αυτά τα μπλοκ μνήμης κατά τη διάρκεια της διαμόρφωσης του FPGA, απλοποιώντας τη διαδικασία σχεδίασης και εξαλείφοντας την ανάγκη για χειροκίνητη κωδικοποίηση ROM.

Ενώ αυτή η προσέγγιση απλοποιεί τη σχεδίαση και διευκολύνει τις ευκολότερες ενημερώσεις μοντέλων, μπορεί επίσης να εισάγει επιβάρυνση. Η χρήση των megafunctions για ενσωματωμένη μνήμη μπορεί να οδηγήσει σε μη αποδοτική χρήση πόρων, καταναλώνοντας ενδεχομένως περισσότερα λογικά στοιχεία από μια προσαρμοσμένη υλοποίηση HDL. Αυτό μπορεί να επηρεάσει τη συνολική χωρητικότητα του νευρωνικού δικτύου που μπορεί να υλοποιηθεί στο FPGA, ιδίως σε περιβάλλοντα με περιορισμένους πόρους, όπως η σειρά Cyclone.

Τα πλεονεκτήματα αυτής της μεθόδου περιλαμβάνουν:

- Απλότητα: Μειώνει την ανάγκη για χειροκίνητη κωδικοποίηση και επιτρέπει ευκολότερες ενημερώσεις του μοντέλου.

- Ευελιξία: Η ενσωματωμένη μνήμη μπορεί εύκολα να αναδιαμορφωθεί, προσαρμόζοντας διαφορετικά μοντέλα ή περιπτώσεις χρήσης.

Τα μειονεκτήματα αυτής της μεθόδου:

- Αυξημένη χωρητικότητα: Με την εκφόρτωση των βαρών και των προκαταλήψεων στην ενσωματωμένη μνήμη, περισσότερα λογικά στοιχεία καθίστανται διαθέσιμα για την υλοποίηση πρόσθετων νευρώνων ή σύνθετης λογικής.

Σύγκριση και εκτιμήσεις

Συνοψίζοντας, ενώ και οι δύο προσεγγίσεις έχουν τα πλεονεκτήματά τους, μια προσαρμοσμένη υλοποίηση HDL προσφέρει καλύτερη βελτιστοποίηση για τις συγκεκριμένες ανάγκες μας. Με τη σκληρή κωδικοποίηση των απαραίτητων παραμέτρων, διατηρούμε αυστηρότερο έλεγχο της χρήσης των πόρων και της απόδοσης, γεγονός ιδιαίτερα κρίσιμο σε σχέδια FPGA. Αυτή η μέθοδος μας επιτρέπει επίσης να μετριάσουμε τα πιθανά μειονεκτήματα που προκύπτουν από τη στήριξη σε προκαθορισμένες megafunctions ή ενσωματωμένη μνήμη, διασφαλίζοντας ότι μεγιστοποιούμε τις δυνατότητες της υλοποίησης της FPGA μας.

4.3.4 < Συναρτήσεις ενεργοποίησης: Hardmax vs. Softmax>

Η επιλογή της συνάρτησης ενεργοποίησης είναι κρίσιμη στο σχεδιασμό νευρωνικών δικτύων, ιδίως για το επίπεδο εξόδου. Ενώ η Softmax είναι μια δημοφιλής επιλογή για προβλήματα ταξινόμησης πολλαπλών κλάσεων, η υλοποίησή της σε FPGA παρουσιάζει σημαντικές προκλήσεις.

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Εικόνα 4.2.4 Ορισμός Softmax

Η Softmax απαιτεί τον υπολογισμό εκθετικών και διαιρέσεων, οι οποίες είναι υπολογιστικά δαπανηρές πράξεις. Η αποτελεσματική υλοποίηση αυτών των πράξεων σε ένα FPGA, ιδίως σε αριθμητική σταθερού σημείου, είναι δύσκολη και απαιτεί πολλούς πόρους. Η ανάγκη υπολογισμού εκθετικών για κάθε κλάση εξόδου και στη συνέχεια κανονικοποίησης αυτών των τιμών εισάγει καθυστέρηση και αυξάνει την

πολυπλοκότητα της σχεδίασης. Δεδομένων των προκλήσεων που σχετίζονται με την Softmax, επιλέξαμε την Hardmax ως συνάρτηση ενεργοποίησης στο στρώμα εξόδου. Η Hardmax απλοποιεί τη διαδικασία λήψης αποφάσεων επιλέγοντας τον νευρώνα με τη μέγιστη τιμή ενεργοποίησης, παρακάμπτοντας την ανάγκη για πολύπλοκες μαθηματικές πράξεις. Στην υλοποίησή μας σε FPGA, το Hardmax λειτουργεί συγκρίνοντας διαδοχικά τις ενεργοποιήσεις εξόδου των νευρώνων, αποθηκεύοντας τη μέγιστη τιμή και τον αντίστοιχο δείκτη. Αυτή η προσέγγιση μειώνει σημαντικά την υπολογιστική επιβάρυνση και τη χρήση πόρων, καθιστώντας την πιο κατάλληλη για ανάπτυξη σε FPGA.

4.4 Εφαρμογή στο Cyclone II

Η υλοποίηση του Βαθύ Νευρωνικού Δικτύου (DNN) στο Cyclone II FPGA ήταν μια λεπτομερής διαδικασία που χωρίσαμε προσεκτικά σε φάσεις προσομοίωσης και ανάπτυξης υλικού. Χρησιμοποιήσαμε έναν συνδυασμό εργαλείων, συμπεριλαμβανομένων των Quartus II, ModelSim και Verilog, για να διασφαλίσουμε ότι ο σχεδιασμός μας δοκιμάστηκε διεξοδικά και βελτιστοποιήθηκε για το Cyclone II EP2C35F672C6 στην πλακέτα DE2.

4.4.1 < Ρύθμιση του περιβάλλοντος ανάπτυξης >

Ξεκινήσαμε με τη ρύθμιση του περιβάλλοντος ανάπτυξης για την υποστήριξη τόσο της προσομοίωσης όσο και της φάσης υλοποίησης υλικού, με έμφαση στη συμβατότητα και τη βελτιστοποίηση για το Cyclone II FPGA.

- **Quartus II 13.0sp1**: Επιλέξαμε το Quartus II 13.0sp1, καθώς είναι η τελευταία έκδοση που υποστηρίζει το Cyclone II EP2C35F672C6 FPGA.Χρησιμοποιήσαμε το Quartus II για ολόκληρη τη διαδικασία σχεδίασης HDL,συμπεριλαμβανομένης της εισαγωγής σχεδίασης, της σύνθεσης και της μεταγλώττισης. Η επιλογή μας εξασφάλισε την ομαλή υλοποίηση χωρίς να αντιμετωπίσουμε προβλήματα που σχετίζονται με την έκδοση.

- **ModelSim**: Χρησιμοποιήσαμε το ModelSim για την προσομοίωση της σχεδίασής μας σε HDL, επιτρέποντάς μας να αναλύσουμε λεπτομερώς τη λειτουργία του

DNN πριν το αναπτύξουμε σε πραγματικό υλικό. Οι προσομοιώσεις ήταν ζωτικής σημασίας για την επαλήθευση της ορθότητας του σχεδιασμού μας, επιτρέποντάς μας να εντοπίσουμε και να επιλύσουμε πιθανά προβλήματα σε πρώιμο στάδιο της διαδικασίας.

- **Python script για τη δημιουργία δεδομένων δοκιμών**: Για τη δημιουργία δεδομένων δοκιμής τόσο για την προσομοίωση όσο και για τη δοκιμή υλικού, γράψαμε ένα προσαρμοσμένο σενάριο Python. Αυτό το σενάριο δημιούργησε εισόδους δοκιμών σε μορφή συμβατή με τον κώδικα Verilog μας, ο οποίος χρησιμοποιεί αναπαράσταση συμπληρώματος δύο για αριθμητική σταθερού σημείου. Αυτοματοποιώντας τη μετατροπή των τιμών εισόδου στην απαιτούμενη δυαδική μορφή, εξασφαλίσαμε ότι τα δεδομένα δοκιμής μας θα αντανακλούσαν με ακρίβεια τον τρόπο με τον οποίο το DNN θα επεξεργαζόταν τις εισόδους κατά την εκτέλεση υλικού. Το σενάριο παρήγαγε ένα ευρύ φάσμα εισόδων, επιτρέποντάς μας να ελέγξουμε διεξοδικά την απόδοση του DNN, ιδίως όσον αφορά τον χειρισμό ακραίων περιπτώσεων και ακραίων τιμών.

- **Python και TensorFlow για προεπεξεργασία**: Χρησιμοποιήσαμε επίσης την Python και το TensorFlow για την εκπαίδευση του DNN μας και την εξαγωγή των βαρών και των bias. Μετά την εκπαίδευση, μετατρέψαμε αυτές τις παραμέτρους σε αναπαράσταση σταθερού σημείου κατάλληλη για υλοποίηση σε FPGA, διασφαλίζοντας ότι θα μπορούσαμε να χρησιμοποιήσουμε αποτελεσματικά τους περιορισμένους πόρους που είναι διαθέσιμοι στην FPGA.

Με την προσεκτική επιλογή και ρύθμιση αυτών των εργαλείων, δημιουργήσαμε μια ισχυρή βάση για την κωδικοποίηση HDL, τη σύνθεση και την ανάπτυξη υλικού. Αυτή η προσέγγιση μας επέτρεψε να βελτιστοποιήσουμε την υλοποίησή μας και να διασφαλίσουμε ότι κάθε βήμα υποστηριζόταν από τα κατάλληλα εργαλεία και μεθόδους για την καλύτερη δυνατή απόδοση στην Cyclone II FPGA.

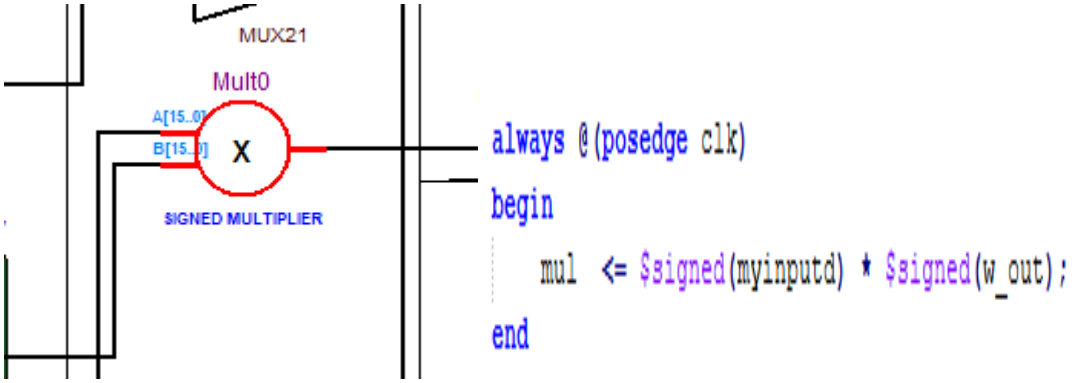
4.4.2 < Κωδικοποίηση και σύνθεση HDL >

Σε αυτό το στάδιο της υλοποίησής μας, σχεδιάσαμε και αναπτύξαμε σχολαστικά τον κώδικα HDL για να αναπαραστήσουμε το βαθύ νευρωνικό δίκτυο (DNN) στο Cyclone II FPGA. Ακολουθήσαμε μια δομημένη προσέγγιση, ξεκινώντας με τα βασικά στοιχεία και σταδιακά φτάνοντας στην πλήρη αρχιτεκτονική του δικτύου, ενσωματώνοντας τελικά τα πάντα σε μια ενότητα που ονομάσαμε cyMLP.

4.4.2.1 < Μονάδα νευρώνα>

Ξεκινήσαμε με την ανάπτυξη της μονάδας νευρώνων, του θεμελιώδους δομικού στοιχείου του DNN μας. Αυτή η ενότητα σχεδιάστηκε για να χειρίζεται όλες τις βασικές λειτουργίες του νευρώνα:

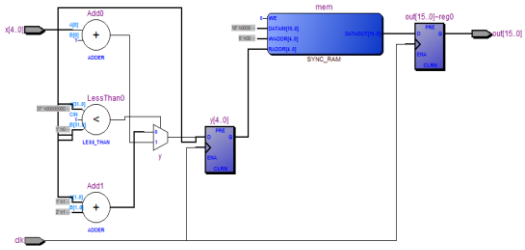
Υπολογισμός αθροίσματος: Η μονάδα νευρώνων λαμβάνει εισόδους, τις πολλαπλασιάζει με τα αντίστοιχα βάρη, αθροίζει τα αποτελέσματα και στο τέλος προσθέτει και το bias . Το υλοποιήσαμε αυτό χρησιμοποιώντας αριθμητική σταθερού σημείου, εξασφαλίζοντας τη συμβατότητα με την αναπαράσταση συμπληρώματος του δύο που χρησιμοποιήθηκε κατά την εκπαίδευση. Αυτή η διαδικασία υλοποιείται αποτελεσματικά στον κώδικα Verilog ως εξής



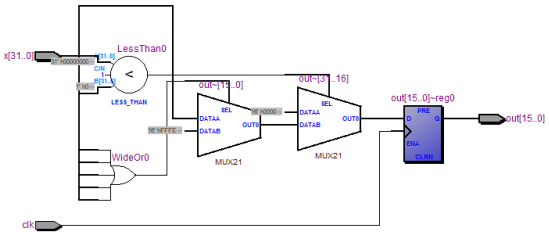
Χρησιμοποιώντας τους ενσωματωμένους πολλαπλασιαστές στο Cyclone II FPGA, εξασφαλίζουμε ότι αυτοί οι πολλαπλασιασμοί εκτελούνται γρήγορα και αποτελεσματικά, βελτιώνοντας έτσι τη συνολική απόδοση των λειτουργιών του νευρώνα.

Συναρτήσεις ενεργοποίησης: Κωδικοποιήσαμε τόσο τη σιγμοειδή όσο και τη συναρτήσεις ενεργοποίησης ReLU εντός της μονάδας νευρώνων. Αυτές οι συναρτήσεις καλούνται μετά τη λειτουργία σταθμισμένου αθροίσματος, επιτρέποντας ευελιξία στη

δοκιμή διαφορετικών συναρτήσεων ενεργοποίησης κατά τη διάρκεια της σύνθεσης και της δοκιμής.



Sigmoid RTL



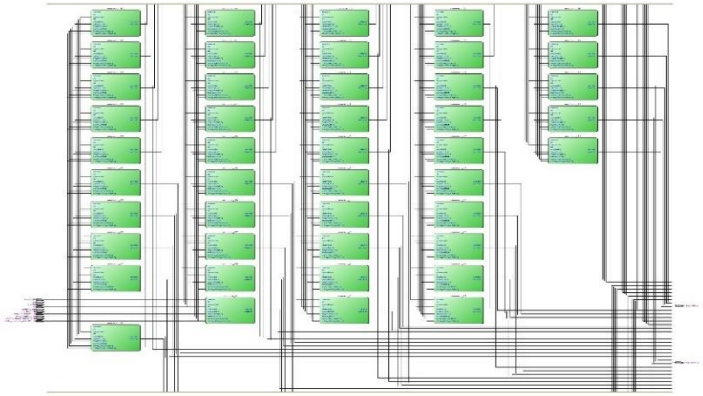
RELU RTL

Ενσωμάτωση μνήμης: Αρχικά, τα βάρη και τα bias ήταν hardcoded στον κώδικα HDL, ο οποίος κατανάλωνε σημαντικά λογικά στοιχεία (LEs). Για να βελτιστοποιήσουμε τη χρήση των πόρων, μεταβήκαμε στη χρήση της ενσωματωμένης μνήμης της FPGA, απελευθερώνοντας σημαντικά LEs και επιτρέποντάς μας να υλοποιήσουμε μεγαλύτερο αριθμό νευρώνων εντός της FPGA.

4.4.2.2 < Μονάδες επιπέδων>

Αφού ολοκληρώσαμε την ενότητα νευρώνων, προχωρήσαμε στις ενότητες επιπέδων, όπου ενσωματώσαμε τον παραλληλισμό για να αυξήσουμε την αποδοτικότητα της επεξεργασίας:

Παράλληλη ενσωμάτωση νευρώνων: Σε κάθε ενότητα στρώματος, υλόποιήσαμε ταυτόχρονα πολλαπλές ενότητες νευρώνων. Αυτή η παράλληλη προσέγγιση μας επέτρεψε να επεξεργαστούμε πολλούς νευρώνες ταυτόχρονα, επιταχύνοντας σημαντικά τη διαδικασία υπολογισμού. Για παράδειγμα, στο πρώτο στρώμα του DNN μας, υλοποιήσαμε παράλληλα 45 μονάδες νευρώνων, καθεμία από τις οποίες επεξεργάζεται ανεξάρτητα τα ίδια δεδομένα εισόδου. Αυτός ο παραλληλισμός ήταν ζωτικής σημασίας για την εκπλήρωση των απαιτήσεων χρονισμού και τη μεγιστοποίηση της απόδοσης του δικτύου.

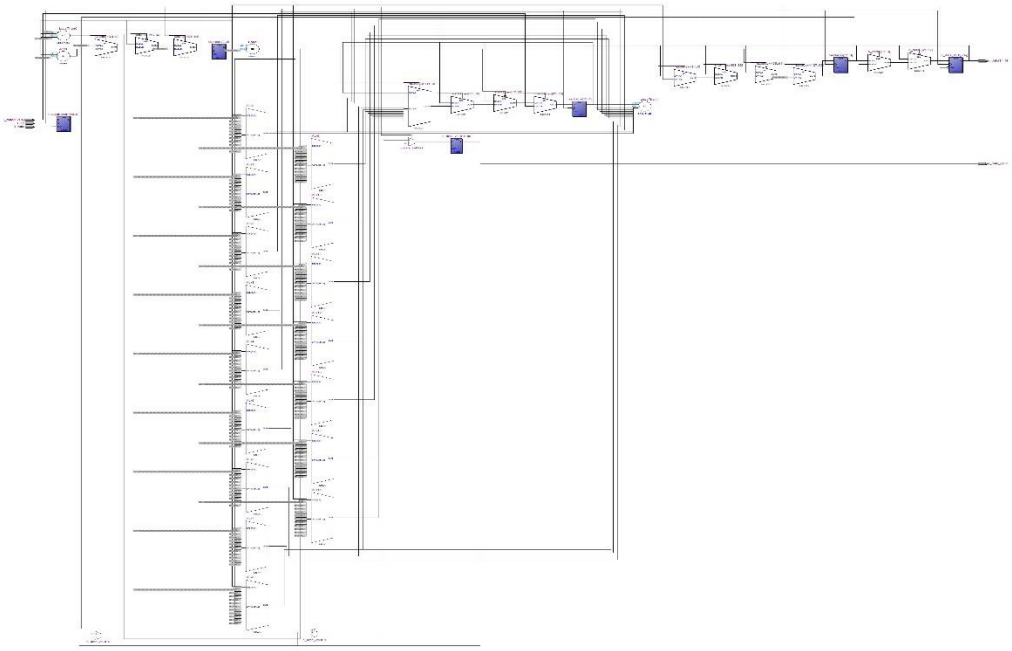


4.4.2.3 < Μονάδα Hardmax>

Στη συνέχεια υλοποιήσαμε τη μονάδα hardmax, η οποία χρησιμεύει ως το τελικό συστατικό λήψης αποφάσεων του DNN:

Επιλογή μέγιστης τιμής: Η μονάδα hardmax συγκρίνει τις εξόδους από τους νευρώνες του τελευταίου στρώματος για να προσδιορίσει τη μέγιστη τιμή, που αντιστοιχεί στην πρόβλεψη του δικτύου.

Παραγωγή εξόδου: Η μονάδα παράγει ένα σήμα εξόδου που αντιπροσωπεύει τον δείκτη του νευρώνα με τη μέγιστη ενεργοποίηση, καθορίζοντας την προβλεπόμενη κλάση.



4.4.2.4 <Μονάδα cyMlp>

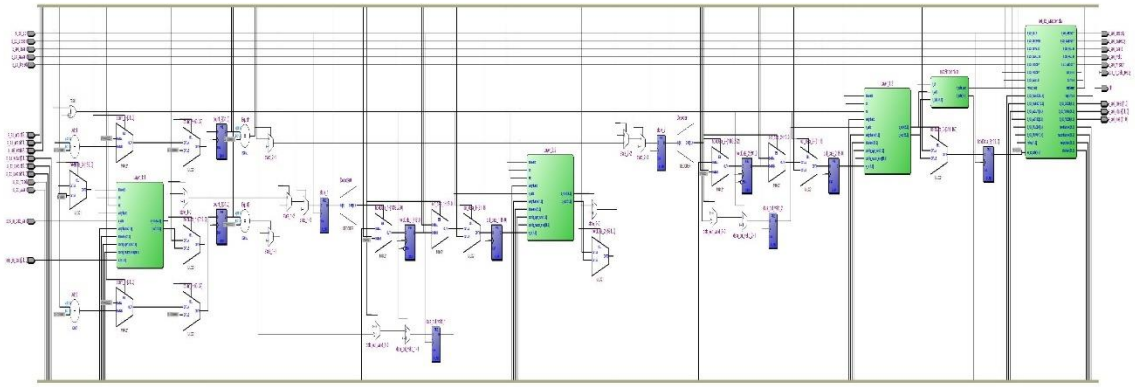
Τέλος, αναπτύξαμε την ενότητα cyMLP για την ενσωμάτωση όλων των στοιχείων του DNN μας:

Ένωση επιπέδων: Η ενότητα cyMLP συνδέει κάθε στρώμα, διασφαλίζοντας ότι οι έξοδοι από ένα στρώμα χρησιμεύουν ως είσοδοι στο επόμενο. Αυτή η ενότητα είναι κεντρική για τη διατήρηση της συνολικής δομής του DNN, διευκολύνοντας την απρόσκοπτη ροή δεδομένων από το στρώμα εισόδου στην έξοδο.

Λογική ελέγχου: Υλοποιήσαμε λογική ελέγχου στο πλαίσιο του cyMLP για τη διαχείριση του χρονισμού και του συγχρονισμού των λειτουργιών σε όλο το δίκτυο. Αυτό εξασφάλιζε ότι τα δεδομένα επεξεργάζονταν σωστά και οι έξοδοι παράγονταν εντός των επιθυμητών χρονικών περιορισμών.

4.4.2.5 < Σύνθεση του Δικτύου>

Μετά την οριστικοποίηση της κωδικοποίησης HDL για το DNN, προχωρήσαμε στη σύνθεση χρησιμοποιώντας το Quartus II. Η διαδικασία σύνθεσης ήταν ζωτικής σημασίας για τη δημιουργία του σχεδιασμού υλικού που αντικατοπτρίζει την αρχιτεκτονική και τη δομή του νευρωνικού μας δικτύου. Ο τελικός συνθετικός σχεδιασμός, που παρουσιάζεται στην παρακάτω προβολή RTL, αντιπροσωπεύει τη βασική διάταξη του DNN μας όπως υλοποιήθηκε στο Cyclone II FPGA.



Αυτή η σχεδίαση αποτελείται από πολλαπλά στρώματα που συνδέονται παράλληλα μεταξύ τους για να σχηματίσουν την πλήρη αρχιτεκτονική του πολυεπίπεδου perceptron (MLP). Κάθε μπλοκ αντιπροσωπεύει τις βασικές λειτουργίες που είναι απαραίτητες για την επεξεργασία των εισόδων και την παραγωγή εξόδων με βάση τα μαθημένα βάρη.

4.4.2.6 < Εξερεύνηση διαμορφώσεων DNN >

Αφού δημιουργήθηκε αυτή η βασική αρχιτεκτονική, η επόμενη φάση περιελάμβανε τη διερεύνηση παραλλαγών στη διαμόρφωση του DNN. Οι παραλλαγές αυτές αποσκοπούσαν στην εξεύρεση της βέλτιστης ισορροπίας μεταξύ του αριθμού των νευρώνων, της χρήσης των πόρων και της απόδοσης. Πειραματιστήκαμε με διαφορετικά βάθη δικτύου (όπως η προσθήκη ενός πρόσθετου κρυμμένου επιπέδου) και διαμορφώσεις νευρώνων για να αξιολογήσουμε τον αντίκτυπό τους τόσο στην ακρίβεια όσο και στην αποδοτικότητα του υλικού.

Το επόμενο κεφάλαιο θα εμβαθύνει σε αυτές τις διαφορετικές διαμορφώσεις, ξεκινώντας από τις ρυθμίσεις προσομοίωσης και ακολουθούμενο από τα αντίστοιχα πειραματικά αποτελέσματα για να εντοπίσουμε την πιο αποτελεσματική δομή DNN για την υλοποίησή μας σε FPGA.

Κεφάλαιο 5. < Πειραματικά αποτελέσματα >

Αυτό το κεφάλαιο παρέχει μια ολοκληρωμένη επισκόπηση της πειραματικής διάταξης, της επαλήθευσης, της χρήσης των πόρων, των μετρικών απόδοσης, της συγκριτικής ανάλυσης και της συζήτησης των αποτελεσμάτων που προέκυψαν από την υλοποίηση του νευρωνικού δικτύου στο Cyclone II FPGA.

5.1 Φάση Προσομοίωσης

Πριν αναπτύξουμε την αρχιτεκτονική MLP στο FPGA, πραγματοποιήσαμε μια σε βάθος προσομοίωση για να επαληθεύσουμε την ορθότητά της και να διασφαλίσουμε ότι το δίκτυο συμπεριφερόταν όπως αναμενόταν σε διάφορες συνθήκες εισόδου. Αυτή η φάση προσομοίωσης ήταν ζωτικής σημασίας για την αποφυγή πιθανών προβλημάτων κατά την υλοποίηση υλικού και για τη λεπτομερή ρύθμιση των παραμέτρων του δικτύου. Για την προσομοίωση, χρησιμοποιήσαμε το ModelSim, ένα ισχυρό εργαλείο για την επαλήθευση κώδικα Verilog. Αναπτύξαμε ένα εκτεταμένο testbench για την προσομοίωση της συμπεριφοράς του βαθύ νευρωνικού δικτύου (DNN) σε ένα ελεγχόμενο περιβάλλον, το οποίο μας επέτρεψε να εισάγουμε διάφορα διανύσματα

εισόδου και να παρατηρήσουμε πώς το δίκτυο επεξεργάστηκε αυτές τις εισόδους. Αυτό εξασφάλισε ότι οι έξοδοι ανταποκρίνονταν στις προσδοκίες μας σε ένα ευρύ φάσμα περιπτώσεων δοκιμής. Το περιβάλλον προσομοίωσης παρείχε την ευελιξία να αξιολογήσουμε διαφορετικές διαμορφώσεις του δικτύου, μεταβάλλοντας τον αριθμό των νευρώνων και των επιπέδων. Κατά τη διάρκεια αυτής της φάσης, αναλύσαμε επίσης την επίδραση της αλλαγής των συναρτήσεων ενεργοποίησης, όπως η ReLU και η Sigmoid, και πώς επηρέασαν τη συμπεριφορά του δικτύου. Επιλέξαμε να προσομοιώσουμε πολλαπλές διαμορφώσεις DNN, τόσο από άποψη αρχιτεκτονικής (αριθμός νευρώνων και επιπέδων) όσο και εσωτερικών παραμέτρων (ακρίβεια σταθερού σημείου, βάρη, bias). Αυτές οι προσομοιώσεις παρείχαν πολύτιμες πληροφορίες σχετικά με τους συμβιβασμούς μεταξύ της χρήσης των πόρων και της ακρίβειας του δικτύου. Για παράδειγμα, η αύξηση του αριθμού των νευρώνων και των στρωμάτων βελτίωσε την ακρίβεια, αλλά απαιτούσε επίσης περισσότερους πόρους FPGA, κάτι που ήταν ένας περιορισμός που έπρεπε να εξισορροπήσουμε προσεκτικά. Το λογισμικό Quartus II χρησιμοποιήθηκε για τη σύνθεση του σχεδιασμού πριν από την προσομοίωση. Μετά από κάθε επανάληψη, εξετάζαμε τη χρήση των πόρων για να διασφαλίσουμε ότι ο σχεδιασμός μας ταίριαζε στους περιορισμούς της Cyclone II FPGA, ιδίως όσον αφορά τα λογικά στοιχεία, τους ενσωματωμένους πολλαπλασιαστές και τα μπλοκ μνήμης. Αυτή η επαναληπτική διαδικασία προσομοίωσης και βελτιστοποίησης μας επέτρεψε να εντοπίσουμε νωρίς πιθανά σημεία συμφόρησης και να κάνουμε προσαρμογές πριν προχωρήσουμε στη φάση του υλικού. Αυτή η φάση ήταν κρίσιμη για την επικύρωση των σχεδιαστικών μας επιλογών και την προετοιμασία μας για την υλοποίηση υλικού.

Με την επιτυχή επικύρωση του DNN μέσω προσομοίωσης, το επόμενο βήμα περιελάμβανε την αντιμετώπιση των περιορισμών των πόρων της FPGA για να διασφαλιστεί ότι ο σχεδιασμός θα μπορούσε να υποστηρίξει την επιθυμητή αρχιτεκτονική δικτύου. Στην επόμενη υποενότητα θα συζητήσουμε τις διάφορες τεχνικές βελτιστοποίησης που εφαρμόστηκαν για τη μεγιστοποίηση του αριθμού των νευρώνων με παράλληλη ελαχιστοποίηση της χρήσης των πόρων. Αυτή η προσπάθεια περιελάμβανε προσεκτική διαχείριση των λογικών στοιχείων, των μπλοκ μνήμης και άλλων κρίσιμων πόρων, διασφαλίζοντας ότι η FPGA θα μπορούσε να διαχειριστεί την πολυπλοκότητα του DNN. Αυτές οι βελτιστοποιήσεις έθεσαν τα θεμέλια για τη διερεύνηση διαφορετικών διαμορφώσεων σε μεταγενέστερα στάδια, επιτρέποντάς μας να διευρύνουμε τα όρια των δυνατοτήτων του υλικού.

5.2 Αξιοποίηση πόρων και βελτιστοποίηση

Η μεγιστοποίηση του αριθμού των νευρώνων με ταυτόχρονη αποτελεσματική χρήση των διαθέσιμων πόρων της FPGA ήταν μια βασική πρόκληση κατά την υλοποίηση του DNN στην Cyclone FPGA. Τα περιορισμένα λογικά στοιχεία (LEs), τα μπλοκ μνήμης και οι ενσωματωμένοι πολλαπλασιαστές μάς επέβαλαν να βελτιστοποιήσουμε κάθε μέρος της σχεδίασης. Στην ενότητα που ακολουθεί περιγράφονται λεπτομερώς οι διάφορες τεχνικές βελτιστοποίησης που χρησιμοποιήθηκαν, οι οποίες μας επέτρεψαν να διευρύνουμε τα όρια του τι μπορούσε να υποστηρίξει η FPGA, διατηρώντας παράλληλα την απαιτούμενη ακρίβεια και λειτουργικότητα.

Αναπαράσταση σταθερού σημείου

Μία από τις σημαντικότερες βελτιστοποιήσεις ήταν η μετάβαση από την αριθμητική κινητής υποδιαστολής στην αριθμητική σταθερής υποδιαστολής. Ενώ οι πράξεις κινητής υποδιαστολής προσφέρουν μεγαλύτερη ακρίβεια, απαιτούν σημαντικά περισσότερους πόρους σε μια FPGA, ιδίως όσον αφορά τα λογικά στοιχεία και τους πολλαπλασιαστές. Με τη μετάβαση στην αναπαράσταση σταθερής υποδιαστολής, μπορέσαμε να μειώσουμε τις απαιτήσεις πόρων για τις μαθηματικές πράξεις χωρίς να θυσιάσουμε ουσιαστικά την ακρίβεια του DNN. Η επιλογή του πλάτους δεδομένων είναι κρίσιμη. Σε μια πειραματική δοκιμή, χρησιμοποιήσαμε μια διαμόρφωση 16-10-10 για το νευρωνικό δίκτυο. Ξεκινήσαμε με πλάτος δεδομένων 8 bit. Ενώ αυτή η διαμόρφωση οδήγησε σε χαμηλότερη χρήση λογικών πόρων, συγκεκριμένα 4621 /33216 λογικά στοιχεία, εισήγαγε επίσης σημαντικές προκλήσεις για την ακρίβεια του DNN.

Total logic elements	4,621 / 33,216 (14 %)
Total combinational functions	4,479 / 33,216 (13 %)
Dedicated logic registers	2,396 / 33,216 (7 %)

Καθώς προχωρούσαμε, αυξήσαμε το πλάτος δεδομένων σε 16 bit, παρατηρώντας ότι αυτή η αλλαγή δεν οδήγησε σε σημαντική αύξηση της κατανάλωσης λογικών πόρων, παρά μόνο 5%, αλλά βελτίωσε και την ακρίβεια.

Total logic elements	6,279 / 33,216 (19 %)
Total combinational functions	5,519 / 33,216 (17 %)
Dedicated logic registers	2,993 / 33,216 (9 %)

Συνεχίζοντας την εξερεύνησή μας, δοκιμάσαμε ένα πλάτος δεδομένων 24 bit. Παρόλο που αυτό το πλάτος παρείχε καλύτερη ακρίβεια λόγω μεγαλύτερου κλασματικού μέρους των δεδομένων μας, διαπιστώσαμε ότι αύξησε εκθετικά τις απαιτήσεις λογικών πόρων, καταναλώνοντας πάνω από το ήμισυ των διαθέσιμων πόρων της Cyclone FPGA.

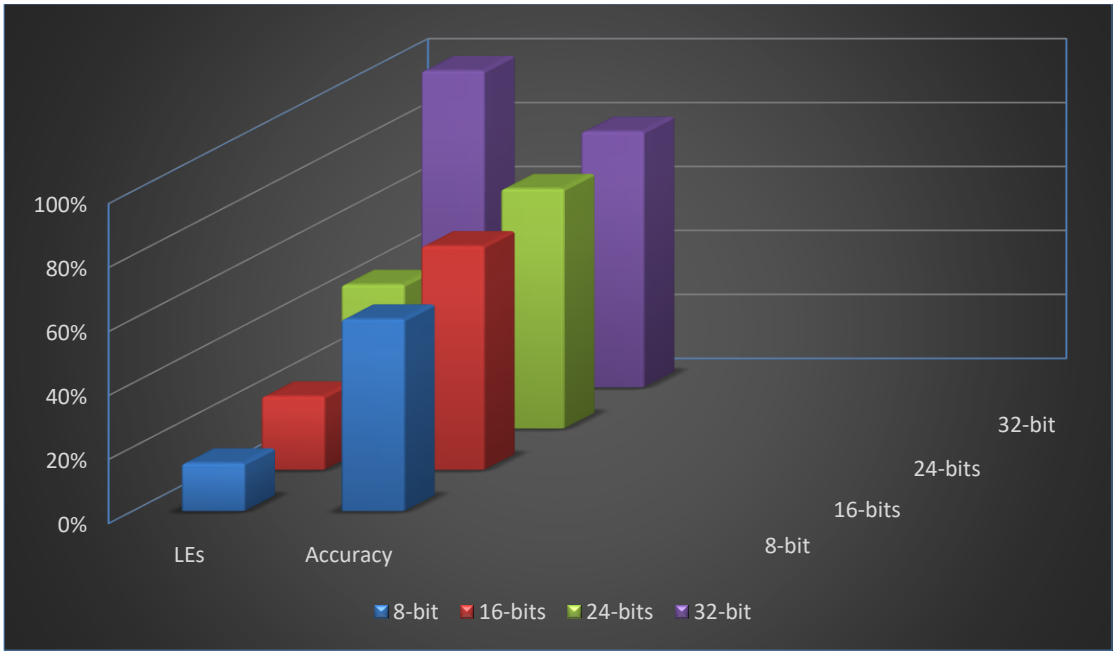
Total logic elements	14,952 / 33,216 (45 %)
Total combinational functions	14,439 / 33,216 (43 %)
Dedicated logic registers	4,840 / 33,216 (15 %)

Τέλος, πειραματιστήκαμε με πλάτος 32-bit, το οποίο θεωρητικά παρείχε το υψηλότερο επίπεδο ακρίβειας. Ωστόσο, οι απαιτήσεις πόρων αυτής της διαμόρφωσης αποδείχθηκαν μη πρακτικές, με 33050/33216 λογικά στοιχεία να χρησιμοποιούνται μόνο για τη μικρή διαμόρφωση.

Total logic elements	33,050 / 33,216 (100 %)
Total combinational functions	33,015 / 33,216 (99 %)
Dedicated logic registers	8,006 / 33,216 (24 %)

Η εξέλιξη αυτή ανέδειξε σαφώς τη λεπτή ισορροπία μεταξύ του εύρους των δεδομένων και της χρήσης των λογικών πόρων.

Τα λεπτομερή ευρήματα από τα πειράματά μας παρουσιάζονται στο παρακάτω διάγραμμα, παρουσιάζοντας τον τρόπο με τον οποίο η κατανάλωση πόρων μεταβάλλεται με κάθε αύξηση του πλάτους δεδομένων στη διαμόρφωση του DNN μας.



Προ-εκπαίδευση στην Python

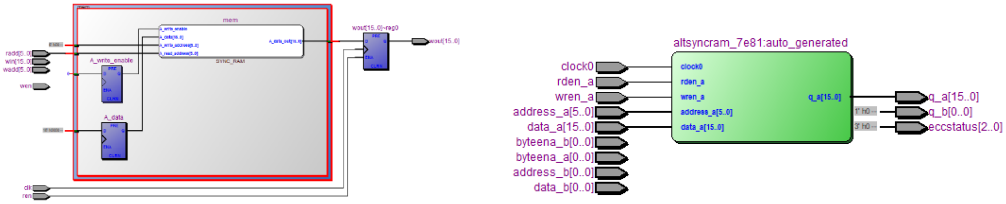
Μια άλλη σημαντική βελτιστοποίηση ήταν η απόφαση να προ-εκπαιδευτεί το νευρωνικό δίκτυο χρησιμοποιώντας Python πριν από τη μεταφορά του στο FPGA. Χρησιμοποιώντας ένα περιβάλλον λογισμικού για την εκπαίδευση, εξαλείψαμε την ανάγκη να εκτελέσουμε την αντίστροφη διάδοση και άλλες λειτουργίες εκπαίδευσης απευθείας στην FPGA, οι οποίες θα κατανάλωναν τεράστιο ποσό πόρων. Αντ' αυτού,

εστιάσαμε την υλοποίηση της FPGA αποκλειστικά στην εξαγωγή συμπερασμάτων, όπου το DNN χρησιμοποιείται για να κάνει προβλέψεις με βάση τα προ-εκπαιδευμένα βάρη.

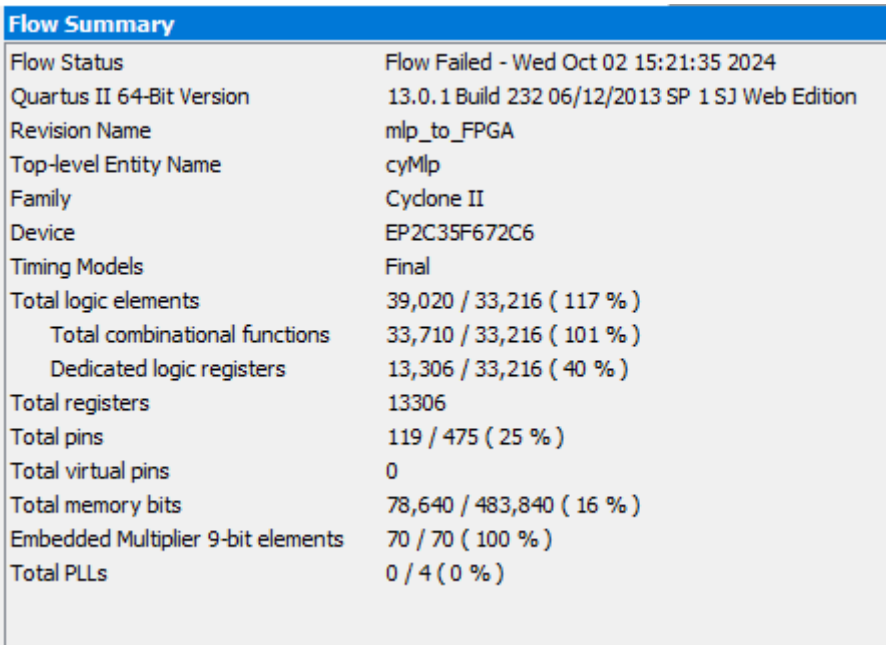
Αυτή η στρατηγική μείωσε σημαντικά τόσο τη μνήμη όσο και την υπολογιστική επιβάρυνση, καθώς μόνο το εμπρόσθιο πέρασμα του δικτύου έπρεπε να υλοποιηθεί σε υλικό. Μεταφέροντας τη φάση εκπαίδευσης στην Python, είχαμε επίσης την ευελιξία να τελειοποιήσουμε τις παραμέτρους του δικτύου πριν από την υλοποίηση του τελικού μοντέλου στο FPGA, διασφαλίζοντας ότι μόνο οι πιο αποδοτικές διαμορφώσεις μεταφέρθηκαν στο υλικό.

Αποδοτική χρήση των μπλοκ μνήμης

Η προσαρμοσμένη λύση που υλοποιήσαμε για τη διαχείριση των βαρών και των προκαταλήψεων στα στρώματα του DNN αποδείχθηκε ανώτερη από τη χρήση των ενσωματωμένων μπλοκ μνήμης που παρέχονται από το FPGA. Χρησιμοποιώντας μια προσαρμοσμένη προσέγγιση διαχείρισης μνήμης, βελτιστοποιήσαμε την κατανομή των πόρων, επιτρέποντάς μας να αποθηκεύουμε αποτελεσματικά και να έχουμε πρόσβαση στα βάρη χωρίς υπερβολική χρήση λογικών στοιχείων (LE).



Όταν προσπαθήσαμε να ενσωματώσουμε την ενσωματωμένη μνήμη στην ίδια διαμόρφωση, διαπιστώσαμε ότι κατανάλωνε περισσότερους πόρους από την προσαρμοσμένη λύση μας, καθιστώντας την ακατάλληλη για τους περιορισμούς της FPGA μας.Συγκεκριμένα, σε διαρρίθμυση που χρησιμοποιούσε το μέγιστο δυνατό του Cyclone τελικά βγήκε από τα όρια του.Αυτό ενίσχυσε τα πλεονεκτήματα της προσαρμοσμένης προσέγγισής μας έναντι της διαμόρφωσης με ενσωματωμένη μνήμη.

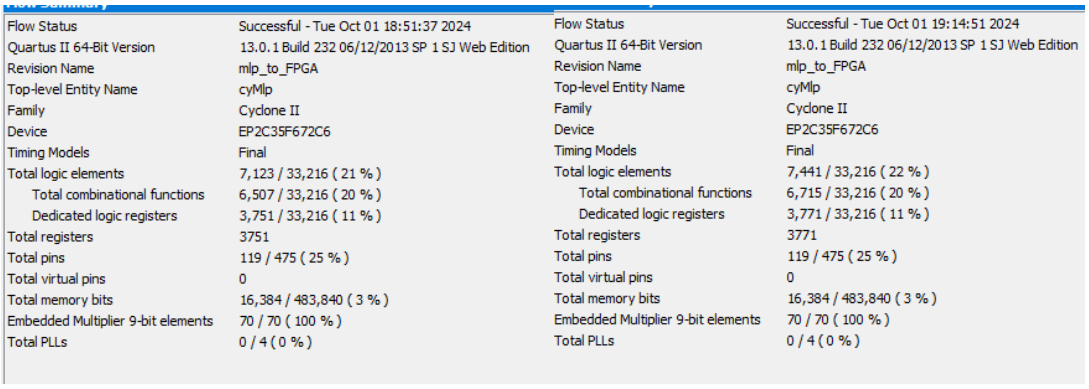


Flow Summary	
Flow Status	Flow Failed - Wed Oct 02 15:21:35 2024
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	mlp_to_FPGA
Top-level Entity Name	cyMlp
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	39,020 / 33,216 (117 %)
Total combinational functions	33,710 / 33,216 (101 %)
Dedicated logic registers	13,306 / 33,216 (40 %)
Total registers	13306
Total pins	119 / 475 (25 %)
Total virtual pins	0
Total memory bits	78,640 / 483,840 (16 %)
Embedded Multiplier 9-bit elements	70 / 70 (100 %)
Total PLLs	0 / 4 (0 %)

Με χρήση έτοιμων megafunctions του λογισμικού

Συναρτήσεις ενεργοποίησης

Μία από τις βελτιστοποιήσεις που εφαρμόσαμε ήταν ο περιορισμός των συναρτήσεων ενεργοποίησης που χρησιμοποιούνται κάθε φορά για την ελαχιστοποίηση της κατανάλωσης πόρων. Αρχικά, τόσο η ReLU όσο και η σιγμοειδής συνάρτηση υπήρχαν στη σχεδίαση FPGA για λόγους ευελιξίας, αλλά η ταυτόχρονη ενεργοποίηση και των δύο κατανάλωνε περιττά λογικά στοιχεία (LE). Για παράδειγμα, σε μια δοκιμή με ένα δίκτυο 16-10-10 νευρώνων, η παρουσία και των δύο συναρτήσεων καταλάμβανε 7.441 από τα 33.216 LE (~22%). Όταν χρησιμοποιήθηκε μόνο μία λειτουργία κάθε φορά, ο σχεδιασμός χρησιμοποίησε 7.123 LEs, μειώνοντας τη χρήση πόρων στο ~21%. Αυτή η προσέγγιση ελαχιστοποίησε τη χρήση πόρων και μας επέτρεψε να πειραματιστούμε με διαφορετικές διαμορφώσεις.



Flow Status	Successful - Tue Oct 01 18:51:37 2024
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	mlp_to_FPGA
Top-level Entity Name	cyMlp
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	7,123 / 33,216 (21 %)
Total combinational functions	6,507 / 33,216 (20 %)
Dedicated logic registers	3,751 / 33,216 (11 %)
Total registers	3751
Total pins	119 / 475 (25 %)
Total virtual pins	0
Total memory bits	16,384 / 483,840 (3 %)
Embedded Multiplier 9-bit elements	70 / 70 (100 %)
Total PLLs	0 / 4 (0 %)

Flow Status	Successful - Tue Oct 01 19:14:51 2024
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	mlp_to_FPGA
Top-level Entity Name	cyMlp
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	7,441 / 33,216 (22 %)
Total combinational functions	6,715 / 33,216 (20 %)
Dedicated logic registers	3,771 / 33,216 (11 %)
Total registers	3771
Total pins	119 / 475 (25 %)
Total virtual pins	0
Total memory bits	16,384 / 483,840 (3 %)
Embedded Multiplier 9-bit elements	70 / 70 (100 %)
Total PLLs	0 / 4 (0 %)

One Activation Function

Both Activation Functions

Επιπλέον, επιλέξαμε να μην χρησιμοποιήσουμε τη συνάρτηση softmax για τις τελικές προβλέψεις. Η softmax, αν και χρησιμοποιείται συνήθως σε εργασίες ταξινόμησης, περιλαμβάνει πολύπλοκες πράξεις όπως ο πολλαπλασιασμός και η διαίρεση, οι οποίες απαιτούν σημαντικούς υπολογιστικούς πόρους. Αντ' αυτού, επιλέξαμε μια προσέγγιση hardmax, η οποία επιλέγει απευθείας τον νευρώνα με την υψηλότερη ενεργοποίηση χωρίς να εκτελεί τη λεπτομερή πιθανολογική κατανομή που παρέχει η softmax. Εξαλείφοντας αυτούς τους ακριβούς υπολογισμούς, μπορέσαμε να εξοικονομήσουμε πόρους και να εξορθολογήσουμε την υλοποίηση για απόδοση σε πραγματικό χρόνο

Ελαχιστοποίηση περιττής λογικής

Κατά τη διάρκεια της διαδικασίας σχεδιασμού, εξετάσαμε προσεκτικά τον κώδικα HDL για να εξαλείψουμε κάθε περιττή λογική που θα μπορούσε να καταναλώσει άσκοπα πόρους. Αυτό περιελάμβανε τον εξορθολογισμό της λογικής ελέγχου, τη μείωση της πολυπλοκότητας των μηχανών κατάστασης και την αφαίρεση τυχόν αχρησιμοποίητων στοιχείων ή ενδιάμεσων σημάτων. Με την απλοποίηση του συνολικού σχεδιασμού, μπορέσαμε να μειώσουμε σημαντικά τον αριθμό των LE που απαιτούνται για το DNN, γεγονός που μας επέτρεψε να διαθέσουμε περισσότερους πόρους για την επέκταση του αριθμού των νευρώνων και των επιπέδων.

Αγωγιμοποίηση και παράλληλη επεξεργασία

Για να μεγιστοποιήσουμε την απόδοση και να βελτιώσουμε τη χρονική απόδοση του DNN, εφαρμόσαμε τεχνικές διοχέτευσης και παράλληλης επεξεργασίας όπου ήταν δυνατόν. Η αγωγιμοποίηση μας επέτρεψε να αναλύσουμε τις μεγάλες λειτουργίες σε μικρότερα, διαχειρίσιμα στάδια, καθένα από τα οποία θα μπορούσε να υποβληθεί σε ταυτόχρονη επεξεργασία.

Αυτό μείωσε τη συνολική κρίσιμη διαδρομή της σχεδίασης και μας επέτρεψε να τηρούμε τους περιορισμούς χρονισμού ακόμη και όταν αυξανόταν η πολυπλοκότητα του δικτύου.

Παρομοίως, η παράλληλη επεξεργασία χρησιμοποιήθηκε μέσα στις μονάδες νευρώνων και επιπέδων για την ταυτόχρονη διεκπεραίωση πολλαπλών υπολογισμών. Εκμεταλλευόμενοι την ικανότητα του FPGA να εκτελεί παράλληλες λειτουργίες,

μπορέσαμε να επιταχύνουμε την εκτέλεση του DNN, μειώνοντας την καθυστέρηση και εξασφαλίζοντας απόδοση σε πραγματικό χρόνο.

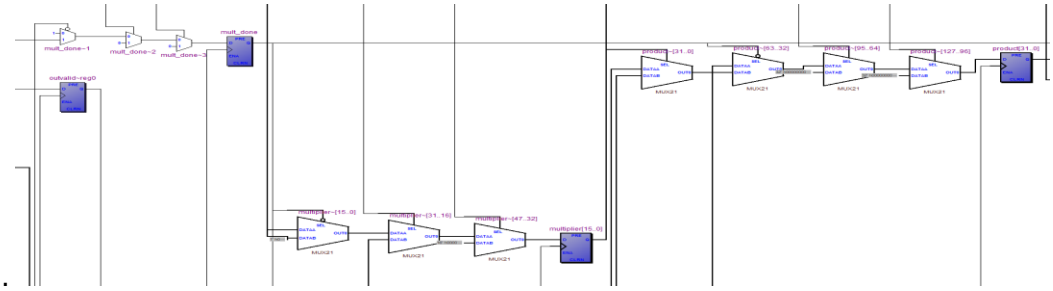
Αφαίρεση αχρησιμοποίητων λογικών και ενεργοποιήσεων

Σε κάθε στάδιο δοκιμών και διαμόρφωσης, επανεξετάσαμε προσεκτικά το σχεδιασμό του FPGA για να αφαιρέσουμε τυχόν αχρησιμοποίητη λογική ή στοιχεία που δεν χρειάζονταν πλέον.

Για παράδειγμα, κατά τη διάρκεια της τελικής υλοποίησης, αφαιρέσαμε ορισμένες λειτουργίες ελέγχου και πρόσθετες λειτουργίες εντοπισμού σφαλμάτων που ήταν χρήσιμες σε προηγούμενα στάδια αλλά δεν ήταν πλέον απαραίτητες. Διατηρώντας μόνο την απαραίτητη λογική για την εξαγωγή συμπερασμάτων, απελευθερώσαμε πρόσθετους πόρους που θα μπορούσαν να χρησιμοποιηθούν για την επέκταση του δικτύου.

Αποδοτικοί πολλαπλασιαστές και αριθμητικές μονάδες

Για να αξιολογήσουμε τον αντίκτυπο της αντικατάστασης των ενσωματωμένων πολλαπλασιαστών, πραγματοποιήσαμε ένα πείραμα όπου αντικαταστήσαμε τον τελεστή άμεσου πολλαπλασιασμού (*) με μια μέθοδο shift-and-add. Αυτή η τεχνική προσομοιώνει τον πολλαπλασιασμό χρησιμοποιώντας μετατοπίσεις bit και πρόσθεση, επιτρέποντας στην FPGA να βασίζεται αποκλειστικά σε LUTs (Look-Up Tables) και καταχωρητές, αποφεύγοντας έτσι τους ενσωματωμένους πολλαπλασιαστές. Αυτή η έκδοση είναι πιο αργή από εκείνη που χρησιμοποιεί πολλαπλασιαστές υλικού, επειδή ο πολλαπλασιασμός εκτελείται τώρα επαναληπτικά, σε κύκλους ρολογιού ίσους με το dataWidth. Παρακάτω η εικόνα απικονίζει το RTL του Quartus και είναι ξεκάθαρο πως τα λογικά στοιχεία που χρησιμοποιούμε για τη την δουλειά εχουν αυξηθεί



Για το πείραμα αυτό, χρησιμοποιήσαμε μια διαμόρφωση DNN με 16-10-10 νευρώνες, εκτελώντας πολλαπλασιασμό σε κάθε νευρώνα με τη μέθοδο shift-and-add. Αυτό το πείραμα παρείχε εικόνα για το πώς αλλάζει η χρήση των πόρων όταν αποφεύγονται οι ενσωματωμένοι πολλαπλασιαστές, καθώς οι LUT και οι καταχωρητές αντικαθιστούν τα πιο αποδοτικά στοιχεία του FPGA. Παρακάτω το Flow Summary του λογισμικού μας δειχνει την αξιοποίηση του Cyclone.

Flow Summary	
Flow Status	Successful - Wed Oct 02 11:48:13 2024
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	mlp_to_FPGA
Top-level Entity Name	cyMlp
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	9,455 / 33,216 (28 %)
Total combinational functions	8,730 / 33,216 (26 %)
Dedicated logic registers	5,566 / 33,216 (17 %)
Total registers	5566
Total pins	119 / 475 (25 %)
Total virtual pins	0
Total memory bits	16,384 / 483,840 (3 %)
Embedded Multiplier 9-bit elements	0 / 70 (0 %)
Total PLLs	0 / 4 (0 %)

Με απενεργοποιημένους τους ενσωματωμένους πολλαπλασιαστές, παρατηρήσαμε μια αξιοσημείωτη αύξηση στη χρήση της FPGA. Ο αριθμός των λογικών στοιχείων αυξήθηκε από 22% σε 28%, με σχετικά χαμηλό αριθμό νευρώνων. Προφανώς αν επιλέγαμε να χρησιμοποιήσουμε μία τετοια τεχνική όσο αυξάναμε τον αριθμό των νευρώνων θα αυξάνονταν αναλογικά και η χρήση του FPGA.

Επίσης, στη σχεδίασή μας, το εργαλείο σύνθεσης Quartus εξάγει αυτόματα ενσωματωμένους πολλαπλασιαστές για τη λειτουργία πολλαπλασιασμού κάθε νευρώνα. Αυτό γίνεται μέσω του megafunction «lpm_mult», η οποία αντιστοιχίζεται στους 70 διαθέσιμους πολλαπλασιαστές υλικού της FPGA. Όταν ο αριθμός των νευρώνων υπερβαίνει τους διαθέσιμους πολλαπλασιαστές, το Quartus τους προγραμματίζει δυναμικά και τους επαναχρησιμοποιεί σε διαφορετικούς κύκλους ρολογιού, βελτιστοποιώντας τη χρήση των πόρων. Αυτό επιτρέπει στη σχεδίαση να διαχειρίζεται περισσότερους νευρώνες από τους διαθέσιμους πολλαπλασιαστές χωρίς χειροκίνητη παρέμβαση, διατηρώντας παράλληλα την αποδοτικότητα των επιδόσεων.

5.3 Εξερεύνηση διαφορετικων διαρρυθμίσεων του DNN

Για να βελτιστοποιήσουμε το νευρωνικό δίκτυο για ανάπτυξη στο Cyclone FPGA, διερευνήσαμε πολλαπλές διαμορφώσεις, μεταβάλλοντας τον αριθμό των νευρώνων σε κάθε επίπεδο και δοκιμάζοντας διαφορετικές συναρτήσεις ενεργοποίησης (ReLU και sigmoid). Αυτή η εξερεύνηση παρείχε βασικές γνώσεις σχετικά με τον τρόπο με τον οποίο οι αρχιτεκτονικές αποφάσεις επηρεάζουν τόσο την απόδοση όσο και τη χρήση των πόρων. Παρακάτω, θα συζητήσουμε τρεις βασικές διαμορφώσεις, παρουσιάζοντας τα αποτελέσματα που προέκυψαν μέσω προσομοιώσεων Python και επαληθεύσεων Quartus (ModelSim).

Διαμόρφωση 1: 16-10-10 (ReLU και sigmoid)

Η αρχική μας διαμόρφωση χρησιμοποίησε μια απλή δομή 3 επιπέδων με 16 νευρώνες στο πρώτο κρυφό επίπεδο, 10 στο δεύτερο και 10 στο επίπεδο εξόδου. Εφαρμόσαμε τόσο τις συναρτήσεις ενεργοποίησης ReLU όσο και τις σιγμοειδείς σε ξεχωριστές δοκιμές για να εκτιμήσουμε τον τρόπο με τον οποίο η καθεμία επηρέαζε την απόδοση.

Αυτή η διαμόρφωση, αν και μικρή σε μέγεθος, παρείχε ένα καλό σημείο εκκίνησης για την αξιολόγηση του τρόπου με τον οποίο συμπεριφέρεται κάθε συνάρτηση όσον αφορά τη χρήση των πόρων και την ακρίβεια. Στις προσομοιώσεις της Python, η συνάρτηση ReLU γενικά υπερείχε έναντι της sigmoid, ιδίως όσον αφορά την ταχύτητα. Ωστόσο, όταν συντέθηκαν για το FPGA, και οι δύο συναρτήσεις κατανάλωναν ελάχιστους πόρους, επιτρέποντας την αποτελεσματική σύνθεση αλλά αποδίδοντας χαμηλότερη συνολική ακρίβεια λόγω του μικρού αριθμού νευρώνων.

Relu:

```
Epoch 17/20
419/419 [==============================] - 1s 1ms/step - loss: 1.1814 - accuracy: 0.5967 - val_loss: 1.2253 - val_accuracy: 0.5852
```

```
# Total execution time               1098900 ns
# Reading file:                          C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 0
# 1000. Accuracy: 41.000000, Detected number: 3, Expected: 0003
# Total execution time               1100000 ns
# Accuracy: 41.000000
```

Sigmoid:

```
Epoch 20/20
419/419 [==============================] - 1s 1ms/step - loss: 1.6780 - accuracy: 0.9021 - val_loss: 1.6776 - val_accuracy: 0.8667
```

```
Total execution time                1258740 ns
Reading file:                           C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
Status: 4
1000. Accuracy: 84.700000, Detected number: 3, Expected: 0003
Total execution time                1260000 ns
Accuracy: 84.700000
```

Διαμόρφωση 2: 30-25-10 (ReLU και σιγμοειδές)

Στη συνέχεια, αυξήσαμε τον αριθμό των νευρώνων σε 30 στο πρώτο στρώμα, 25 στο δεύτερο και 10 στο στρώμα εξόδου. Και πάλι, δοκιμάστηκαν τόσο οι συναρτήσεις ReLU όσο και οι σιγμοειδείς συναρτήσεις. Όπως αναμενόταν, ο αυξημένος αριθμός νευρώνων βελτίωσε σημαντικά την ακρίβεια στις προσομοιώσεις της Python, ιδιαίτερα με την ReLU, η οποία συνέχισε να υπερτερεί της sigmoid όσον αφορά την ταχύτητα σύγκλισης και την ακρίβεια.

Όταν υλοποιήθηκε στην FPGA, αυτή η διαμόρφωση χρησιμοποίησε μεγαλύτερο μέρος των διαθέσιμων λογικών στοιχείων και παρατηρήσαμε αξιοσημείωτη αύξηση στη χρήση πόρων. Το αντιστάθμισμα ήταν μια αξιοσημείωτη βελτίωση των επιδόσεων, με τη συνάρτηση ενεργοποίησης ReLU να παρουσιάζει μια πιο ευνοϊκή ισορροπία μεταξύ χρήσης πόρων και ακρίβειας. Το Sigmoid, από την άλλη πλευρά, δυσκολεύτηκε με την αυξημένη πολυπλοκότητα, απαιτώντας περισσότερους πόρους FPGA και πιο αργούς χρόνους εκτέλεσης.

Relu:

```
Epoch 15/20
419/419 [==============================] - 1s 1ms/step - loss: 0.5013 - accuracy: 0.7955 - val_loss: 0.5086 - val_accuracy: 0.7907
```

```
# Total execution time                1198800 ns
# Reading file:                           C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 4
# 1000. Accuracy: 76.200000, Detected number: 3, Expected: 0003
# Total execution time                1200000 ns
# Accuracy: 76.200000
```

Sigmoid:

```
Epoch 20/20
419/419 [==============================] - 1s 1ms/step - loss: 1.5485 - accuracy: 0.9570 - val_loss: 1.5546 - val_accuracy: 0.9370
```

```
# Total execution time              1548450 ns
# Reading file:                         C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 4
# 1000. Accuracy: 93.900000, Detected number: 3, Expected: 0003
# Total execution time              1550000 ns
# Accuracy: 93.900000
```

Διαμόρφωση 3: Δίκτυο 4 επιπέδων (30-25-20-10)

Σε αυτή τη διαμόρφωση, πειραματιστήκαμε με ένα βαθύτερο δίκτυο 4 επιπέδων, χρησιμοποιώντας 30 νευρώνες στο πρώτο επίπεδο, 25 στο δεύτερο, 20 στο τρίτο και 10 στο επίπεδο εξόδου. Στόχος μας ήταν να αξιολογήσουμε κατά πόσον η προσθήκη ενός επιπλέον στρώματος θα ενίσχυε την ικανότητα του δικτύου να μαθαίνει σύνθετα μοτίβα και να βελτιώνει την ακρίβεια. Ωστόσο, τα αποτελέσματα τόσο από τις προσομοιώσεις Python όσο και από τη σύνθεση FPGA έδειξαν ότι αυτό το βαθύτερο δίκτυο δεν είχε την αναμενόμενη απόδοση. Παραδόξως, το πρόσθετο στρώμα οδήγησε σε χειρότερα αποτελέσματα, με αξιοσημείωτη μείωση της ακρίβειας και αύξηση της υπερπροσαρμογής κατά τη φάση εκπαίδευσης στην Python. Στην υλοποίηση FPGA, το δίκτυο 4 επιπέδων επιβάρυνε σημαντικά τους διαθέσιμους πόρους, καταναλώνοντας πολύ περισσότερο από τις προηγούμενες διαμορφώσεις 3 επιπέδων. Η πρόσθετη πολυπλοκότητα όχι μόνο κατανάλωσε περισσότερα λογικά στοιχεία, αλλά οδήγησε και σε προβλήματα χρονισμού, δυσχεραίνοντας τελικά την απόδοση του δικτύου. Αυτή η διαμόρφωση κατέδειξε ότι η απλή προσθήκη περισσότερων στρωμάτων χωρίς αύξηση της χωρητικότητας των νευρώνων σε κάθε στρώμα οδηγεί σε φθίνουσες αποδόσεις. Στην πραγματικότητα, κατέδειξε ότι οι πόροι του Cyclone FPGA είναι καταλληλότεροι για μια αρχιτεκτονική 3 επιπέδων με υψηλότερο αριθμό νευρώνων σε κάθε επίπεδο, αντί να τεντώνονται για να υποστηρίξουν μια σχεδίαση 4 επιπέδων.

Τα ευρήματα αυτού του πειράματος τόνισαν ότι, για το έργο μας, η προσήλωση σε ένα καλά βελτιστοποιημένο δίκτυο 3 επιπέδων απέδωσε πολύ καλύτερα αποτελέσματα από την προσθήκη επιπλέον επιπέδων.

Relu:

```
Epoch 20/20
419/419 [==============================] - 1s 1ms/step - loss: 0.9469 - accuracy: 0.6881 - val_loss: 1.0227 - val_accuracy: 0.6778
```

```
# Total execution time              1178820 ns
# Reading file:                         C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 4
# 1000. Accuracy: 57.800000, Detected number: 3, Expected: 0003
# Total execution time              1180000 ns
# Accuracy: 57.800000
```

Sigmoid:

```
Epoch 20/20
419/419 [==============================] - 1s 1ms/step - loss: 1.6533 - accuracy: 0.7319 - val_loss: 1.6514 - val_accuracy: 0.7074
```

```
# Total execution time              1198800 ns
# Reading file:                         C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 4
# 1000. Accuracy: 71.900000, Detected number: 3, Expected: 0003
# Total execution time              1200000 ns
# Accuracy: 71.900000
```

Τελική διαμόρφωση: 50-40-10 (πιέζοντας τα όρια της FPGA)

Με βάση τα αποτελέσματα των προηγούμενων διαμορφώσεων, επιστρέψαμε σε ένα δίκτυο 3 επιπέδων και μεγιστοποιήσαμε τον αριθμό των νευρώνων που μπορούσε να διαχειριστεί η FPGA. Η τελική διαμόρφωση χρησιμοποίησε 50 νευρώνες στο πρώτο στρώμα, 40 στο δεύτερο στρώμα και 10 στο στρώμα εξόδου. Αυτή η ρύθμιση ώθησε τους πόρους της FPGA στα όριά της, χρησιμοποιώντας το 99% των διαθέσιμων λογικών στοιχείων.

Στην Python, αυτή η διαμόρφωση πέτυχε την υψηλότερη ακρίβεια, ιδίως με τη συνάρτηση ενεργοποίησης Sigmoid . Κατά τη σύνθεση για την FPGA, αυτή η διαμόρφωση κατανάλωσε σχεδόν όλους τους διαθέσιμους πόρους, αλλά παρείχε σημαντική βελτίωση της ακρίβειας σε σύγκριση με τις μικρότερες διαμορφώσεις. Ωστόσο, η Relu συνάρτηση, απέδωσε υποβέλτιστα, απαιτώντας λιγότερους υπολογιστικούς πόρους και παρέχοντας χειρότερα αποτελέσματα.

```
Epoch 20/20
419/419 [==============================] - 1s 1ms/step - loss: 1.5118 - accuracy: 0.9730 - val_loss: 1.5192 - val_accuracy: 0.9574
```

```
# Total execution time              1818180 ns
# Reading file:                         C:/Users/Ohmen/Desktop/Desktop/School/Thesis/test_data/test_data_0999.txt
# Status: 4
# 1000. Accuracy: 94.500000, Detected number: 3, Expected: 0003
# Total execution time              1820000 ns
# Accuracy: 94.500000
```

5.4 Πειραματική Διαρρύθμιση

Στην υλοποίηση του υλικού μας, οι διακόπτες και η οθόνη LCD της πλακέτας DE2 έπαιξαν κρίσιμο ρόλο στη διασύνδεση με το νευρωνικό δίκτυο και στην επαλήθευση της λειτουργίας του.

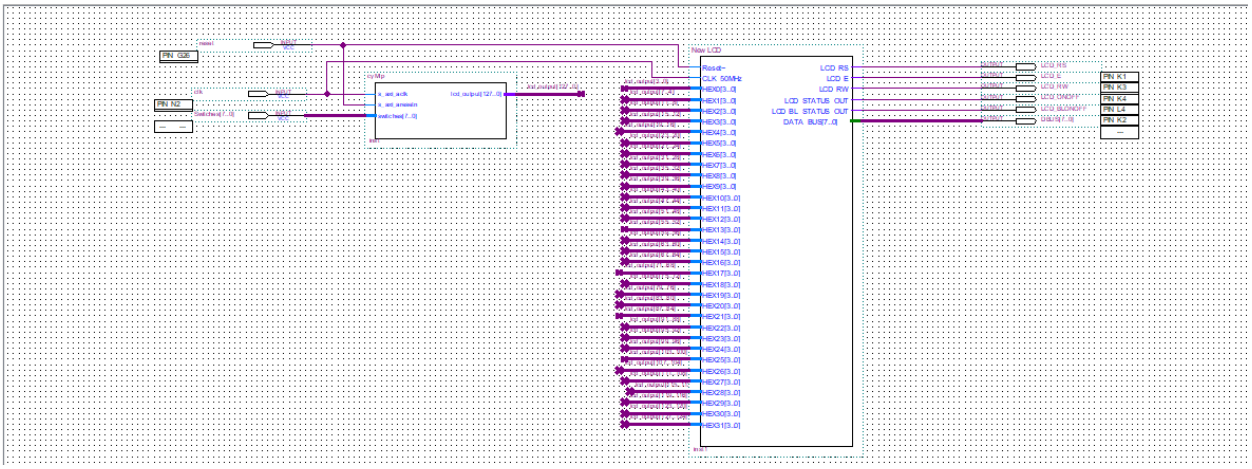
Διακόπτες: Οι διακόπτες στην πλακέτα DE2 χρησιμοποιήθηκαν για να ελέγχουν πόσες εισόδους επεξεργάζεται το νευρωνικό δίκτυο. Αντί να παρέχουμε τις εισόδους απευθείας σε πραγματικό χρόνο, τις αποθηκεύσαμε μέσα στο FPGA, όπως ακριβώς

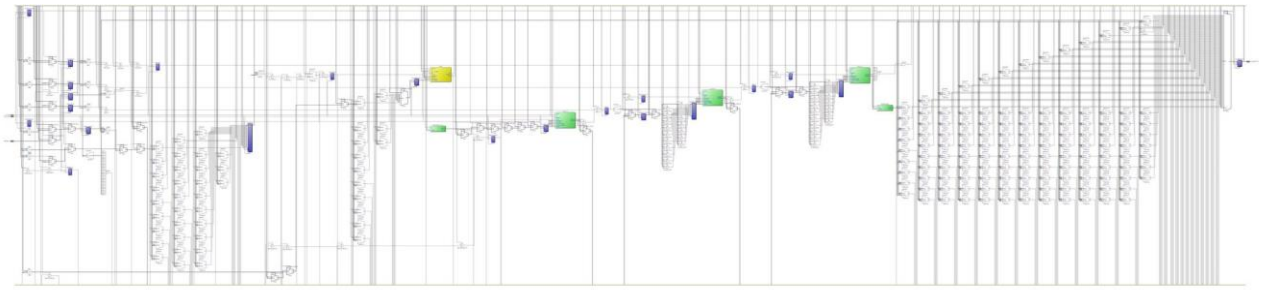
αποθηκεύουμε τα βάρη και τα bias. Αυτό επέτρεψε ταχύτερη πρόσβαση και επεξεργασία. Οι διακόπτες μας επέτρεψαν να καθορίσουμε δυναμικά πόσες αποθηκευμένες εισόδους θα έπρεπε να επεξεργαστεί το δίκτυο, ξεκινώντας από ένα καθορισμένο σημείο. Δόθηκε ιδιαίτερη προσοχή στον τρόπο με τον οποίο αντιστοιχίσαμε τους διακόπτες στα pins της FPGA, εξασφαλίζοντας ακριβή έλεγχο και επιλογή εισόδου.

Συνδέοντας τα κατάλληλα pins με τους καθορισμένους διακόπτες, δώσαμε τη δυνατότητα στο FPGA να ερμηνεύει σωστά τις θέσεις των διακοπτών. Αυτή η ρύθμιση μας επέτρεψε να διαχειριστούμε αποτελεσματικά το μέγεθος των εισόδων χωρίς να χρειάζεται να επαναπρογραμματίσουμε την FPGA, καθιστώντας την μια ευέλικτη προσέγγιση για τον πειραματισμό με διάφορες διαμορφώσεις εισόδου.

Οθόνη LCD: Η οθόνη LCD χρησιμοποιήθηκε για την εμφάνιση των προβλέψεων του δικτύου σε πραγματικό χρόνο. Για να το επιτύχουμε αυτό, αξιοποιήσαμε μια υπάρχουσα βιβλιοθήκη για βασικές λειτουργίες LCD. Ωστόσο, αναπτύχθηκε προσαρμοσμένη λογική για τη διασύνδεση του νευρωνικού δικτύου με την οθόνη, μετατρέποντας την έξοδο του νευρωνικού δικτύου σε μορφή κατάλληλη για την οθόνη. Δημιουργήσαμε μια προσαρμοσμένη μονάδα για να ενεργεί ως ενδιάμεσος μεταξύ της εξόδου του νευρωνικού δικτύου και της οθόνης LCD, διασφαλίζοντας ότι το FPGA θα μπορούσε να επικοινωνεί απρόσκοπτα με την οθόνη.

Μόλις αναπτύχθηκε η μονάδα, τη συνδέσαμε στους κατάλληλους ακροδέκτες της πλακέτας DE2, διευκολύνοντας την ομαλή επικοινωνία μεταξύ της FPGA και της LCD. Η εμφάνιση των αποτελεσμάτων στην οθόνη σε πραγματικό χρόνο παρείχε άμεση ανατροφοδότηση σχετικά με τις προβλέψεις του δικτύου, η οποία ήταν ζωτικής σημασίας για τη δοκιμή και την αποσφαλμάτωση της υλοποίησης υλικού.

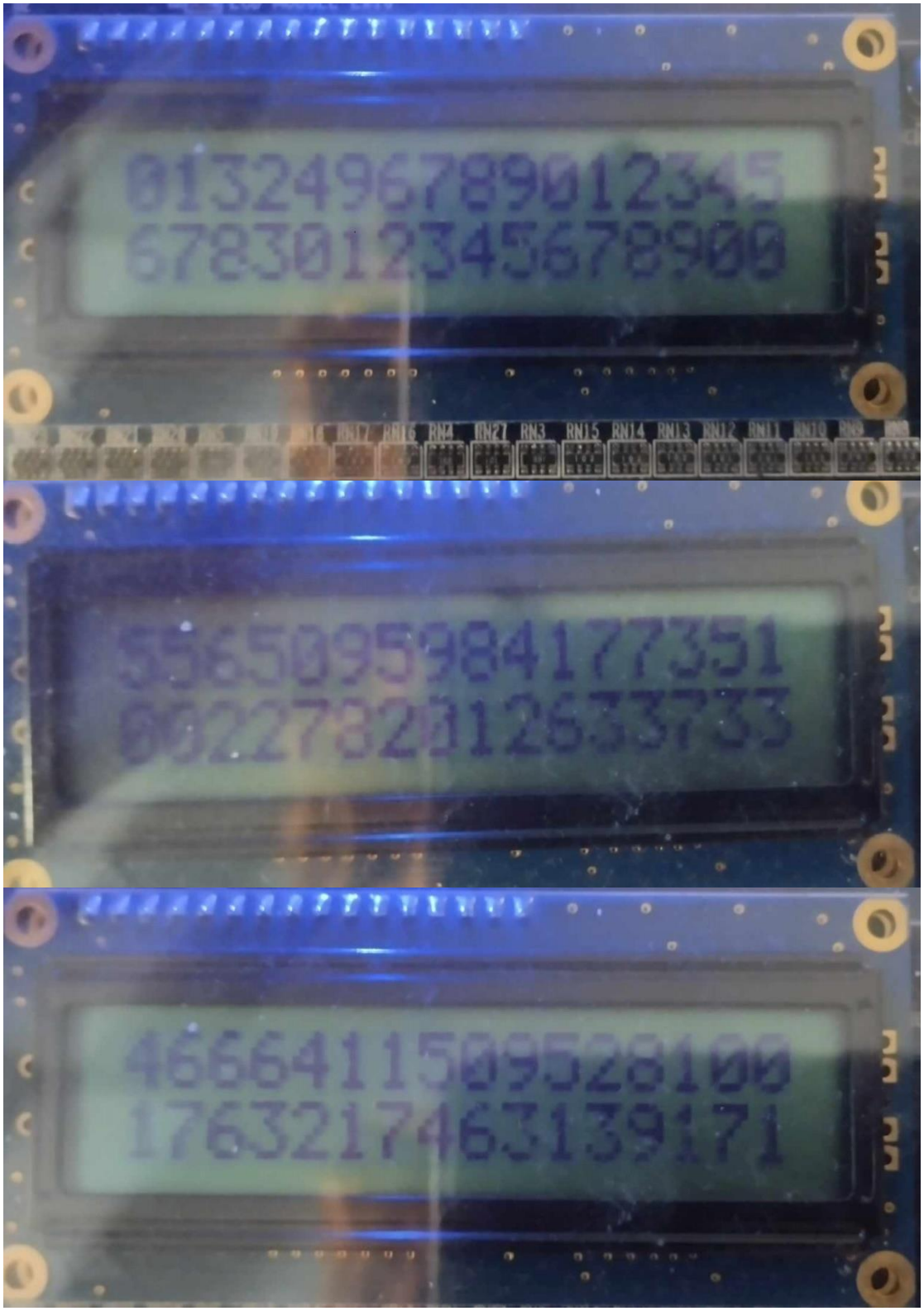




5.5 Πειραματικά Αποτελέσματα

Αφού οριστικοποιήσαμε τη βέλτιστη διαμόρφωση του νευρωνικού δικτύου, προχωρήσαμε στη δοκιμή της απόδοσής του στο Cyclone II FPGA. Η τελική υλοποίηση μας επέτρεψε να αποθηκεύσουμε τόσο τις εισόδους όσο και τα βάρη/bias απευθείας μέσα στο FPGA, με τους διακόπτες στην πλακέτα DE2 να χρησιμοποιούνται για να ελέγχουν δυναμικά πόσες εισόδους θα επεξεργαστούν και πότε θα ξεκινήσει η επεξεργασία. Πραγματοποιήσαμε τρεις δοκιμαστικές εκτελέσεις στην FPGA, επεξεργαζόμενοι κάθε φορά 32 εισόδους, καθώς το σύστημα μπορούσε να εμφανίζει ταυτόχρονα έως και 32 εξόδους. Κατά τη διάρκεια αυτών των δοκιμών, συγκρίναμε τα προβλεπόμενα αποτελέσματα με τις αναμενόμενες τιμές, επιτυγχάνοντας συνολική ακρίβεια 91,67%. Η ακρίβεια υπολογίστηκε χρησιμοποιώντας τον τύπο Ακρίβεια = (88/96) × 100%, όπου 88 αντιπροσωπεύει τον αριθμό των σωστών προβλέψεων από το σύνολο των 96 εισόδων στις τρεις δοκιμαστικές εκτελέσεις. Το FPGA εκτέλεσε με επιτυχία τις προβλέψεις σε πραγματικό χρόνο, με τις εξόδους να εμφανίζονται στην οθόνη LCD. Τα αποτελέσματα ταίριαζαν απόλυτα με τα αναμενόμενα αποτελέσματα με βάση τις προηγούμενες προσομοιώσεις μας. Ωστόσο, όπως αναμενόταν, η υλοποίηση

της FPGA παρουσίασε μικρές αποκλίσεις από την προσομοίωση λόγω της χρήσης αριθμητικής σταθερής υποδιαστολής, η οποία εισήγαγε κάποια σφάλματα κβαντισμού. Παρόλα αυτά, το σύστημα εξακολουθούσε να λειτουργεί εντός αποδεκτών ορίων ακρίβειας. Παρακάτω, μπορείτε να δείτε τα αποτελέσματα, και των τριων εκτελέσεων, απευθειας από την οθονη LCD του DE2 Board.



Στον Πίνακα 5.1 βλέπουμε τα πρώτα είκοσι αποτελέσματα της υλοποίησης στο FPGA

Input Pattern	Output	Result
0	**0**	**correct**
1	**1**	**correct**
3	**3**	**correct**
2	**2**	**correct**
4	**4**	**correct**
5	**9**	**incorrect**
6	**6**	**correct**
7	**7**	**correct**
8	**8**	**correct**
9	**9**	**correct**
0	**0**	**correct**
1	**1**	**correct**
3	**3**	**correct**
2	**2**	**correct**
4	**4**	**correct**
5	**9**	**correct**
6	**6**	**correct**
7	**7**	**correct**
8	**8**	**correct**
9	**3**	**incorrect**

0	**0**	**correct**

Πίνακας 5.1. Μερικά Αποτελέσματα Εκδοχής FPGA

Συμπερασματικά, αν και η ακρίβεια ήταν ελαφρώς χαμηλότερη από τις προσομοιώσεις λογισμικού, η υλοποίηση FPGA έδειξε ότι μπορεί να χειριστεί αποτελεσματικά την εξαγωγή συμπερασμάτων νευρωνικών δικτύων σε πραγματικό χρόνο, ακόμη και υπό περιορισμούς υλικού. Η ευελιξία που παρέχεται από τους διακόπτες εισόδου της πλακέτας DE2 μας επέτρεψε να ελέγχουμε την επεξεργασία δυναμικά, καθιστώντας το FPGA μια βιώσιμη πλατφόρμα για εφαρμογές νευρωνικών δικτύων σε περιβάλλοντα με περιορισμένους πόρους.

5.6 Σύγκριση αποτελεσμάτων

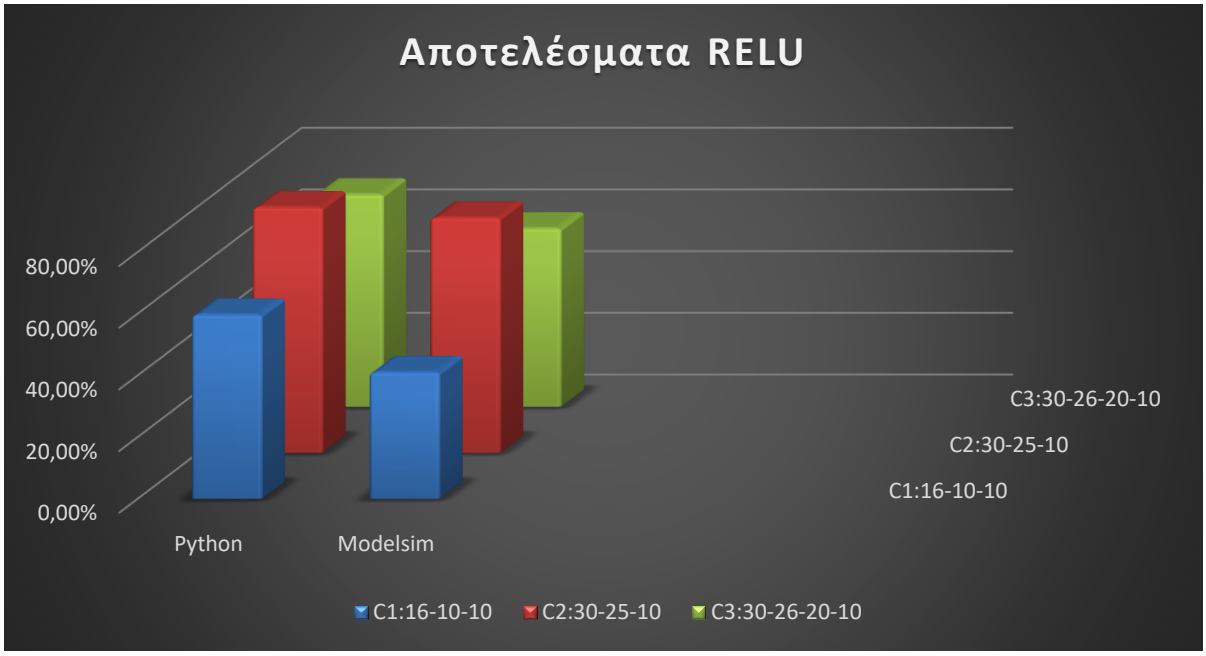
Με την επικύρωση της υλοποίησης FPGA, είναι σημαντικό να συγκρίνετε τις επιδόσεις σε όλα τα στάδια ανάπτυξης: Verilog και την υλοποίηση FPGA. Κάθε στάδιο παρείχε πολύτιμες γνώσεις σχετικά με τη συμπεριφορά του νευρωνικού δικτύου και συγκρίνοντας τα αποτελέσματά τους, μπορέσαμε να αξιολογήσουμε τους συμβιβασμούς μεταξύ της ακρίβειας, της ταχύτητας και της αποδοτικότητας των πόρων.

Αποτελέσματα ενεργοποίησης ReLU

Ο Πίνακας 5.2 παρουσιάζει τα αποτελέσματα για διαφορετικές διαμορφώσεις του δικτύου με τη χρήση της συνάρτησης ενεργοποίησης ReLU. Η ακρίβεια που επιτυγχάνεται στην προσομοίωση Python συγκρίνεται με την ακρίβεια από το ModelSim, καταδεικνύοντας μια αξιοσημείωτη πτώση στο ModelSim λόγω των περιορισμών της αριθμητικής σταθερού σημείου. Τα αποτελέσματα της Python αντιπροσωπεύουν την υψηλότερη ακρίβεια που μπορεί να επιτευχθεί με ακρίβεια κινητής υποδιαστολής.

	Ακρίβεια Python	**Ακρίβεια Modelsim**
16-10-10	59,67%	41%
30-25-10	79,55%	76,2%
30-25-20-10	68,81%	57,8%

Πίνακας 5.2. Αποτελέσματα διαφορετικών εκδοχών με Relu

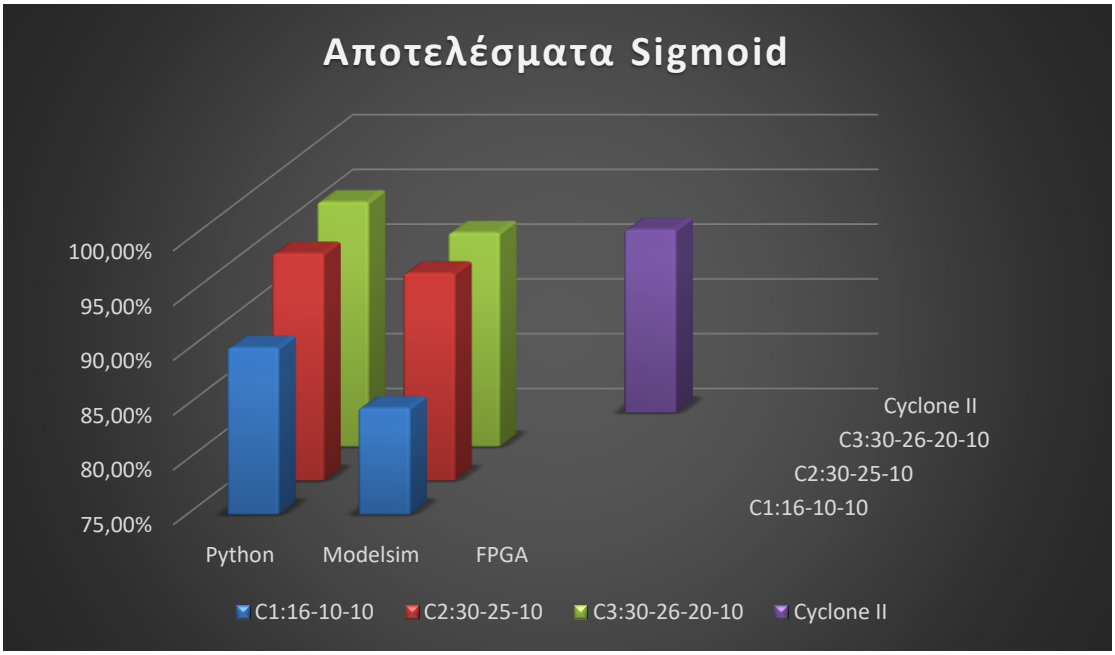


Αποτελέσματα ενεργοποίησης του σιγμοειδούς

Ο Πίνακας 5.3 παρέχει τα αποτελέσματα ακρίβειας για τις ίδιες διαμορφώσεις δικτύου αλλά με τη χρήση της συνάρτησης ενεργοποίησης Sigmoid. Όπως αναμενόταν, τα αποτελέσματα ήταν υψηλότερα στις περισσότερες περιπτώσεις σε σύγκριση με το ReLU, ιδίως για βαθύτερες διαμορφώσεις δικτύου. Επιπλέον, ο πίνακας παρουσιάζει την ακρίβεια της υλοποίησης Cyclone II FPGA, η οποία πέτυχε αξιοσημείωτη ακρίβεια 91,67%, αν και ελαφρώς χαμηλότερη από την ακρίβεια που επιτεύχθηκε στο ModelSim λόγω σφαλμάτων κβαντισμού στην υλοποίηση υλικού.

	Ακρίβεια Python	**Ακρίβεια Modelsim**	**Ακρίβεια FPGA**
16-10-10	90.21%	84.7%	-
30-25-10	95.7%	93.9%	-
30-25-20-10	73.19%	71.9%	-
50-40-10	97.3%	94.5%	-
Cyclone II	-	-	91.67%

Πίνακας 5.3. Αποτελέσματα διαφορετικών εκδοχών με Sigmoid



Όσον αφορά τη χρήση των πόρων, βελτιστοποιήσαμε επιτυχώς την υλοποίηση υλικού ώστε να αξιοποιήσουμε πλήρως τους πόρους του Cyclone II FPGA. Υλοποιήσαμε μια δομή νευρωνικού δικτύου με 50 νευρώνες στο πρώτο στρώμα, 40 στο δεύτερο κρυφό στρώμα και 10 στο στρώμα εξόδου. Αυτή η σχεδίαση απαιτούσε σημαντικό αριθμό λογικών στοιχείων (LEs) και μέσω των βελτιστοποιήσεων που αναφέρθηκαν προηγουμένως, καταφέραμε να μεγιστοποιήσουμε τον αριθμό των νευρώνων επιτυγχάνοντας σχεδόν πλήρη αξιοποίηση των πόρων της FPGA.

Η σύνοψη της ροής δείχνει ότι χρησιμοποιήθηκε το 99% του συνόλου των διαθέσιμων λογικών στοιχείων, που αντιστοιχεί σε 32.905 από τα 33.216 LEs. Αυτό αναδεικνύει πόσο αποτελεσματικά χρησιμοποιήθηκε η FPGA, ιδίως αν ληφθεί υπόψη ότι χρησιμοποιήθηκε επίσης το 99% των συνδυαστικών συναρτήσεων, με 32.808 από τις 33.216 συνδυαστικές συναρτήσεις σε χρήση. Επιπλέον, χρησιμοποιήθηκαν 8.322 λογικοί καταχωρητές, που αντιστοιχούν στο 25% της διαθέσιμης χωρητικότητας καταχωρητών. Η διαχείριση της μνήμης ήταν επίσης μια κρίσιμη πτυχή, με το 24% των συνολικών bits μνήμης να χρησιμοποιείται, που αντιστοιχεί σε 114.048 bits από τα διαθέσιμα 483.840. Αυτή η προσεκτική διαχείριση τόσο της μνήμης όσο και των λογικών πόρων μας επέτρεψε να υλοποιήσουμε μια πιο σύνθετη αρχιτεκτονική νευρωνικού δικτύου, ξεπερνώντας τα όρια του εφικτού με τη συγκεκριμένη FPGA. Τελικά, πετύχαμε έναν σχεδιασμό που μεγιστοποίησε τη χωρητικότητα της FPGA, εξασφαλίζοντας τη βέλτιστη απόδοση εντός των περιορισμών του υλικού.

Flow Summary	
Flow Status	Successful - Fri Sep 13 20:52:25 2024
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	cyMlp
Top-level Entity Name	cyMlp
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Total logic elements	32,905 / 33,216 (99 %)
Total combinational functions	32,808 / 33,216 (99 %)
Dedicated logic registers	8,322 / 33,216 (25 %)
Total registers	8322
Total pins	138 / 475 (29 %)
Total virtual pins	0
Total memory bits	114,048 / 483,840 (24 %)
Embedded Multiplier 9-bit elements	70 / 70 (100 %)
Total PLLs	0 / 4 (0 %)

5.7 Συζήτηση των αποτελεσμάτων

Τα αποτελέσματα που προέκυψαν τόσο από τις προσομοιώσεις Python όσο και από τις υλοποιήσεις FPGA υπογραμμίζουν τον κεντρικό στόχο της παρούσας διατριβής: να διερευνήσει πόσο μακριά μπορεί να φτάσει η Cyclone II FPGA όσον αφορά το μέγεθος και την πολυπλοκότητα του βαθύ νευρωνικού δικτύου (DNN), διατηρώντας παράλληλα υψηλή ακρίβεια. Η προσέγγισή μας εξέτασε τα όρια της χρήσης των πόρων υλοποιώντας ολοένα και μεγαλύτερα νευρωνικά δίκτυα για να αξιολογήσει τη σκοπιμότητα και την απόδοσή τους στο Cyclone II FPGA, με έμφαση στην εξισορρόπηση των περιορισμών των πόρων με την ακρίβεια.

1. **Πιέζοντας το Cyclone II στα όριά του**

Το πρωταρχικό μας επίτευγμα ήταν η επιτυχής υλοποίηση μιας δομής DNN 50-40-10 στο Cyclone II FPGA. Αυτή η σχεδίαση ώθησε την FPGA στο 99% της χρήσης των διαθέσιμων λογικών στοιχείων (LEs) της, αποδεικνύοντας ότι ακόμη και μια FPGA με περιορισμένους πόρους μπορεί να χειριστεί πολύπλοκα νευρωνικά δίκτυα με προσεκτική βελτιστοποίηση. Η επίτευξη σχεδόν πλήρους αξιοποίησης των πόρων με ελάχιστη υποβάθμιση των επιδόσεων είναι ένα βασικό στοιχείο, καθώς δείχνει ότι

μεγιστοποιήσαμε αποτελεσματικά τις δυνατότητες της FPGA χωρίς να υπερβούμε τα όριά της.

Ένα από τα πιο εντυπωσιακά αποτελέσματα των δοκιμών μας ήταν το γεγονός ότι παρά τον σχεδόν κορεσμό των διαθέσιμων πόρων, το δίκτυο εξακολουθούσε να επιτυγχάνει υψηλή ακρίβεια τόσο στην Python (97,3%) όσο και στο ModelSim (94,5%) όταν χρησιμοποιούσε ενεργοποίηση Sigmoid. Αυτό αποτελεί απόδειξη ότι, με βελτιστοποιημένες τεχνικές σχεδίασης -συμπεριλαμβανομένης της αριθμητικής σταθερής υποδιαστολής και της αποδοτικής διαχείρισης μνήμης- η Cyclone II FPGA είναι ικανή να εκτελεί προηγμένα νευρωνικά δίκτυα.

2. Επίδραση του βάθους του δικτύου και της δομής των στρωμάτων

Τα πειράματα αποκάλυψαν επίσης σημαντικές γνώσεις σχετικά με το βάθος και την πολυπλοκότητα του δικτύου. Αρχικά, υποθέσαμε ότι τα βαθύτερα δίκτυα με περισσότερα στρώματα (π.χ. τέσσερα στρώματα) θα παρείχαν καλύτερες επιδόσεις. Ωστόσο, τα αποτελέσματά μας έδειξαν ότι μια διαμόρφωση τριών στρωμάτων, με μεγαλύτερο αριθμό νευρώνων ανά στρώμα, απέδωσε ανώτερα αποτελέσματα για τα δεδομένα της δοκιμής που εξετάσαμε. Τα δεδομένα δοκιμής δεν ήταν αρκετά πολύπλοκα ώστε να δικαιολογούν ένα βαθύτερο δίκτυο και η αύξηση του αριθμού των στρωμάτων προσέθετε περιττή πολυπλοκότητα χωρίς να βελτιώνει την ακρίβεια.

Η διαμόρφωση 50-40-10, η οποία διαθέτει τρία στρώματα με αυξημένο αριθμό νευρώνων, είχε βέλτιστες επιδόσεις, επιτυγχάνοντας τη σωστή ισορροπία μεταξύ βάθους και αριθμού νευρώνων. Αυτό υποδηλώνει ότι, σε πολλές εφαρμογές του πραγματικού κόσμου όπου τα δεδομένα δεν είναι υπερβολικά πολύπλοκα, η προσθήκη στρωμάτων μπορεί να μην είναι επωφελής. Αντίθετα, η εστίαση στη μεγιστοποίηση του αριθμού των νευρώνων σε λιγότερα επίπεδα μπορεί να επιτύχει καλύτερα αποτελέσματα, όπως παρατηρήθηκε με την αρχιτεκτονική μας.

Επιπλέον, η διαμόρφωση 30-25-20 προσέφερε μια καλή μέση λύση μεταξύ αποδοτικότητας πόρων και ακρίβειας, τονίζοντας ότι ακόμη και μικρότερα δίκτυα μπορούν να επιτύχουν αποδεκτή ακρίβεια. Ωστόσο, η επιπλέον ακρίβεια 5-6% που κερδίζεται με τα μεγαλύτερα δίκτυα, ειδικά όταν ωθούνται σε σχεδόν πλήρη χρήση, καθίσταται κρίσιμη σε πραγματικές περιπτώσεις, όπου η κάλυψη αυτού του χάσματος επιδόσεων μπορεί να είναι η διαφορά μεταξύ ικανοποιητικής και βέλτιστης απόδοσης.

3. Ακρίβεια έναντι αποδοτικότητας πόρων

Ένα βασικό συμπέρασμα από τη σύγκρισή μας μεταξύ διαφορετικών αρχιτεκτονικών είναι ο συμβιβασμός μεταξύ ακρίβειας και χρήσης πόρων. Οι μεγαλύτερες διαμορφώσεις, όπως η 50-40-10, παρείχαν σαφώς καλύτερη ακρίβεια, αλλά με το κόστος του σχεδόν κορεσμού των πόρων της FPGA. Από την άλλη πλευρά, οι μικρότερες διαμορφώσεις, όπως το 16-10-10, πέτυχαν αξιοπρεπή ακρίβεια με πολύ μικρότερη κατανάλωση πόρων, η οποία θα μπορούσε να είναι ιδανική για εφαρμογές με αυστηρότερους περιορισμούς υλικού.

Τούτου λεχθέντος, ενώ η διαφορά στην ακρίβεια μεταξύ αυτών των αρχιτεκτονικών μπορεί να φαίνεται οριακή (μόνο 5-6%), η διαφορά αυτή μπορεί να είναι σημαντική σε πραγματικές εφαρμογές. Στην πράξη, κάθε πρόσθετη ποσοστιαία μονάδα ακρίβειας μπορεί να οδηγήσει σε καλύτερη λήψη αποφάσεων και λιγότερα σφάλματα, ιδίως σε κρίσιμα συστήματα όπου η ακρίβεια αποτελεί κλειδί. Έτσι, ενώ τα μικρότερα δίκτυα είναι πιο αποδοτικά ως προς τους πόρους, η ώθηση της FPGA για την υποστήριξη μεγαλύτερων, ακριβέστερων δικτύων αξίζει τον κόπο σε εφαρμογές που απαιτούν τις υψηλότερες δυνατές επιδόσεις.

4. Απόδοση ReLU vs. Sigmoid

Μια άλλη σημαντική παρατήρηση από τα αποτελέσματα είναι η σύγκριση μεταξύ των συναρτήσεων ενεργοποίησης ReLU και Sigmoid. Όπως δείχνουν οι πίνακες, η Sigmoid υπερτερεί σταθερά έναντι της ReLU, ιδίως όσον αφορά τη διατήρηση της ακρίβειας τόσο στις προσομοιώσεις Python όσο και στις προσομοιώσεις ModelSim. Η μεγαλύτερη πτώση της ακρίβειας για τα δίκτυα που βασίζονται στην ReLU κατά τη μετάβαση από την Python (κινητής υποδιαστολής) στην ModelSim (σταθερής υποδιαστολής) υποδηλώνει ότι η ReLU είναι πιο ευαίσθητη στην απώλεια ακρίβειας κατά τη σύνθεση FPGA.

Η Sigmoid, με τις πιο ομαλές κλίσεις της, αποδείχθηκε πιο ανθεκτική στους περιορισμούς της αριθμητικής σταθερής υποδιαστολής στην FPGA, καθιστώντας την πιο κατάλληλη επιλογή για νευρωνικά δίκτυα βασισμένα σε FPGA, ιδίως σε περιβάλλοντα με περιορισμένους πόρους όπως το δικό μας. Αυτό το εύρημα είναι κρίσιμο, καθώς ενημερώνει τους μελλοντικούς σχεδιασμούς σχετικά με το ποιες συναρτήσεις ενεργοποίησης είναι πιο αξιόπιστες σε σενάρια περιορισμένης ακρίβειας.

Κεφάλαιο 6. < Συμπεράσματα και μελλοντικές εργασίες>

6.1 Συμπέρασμα

Η παρούσα διατριβή είχε ως στόχο να διερευνήσει τα όρια του Cyclone II FPGA στην υλοποίηση ενός βαθύ νευρωνικού δικτύου (DNN). Ωθώντας την αρχιτεκτονική στις δυνατότητές της, αποδείξαμε με επιτυχία ότι μπορεί να υλοποιηθεί ένα DNN 3 επιπέδων με σημαντικό αριθμό νευρώνων, διατηρώντας παράλληλα καλή ακρίβεια. Η προσέγγισή μας εξισορρόπησε τη χρήση των πόρων με την ακρίβεια, δείχνοντας ότι η μεγιστοποίηση του αριθμού των νευρώνων σε λιγότερα επίπεδα παρείχε καλύτερα αποτελέσματα για την πολυπλοκότητα των δεδομένων δοκιμής που χρησιμοποιήθηκαν. Επιπλέον, κατέστη προφανές ότι ενώ θα μπορούσε να υλοποιηθεί ένα DNN 4 επιπέδων, τα κέρδη απόδοσης ήταν οριακά σε σύγκριση με την αύξηση της κατανάλωσης πόρων.

Αυτή η έρευνα αναδεικνύει τις αντισταθμίσεις μεταξύ του βάθους του δικτύου, της ακρίβειας και των περιορισμών του υλικού. Η επίτευξη υψηλής ακρίβειας, ακόμη και σε σχετικά απλά σύνολα δεδομένων, απαιτεί μια λεπτή ισορροπία μεταξύ της

πολυπλοκότητας του δικτύου και των διαθέσιμων πόρων FPGA. Τελικά, αυτή η εργασία συμβάλλει σε ένα αυξανόμενο σώμα έρευνας σχετικά με αποτελεσματικές υλοποιήσεις DNN σε πλατφόρμες περιορισμένου υλικού, όπως οι FPGA.

6.2 Μελλοντική εργασία

Ενώ η παρούσα έρευνα κατέδειξε τις δυνατότητες του Cyclone II FPGA για υλοποιήσεις DNN, παραμένουν ανοιχτοί αρκετοί δρόμοι για μελλοντική διερεύνηση:

- Εξερευνώντας μεγαλύτερες και πιο προηγμένες πλατφόρμες FPGA:

Η Cyclone II FPGA, αν και αποτελεί μια ισχυρή πλατφόρμα για τη μελέτη μας, έχει τους περιορισμούς της όσον αφορά τη μνήμη, τα λογικά στοιχεία και την επεξεργαστική ισχύ. Η μετάβαση σε πιο προηγμένες FPGA (όπως η σειρά Cyclone V ή Stratix) θα επέτρεπε βαθύτερα δίκτυα με περισσότερους νευρώνες ανά στρώμα. Αυτές οι πλατφόρμες παρέχουν βελτιωμένους πόρους, όπως ενσωματωμένους πολλαπλασιαστές και μπλοκ DSP, οι οποίοι θα μπορούσαν να υποστηρίξουν πιο εξελιγμένες αρχιτεκτονικές, συμπεριλαμβανομένων των συνελικτικών νευρωνικών δικτύων (CNN) ή των επαναλαμβανόμενων νευρωνικών δικτύων (RNN). Αυτή η μετατόπιση θα μπορούσε να επιτρέψει εργασίες που απαιτούν υψηλότερη υπολογιστική πολυπλοκότητα, όπως η επεξεργασία εικόνας ή η αναγνώριση ομιλίας.

- Υβριδικές αρχιτεκτονικές (Co-Design CPU-FPGA):

Οι FPGA υπερέχουν στην παράλληλη επεξεργασία, αλλά ορισμένες πτυχές των νευρωνικών δικτύων, όπως η πολύπλοκη λογική ελέγχου και οι μη παραλληλοποιήσιμες λειτουργίες, μπορεί να αντιμετωπίζονται καλύτερα από CPU. Μια υβριδική αρχιτεκτονική που συνδυάζει την ισχύ παράλληλης επεξεργασίας της FPGA με την ευελιξία μιας CPU θα μπορούσε να αποδώσει καλύτερες επιδόσεις για εργασίες που περιλαμβάνουν πιο περίπλοκες διαδικασίες λήψης αποφάσεων. Αυτό θα μπορούσε επίσης να εκφορτώσει ορισμένες μη κρίσιμες εργασίες στη CPU, ενώ η FPGA θα επικεντρωνόταν στην επιτάχυνση των βασικών νευρωνικών υπολογισμών, βελτιστοποιώντας περαιτέρω την απόδοση του συστήματος.

- Συστήματα Multi-FPGA για κατανεμημένα νευρωνικά δίκτυα:

Για ιδιαίτερα μεγάλα ή πολύπλοκα DNN, οι μεμονωμένες υλοποιήσεις FPGA μπορεί να μην επαρκούν. Σε τέτοιες περιπτώσεις, θα μπορούσαν να εξεταστούν συστήματα πολλαπλών FPGA, όπου το νευρωνικό δίκτυο κατανέμεται σε πολλές FPGA. Αυτή η προσέγγιση μπορεί να παραλληλίσει τον υπολογισμό ακόμη περισσότερο, επιτρέποντας την εκτέλεση βαθιών δικτύων σε πραγματικό χρόνο σε ενσωματωμένες ή ακραίες συσκευές. Επιπλέον, η έρευνα σχετικά με τα πρωτόκολλα επικοινωνίας μεταξύ των FPGA και τον τρόπο με τον οποίο μοιράζονται αποτελεσματικά τα ενδιάμεσα δεδομένα θα μπορούσε να καταστήσει αυτή την προσέγγιση πιο βιώσιμη.

- Επεξεργασία σε πραγματικό χρόνο σε εφαρμογές ακμής:

Ένας κρίσιμος τομέας της μελλοντικής έρευνας θα μπορούσε να περιλαμβάνει την ανάπτυξη αυτών των νευρωνικών δικτύων που επιταχύνονται από FPGA σε πραγματικό χρόνο, σε εφαρμογές πραγματικού κόσμου. Αυτό περιλαμβάνει αυτόνομα οχήματα, μη επανδρωμένα αεροσκάφη, ιατρικές συσκευές ή άλλες συσκευές άκρων όπου η εξαγωγή συμπερασμάτων με χαμηλή καθυστέρηση είναι κρίσιμη. Η δοκιμή του συστήματος σε δυναμικά περιβάλλοντα με μεταβλητές εισόδους και συνθήκες θα παρείχε πολύτιμη ανατροφοδότηση σχετικά με τον τρόπο περαιτέρω βελτίωσης της αρχιτεκτονικής όσον αφορά την ευρωστία, την αποδοτικότητα ισχύος και την ταχύτητα. Επιπλέον, η ενσωμάτωση DNNs με βάση FPGA με συστήματα απόκτησης δεδομένων σε πραγματικό χρόνο θα μπορούσε να ανοίξει νέες πόρτες για τη λήψη αποφάσεων με βάση την τεχνητή νοημοσύνη σε βιομηχανίες όπως η μεταποίηση, η υγειονομική περίθαλψη και η ρομποτική.

- Μελέτες ενεργειακής απόδοσης:

Η ενεργειακή αποδοτικότητα είναι υψίστης σημασίας, καθώς οι FPGAs χρησιμοποιούνται συχνά σε περιβάλλοντα με περιορισμένους πόρους, όπως τα μη επανδρωμένα αεροσκάφη ή οι φορητές συσκευές. Η μελλοντική έρευνα θα μπορούσε να διερευνήσει τον τρόπο περαιτέρω βελτιστοποίησης της κατανάλωσης ενέργειας, ενδεχομένως με μείωση των ταχυτήτων ρολογιού κατά τη διάρκεια περιόδων αδράνειας, με χρήση παραλλαγών FPGA χαμηλότερης ισχύος ή με δυναμική προσαρμογή της χρήσης των πόρων ανάλογα με το φορτίο του δικτύου. Η ακριβής σκιαγράφηση της κατανάλωσης ενέργειας στα διάφορα στάδια της συμπερασματολογίας του DNN θα μπορούσε να οδηγήσει σε ακόμη πιο αποδοτικούς σχεδιασμούς.

- Γενίκευση της προσέγγισης σε άλλα FPGA:

Ενώ η παρούσα διατριβή επικεντρώθηκε στο Cyclone II, μελλοντικές εργασίες θα μπορούσαν να εξετάσουν κατά πόσον οι βελτιστοποιήσεις και οι τεχνικές που αναπτύχθηκαν μπορούν να γενικευτούν σε άλλες οικογένειες και κατασκευαστές FPGA (π.χ. Xilinx ή Lattice FPGA). Αυτό θα καθιστούσε την εργασία εφαρμόσιμη σε ένα ευρύτερο φάσμα εφαρμογών και βιομηχανιών, καθώς οι διάφορες FPGA διαθέτουν διαφορετικά πλεονεκτήματα όσον αφορά την αποδοτικότητα ισχύος, τη διαθεσιμότητα πόρων και την ταχύτητα επεξεργασίας.

Με τη διερεύνηση αυτών των μελλοντικών κατευθύνσεων, μπορούμε να συνεχίσουμε να διευρύνουμε τα όρια των υλοποιήσεων DNN που βασίζονται σε FPGA, βελτιώνοντας τόσο την αποδοτικότητα όσο και την επεκτασιμότητα αυτών των συστημάτων. Τέτοιες εργασίες έχουν τη δυνατότητα να φέρουν επανάσταση στις εφαρμογές τεχνητής νοημοσύνης πραγματικού χρόνου, ιδίως σε τομείς όπου τα συστήματα χαμηλής καθυστέρησης και υψηλής απόδοσης είναι κρίσιμα.

6.3 Τελικές σκέψεις

Η υλοποίηση ενός νευρωνικού δικτύου στο Cyclone II FPGA αναδεικνύει τη διασταύρωση του σχεδιασμού υλικού και της μηχανικής μάθησης. Το έργο αυτό αναδεικνύει τη δυνατότητα μεταφοράς της βαθιάς μάθησης σε περιβάλλοντα με περιορισμένους πόρους, όπου οι παραδοσιακές CPU ή GPU μπορεί να μην είναι βιώσιμες. Καθώς οι FPGA συνεχίζουν να εξελίσσονται, ο ρόλος τους στη μηχανική μάθηση είναι πιθανό να επεκταθεί, προσφέροντας νέες δυνατότητες για την ανάπτυξη ευφυών συστημάτων σε ένα ευρύ φάσμα εφαρμογών. Η παρούσα εργασία χρησιμεύει ως βάση για την περαιτέρω διερεύνηση της βαθιάς μάθησης με βάση FPGA και τα διδάγματα που αντλήθηκαν εδώ θα ενημερώσουν μελλοντικά έργα που στοχεύουν να διευρύνουν τα όρια του τι είναι δυνατό να γίνει με νευρωνικά δίκτυα που επιταχύνονται με υλικό.

Βιβλιογραφία

[GBC16] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. Available at https://www.deeplearningbook.org/.

[Hay99] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation* (2nd ed.). Prentice Hall. Available at https://archive.org/details/neuralnetworksco0000hayk_2ed/page/n7/mode/2up.

[RHW86] Rumelhart, D.E., Hinton, G.E., & Williams, R.J. (1986). Learning representations by back-propagating errors. *Nature*. Available at https://www.nature.com/articles/323533a0

[BM94] Bourlard, H.A., & Morgan, N. (1994). *Multilayer Perceptrons*. In: *Connectionist Speech Recognition*. Springer, Boston. Available at https://link.springer.com/chapter/10.1007/978-1-4615-3210-1_4

[AH18] Amano, H. (2018). *Principles and Structures of FPGAs*. Springer Singapore. Available at https://link.springer.com/book/10.1007/978-981-13-0824-6

[LW15] Liu, H., & Wu, J. (2015). FPGA-Based Reconfigurable Computing: A Survey. *IEEE Transactions on Computers, 64*(8), 2232-2245.

[IC18] Intel Corporation. (2017). *Cyclone II Device Handbook*. Available at https://www.intel.com/

[AC12] Altera Corporation. (2012). DE2 Development and Education Board User Manual. Available at https://www-ug.eecg.utoronto.ca/desl/manuals/DE2_User_Manual.pdf