ΕΘΝΙΚΟ ΜΕΤΣΟΒΕΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ. ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Υπολογιστών και Υπολογιστών
Εργαστήριο Μικροϋπολογιστών και Ψηφιακών Συστημάτων

Σχεδιασμός Ενσωματωμένων Συστημάτων
9ο Εξάμηνο ΗΜΜΥ

3η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

Εισαγωγική εργασία στο High Level Synthesis

Ο σκοπός της άσκησης είναι να γίνει εισαγωγή στον προγραμματισμό Field Programmable Gate Arrays (FPGA) με High Level Synthesis (HLS). Η άσκηση αυτή παρουσιάζει ένα Recommendation System (RS) για ταινίες βασισμένο στον αλγόριθμο K Nearest Neighbors (KNN), χρησιμοποιώντας ένα υποσύνολο του MovieLens dataset. Με τη βοήθεια του HLS θα επιταχύνετε τον χρόνο εκτέλεσης του RS, χρησιμοποιώντας το περιβάλλον ανάπτυξης SDSoC 2016.4. Τέλος, θα εκτελέσετε την εφαρμογή στο Zynq-7000 ARM/FPGA SoC (Zybo). Η εφαρμογή βρίσκεται στο Github link. Πέρα από τον κώδικα της άσκησης εκεί θα βρείτε το jupyter notebook το οποίο χρησιμοποιήθηκε για την κατανόηση των δεδομένων εισόδου και τη δημιουργία του dataset καθώς και μια υλοποίηση του αλγορίθμου μόνο για CPU με στόχο να πειραματιστείτε (αν το επιθυμείτε) με διαφορετικές μετρικές απόστασης για τον αλγόριθμο.

Βοηθητικό Υλικό

Xilinx SDSoC 2016.4 (link): Περιβάλλον για να αναπτύξετε την εφαρμογή ώστε να τρέξει στο Zybo. Οδηγίες για την εγκατάσταση του Xilinx SDSoC 2016.4 υπάρχουν στη σελίδα του μαθήματος.

Σχετικά με την Εφαρμογή

Όπως αναφέρθηκε παραπάνω, η εφαρμογή αφορά ένα RS για ταινίες βασισμένο στον αλγόριθμο KNN, χρησιμοποιώντας ένα υποσύνολο του MovieLens dataset. Κάθε γραμμή του dataset (εκτός της πρώτης) ξεκινάει με το id της ταινίας ενώ ακολουθούν οι βαθμολογίες των διαφόρων χρηστών. Η βαθμολογία είναι ένα αριθμός από το σύνολο:

{ 0, 0.5, 1, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0 }

Η εφαρμογή δέχεται ως είσοδο τμήμα του dataset και προτείνει τις 10 πιο συναφείς ταινίες για την ταινία με id 0 (Toy Story του 1995). Μετά την εκτέλεση θα φανεί ο μέσος χρόνος εκτέλεσης σε κύκλους για τη software (SW) και τη hardware (HW) υλοποίηση του computational intensive τμήματος της εφαρμογής (υπολογισμός ευκλείδειων αποστάσεων) καθώς και το speedup. Σκοπός είναι να μεγιστοποιήσετε το speedup με HLS optimizations στη HW υλοποίηση. Ενδεικτικό παράδειγμα εκτέλεσης της εφαρμογής στο Zybo (το .elf είναι το ARM εκτελέσιμο).

```
root@Avnet-Digilent-ZedBoard-2016_3:/mnt# ./Embedded-Systems-2021-HLS-01-01.elf
Started reading dataset...
Finished reading dataset...
Started reading name id mapping...
Finished reading name id mapping...
Input movie id = 0
Started distance calculations on software...
Finished distance calculations on software...
Started distance calculations on hardware...
Finished distance calculations on hardware...

Recommendation system start to make inference
...
Recommendations for movie with id 0:
0. Jurassic Park (1993), with distance of 10.7121
1. Fish Called Wanda  A (1988), with distance of 11.0793
2. Back to the Future (1985), with distance of 11.6404
3. Star Wars: Episode VI - Return of the Jedi (1983), with distance of 11.8849
4. Lion King  The (1994), with distance of 12.0623
5. Raising Arizona (1987), with distance of 12.1552
6. Wizard of Oz  The (1939), with distance of 12.1861
7. Batman (1989), with distance of 12.2066
8. Princess Bride  The (1987), with distance of 12.3085
9. E.T. the Extra-Terrestrial (1982), with distance of 12.3996

Hardware cycles : 3263413
Software cycles : 1363445
Speedup         : 0.417797
Correct = 1024, Score = 1.000000
```

Αρχεία Εφαρμογής

main.cpp Τρέχει σε ARM CPU. Καλεί τις συναρτήσεις που υπολογίζουν τις ευκλείδειες αποστάσεις σε SW και HW και μετράει την επίδοση.

calcDist.cpp Είναι το computational intensive τμήμα της εφαρμογής. Η συνάρτηση αυτού του αρχείου θα πρέπει να βελτιστοποιηθεί με HLS directives για να τρέξει στο HW.

calcDist.h Περιέχει definitions καθώς και τον τρόπο με τον οποίο θα μεταφερθούν τα δεδομένα στη συνάρτηση στο HW.

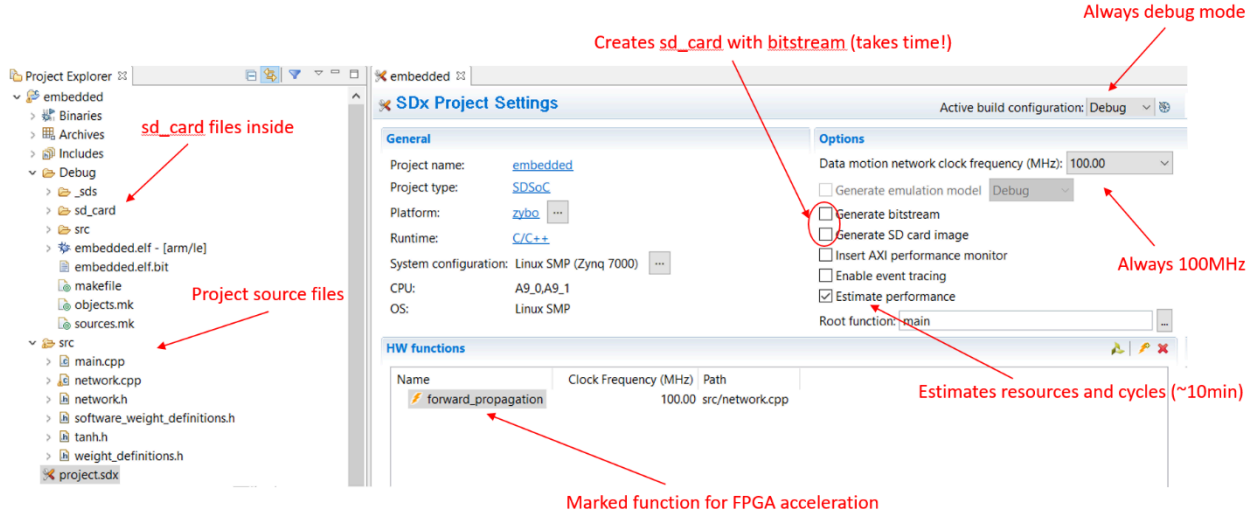
dataset.csv Περιέχει το input dataset.

nameIdMapping.csv Περιέχει την αντιστοίχιση των ids των ταινιών με τα ονόματά τους.

Χρήση του εργαλείου SDSoC 2016.4

Μέσω του εργαλείου SDSoC θα μπορέσετε να αναπτύξετε την εφαρμογή για το Zybo FPGA. Συγκεκριμένα το SDSoC παρέχει ένα Eclipse τύπου περιβάλλον όπου μπορείτε να γράφετε C/C++ κώδικα και να αναπτύξετε έναν accelerator που θα τρέξει σε FPGA με χρήση HLS optimizations. Επίσης μπορείτε να κάνετε performance και resource estimation, να φτιάξετε το bitstream του υλικού αλλά και τελικώς την sd boot card όπου θα "bootάρει" το σύστημα (τα παραγόμενα αρχεία μεσα στην sd_card περιλαμβάνουν το εκτελέσιμο (.elf), το bitstream και το λειτουργικό petalinux του συστήματος).

SDSoC project



Αφού φτιάξετε ένα Xilinx SDx project σύμφωνα με το πάνω configuration (zybo, linux, debug mode, 100 MHz) και εισάγετε τα source files μαζί με τα δεδομένα που σας δίνονται, μαρκάρετε την συνάρτηση που θέλετε να υλοποιηθεί στο HW. Στη συγκεκριμένη περίπτωση θα είναι η *calcDistancesHW* η οποία στο τέλος θα έχει όλα τα HLS optimizations που θα πρέπει να προσθέσετε. Τέλος, κάνετε build το project κάθε φορά για να υλοποιηθεί αυτό που επιθυμείτε.

Σημαντικές Παρατηρήσεις

1. Δε θα χρειαστεί να κάνετε κάποια αλλαγή στο εκτελέσιμο που θα τρέξει στον ARM (main.cpp).
2. Πριν φτιάξετε bitstream θα πρέπει να κάνετε estimate performance ώστε να δείτε τα resources αν χωράνε στο zybo αλλά και τον αριθμό κύκλων του HW για να αποφανθείτε για το estimated performance.
3. Όταν θα θέλετε να φτιάξετε νέο bitstream καλό θα είναι να κάνετε clean ή ακόμα και να σβήνετε το προηγούμενο debug folder πριν κάνετε νέο build.
4. Για να φτιάξετε sd με bitstream θα τικάρετε μαζί τις 2 επιλογές όπως φαίνεται στη φωτογραφία. Να γνωρίζετε ότι ο χρόνος εξαγωγής bitstream μπορεί να φτάσει τα 45min.
5. Τέλος, όταν είναι έτοιμα τα αρχεία της sd_card θα πρέπει να μεταφέρετε ό,τι έχει μέσα όπως και τα δεδομένα στο zybo σας και να αντικαθιστάτε τα παλιά αρχεία στην zybo sd_card με τα νέα. Έπειτα αρκεί να εκκινήσετε το board. Το περιεχόμενο της sd_card βρίσκεται στο /mnt. Μόνο τα αρχεία στην sd παραμένουν μετά από κάθε reboot, τα αρχεία στο Petalinux OS δεν είναι persistent! (πχ. home/)
6. Για τη σύνδεσή σας στα boards προτείνεται η χρήση του προγράμματος minicom. **Μόνο για χρήστες Linux:** εναλλακτική λύση είναι και η χρήση της πλατφόρμας **screen**, την οποία (στα Debian based Linux, Ubuntu κτλ.) κάνετε εγκατάσταση με `sudo apt install screen` και για να μπορέσετε να κάνετε συνδεθείτε σε κάποιο UART device η μέση εκτέλεση μοιάζει κάπως έτσι – `sudo screen /dev/ttyUSBx <baud_rate>` - ενώ παραπάνω documentation μπορείτε να βρείτε εδώ.

Ασκήσεις

Η συνάρτηση calcDistancesHW είναι ο κώδικας C που υπολογίζει τις ευκλείδιες αποστάσεις μίας ταινίας από όλες υπόλοιπες του dataset. Συγκεκριμένα, φέρνει τα δεδομένα στο FPGA, εκτελεί τον υπολογισμό και τέλος επιστρέφει τα δεδομένα πίσω στον host.

1. Α) Δοκιμάστε να την θέσετε ως HW function και κάντε Estimate performance χωρίς να κάνετε κάποιο optimization. Καταγράψτε το report που δείχνει τα estimated resources και clock cycles της HW function καθώς και τα αποτελέσματα του HLS report για κάθε loop (Latency Detail Loop). Ποιό loop απαιτεί τους περισσότερους κύκλους;
   Β) Δοκιμάστε τώρα να φτιάξετε την sd_card με το bitstream και να τα περάσετε στην sd του zybo. (Αγνοείστε πιθανά warnings κατά το compilation (unused labels, κτλ)). Μη ξεχάσετε να περάσετε και τα δεδομένα εισόδου. Κάντε reboot και τρέξτε την εφαρμογή στο board. Σε πόσους κύκλους τρέχει; Συμφωνεί με το estimation; Υπάρχει speedup σε σχέση με την SW εκτέλεση στον ARM;
   Γ) Σε αυτό το σημείο καλείσθε να κάνετε optimizations ώστε να επιταχυνθεί ο αλγόριθμος. Δοκιμάστε διάφορα HLS directives και δείτε το αποτέλεσμα του estimation. Παράλληλα, να βλέπετε τα HLS reports ώστε να ελέγχετε αν το design σας είναι pipelined (όλα τα loops να έχουν II = 1). Μπορείτε να πετύχετε pipelined design; Τί προβλήματα αντιμετωπίζετε; Αν σας ικανοποιεί το αποτέλεσμα παράξτε την sd με το bitstream και τρέξτε την εφαρμογή στο board. Καταγράψτε ξανά κύκλους και speedup από το board. Συγκρίνετε αυτή την υλοποίηση με την unoptimized.
2. Στο σημείο αυτό θα χρησιμοποιήσουμε τη βιβλιοθήκη ap_fixed με στόχο να δημιουργήσουμε custom datatypes και να βελτιώσουμε το design μας. Στο αρχείο calcDist.h βρίσκεται ο ορισμός του τύπου DTYPE1 για τον οποίο πρέπει να βρείτε τις βέλτιστες τιμές για το ακέραιο και το δεκαδικό του τμήμα, αλλάζοντας τις τιμές των μεταβλητών INT_BITS και DEC_BITS (ποιες είναι οι τιμές αυτές θεωρητικά;). Μόλις εντοπίσετε τις βέλτιστες τιμές κάντε Estimate performance προσθέτοντας και τα HLS directives που είχατε βάλει στο προηγούμενο ερώτημα και καταγράψτε τα αποτελέσματα. Στη συνέχεια παράξτε την sd_card με το bitstream και εκτελέστε στο FPGA. Συγκρίνετε αυτή την υλοποίηση με την optimized του προηγούμενου ερωτήματος.
3. Δοκιμάστε να βελτιστοποιήσετε περαιτέρω την έκδοση της προηγούμενης άσκησης. Μπορείτε να πετύχετε τώρα pipelined design; Αν σας ικανοποιεί το αποτέλεσμα παράξτε την sd με το bitstream και τρέξτε την εφαρμογή στο board. Καταγράψτε το speedup. Παρατηρείτε κάποια αύξηση στο speedup; Εξηγείστε.

Για την εργασία αυτή θα πρέπει να παραδώσετε την αναφορά με όσα σας ζητούνται στα παραπάνω ερωτήματα καθώς και τα αρχεία calcDist.cpp και calcDist.h που προκύπτουν στο τέλος κάθε ερωτήματος.