Ομάδα 15

ΣΧΕΔΙΑΣΜΟΣ ΕΝΣΩΜΑΤΩΜΕΝΩΝ ΣΥΣΤΗΜΑΤΩΝ

Μαρία Αρετή Γερμανού 57807
Διαλεχτή Πιστιόλη 57859

Περιγραφή εξισορρόπησης ιστογράμματος με βάση την μέση τιμή

- 1ο Μέρος Εργασίας: Μετρήσεις και μέθοδοι βελτιστοποίησης μεταφοράς και αποθήκευσης δεδομένων στον αλγόριθμο.
- 2ο Μέρος Εργασίας: Επεξεργασία δομής μνήμης.
- 3ο Μέρος Εργασίας: Επαναχρησιμοποίηση δεδομένων.

Μετρήσεις και μέθοδοι βελτιστοποίησης μεταφοράς και αποθήκευσης δεδομένων στον αλγόριθμο.

Σκοπός της εργασίας είναι να υλοποιηθεί σε γλώσσα C ο προτεινόμενος αλγόριθμος και στη συνέχεια να εφαρμοστεί η μεθοδολογία βελτιστοποίησης μεταφοράς και αποθήκευσης δεδομένων στον αλγόριθμο αυτό. Κατά την εκτέλεση του προγράμματος εμφανίζεται το ιστόγραμμα των εικονοστοιχείων έπειτα από έλεγχο και εύρεση μέσης τιμής, καταλήγοντας στην εξισορρόπηση ιστογράμματος.

ΒΗΜΑΤΑ ΕΚΤΕΛΕΣΗΣ ΑΡΧΙΚΟΥ ΚΩΔΙΚΑ

- **Βήμα 1ο: Σάρωση της εικόνας από αριστερά προς τα δεξιά και από πάνω προς τα κάτω.**
- **Βήμα 2ο: Υπολογισμός αρχικού ιστογράμματος.**
- **Βήμα 3ο: Υπολογισμός μέσης τιμής εικόνας.**
- **Βήμα 4ο: Εφαρμογή εξισορρόπησης στο πρώτο κομμάτι του ιστογράμματος.**
- **Βήμα 5ο: Εφαρμογή εξισορρόπησης στο δεύτερο κομμάτι του ιστογράμματος.**
- **Βήμα 6ο: Μετασχηματισμός της αρχικής εικόνας με βάση το νέο ιστόγραμμα.**

ΕΠΕΞΗΓΗΣΗ ΔΕΔΟΜΕΝΩΝ ΜΕΤΡΗΣΕΩΝ

Code: Δείχνει πόσα byte καταλαμβάνει ο κώδικας.

RO-Data: Δείχνει πόσα byte καταλαμβάνονται από δεδομένα μόνο για ανάγνωση.

RW-Data: Δείχνει πόσα byte καταλαμβάνονται από δεδομένα ανάγνωσης-εγγραφής.

ZI-Data: Δείχνει πόσα byte καταλαμβάνονται από μηδενικά αρχικοποιημένα δεδομένα.

Debug: Δείχνει πόσα byte καταλαμβάνονται από δεδομένα εντοπισμού σφαλμάτων, για παράδειγμα, εισαγωγή εντοπισμού σφαλμάτων τμήματα και τον πίνακα συμβόλων και συμβολοσειρών.

ΜΕΤΡΗΣΕΙΣ ΑΡΧΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Πρόγραμμα "original.c"

Grand Totals				
Total RO Size		Total RW Size		Debug
Code	RO-Data	RW-Data	ZI-Data	
17436	518	0	2953620	13620
Total ROM Size				

ΕΠΕΞΗΓΗΣΗ ΔΕΔΟΜΕΝΩΝ ΜΕΤΡΗΣΕΩΝ

Instructions: Δείχνει το σύνολο των εντολών

Core cycles: Δείχνει τους βασικούς κύκλους.

S_cycles: Δείχνει τους διαδοχικούς κύκλους. Ο διαδοχικός κύκλος απαιτεί τη μεταφορά από ή προς μια διεύθυνση που είναι είτε το ίδιο, μια λέξη, ή μια μισή λέξη μεγαλύτερη από την διεύθυνση που χρησιμοποιήθηκε στον προηγούμενο κύκλο

N_cycles: Δείχνει τους μη διαδοχικούς κύκλους. Ο μη διαδοχικός κύκλος απαιτεί μεταφορά από ή προς μια διεύθυνση που δεν σχετίζεται με τη διεύθυνση που χρησιμοποιήθηκε στον προηγούμενο κύκλο.

I_cycles: Δείχνει τους εσωτερικούς κύκλους. Ο εσωτερικός κύκλος δεν απαιτεί μεταφορά επειδή εκτελεί μια εσωτερική λειτουργία και καμία χρήσιμη προ-ανάκτηση δεν μπορεί να πραγματοποιηθεί την ίδια ώρα.

Wait states: Κύκλοι που απαιτούνται να γίνουν κλήσεις στη μνήμη.

ΜΕΤΡΗΣΕΙΣ ΑΡΧΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Πρόγραμμα "original.c"

Προτιμάμε να έχουμε όσο το δυνατόν λιγότερους μή διαδοχικούς κύκλους γίνεται σε σχέση με τους διαδοχικούς.

Statistics							
Instructions	Core cycles	S_cycles	N_cycles	I_cycles	Total	Cpu-time	Sys clock
428240399	893266344	476490409	313117298	104421657	894029364	894029364	8769

ΜΕΘΟΔΟΙ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΜΕΤΑΦΟΡΑΣ ΚΑΙ ΑΠΟΘΗΚΕΥΣΗΣ ΔΕΔΟΜΕΝΩΝ

Αρχικές μέθοδοι που εφαρμόστηκαν ξεχωριστά στο πρόγραμμα:

- Loop Fusion and reversal
- Loop Interchange
- Loop Unrolling

Συνδυασμός μεθόδων που εφαρμόστηκαν στο πρόγραμμα:

- Fusion revearsal and Interchange
- Fusion reversal and Unrolling
- Unrolling and Interchange

LOOP FUSION

Συγχωνεύουμε τις γειτονικές λούπες σε μια λούπα για να μειωθεί η επιβάρυνση της μνήμης και να βελτιωθεί η απόδοση του χρόνου εκτέλεσης.

```
int main(){
  read();
  h(current_y);
  printhist(hist);
  mean(hist);
  check(current_y);
  write();
}
```

```
int main(){
  read();
  printhist_mean
(hist,current_y);
  check(current_y);
  write();
}
```

LOOP REVERSAL

Μέθοδος που αντιστρέφει και φθήνει την επανάληψη με αρνητικό βήμα. Μπορεί να χαρακτηριστεί ως μειωτική επανάληψη.

```
void printhist_mean(int histogram[256],int
channel[N][M])
{
  for(k=255;k>=0;k--)
  {
    for(i=N-1;i>=0;i--)
    {
      for(j=M-1;j>=0;j--)
```

LOOP INTERCHANGE

Ανταλλάσσει τη σειρά εκτέλεσης από δυό λούπες. Πραγματοποιείται αλλαγή της σειράς προσπέλασης των δύο επαναλήψεων σε περισσότερα “sequential” αντι για “non sequential” στοιχεία. Έτσι βελτιώνεται ο χρόνος προσπέλασης όταν τα στοιχεία βρίσκονται σε διαδοχικές μνήμες. Ανάλογα με τον debugger η μνήμη εγγράφεται με διαφορετικό τρόπο και σειρά.

```
for(i=0;i<N;i++)
{
  for(j=0;j<M;j++)
  {
     current_y[i][j]=
     fgetc(frame_c);
  }
}
```

```
for(j=0;j<M;j++)
{
  for(i=0;i<N;i++)
  {
     current_y[i][j]=
     fgetc(frame_c);
  }
}
```

LOOP UNROLLING

Είναι μια τεχνική μετασχηματισμού λούπας που βοηθά στη βελτιστοποίηση του χρόνου εκτέλεσης ενός προγράμματος αυξάνοντας την μνήμη που καταλαμβάνει ο κώδικας. Βασικά μειώνουμε τις επαναλήψεις ξεδιπλώνοντας εντολές της επανάληψης και μειώνοντας τον αριθμό βημάτων των επαναλήψεων. Συγκεκριμένα γίνεται εγγραφή περισσότερων εντολών για την εκτέλεση λιγότερων επαναλήψεων με βήμα ακέραιου πολλαπλάσιου με τον αριθμό των επαναλήψεων. Έτσι αποφεύγουμε το index out of range error.

```
for(j=0;j<M;j++)
{
current_y[i][j]=fgetc(frame_c);
}
```

```
for(j=0;j<M;j=j+4)
{
current_y[i][j]=fgetc(frame_c);
current_y[i][j+1]=fgetc(frame_c);
current_y[i][j+2]=fgetc(frame_c);
current_y[i][j+3]=fgetc(frame_c);
}
```

ΣΥΝΔΥΑΣΜΟΙ ΠΡΟΓΡΑΜΜΑΤΩΝ

Θα συνδυάσουμε τις μεθόδους ανά δύο για να παρατηρήσουμε τα αποτελέσματα πραγματοποιώντας παραπάνω από μια βελτιστοποίηση.

- Fusion_revearsal_and_Interchange
- Fusion_reversal_and_Unrolling
- Unrolling_and_Interchange

ΣΥΝΟΛΙΚΕΣ ΜΕΤΡΗΣΕΙΣ

Αρχείο Κώδικα	Total (Statistics)	System Clock	Total ROM
original	894029364	8769	17.53kB
improved_fusion_reversal	894043357	10171	17.51kB
improved_interchange	988635162	7254	17.53kB
improved_unrolling	544199295	6810	19kB
improved_fusion_reversal_and_interchange	988647917	19772	17.52kB
improved_fusion_reversal_and_unrolling	570584468	3174	19.75kB
improved_interchange_and_unrolling	580052068	11601	19.73kB

ΣΥΜΠΕΡΑΣΜΑ

Η μέθοδος fusion reversal δεσμεύει λιγότερη μνήμη. Όταν συνδυάζουμε την μέθοδο fusion reversal με την μέθοδο "unrolling" η μνήμη που δεσμεύεται αυξάνεται λόγω της ξεδίπλωσης. Άρα, για την υλοποίηση του δεύτερου μέρους της εργασίας θα επιλέξουμε συνδυασμό "fusion reversal unrolling" καθώς μας δίνει τα καλύτερα αποτελέσματα.

Επεξεργασία δομής μνήμης.

Για το δεύτερο εργαστήριο θα επιλέξουμε τον βελτιστοποιημένο κώδικα Fusion Reversal Unrolling αφού έχουμε καλύτερα αποτελέσματα στο System Clock και τα δεύτερα καλύτερα στα συνολικά Statistics. Εφαρμόζονται τα αρχεία δομής μνήμης που δίνονται στο αρχικό πρόγραμμα και μετά τα αλλάζω σύμφωνα με τον επιλεγμένο και βελτιστοποιημένο κώδικα, για να επαναδομήσω την μνήμη. Παρουσιάζονται τελικές μετρήσεις με αλλαγές στην αποθήκευση μεταβλητων και επαναληπτών του προγράμματος.

ΑΡΧΕΙΑ ΔΟΜΗΣ ΜΝΗΜΗΣ

SCATTER FILE

Σύμφωνα με το αντίστοιχο αρχείο εργαστηρίου:

- Name of region: ROM
- Start address for load region: 0x0
- Maximum size of load region: 0x080000
- Name of first execution region: ROM
- Start address for execution region: 0x0
- Maximum size of execution region: 0x80000
- Name of second execution region: SRAM
- Start address for execution region: 0x80000
- Maximum size of execution region: 0x8000000

MEMORY MAP

Σύμφωνα με το αντίστοιχο αρχείο εργαστηρίου:

- Start address: 00000000
- Size: 00080000
- Name: ROM
- Use of memory: R
- Bandwidth (Bytes): 4
- Read Times , Write Times: 1/1 , 1/1
- Start address: 00080000
- Size: 08000000
- Name: RAM
- Use of memory: RW
- Bandwidth (Bytes): 4
- Read Times , Write Times: 250/50 , 250/50

MEMORY VIEWS

Load View: Προβολή μνήμης κατά την πρώτη φόρτωση του προγράμματος και των δεδομένων.

Execution View: Προβολή μνήμης αφού ο κώδικας μετακινηθεί στην κανονική θέση εκτέλεσής του.

	Load View	Execution View
RAM		
RAM		ZI
RAM		RW
ROM	RW	
ROM	RO	RO

ΜΕΤΡΗΣΕΙΣ ΑΡΧΙΚΟΥ ΑΡΧΕΙΟΥ

Πραγματοποιούμε μετρήσεις στο αρχικό μας αρχείο με τα υπάρχοντα αρχεία του 2ου εργαστηρίου πριν την αλλαγή "original.c". Πλέον έχουμε scatter file memory map και “stack.c” αρχείο όπως δίνεται.

Total RO Size		Total RW Size		Total ROM Size	
17934 (17.51kB)		3692692 (3606.14kB)		17934 (17.51kB)	
RO Data	498	ZI Data	3692692	Code	15564

ΣΥΝΟΛΙΚΕΣ ΜΕΤΡΗΣΕΙΣ

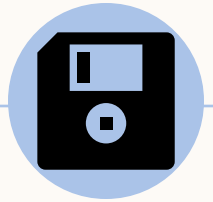
Στοιχεία επιλεγμένου προγράμματος:
"improved_fusion_revearsal_and_unrolling".Χρήση scatter file και memory map χωρίς “stack.c” αρχείο.

Total RO Size		Total RW Size		Total ROM Size	
20226 (19.97kB)		2953620 (2884.39kB)		20226 (19.97kB)	
RO Data	518	ZI Data	3692692	Code	19708

Ανάλυση αρχείων “stack” & “heap”:

Στην στατική τύπου μνήμη αποθηκεύονται οι τοπικές μεταβλητές κατα το compilation του αρχείου, έχουν καθορισμένο μέγεθος, είναι εύκολα προσβάσιμες και δεν χρειαζεται να ελευθερωθούν οι θέσεις μνήμης πριν τον τερματισμό του προγράμματος. Ενώ αντίθετα, στην δυναμική τύπου μνήμη αποθηκεύονται pointers που δείχνουν στις θέσεις μνήμης που έχουν δεσμευτεί για να αποθηκευτούν δεδομένα, είναι τύπου global, δεν έχουν καθορισμένο μέγεθος και λόγω των pointers δεν είναι εύκολα προσβάσιμες.

ΕΠΕΞΕΡΓΑΣΊΑ ΔΟΜΉΣ ΜΝΗΜΗΣ



ROM

- Η μνήμη ROM είναι μνήμη μόνο ανάγνωσης.
- Τα περιεχόμενα της οποίας δε μεταβάλλονται.
- Η ταχύτητα μεταφοράς δεδομένων από μια μνήμη ROM είναι μικρότερη από αυτή μιας μνήμης RAM.

SRAM

- Είναι σχετικά ταχύτερη από άλλους τύπους RAM.
- Καταναλώνει λιγότερη ενέργεια και έχει μικρότερη χωρητικότητα.
- Το SRAM μπορεί να διατηρεί τα δεδομένα για όσο διάστημα τροφοδοτείται με ενέργεια.

DRAM

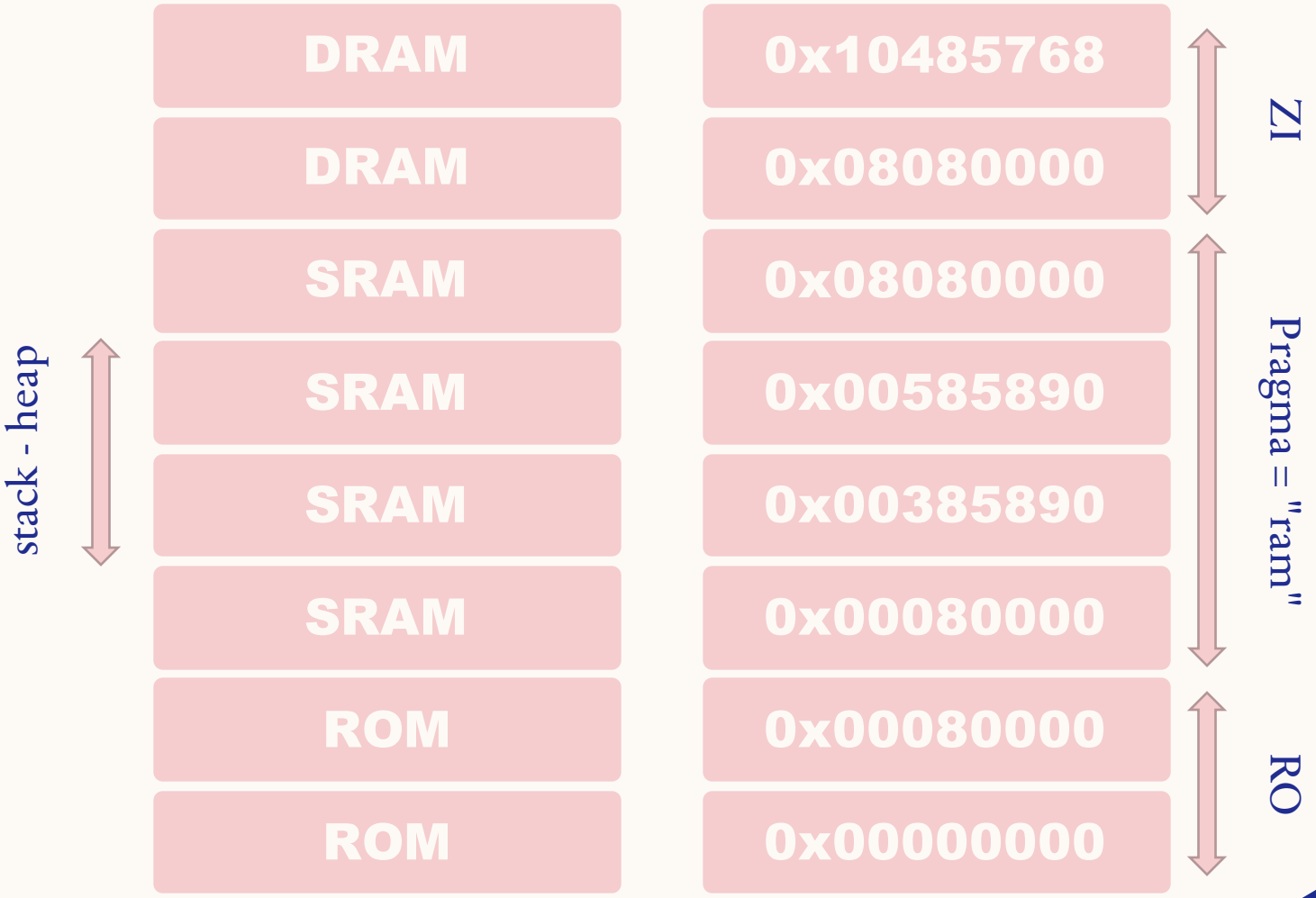
- Η DRAM είναι αργή.
- Καταναλώνει περισσότερη ισχύ από την SRAM και έχει μεγαλύτερη χωρητικότητα.
- Χάνει τα δεδομένα όταν σταματήσει η τροφοδότηση.

ΕΠΕΞΕΡΓΑΣΙΑ ΔΟΜΗΣ ΜΝΗΜΗΣ

Στοιχεία επιλεγμένου προγράμματος:
"improved_fusion_revearsal_and_unrolling".Χρήση scatter file και memory map, πλέον έχουμε προσθέσει και το “stack.c” αρχείο:

Total RO Size		Total RW Size		Total ROM Size	
20194 (19.72kB)		3692692 (3606.14kB)		20194 (19.72kB)	
RO Data	518	ZI Data	3692692	Code	19676

ΣΧΗΜΑΤΙΚΗ ΔΟΜΗ ΜΝΗΜΗΣ



ΕΠΕΞΕΡΓΑΣΙΑ ΟΡΙΩΝ STACK & HEAP

- Σύμφωνα με την δομή της μνήμης που σχολιάστηκε παραπάνω, το διάστημα που μου δίνεται από το heap & stack base θα ξεκινάει από την θέση μνήμης με αριθμό **0x00260000**.
- Επειδή μέχρι το αρχείο stack χωράνε περίπου **2432kB** και ο κώδικας μου χρειάζεται τουλάχιστον **3606.14kB**, θα αναγκαστώ να μετακινήσω τα όρια του αρχείου stack για να μην υπάρχει επικάλυψη.
- Μετακινώ τα όρια σε **0x00385890** και **0x00585890** ώστε να τοποθετούνται ακριβώς μετά τον κώδικα που επιλέχθηκε.
- Από την θέση **0x00000000** μέχρι **0x00080000** θα πρέπει να αποθηκευτεί το αρχείο κώδικα που φαίνεται από το "Total RO Size".
- Το μέγεθος του κώδικα είναι περίπου **19.90kB** και χωράει στα **19.96kB** που έχω διαθέσιμα στην ROM. Οπότε δεν αλλάζω κάτι στην χωρητικότητα της ROM στο scatter file.
- Φτιάχνω ξεχωριστή DRAM για να αποθηκεύσω τα δεδομένα "ZI", τα οποία είναι μεγέθους περίπου **3606.14kB** και γράφω το επόμενο μέρος τέτοιας χωρητικότητας στο scatter file και memory map αντιστοιχα.

ΑΡΧΕΙΑ ΔΟΜΗΣ ΜΝΗΜΗΣ

SCATTER FILE

Σύμφωνα με το αντίστοιχο αρχείο που επεξεργαστήκαμε:

- Name of region: ROM
- Start address for load region: 0x0
- Maximum size of load region: 0x080000
- Name of first execution region: ROM
- Start address for execution region: 0x0
- Maximum size of execution region: 0x80000
- Name of second execution region: SRAM
- Start address for execution region: 0x80000
- Maximum size of execution region: 0x8000000

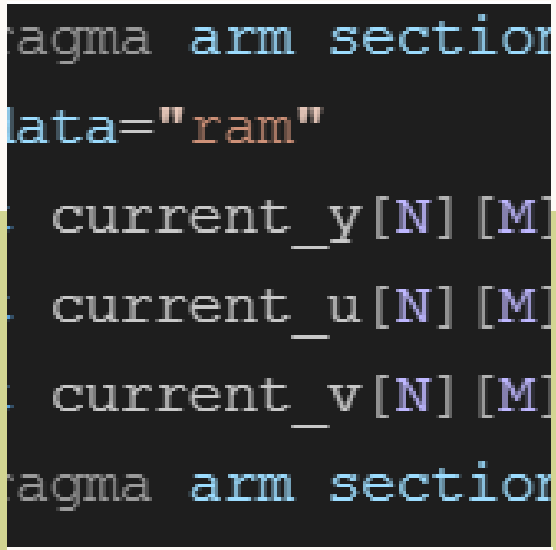
MEMORY MAP

Σύμφωνα με το αντίστοιχο αρχείο αρχείο που επεξεργαστήκαμε, τα χαρακτηριστικα της ROM παραμένουν ίδια:

- Start address: 00080000
- Size: 08000000
- Name: SRAM
- Use of memory: RW
- Bandwidth (Bytes): 4
- Read Times , Write Times: 1/1 , 1/1
- Start address: 08080000
- Size: 08405768
- Name: DRAM
- Use of memory: RW
- Bandwidth (Bytes): 4
- Read Times , Write Times: 250/50 , 250/50

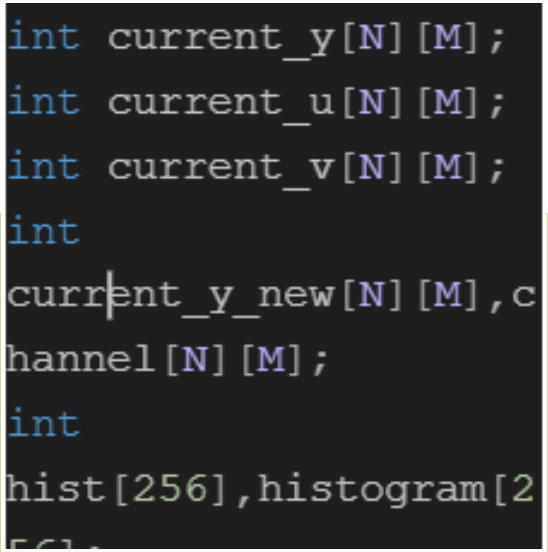
ΚΑΤΑΣΤΑΣΕΙΣ ΔΟΜΗΣ ΜΝΗΜΗΣ

Σε αυτό το στάδιο θα αλλάξουμε τα δεδομένα που θα βάζουμε στον κώδικα του armulator.



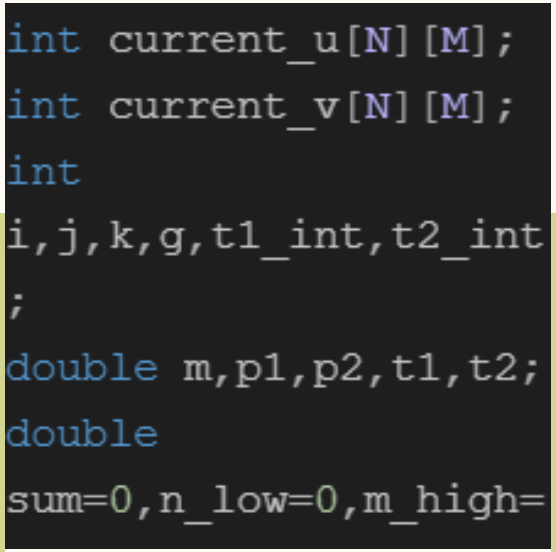
STATE

Default μεταβλητές στον armulator



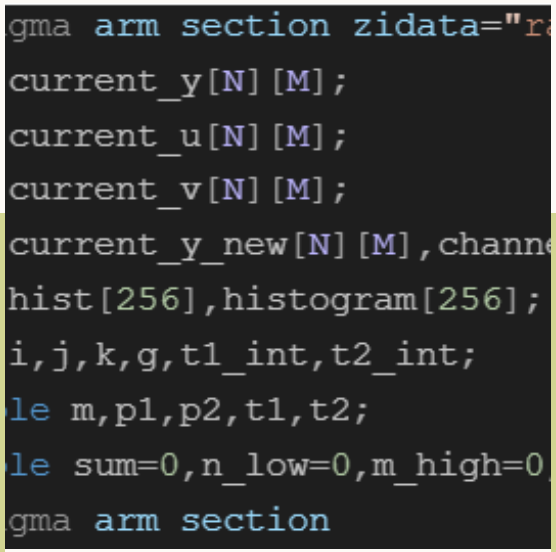
STATE 1

Μεταβλητές πίνακα στον armulator



STATE 2

Μεταβλητές επαναληπτών στον armulator



STATE 3

Όλες οι μεταβλητές στον armulator

ΠΕΡΙΓΡΑΦΗ ΚΑΤΑΣΤΑΣΕΩΝ ΔΟΜΗΣ ΜΝΗΜΗΣ

- State: Στην περιοχή του κώδικα για τον armulator υπάρχουν οι default μεταβλητες. Δηλαδή δεν έχει γίνει ακόμα καμία προσθήκη.
- State_1: Τοποθετώ όλες τις μεταβλητές που είναι μορφής πίνακα μέσα στην περιοχή.
- State_2: Τοποθετώ τους επαναλήπτες μέσα στην περιοχή και βγάζω τις μεταβλητές που είχα προσθέσει στην κατάσταση 2.
- State_3: Τοποθετώ όλες τις μεταβλητές στην περιοχή.

ΕΠΕΞΕΡΓΑΣΙΑ ΔΟΜΗΣ ΜΝΗΜΗΣ

Οι μετρήσεις ανάλογα με τα δεδομένα που θα βάζουμε στον κώδικα του armulator και της κάθε κατάστασης.

	Core Cycles	Wait States	Total
State	570935862	360651182	932344272
State 1	570935862	347349902	919042992
State 2	570937619	17268544	588963392
State 3	570937535	3967252	575662015

ΣΥΜΠΕΡΑΣΜΑ

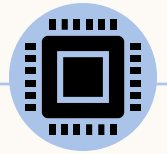
Φαίνεται πως η τέταρτη κατάσταση είναι πιο αποδοτική στα Wait States και στα συνολικά στατιστικά. Στόχος μας είναι να μειώσουμε τον χρόνο εκτέλεσης, ή κύκλων που απαιτούνται να γίνουν κλήσεις στη μνήμη, του βελτιστοποιημένου προγράμματος.

Μετασχηματισμοί Επαναχρησιμοποίησης Δεδομένων

Ο βασικός στόχος των μετασχηματισμών επαναχρησιμοποίησης δεδομένων είναι να μειωθούν οι περιττές προσπελάσεις στην μνήμη δεδομένων, εισάγοντας μικρότερου μεγέθους μνήμες, με σκοπό την μείωση των μεταφορών δεδομένων από και προς την μνήμη. Έτσι μειώνεται ο χρόνος εκτέλεσης.

ΙΕΡΑΡΧΙΑ ΜΝΗΜΗΣ ΣΕ ΕΠΙΠΕΔΑ

Παρακάτω φαίνεται η μεταφορά δεδομένων προς την μνήμη με τα επίπεδα μνημών που υπάρχουν πιο κοντά στον επεξεργαστή



CPU

CPU

LAYER 1

Buffer 1

LAYER 2

Buffer 2

LAYER 3

Buffer 3

ΠΡΟΒΛΗΜΑ

- Οι συναρτήσεις υπολογισμού ιστογράμματος και ελέγχου εξισορρόπησης με βάση την φωτεινότητα, δέχονται δισιάστατους πίνακες.
- Για τον υπολογισμό ιστογράμματος και μέσης τιμής είναι απαραίτητη η συνολική ανάγνωση και εγγραφή του πίνακα.
- Οι μετρητές στον έλεγχο κάθε εικονοστοιχείου αυξάνονται με βάση ένα κατώφλι συσχετιζόμενο με την μέση τιμή, άρα υπάρχει "dependency" μεταξύ αυτού και της μέσης τιμής.

ΣΥΜΠΕΡΑΣΜΑ

Άρα, για επαναχρησιμοποίηση δεδομένων πρέπει να αλλαχθεί όλος ο κώδικας και ο τρόπος που διαβάζει και γράφει τα στοιχεία από τους πίνακες. Κάτι που δεν μπορεί να συμβεί στην υπάρχουσα δομή του κώδικά μας.