3ο Εργαστήριο Αρχιτεκτονικής Η/Υ: Μετατροπή εικόνας από RGB888 σε RGB565

Α. Ευθυμίου

Παραδοτέο: Δευτέρα 7 Νοέμβρη, 23:59

Με αυτή την εργαστηριακή άσκηση θα γράψετε και θα ελέγξετε ένα λίγο πιο σύνθετο πρόγραμμα Risc-V assembly με τον Ripes. Θα πρέπει να έχετε μελετήσει μέχρι και το 4ο μάθημα για τη γλώσσα assembly του Risc-V (7η διάλεξη του μαθήματος), που αναφέρεται σε υπορουτίνες στον Risc-V.

1 Παραλαβή του σκελετού της άσκησης

Ακολουθήστε τον σύνδεσμο https://classroom.github.com/a/9AJztBkz για να δημιουργηθει το δικό σας αποθετήριο της άσκησης και κλωνοποιήστε το.

Στον κατάλογο που θα δημιουργηθεί, `lab03-ghUsername`, θα βρείτε το αρχείο `lab03.s`, με έναν μικρό σκελετό κώδικα και και μια δοκιμαστική εικόνα για είσοδο, καθώς και το `Lab03Test.py`, για έλεγχο που περιέχει αρκετές περιπτώσεις εισόδων για έλεγχο.

2 Αναπαράσταση εικόνων σε υπολογιστές

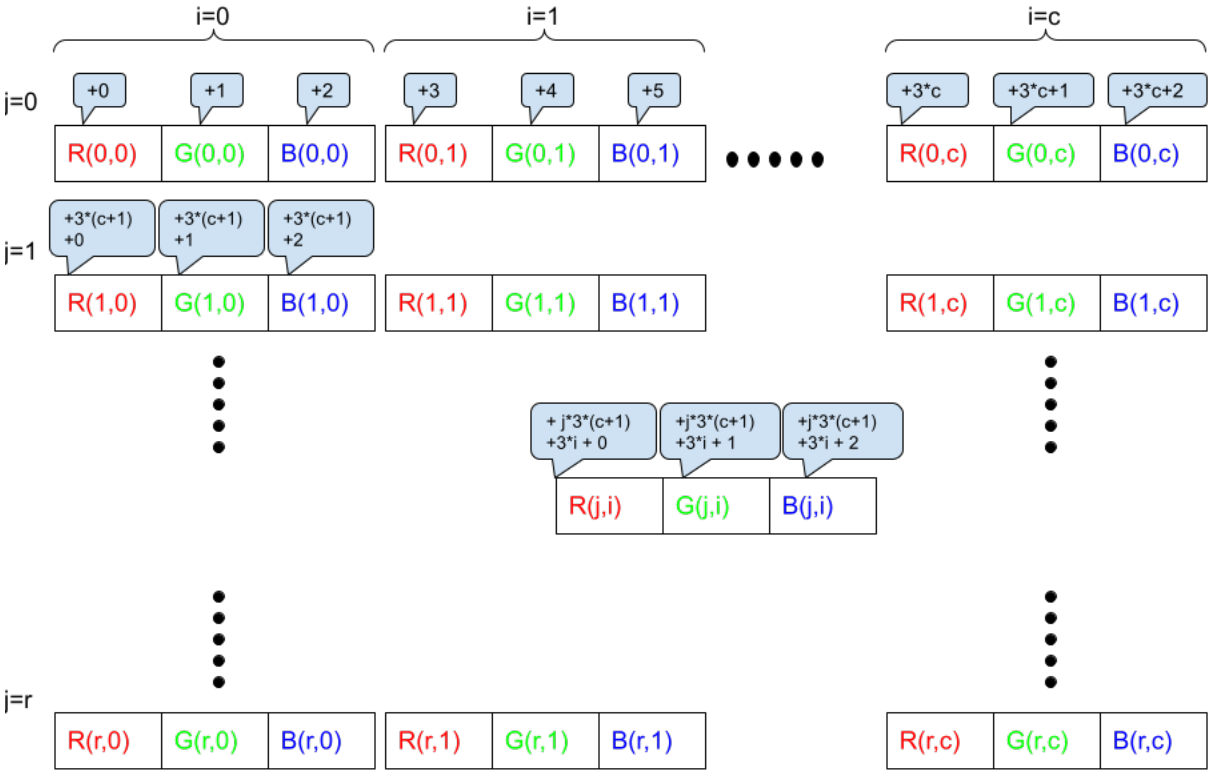
Μια πολύ συνηθισμένη μορφή αποθήκευσης εικόνων σε υπολογιστές είναι η RGB888 όπου κάθε pixel παριστάνεται από μια τριάδα από bytes, με το πρώτο να αντιστοιχεί στην τιμή, ένταση-φωτεινότητα, του κόκκινου (Red), το δεύτερο στην τιμή του πράσινου (Green) και το τρίτο στην τιμή του μπλέ (Blue). Η παραπάνω σειρά είναι σημαντική και από αυτήν παίρνει το πρώτο συστατικό του το όνομα της μορφής RGB888. Κάθε ένα από αυτά τα τρία βασικά χρώματα μπορεί να πάρει τιμές από 0 (δεν υπάρχει καθόλου το χρώμα αυτό) έως 255 (το χρώμα έχει την πιο έντονη τιμή του). Έτσι ο συνδιασμός τους μπορεί να σχηματίσει 2^{24} διαφορετικά χρώματα για κάθε pixel. Για παράδειγμα το απόλυτο μαύρο είναι η τριάδα 0,0,0, το λευκό: 255,255,255, το σκέτο, έντονο, κίτρινο: 255,255,0.

Μια εικόνα είναι ένας πίνακας δύο διαστάσεων (στήλες στην οριζόντια διάσταση και γραμμές στην κατακόρυφη), από pixels. Με την παραπάνω μορφή αναπαράστασης του χρώματος κάθε pixel (RGB888), έχουμε έναν δισδιάστατο πίνακα από τριάδες bytes (με την σειρά R, G, B). Ο τρόπος που αυτός ο πίνακας αποθηκεύεται στη μνήμη είναι ανά γραμμή - σειρά. Δηλαδή πρώτα αποθηκεύεται το byte του κόκκινου του επάνω-αριστερά pixel. Ακολουθεί το byte του πράσινου του ίδιου pixel και το byte του μπλέ, πάλι του ίδιου pixel. Μετά αποθηκεύονται τα τρία bytes του 2ού από αριστερά pixel της πρώτης (επάνω) γραμμής, κ.ο.κ. Στο σχήμα 1 φαίνονται πως είναι οργανωμένα τα bytes κάθε pixel της εικόνας καθώς και η απόκλιση (offset) σε bytes από την αρχή του πίνακα όπου είναι αποθηκευμένη η εικόνα. Η εικόνα του σχήματος έχει $c+1$ στήλες και $r+1$ γραμμές, καθώς η αρίθμηση στηλών και γραμμών ξεκινάει από το 0.

Οι οικονομικές οθόνες που χρησιμοποιούνται σε ενσωματωμένους (embedded) υπολογιστές δεν έχουν τη δυνατότητα να αναπαραστήσουν τόσο πολλά χρώματα. Επιπλέον ο τρόπος με τον οποίο ανανεώνεται η οθόνη είναι με σειριακή αποστολή των bytes όλων των pixel. Συνεπώς, χρησιμοποιούν έναν άλλο τρόπο αναπαράστασης χρώματος που αντί για 3 bytes χρειάζεται 2. Μια πολύ συνηθισμένη μορφή είναι η RGB565, όπου το κόκκινο και το μπλέ έχουν 5 bits το καθένα, ενώ το πράσινο έχει 6 bits, αντί των 8 bit ανά χρώμα της RGB888. Έτσι και περιορίζεται ο αριθμός των bytes που στέλνονται σειριακά κατά το 1/3 και οι διαθέσιμοι συνδιασμοί χρωμάτων είναι μικρότεροι (2^{16} αντί για 2^{24}). Το πράσινο έχει 1 bit περισσότερο γιατί το ανθρώπινο μάτι είναι πιο ευαίσθητο στο πράσινο χρώμα και έτσι ξεχωρίζει περισσότερες αποχρώσεις του πράσινου σε σχέση με τα άλλα χρώματα.

Η αντιστοίχιση της μορφής RGB888 σε RGB565 φαίνεται στο σχήμα 2. Προσέξτε ότι τα περισσότερο σημαντικά bits κάθε χρώματος είναι αυτά που περνάνε από την RGB888 στην RGB565[1]. Επιπλέον παρατηρήστε ότι, καθώς τα bytes του pixel είναι πλέον δύο, τα 5 ή 6 bits που κωδικοποιούν την ένταση

[1]Προσοχή παρόλο που, για παράδειγμα τα 5 bits του κόκκινου στο RBG565 αναφέρονται ως R4 - R0, στην πραγματικότητα προέρχονται από τα R7 - R3 της μορφής RGB888, δηλαδή τα περισσότερο σημαντικά. Αντίστοιχα και για τα υπόλοιπα χρώματα.



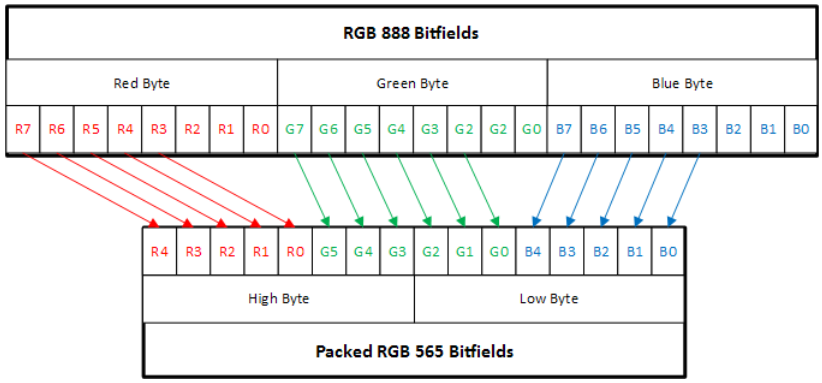
Σχήμα 1: Αναπαράσταση εικόνας σε μορφή RGB888.

των χρωμάτων μοιράζονται στα δύο bytes και χρειάζονται ολισθήσεις, μάσκες κλπ για να ξεχωρίσουν.

Μια εικόνα με την μορφή RGB565 είναι ένας δισδιάστατος πίνακας από ζεύγη bytes αντί για τις τριάδες bytes της μορφής RGB888. Στους περισσότερους υπολογιστές αυτή η ποσότητα πληροφορίας, ονομάζεται half-word και υπάρχουν ειδικές εντολές προσπέλασης μνήμης για αυτή. Στον Risc-V υπάρχουν οι lh, lhu, sh που συντάσονται όπως και οι lb, lbu, sb. Όπως και στην φόρτωση byte, υπάρχει η παραλλαγή (lh) που κάνει επέκταση προσήμου από τα 16 bit στα 32 και η παραλλαγή (lhu) που μηδενίζει τα πιο σημαντικά bits του καταχωρητή που αποθηκεύει τα δεδομένα. Οπότε η αποθήκευση στη μνήμη μιας εικόνας μοιάζει με τη μορφή του Σχήματος 1 αλλά είναι απλούστερο να θεωρηθεί ως πίνακας 2 διαστάσεων από half-words. Για να υπολογιστεί η διεύθυνση της half-word που κωδικοποιεί, σε RGB565, το χρώμα του pixel της γραμμής j και στήλης i, πρέπει να γινει η πράξη $j \times 2 \times (c+1) + 2 \times i$, όπου ο αριθμός στηλών είναι $c+1$ pixels, όπως και στο Σχήμα 1. Αν θέλει κανείς να υπολογίσει την διεύθυνση των μεμονωμένων bytes θα πρέπει να λάβει υπόψη και το endianism του συστήματος. Αν είναι litle endian, η παραπάνω παράσταση δίνει τη διεύθυνση του λιγότερο σημαντικού byte (low byte στο Σχήμα 2) και το πιο σημαντικό βρίσκεται μία θέση μετά.

3 Η άσκηση

Θα γράψετε μια υπορουτίνα που μετατρέπει μια εικόνα από τη μορφή RGB888 στην RGB565. Η αρχική εικόνα θα είναι αποθηκευμένη στη μνήμη και θα ξεκινά από την διεύθυνση που θα περνάει ως είσοδος στην υπορουτίνα σας στον καταχωρητή a0. Οι διαστάσεις της εικόνας θα δίνονται ως εξής: ο αριθμός στηλών (οριζόντια διάσταση - πλάτος της εικόνας), στον καταχωρητή a1, και ο αριθμός γραμμών-σειρών (κατακόρυφη διάσταση - ύψος της εικόνας) στον καταχωρητή a2. Και οι δύο αυτές τιμές είναι σε pixels, από τις οποίες μπορείτε να υπολογίσετε το μέγεθος του πίνακα εισόδου σε bytes. Τέλος ο πίνακας στη μορφή RGB565 θα αποθηκεύεται στη μνήμη, με αρχική διεύθυνση την τιμή του καταχωρητή a3.



Σχήμα 2: Αντιστοίχισης μορφής RGB888 με RGB565.

Δεν υπάρχει τιμή επιστροφής της υπορουτίνας.

Μπορείτε να υποθέσετε ότι οι διευθύνσεις (a0, a3) δεν θα είναι 0 (null), καθώς η άδεια εικόνα δίνεται από μηδενικά στους a1, a2. Επίσης μπορείτε να υποθέσετε ότι οι διαστάσεις (a1, a2) δεν θα είναι ποτέ αρνητικοί αριθμοί.

Το πρόγραμμά σας θα πρέπει να αποτελεί γενική λύση: να μπορεί να λειτουργήσει σωστά για οποιεσδήποτε τιμές των a0, a1, a2, a3. Ένα λάθος που έχει παρατηρηθεί στο παρελθόν είναι να χρησιμοποιούνται labels από πίνακες εισόδου αντί για τους καταχωρητές που περνούν τις αντίστοιχες τιμές. Αυτό δεν είναι σωστό γιατί περιορίζει την χρήση της υπορουτίνας σε συγκεκριμένους πίνακες και θα προκαλέσει λάθη κατά τον έλεγχο.

Επιπλέον θα πρέπει να ακολουθείτε την σύμβαση του Risc-V για την χρήση των καταχωρητών. Ενδεικτικά, αν χρησιμοποιείτε κάποιους από τους καταχωρητές s θα πρέπει να αποθηκεύετε στη στοίβα τις τιμές τους στην αρχή της υπορουτίνας και να τους επαναφέρετε λίγο πριν την επιστροφή.

4 Παραδοτέο και κριτήρια αξιολόγησης

Το παραδοτέο της άσκησης είναι το αρχείο lab03.s που περιέχει το πρόγραμμά σας. Μην αλλάξετε το όνομα του αρχείου, γράψετε εκτός της περιοχής που δείχνουν τα σχόλια, γιατί δεν θα το βρίσκει ο αυτόματος έλεγχος! Προεραιτικά αλλάξτε και το README.md.

Πρέπει να κάνετε commit τις αλλαγές σας και να τις στείλετε (push) στο αποθετήριό σας στο GitHub για να βαθμολογηθούν πριν από την καταληκτική ημερομηνία!

Το πρόγραμμά σας θα βαθμολογηθεί για την ορθότητά του (περιλαμβάνει τις απαιτήσεις για γενική λύση και τήρηση της σύμβασης χρήσης καταχωρητών), την ποιότητα σχολίων και την ταχύτητα εκτέλεσής του. Το τελευταίο σημαίνει ότι πρέπει να είναι σύντομο και ο αριθμός εντολών, ειδικά μέσα σε βρόχο, να είναι όσο γίνεται μικρότερος. Μπορείτε να βρείτε πόσες εντολές εκτελούνται στο tab processor του Ripes (μοιάζει με ολοκληρωμένο κύκλωμα), κάτω δεξιά στο Execution info, πεδίο Instrs. retired.