ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΡΟΥΣΙΑΣΗ / ΕΞΕΤΑΣΗ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Νικολαΐδης Φώτιος

Μεταπτυχιακός Φοιτητής

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Επόπτης Μεταπτ. Εργασίας: Καθηγητής Α. Τραγανίτης

Δευτέρα, 30/11/2015, 14:30

Αίθουσα B108, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

"Calypso : Ένα πλαίσιο επεξεργασίας ροών δεδομένων σε επεξεργαστή γραφικών, για εφαρμοφές ραδιοπομπών προγραμματιζόμενων σε λογισμικό"

ΠΕΡΙΛΗΨΗ

Από τους Αρχαίους χρόνους ο άνθρωπος προσπαθεί να βρει τρόπους να επικοινωνήσει με άλλους από απόσταση. Ανεξάρτητα από το αν πρόκειται για φρυκτωρίες στην Αρχαία Ελλάδα, τηλέγραφο στην Αμερική του 17 αιώνα ή ακόμα και σύγχρονες συσκευές ολοκληρωμάτων κυκλωμάτων, όλα έχουν ένα κοινό μειονέκτημα. Είναι υλοποιημένα σε υλικό, με αποτέλεσμα να είναι δύσκολη η αντικατάσταση τους, η αναβάθμιση τους, καθώς και η χρησιμοποίηση τους με τρόπο άλλου του προβλεπόμενου.

Τα τελευταία χρόνια έχουν αναπτυχθεί οι συσκευές SDR (Software Defined Radios). Είναι ελαφριές συσκευές οι οποίες υλοποιούν σε υλικό μόνο τα πολύ βασικά ηλεκτρονικά μέρη (όπως ενισχυτές, RF-frontend) ενώ το πρωτόκολλο επικοινωνίας αναπτύσσεται σε λογισμικό. Η μεταφορά από υλικό σε λογισμικό ανοίγει νέους ορίζοντες στην δημιουργία επαναπρογραμματίζόμενων δικτύων, χαμηλής κατανάλωσης συσκευές, κατασκευή all-in-one συστημάτων, και όλα αυτά με μειωμένο κόστος.

Αν και ακούγεται ελκυστικό, θέτει νέες προκλήσεις στην εξίσωση της απόδοσης του λογισμικού με αυτή του υλικού. Προς αυτό το σκοπό, έχουν αξιοποιηθεί τεχνολογίες όπως application-specific integrated circuit (ASIC), reconfigurable hardware (FPGA),

programmable Digital Signal Processors (DSP) and General Purpose Processors (GPP), με την πρώτη να στοχεύει στην αυξημένη απόδοση ενώ η τελευταία στην ευκολία προγραμματισμού.

Σε μία προσπάθεια εύρεσης της χρυσής τομής μεταξύ των δυο, έχει προταθεί η χρήση της υπολογιστικής δύναμης των καρτών γραφικών (GPU). Όμως για την πλήρη αξιοποίηση των αυτών των δυνατοτήτων είναι απαραίτητο το αντίστοιχο λογισμικό. Οι τρέχουσες προσπάθειες περιορίζονται στο να¨”μπαλώνουν” την υποστήριξη GPU σε λογισμικό το οποίο είχε κατασκευαστεί για χρήση μόνο σε CPU.

Σε αυτή την εργασία έχουμε αναπτύξει ένα ευέλικτο σύστημα ικανό να χειριστεί δεδομένα είτε ως πακέτα (για χρήση σε CPU) είτε ως πλαίσια (για χρήση σε GPU). Το πρωτόκολο συγχρονισμού υλοποιείται σε CPU ενώ τα νήματα εργάτες μπορούν να βρίσκονται είτε στην GPU είτε στην CPU. Είναι σχεδιασμένο να διευκολύνει τον προγραμματιστή σε θέματα συγχρονισμού, ενώ την ίδια στιγμή να διευκολύνει την ενσωμάτωση νέων συναρτήσεων ακόμα και αν αυτές βρίσκονται σε κλειστό κώδικα.

Ως μέσον μέτρησης της απόδοσης υλοποιήσαμε το πρωτόκολλο μετάδοσης επίγειου ψηφιακού σήματος τηλεόρασης (DVB-T) στην Calypso και στο Gnuradio. Συνολικά, η Calypso υπερτερεί του Gnuradio και σε απόδοση και σε αξιοποίηση πόρων στις μοναδικές ροές. Εκεί που το Gnuradio αποδίδει καλύτερα είναι στις πολλαπλές ροές κυρίως επειδή το Gnuradio εκτελέιται στο “εύπορο” περιβάλλον της CPU ενώ η Calypso εκτελέιται σε περιβάλλον GPU το οποίο έχει αναλογικά περιορισμένους πόρους.

Επιπρόσθετα, παρέχουμε συμπεράσμα ως προς το πότε ένας κώδικας είναι καλύτερο να εκτελείται σε CPU και κάτω από ποιες συνθήκες είναι καλύτερα να μεταφέρεται σε GPU.

Nikolaidis Fotis

M.Sc. Thesis

Computer Science Department

University of Crete

Master's Thesis Supervisor: Professor A. Traganitis

Monday, 30/11/2015, 14:30

Room B108, Computer Science dept., University of Crete

“Calypso : A stream framework in Graphics processor, for Software Defined Radio applications”

ABSTRACT

Since ancient times, humans have been trying to find ways to remotely communicate one another. No matter whether we refer to fryctories in ancient Greece, telegraphs in the States of the 17th century or contemporary chip-based devices, they all share a common fault: they are built on hardware; something that renders them difficult to be replaced, updated or be used in any way other than expected.

Lately, Software Defined Radios (SDR) have emerged. They are lightweight devices implementing in hardware only the most essential electronic parts (amplifiers, RF-front end, and so forth) while the protocol is implemented purely in software. Transposing logic from hardware to software opens up new horizons for reconfigurable networks, energy efficient devices, all-in-one devices, and all that at a low cost.

Charming as it sounds, it sets new challenges on the equalization of software performance to that of dedicated hardware. Towards this goal, various underlying hardware technologies have been used, like application-specific integrated circuit (ASIC), reconfigurable hardware (FPGA), programmable Digital Signal Processors (DSP) and General Purpose Processors (GPP), the former outmatching in performance, while the latter in programmability.

In an effort to find the golden section between the two, it has been proposed to use the computational capabilities of Graphics Cards. But to fully leverage its capabilities additional software is required. Contemporary approaches try to hack GPU support on existing software, which was originally built for CPU purposes.

In this project we have developed a versatile streaming engine able to handle both item-based process chains (implemented in CPU) and frame based process chains (implemented in GPU). The synchronization protocol is implemented in CPU side while worker threads can be located either in CPU or GPU. It is designed to be relieve the programmer from any synchronization issues, while at the same it is function-agnostic so that new functions (even in binary libraries) can be integrated easily.

As a measurement mean, we have implemented the Digital Video Broadcast – Terrestial (DVBT) both to Calypso and to Gnuradio. Overall, Calypso outperforms Gnuradio both in terms of throughput and CPU utilization on a single stream. Where Gnuradio is better, is when multiple streams are in use, mostly because Gnuradio run in the well off of CPU side while Calypso relies to the limited GPU side resources.

In addition, conclusions are provided as of when a module is better to be implemented in CPU and under which circumstances is better to be offloaded to GPU.