ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΡΟΥΣΙΑΣΗ / ΕΞΕΤΑΣΗ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ψιστάκης Αντώνιος
Μεταπτυχιακός Φοιτητής

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης
Επόπτης Μεταπτ. Εργασίας: Καθηγητής, Μ. Κατεβαίνης

Δευτέρα, 08/07/2019, 17:00
Αίθουσα Α121,Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

" Χειρισμός των Σφαλμάτων Σελίδας Μνήμης στη Διάρκεια Άμεσων Απομακρυσμένων Προσβάσεων Μνήμης με Εικονικές Διευθύνσεις"

ΠΕΡΙΛΗΨΗ

Στις ημέρες μας, η αποφυγή των κλήσεων συστήματος κατά την διάρκεια επικοινωνίας συστάδων υπολογιστών (π.χ. Κέντρα Δεδομένων, Υπολογισμοί Υψηλής Επίδοσης κ.ά.) στα μοντέρνα δίκτυα διασύνδεσης υψηλής-ταχύτητας προκύπτει ως ανάγκη, λόγω του υψηλού κόστους των πολλαπλών αντιγράφων (πυρήνα-σε-χρήστη και χρήστη-σε-πυρήνα). Οι τεχνολογίες Άμεσων Απομακρυσμένων Προσβάσεων Μνήμης (Remote Direct Memory Accesses – RDMA), οι οποίες είναι επιπέδου-χρήστη και μηδενικών-αντιγράφων, ξεπερνούν αυτό το πρόβλημα και ως αποτέλεσμα αυξάνουν την επίδοση και μειώνουν την κατανάλωση ενέργειας του συστήματος. Κοινές μηχανές RDMA όπως αυτές, δεν ανέχονται τα σφάλματα σελίδας μνήμης που προκύπτουν από εκείνες.

Οι τελευταίας τεχνολογίας τεχνικές συνήθως περιλαμβάνουν «καρφίτσωμα» (pinning) των χώρων διευθύνσεων ή πολλαπλών σελίδων μνήμης για κάθε εφαρμογή. Αυτή η προσέγγιση έχει κάποια μειονεκτήματα μακροπρόθεσμα, ως συνέπεια της πολυπλοκότητας, η οποία προκαλείται στο προγραμματιστικό μοντέλο («καρφίτσωμα»/«ξε-καρφίτσωμα» ενταμιευτών), του ορίου από bytes τα οποία μία εφαρμογή επιτρέπεται να κάνει «pin» και της συνολικής χρήσης της μνήμης. Επιπλέον, το «καρφίτσωμα» σελίδων μνήμης δεν εξασφαλίζει ότι κάποιος δεν θα αντιμετωπίσει κανένα απολύτως σφάλμα σελίδας, εξαιτίας των εσωτερικών μηχανισμών βελτιστοποίησης, όπως ο Transparent Huge Pages (THP), ο οποίος είναι ενεργοποιημένος ως προεπιλογή στα μοντέρνα λειτουργικά συστήματα Linux.

Αυτή η εργασία υλοποιεί έναν μηχανισμό διαχείρισης των σφαλμάτων σελίδας μνήμης σε συνεργασία με την μηχανή DMA του έργου ExaNeSt. Πρώτα, ανιχνεύεται το λάθος από τον διαχειριστή σφαλμάτων του προγράμματος οδήγησης (driver) της Μονάδας Διαχείρισης Μνήμης Εισόδων/Εξόδων (IOMMU) της ARM, η οποία ονομάζεται SMMU. Έπειτα, η λύση υλικού-λογισμικού μας επιλύει το σφάλμα. Τέλος, στέλνεται ένα αίτημα ώστε να γίνει επαν-αποστολή, όταν χρειάζεται. Στο σύστημα μας, ο μηχανισμός αυτός χρειάστηκε τροποποιήσεις στο πρόγραμμα οδήγησης (driver) Linux για την SMMU, την υλοποίηση μίας νέας βιβλιοθήκης λογισμικού, αλλαγές στο υλικό της μηχανής Άμεσων Απομακρυσμένων Προσβάσεων Μνήμης και τροποποιήσεις στον χρονοπρογραμματιστή των μεταφορών Απομακρυσμένων Προσβάσεων Μνήμης. Οι δοκιμές μας έγιναν επάνω στο Quad-FPGA Daughter Board (QFDB) του ExaNeSt, το οποίο εμπεριέχει τα Xilinx Zynq UltraScale+ MPSoCs.

Εκτιμούμε το κόστος του μηχανισμού μας και κάνουμε σύγκριση με τις εναλλακτικές επιλογές όπως το «καρφίτσωμα» ή την πρώιμη πρόκληση σφαλμάτων σελίδας μνήμης, και συζητάμε τα οφέλη της δικής μας προσέγγισης.

Psistakis Antonis

M.Sc. Thesis

Computer Science Department
University of Crete
Master's Thesis Supervisor: Professor, M. Katevenis

Monday, 08/07/2019, 17:00
Room A121, Computer Science Dept., University of Crete

“Handling Of Memory Page Faults During Virtual-Address RDMA”

ABSTRACT

Nowadays, avoiding system calls during cluster communication (e.g. in Data Centers, High Performance Computing etc) in modern high-speed interconnection networks comes as a necessity, due to the high overhead of the multiple copies (kernel-to-user and user-to-kernel). User-level zero-copy Remote Direct Memory Access (RDMA) technologies overcome this problem and, as a result, increase the performance and reduce the energy consumption of the system. Common RDMA Engines like these cannot tolerate page faults caused by them and choose different ways to circumvent them.

The state-of-the-art RDMA techniques usually include pinning address spaces or multiple pages per application. This approach has some disadvantages in the long run, as a consequence of the complexity induced in the programming model (pinning/unpinning buffers), the limit of bytes that an application is allowed to pin and the overall memory utilization. Furthermore, pinning does not guarantee that someone will not experience any page faults, due to internal optimization mechanisms, such as Transparent Huge Pages (THP), which is enabled by default in modern Linux operating systems.

This thesis implements a page fault handling mechanism in association with the DMA Engine of the ExaNeSt project. First, the fault is detected by the fault handler of the ARM System Memory Management Unit (SMMU). Then, our hardware-software solution resolves the fault. Finally, a retransmission is requested by the mechanism, if needed. In our system, this mechanism required modifications to the Linux driver of the SMMU, a new library in software, alterations to the hardware of the DMA engine and adjustments to the scheduler of the DMA transfers. Our tests were run on the Quad-FPGA Daughter Board (QFDB) of ExaNeSt, which contains Xilinx Zynq UltraScale+ MPSoCs.
We evaluate our mechanism and we compare against alternatives such as pinning or “pre-faulting” pages, and we discuss the merits of our approach.