CPUless PCs inside networked control systems

PETER FODREK
Slovak University of Technology
Institute of Control and Industrial Informatics
Ilkovicova 25, 812 19 Bratislava
SLOVAKIA
peter.fodrek@stuba.sk

TOMAS MURGAS
RT Systems, s.r.o.
Kopcianska 14, 851 01 Bratislava
SLOVAKIA
tomas.murgas@rtsystems.sk

MICHAL BLAHO
RT Systems, s.r.o.
Kopcianska 14, 851 01 Bratislava
SLOVAKIA
michal.blaho@stuba.sk

Abstract: This paper represents results of adavancing our previous WSEAS paper[1] and is aimed to basics for design framework that helps design hard real-time control systems using Unix/Unix like operating systems. This framework is designed while solving research project supported by the Slovak Research and Development Agency under the contract No. VMSP-II-0034-09. This framework contains layer based on OpenCL. OpenCL in control systems was f rst time mentioned in WSEAS paper of Tomas Bata University of Zlin employs[2]. This paper was based on image procesing usingOpenCl WSEAS paper by employs of two Universities in Brno[3]. OpenCL is used for fastening scheduling algorithms as well as for using VGA port‘s high speed digital to analog converters. We are testing use of DisplayPort as high speed industrial network for GPGPUs. That all takes us high advantage of Networked control. As of latest Linux kernel additions of openCL into mainline kernel therefore we are allowed to run Linux kernel parts on GPGPU. Final goal of this is to run full kernel and framework directly on GPGPU instead of CPU. This allow developers to use novel principle of ”GPU only computing” in the hard real-time control. This means computer without CPU but with GPU only. This is future of the computing.

Key–Words: OpenCL, Linux kernel, GPU only computing, CPUless computing

1 Introduction

In this paper we are to introduce reader into an pre-requistries of the Universal framework for industrial netwroked systems using General Purpose Graphics Processing Units via OpenCL version 1.2. This framework is designed to work on Unix/Unix like operating systems to avoid attacks to the industrial systems like Stuxnet and Flame/Flamer malware that are targeted to malfuncion control systems running on Microsoft Windows. Chapter descripes all aspect of such framework design and measurement of the performance gain. The concept of Central Processing Unit(CPU) less computer for industrial networked controlled system is discussed as result of such measuremnet as well.

For this purpose we have metiod how this will work in the real world scenario of industrial net-worked control as novel idea.

□

2 Operating system for boot to GPU

In this section we are to describe how to add capabil-ity to run on GPGPU for any operating system that use open sourced source code.

First part of this section is dealing of operating system design. Next part deals of main part of the operating systems rewrite. Another part deals of rewriting parts of the Operating system into OpenCL standard. Next part deals of howto write loader of these GPPGPU parts into kernel. Last part is to deal about modifying Graphic card Basic Input Output System (BIOS) to load openCL code into GPU via Network Attached Storage connecvted to GPU via Display port‘s Ethernet interface.

2.1 Operating system architecture

Operating system architecture is divided into two principals

- monolitic kernel
- μ kernel with user space servers

First type of the operating system is designed to be less f exible and is much harder to transfer its tassks into GPGPU version. Second type is much more f exible becuase we are able to port each server as single task into GPGPU OpenCL[1] version After this

[1]OpenGL 4.3 released at August, the8^{th} 2012 integrates some parts of OpenCL into itself using compute shaders feature. This

ISBN: 978-1-61804-134-0

we have all kernel tasks ported and we are only to port simple microkernel as as last part. For first case we need to port main parts of the system kernel at the begining stage of the kernel port.

As Linux kernel is second case therefore we are to use second way to achieve our goal[2]. This is case we have been done previous research.

2.2 Linux kernel architecture

First of all we need to be introduced into Linux kernel architecture to know how is kernel divided into parts. There are several parts like

- Bootloader[3]
- Bus layer[4]
- Device driver layer[5]
- Input/Output operation schedulers [6]
- Process schedulers [7]
- Application Binary Interface layer [8]
- Interfacing layer[9]

This is the case why we are to start with process schedulers. Changes in this part does not break kernel's interaction with hardware and application to kernel interfaction is not influenced as well. Some of the problems for this approach are discuses in WSEAS paper form Indian researchers[4]

We are not to change last two parts anymore. Changes of these parts will cause damage of operating system compatibility and therefore user of the changed operating system will not be able to use application ecosystem. We are to avoid this divide.

For the lower end parts we need to change device driver layer to ensure it will support OpenCL or OpenGL version 4.3 with open source drivers that are included in default linux kernel. OpenCL is to be supported in mailine kernel in year of 2012 or in the first half of 2013. Oposite to this there is only support for OpenGL version 3.0 inside open source drivers and OpenGL version 4.3 is initialy supported by nVidia proprietary device drivers just now with AMD to do so with their proprietary device drivers in the near future[10]. After all there is possibility to assume that OpenGL 4.3 will be implemented soon. It is because main parts of this standard are royality free. These parts replaces old ones that are not royality free.

There is project that produces platform for Java based operating system on GPGPU, yet[5] [11]. We are not to go same but we are to take similiar way.

Our approach is to design programming language independent system that uses full operating system functions and capabilities inside GPU. This priciple contains all of the parts listed above in the section.

2.3 Implementation details

First step we need to do is to identify parts of Linux kernel dealing with scheduling algorithms and rewrite them into OpenCL. This was partialy done in the test of their performance in the previous parts of the paper. Now we are to describe howto debug these algorithms using AMD gDEBugger tool and howto implement openCL starting code into kernel for OpenCL enabled drivers.

2.3.1 gDEBugger

This program was made by independent vendor that was bought by Advanced Micro Devices in 2011. After this there was long time lag of Linux support but form the Q2 2012 AMD was made come back of the Linux support with new features. The look of the new version is on Fig.1 This screenshot is taken form

helps us to maintain OpenCL/OpenGL interference to achieve better performance

[2]We are to port of ine schuduling as the first part of the kernel then

[3]Part that deals with boot proces of The Basic Input Output System(BIOS) or Unversal Extensible Firmware Interface(UEFI)

[4]Layer that adds support for System bus/CPU bus, PCI,PCI Express, USB, serial, System Management Bus (SMB and other bus interfaces into kernel)

[5]Layer that add support for device drivers and devioce drivers alone

[6]Layer that schedules input and output operations to achieve desired system I/O performance

[7]Layer that schedules input and output operations to achieve desired system computational performance

[8]Layer that realizes interconection bettween user space applications and kernel when application needs to use kenrel services and vice versa

[9]Additional layer to add command line interfacing shell into kernel as well as debuging, logging and similar interfacing

[10]It is assumed that OpenGL 4.3 to be in the next refresh of AMD's proprietadry device drivers for AMD graphics adapters. But deadline of this is not know after AMD stoped to release them once each month. This stop was made in the June 2012.

[11]After removing Operating system dependency as of August, the 13^{th}, 2012 state of art

Figure 1: gDEBugger screenshot

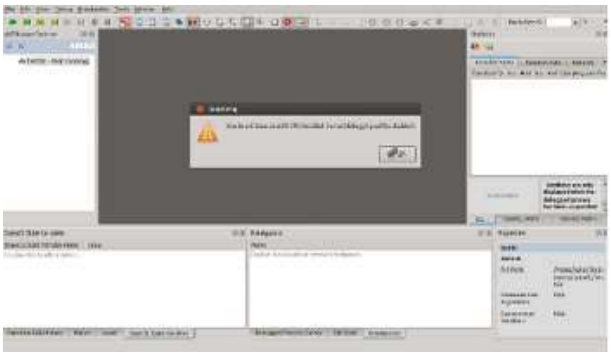

Figure 2: gDEBugger works only for AMD

Ubuntu 12.04LTS inside Unity graphics environment via KSnapshot KDE based program.The program problem is that there is problem to f nd libGL.so.1 f le inside /usr/lib directory. In the previous versions there was same problem when 64-bit version of gDEBugger needed link for 32-bit version of mention openGL library and OpenCL as well. We are not to f nd problem directly via ldd tool as gDEBugger is Bourne again shell (bash) script and gDEBugger-bin does not deal of such library.

The new version does not allow to debug OpenCL code for non-AMD GPUs as of Fig. 2 . It is able to run OpenCL program only when you use 32-bit OpenGL library form nVidia‘s device driver directory. Prof ling is to work correctly on nVidia GPUs but some information are not but suf cient for us, yet. This may be big problem when debugging and prof ling scheduling algorithms in the near future.

2.4 State of art at Slovak Univerzity od Technology and worldwide

We will try to do our best to fullf ll all needs of the implementation team that will be formed 16 days after Slovak University of Technology in Bratislava was named as 101^{th} worlds‘s best University University in branch of Computer Science as of Academic Ranking of World Universities in Computer Science - 2012 ranking `http://www.shanghairanking.com/SubjectCS2012.html`(23^{rd} in percentage of the publication in 20% of the best rated scientif c journals) on August, the 15^{th}, 2012 as we hire only μsatelite operating system designer at Slovakia as PhD candidate on September, the 1^{st}, 2012.

He was graduted as MScEECS(Electrical Engineering and Computer Science) at our Institute of Control and Industrial Informatics in June 2012, yet and as BScEECS in July 2010. He will rely on Operating system research group of the Department of Information and Communication Systems at Apple Authorized Training Center and Real-Time Programming Tools Laboratory. Last mentioned laboratory deals with Linux and Mac OS X based oprating systems and its kernel including schedulers, yet.

in this case there is an project that will help such transitions[6][12] This set of paches allows us to implement OpenCL enabled kernel as its part. OpenCL code via support of LLVM‘s Clang libraries does the same as LTO but allows us to optimize binary for runinng at GPGPU, too. Gcc allows only CPU optimizations. This shows that our way of the research presented in papers, book chapters and other publications form yeras of 2011 and 2012 were to contain interesting ideas of the implementation operating system kernels.

There is another step that supports this it is building whole Linux distribution with LLVM/Clang [7] [13] As of August, the 21^{st}, 2012 there are several interesting conclusioin for LLVM/Clang performance

- ”Starting off with the BYTE Dhyrystone 2 test, this computational test running on the Ivy Bridge system did benef t greatly from GCC’s Link-Time Optimization support: the LTO-optimized binary was about 34% faster”[8]

[12] August 19, 2012 GCC’s Link-Time Optimization support that’s new in the past few years and allows performing various compile-time optimizations across the binary as a whole, dropping dead code, and running other checks across the entire binary. LTO’ing the Linux kernel increases the compile time by two to four times and needs 4 9GB of memory to complete. More details on the Link-Time Optimization support for the Linux kernel can be found LKML this mailing list thread.`https://lkml.org/lkml/2012/8/18/273`

[13] as of August, the 20^{th}, 2012 Building Gentoo Linux With LLVM/Clang - decoupling GCC from the Linux distribution so that LLVM/Clang can build most Debian packages

- ”GCC’s Link-Time Optimization support isn’t able to improve the performance across the board, but for workloads where it did improve, the speed-ups were very noticeable. The downside to GCC LTO is that compiling the binaries takes signif cantly longer and goes through a greater amount of RAM, but if the LTO-provided performance improvement is signif cant, it really is worthwhile. It will be interesting to see how the LTO-optimized Linux kernel performs and other software as more extensive GCC LTO benchmarks are completed.” [8]

The next help for us is what was mentioned at August, the 21^{st}, 2012, too. It is that Intel‘s Mesa driver for Sandy Bridge and Ivy Bridge graphics are now off cially compliant against the Khronos OpenGL ES 2.0 specif cation and to get working on OpenGL ES 3.0 support, which they hope to have mainline in Mesa by early next year.[9] this is another step for us, because OpenGL ES 3.0 is embedded subset of OpenGL 4.3 that includes OpenCL variant inside and OpenCL to OpenGL and vice versa interaction minimization optimization as well.

All these facts allow us to conclude that our approach is actual and to be realizable soon. First of all we are to study LTO patches for Linux kernel. Then we are to modify them to be usable for OpenCL

2.5 Problems regarding design

As of mentioned above we are to face main problem that is data transfer between CPU (Central Processing Unit) and the GPU. This may be solved by AMD‘s Heterogeneous Systems Architecture (HSA) enhancement is 2013 and 2014 as of their roadmap. The unif ed memory interface was f rst publised inside AMD‘s Comal platform release on May, the 15^{th}, 2012.

It is connected to both RAM controller and Unif ed North Bridge(UNB). Its placement shows that UNB is connected to GMC via special bus named channel and there is posibility to connect GMC to the Level cache of the one of the hybrid dual core modules name Piledriver. Main advantage of the Piledriver to the Bulldozer dual core module is use of one more MultiMedia eXtenesion(MMX) unit in the newer module. This enhances idea of the use shared Level 2 cache between Piledriver module and the GPU engine.

It is because Multimedia accelerator named Universial Video Decoder in version 3 is part of the GPU core as well as dedicated acceleration unit for the CPU connected to the GMC. Therefore it is more then anticipaed to share data between CPU module and the GMC. This is able to enhance MMX performance of the CPU alone. Another sign of this solution is that instruction and data caches are divided at f rst level inside the Piledriver module as such and shared f oating point unit(FPU) for both Piledriver module cores is not equiped by cache of the f rst level. It is natural that GMC is data cache of the shared FPU as such.

This solution niched commmunication overhead between CPU and GPU for data hungry applications. Such data hungry applications include discrete PID(Proportional, Integral and Derivative) controller.

Comal system architecture contains another features that are superior inside control systems. It is network controller for high speed optical data transfer inside display port. They are HDMI/DVI connectors as well. As we are able to buy DisplayPort to VGA Video Adapter Converter for as few as 43 Euros plus VAT this allows us to use high speed Digital to Analog converters inside VGA convertor as remote analog outputs inside networked control system. HDMI to VGA converter cost as few as 25 to 27 Euros and DVUI to VGA costs about 3.50 Euros. Both these is allowed to be used as local Digital to analog converters. Analog to digital converters via VGA to HDMI and VGA to DVI costs about 10 Euros more than oposite converters. Is is not problem because it exists and therefore we are able to use them as high speed Analog to digital local inputs. But there is not known VGA to Display port converter and therefore we are not able to use remote Analog inputs for Comal, yet. Itmay be blocker feature to use Piledriver as unique industrial remote/distributed/networked control system.

3 Conclusion

We have been showed suitability of the openCL as an universal framework platform to unify CPU and GPU capabilities inside the framework and for netwoked control applications.

We have been to describe software problems form control theory and control systems point of view. Then we were intrested into realisation and deplyment problems.

By this we were disccuss oportunity of The concept of Central Processing Unit(CPU) less computer for

industrial networked controlled system is discussed as result of such measuremnet as well.

Acknowledgements: This work was supported by the Slovak Research and Development Agency under contract No. VMSP-II-0034-09. It has been supported by the project Req-00048-001 and VEGA-1/1256/12 project too.

References:

[1] P. Fodrek, L. Farkas, and T. Murgas, "Realtime scheduling using gpus - proof of feasibility," *Recent Researches in Circuits, Systems, Communications and Computers(by WSEAS) - ISBN:978-1-61804-056-5*, pp. 285–289, 2011.

[2] E. Kral, P. Capek, and L. Vasek, "Opencl-based algorithm for heat load modelling of district heating system," *Recent Researches in Circuits and Systems(by WSEAS) - ISBN:978-1-61804-108-1*, pp. 401–403, 2012.

[3] V. Skorpil, K. Zidek, T. Koubek, J. Landa, and P. Endrle, "Image processing methods optimization by means of gpu computing," *Recent Researches in Communications and Computers(by WSEAS) - ISBN:978-1-61804-109-8*, pp. 95–100, 2012.

[4] G.Muneeswari and K.L.Shunmuganathan, "Agent based load balancing scheme using affinit processor scheduling for multicore architectures," *WSEAS TRANSACTIONS on COMPUTERS*, vol. 10, pp. 247–258, 2011.

[5] M. Larabel, "Rootbeer," *Phoronix*, p. http://www.phoronix.com/scan.php?page=news_item&px=MTE1ODk, 2012.

[6] M. Larabel, "Link-time optimization to speed up the linux kernel," *Phoronix*, p. http://www.phoronix.com/scan.php?page=news_item&px=MTE2MzY, 2012.

[7] M. Larabel, "Building gentoo linux with llvm/clang," *Phoronix*, p. http://www.phoronix.com/scan.php?page=news_item&px=MTE2NDE, 2012.

[8] M. Larabel, "Gcc 4.7 link-time optimization performance," *Phoronix*, p. http://www.phoronix.com/scan.php?page=article&item=gcc_471_lto, 2012.

[9] M. Larabel, "Intel mesa now officiall opengl es 2.0 conformant," *Phoronix*, p. http://www.phoronix.com/scan.php?page=news_item&px=MTE2NDg, 2012