RESEARCH ARTICLE OPEN ACCESS

GPU Implementation over IPTV Software Defined Networking

Esmeralda Hysenbelliu*
Information Technology Faculty, Polytechnic University of Tirana, Sheshi "Nënë Tereza", Nr.4, Tirana, Albania
Corresponding author: Esmeralda Hysenbelliu

ABSTRACT
One of the most important issue in IPTV Software defined Network is Bandwidth Issue and Quality of Service at the client side. Decidedly, it is required high level quality of images in low bandwidth and for this reason it is needed different transcoding standards (Compression of image as much as it is possible without destroying the quality of it) as H.264, H265, VP8 and VP9. During a test performed in SMC IPTV SDN Cloud Network, it was observed that with a server HP ProLiant DL380 g6 with two physical processors there was not possible to transcode in format H.264 more than 30 channels simultaneously because CPU's achieved 100%. This is the reason why it was immediately needed to use Graphic Processing Units called GPU's which offer high level images processing. After GPU superscalar processor was integrated and done functional via module NVENC of FFEMPEG Program, number of channels transcoded simultaneously was tremendous increased (more than 100 channels). The aim of this paper is to real implement GPU superscalar processors in IPTV Cloud Networks by achieving improvement of performance to more than 60%.

Keywords **-** GPU superscalar processor, Performance Improvement, NVENC, CUDA

Date of Submission: 01 -05-2017 Date of acceptance: 19-08-2017

I. INTRODUCTION

In existing SMC IPTV SDN Cloud network which offer IPTV services, was not possible to tran code channels because it was needed too much physical resources in the same time (Servers) leading directly in high cost. It was thought to test new techniques included integration of CPU with GPU which resulted [4] in high level quality of images processing and high performance of channels delivering at the client side. GPU graphic cards from the architectural overview contains hundreds to thousands cores running slower and are very much suitable to perform intensive computation operation as video processing, image analysed and signal processing delivering the best value of system performance, price and power. In SMC IPTV Cloud network, it was required to enhance performance of IPTV services through implementation of GPU with superscalar processors which employ multiple functional units increasing and achieving high throughput [1]. GPU trends in designee have increased the complexity of processors and have increased the performance in the same time by increasing number of instructions through a pipeline at a time.

This paper attempts to show the implementation of GPU superscalar processor in IPTV SDN Cloud Network (Our Smart Media Communication IPTV Cloud Network) [2] achieving in high improvement of performance to over than 60%. The rest of this paper is organized as follow: Section II describes the implementation of GPU pipeline architecture; Section III describes performance test results after GPU implemented in IPTV Cloud Network and Section IV and V we have Conclusions and References.

II. GPU ARCHITECTURE IMPLEMENTATION

In IPTV SDN Cloud Network, it is needed high and fast video images processing. Last GPU processor version generated by NVIDIA vendor was at the end of February 2013 and it is delivered significantly higher CPU and GPU performance while improving its architectural and power efficiency. These processors have enabled amazing full-featured such as Web browsing, console class gaming, fast UI and multitasking responsiveness, and Blu-ray quality video playback [3]. This processor is the highest performing single-chip smartphone processor in its class and integrates an NVIDIA i500 LTE modem and Quad Cortex-A9 r4 cores, plus a 60-core GPU.

2.1Implementation of GPU Logical Pipeline Flow

Fig. 1 shows the implementation of GPU Logical Pipeline Flow in order to designee an immediate-mode pipeline adapted for high-performance and power efficiency. GPU processor is a superscalar processor that includes on-chip vertex,

texture and pixels caches. It makes possible and reality the reducing off-chip memory accesses which is a very critical point to power efficiency and high performance of images for IPTV Service over SDN (Software Defined Networking). Also this GPU processor insures high bit rates when servicing data requests.

"OpenGL API calls" block in Fig. 1 feeds the Primitive Processing block. Application-level, API calls are interpreted by the graphics driver and the driver sends various commands and pointers to vertex and texture data to the GPU. Vertex buffers are fetched and cached in a VBO Cache for subsequent reuse. Vertex shader programs called also Vertex Shader Pipeline or Vertex Processing Engine (VPE) execute in the Vertex Shader perform operations such as transforms and deformations on character and object geometry. The Primitives Assembly stage combines vertices to assemble primitives such as lines and triangles, and any primitives that reside outside of the camera's viewing area (frustum), or are back-facing, are removed from the pipeline. Those that are contained both inside and outside the viewing frustum have their outside portions clipped because they should not be rendered. Meanwhile edge and plane equations are calculated on the primitives in preparation for rasterizing

The Rasterizer converts primitives to pixels fragments to feed the pixel shader pipelines. The Early-Z unit can reject pixels that have depth (Z) values and place them behind pixels in the frame buffer. Pixel Shader pipes then operate on pixel fragments that pass the Ztest by running pixel shader programs on each pixel fragment. A programmable blend stage is incorporated in the Pixel Shader allowing any type of blend mode to be implemented, not only those found in the OpenGL spec.

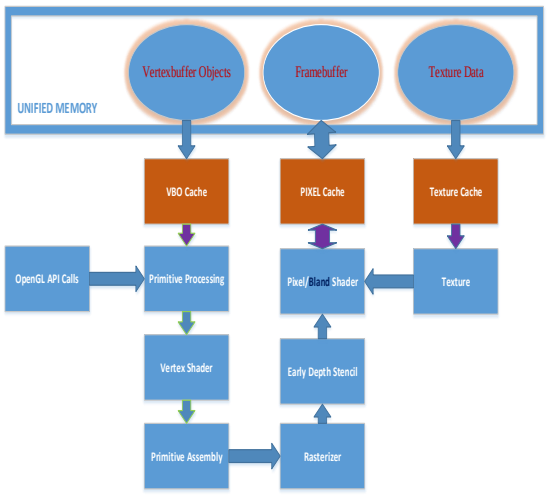

Figure 1.1 GPU logical Pipeline Flow

The Pixel Cache (also called the Fragment Data Cache) caches, writes to the frame buffer and in addition to Early –Z testing, it is used to reduce off-chip frame buffer traffic for user interface pixels or other areas that have high reuse.

The texture unit fetches and filters texture data to apply to a pixel. The accessed texture data constantly is cached in both: a) L1 texture caches present in each Texture Unit and b) L2 texture cache that is shared by all four texture units.

Finally, the processed pixels can be blended with existing frame buffer pixel information, or they can overwrite the current frame buffer pixel data.

1.2 GPU Pipeline Architecture

Fig. 2 presents in more details the actual physical implementation of GPU Processor subsystem. Starting from the top, rendering commands are fetched through Host/Front End units. Next, indices and vertices are fetched directly from memory and cached by the IDX unit. IDX then passes vertices to multiple Vertex Processing Engines (VPEs). The IDX unit also supports DX9-level instancing, where a single draw command can make multiple instances of a model, with each model using a different set of per-instance data.

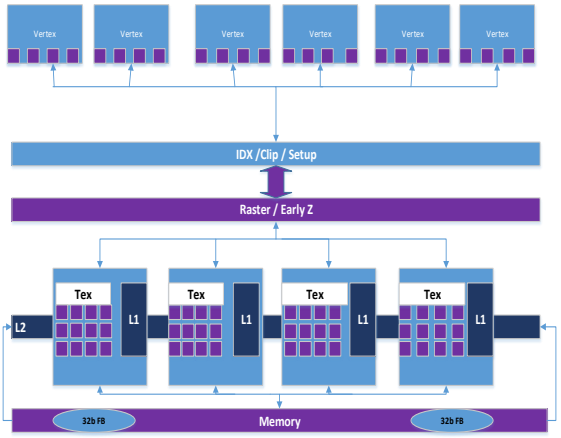

Figure 1.2 GPU Processor Architecture Diagram

Vertex Processing Engines - In our GPU processor, Vertices are processed by six VPE units where each of them includes a VEC4 ALU (arithmetic logic unit) that contains four MAD (Multiply-Add) units (where MAD units are more commonly known as Vertex Cores). GPU Processor includes a total of 24 vertex cores. A 96-entry Vertex Buffer Object cache split 16KB per VPE, allows vertex reuse and reduces off-chip memory accesses. So using GPU processors for providing IPTV Service bring very architectural improvements resulting in a performance increased over 65% where 3 processor per vertex pipe run at the same clock. The vertex shader cores use FP32 precision for their computation to ensure geometric accuracy. As it is described in the GPU Logical

Pipeline section, primitives are assembled from the vertices. Any primitives that aren't visible or back-facing are culled, or if they extend beyond the view frustum they are clipped. On primitives, during preparation for rasterizing are calculated Edge and plane equations.

Raster Engine Early- Z - The Raster Engine generates pixel fragments from the primitives and can provide eight pixel fragments per clock to the Pixel Shader pipes:

Our GPU processor now support 2x and 4x Multisample Antialiasing (MSAA), 24-bit Z, and 8-bit Stencil processing. The Raster Unit generates pixel fragments (Samples when MSAA is enabled) with associated 24-bit Z and 8-bit stencil values.
The Early-Z unit collaborates with the Raster Unit (have an Agreement). The Early-Z operation tests all the pixels or samples for Z-depth and it passes only the pixels or samples that are visible. Early-Z is able to detect and discard hidden pixels at a rate of eight pixels/clock (8ppc). Early-Z processing unit delivers improved performance and power savings by reducing memory traffic between the GPU and off-chip memory.

VLIW Pixel Fragment Shader Pipelines - Each of the four Pixel Fragment Shader pipes in our GPU processor includes three ALUs, and each ALU contains four MAD units, for a total of 48 pixel shader cores (4 x 3 x 4). Each of ALU contains a single MFU (Multi-Function Unit) which has in total 12 MFU units. The MFU units process all transcendental math (logs, exponents, trigonometric, functions), reciprocals, square roots, and MOV operations. The Fragment Shader pipelines implement a VLIW architecture, in which a mix of different instructions can be issued to the four MAD units and affected also the MFU unit in each ALU. Examples of different VLIW instruction mixes are 4x MAD, 2x DP2A + MFU, 1x DP3A + 1xMAD + MFU, 1x DP2A + 2xMAD + MFUP, 1x DP4 + MFU. A total of 16KB of Pixel Cache is divided into four 4K L1 cache slices depicted in the architecture diagram. It brings benefits in some cases like reducing off-chip frame buffer access by over 50%. It is also used rather than unified architecture for power efficiency reasons separate vertex and pixel shader architecture. The power savings derived from separate vertex and pixel shaders in the GPU processor architecture outweigh the workload flexibility advantages of unified shaders.

Texture Filtering Units - Each pixel shader unit also includes a Texture filtering unit capable of FP16 texture filtering, which enables High Dynamic Range (HDR) rendering. Each of the four texture units have their own L1 cache. The 16K L2 texture cache improves performance by reducing texture fetches from external memory. The combination of L1 and L2 texture caches reduces off-chip texture accesses by over 80% in most cases because of the typical locality of texture memory accesses by the four texture units.
Some of additional GPU Features include many other enhancements and features that help deliver richer visuals, higher performance, and more immersive graphics experiences in mobile devices.

III. IMPLEMENTATION OF GPU SMC CLOUD NETWORK

In SMC Cloud Network Data Center, requests for delivering IPTV Services are increased rapidly and in the same time the quality of images processing above client requests. It was emergently needed to implement GPU processors in order to process big data instructions in pipeline mode and to decrease existing CPU of processor. The new idea developed in this paper is the possibility to control quality levels of services delivered in clients (like Quality of service, Performance of service, low latency, high quality of service, high performance and Bluray Disk) via new features added and used by NVENC of FFEMPEG program. NVENC is an API developed by NVIDIA which enables the use of GPU cards to perform H.264 and HEVC encoding. After installation of FFEMPEG program [5] together with applications, libraries and respective drivers required, there are performed follow comparison:

3.1.1.1 Before implementing GPU superscalar processors, CPU utilization was very high (us column):

root@iptv1.smc.com.al:#vmstat -w -n -1

Procs		cpu						
r	b	cs	us	sy	id	wa	st	
9	0	10101	81	0	19	0	0	
5	0	11630	80	0	19	0	0	
12	0	11792	75	0	25	0	0	
9	0	10742	83	0	17	0	0	
14	0	10767	84	0	15	0	0	
15	1	10504	80	0	20	0	0	

3.1.1.2 GPU Card Implementation and Utilization

The NVENC utilization can be seen in the "enc" column. As it is showed, all work for transcoding channels now belongs to GPU Processor and CPU operates in low levels of usage. The "sm" column is the CUDA workload. CUDA [6] is an excellent choice for computationally datasets and designed to work on Nvidia's graphic cards. It can also run on CPUs but not as fast as in GPU.
root@iptv1.smc.com.al:# nvidia-smi dmon -i 0

# gpu	pwr	temp	sm	enc	dec	mclk	pclk
# Idx	W	C	%	%	%	MHz	MHz
0	82	35	12	86	0	3304	1151
0	82	35	11	90	0	3304	1151
0	83	35	11	93	0	3304	1151
0	83	35	12	92	0	3304	1151
0	83	36	11	94	0	3304	1151
0	83	36	9	96	0	3304	1151

CUDA operates in such a way that achieves to support other computational interfaces including Khrono's Group's OpenCL, Microsoft's direct compute. Also by using third party wrappers, it is made possible to support Python, Perl, Fortan, Java, Ruby, Lua, Haskell, MATLAB and IDL.

From the CPU overview, it was observed that usage of software resize caused a CPU intensive and bottlenecks quickly the ability to encode. In many scenarios multiple output formats are created at the same time from the input format. For that reason, NVIDIA has implemented a GPU zero-copy engine to share frames between plugins as well as a video filter that does GPU resize ("nvresize"). In the Fig. 2 follow, the video is resized into 7 formats and combined as different video streams in a single output container.

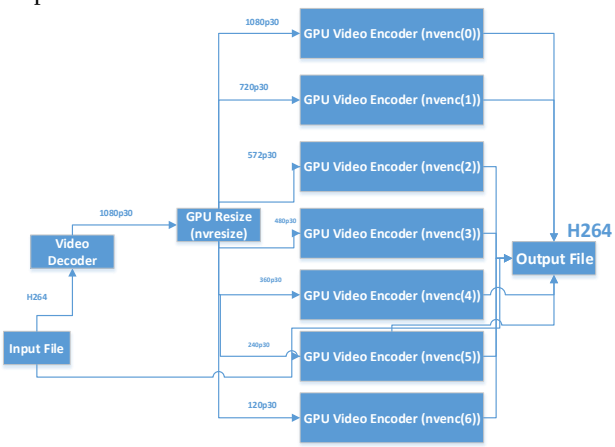

Fig 2 A typical 1:n resize scenario

The audio stream is copied from the input container to the new output container.

IV. TEST RESULTS

In the following example we will take a 1080p30 input file and downsize it to 5 formats, each stream is encoded with NVENC and then the output stream is put in its own container output file along with a copy of the audio (if present).

a) Software resize provided resized frames at 75fps (375fps total - using 37% NVENC utilization).

b) GPU resize provided resized frames at 190fps (950fps total - capped by 100% NVENC utilization).

Software bases resize figure is given follow:

```
~/ $ time ffmpeg -y -i INPUT.mp4 \
-acodec copy -vcodec nvenc -b:v 5M   -s hd1080
out1sw.mkv \ -acodec copy -vcodec nvenc -b:v 4M
-s hd720 out2sw.mkv \ -acodec copy -vcodec nvenc -
b:v 3M   -s hd480 out3sw.mkv \-acodec copy -
vcodec nvenc -b:v 2M   -s wvga out4sw.mkv \
-acodec copy -vcodec nvenc -b:v     M   -s cif
                out5sw.mkv
```

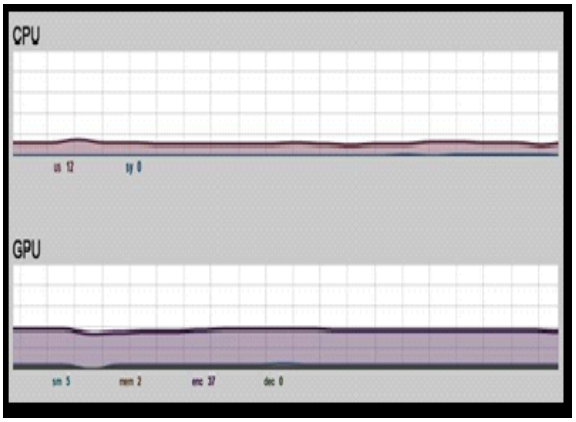

Fig 3. 1 Software based resize

GPU accelerate resize figure is given follow:

```
~/ $ time ffmpeg -y -i INPUT.mp4 -filter_complex \
nvresize=5:s=hd1080\|hd720\|hd480\|wvga\|cif:readb
ack=0[out0][out1] [out2][out3][out4] \
-map [out0] -acodec copy -vcodec nvenc -b:v 5M
out0nv.mkv \ -
map [out1] -acodec copy -vcodec nvenc -b:v 4M
out1nv.mkv \ -
map [out2] -acodec copy -vcodec nvenc -b:v 3M
out2nv.mkv \ -
map [out3] -acodec copy -vcodec nvenc -b:v 2M
out3nv.mkv \ -
map [out4] -acodec copy -vcodec nvenc -b:v 1M
out4nv.mkv
```

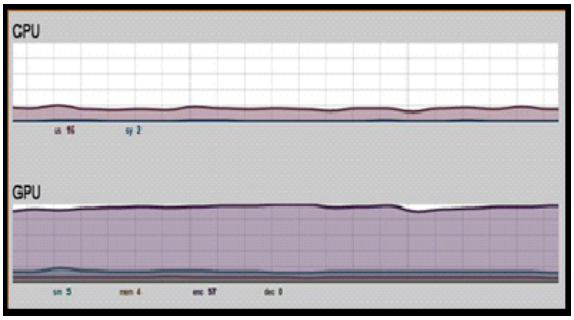

Fig 3. 1 GPU Resize

From the graphics in Fig. 3.1 and Fig. 3.2, it is clear that with GPU resize we achieve performance improvement with 63%, that means all the functionalities for transcoding input files like channels, passed directly from GPU and its high utilization shows the increased number of channels transcoded (more than 100 channels) simultaneous.

V. CONCLUSIONS

In this paper, we implemented a Superscalar GPU processor over SMC SDN Cloud Network due to bottleneck issue faced with CPU utilization (100%). GPU is a powerful unit used mostly in the field where IPTV and Video streaming services are delivered with high quality of images processing, high power efficiency and high performance. GPU resize feature applied in our example lighted out the key benefits of GPU which is improvement of performance with more that 60%. Even though the cost of GPU is high, this might be to the key for the future of Processor technology.

REFERENCES

[1] Kwang-yeob Lee, Nak-woong Eum, Jae-chang Kwak 'Superscalar GP-GPU design of SIMT architecture for parallel processing in the embedded environment', Advanced Science and Technology Letters (Vol.43 Multimedia2013), pp.67-70
[2] Esmeralda Hysenbelliu "A cloud Based architecture for IPTV as a Service," Proceeding of 2015 Balkan Conference on Informatics: advances in ICT, pp. 59–64, 2015.
[3] https://www.nvidia.com/
[4] Sukanya.R, Swaathikka.K, Soorya.R 'Enhancing Computational Performance using CPU-GPU Integration', International Journal of Computer Applications (0975 – 8887) Volume 111 – No 7, February 2015.
[5] https://www.ffmpeg.org/
[6] http://www.nvidia.com/object/cuda_home new.html

International Journal of Engineering Research and Applications (IJERA) is **UGC approved** Journal with Sl. No. 4525, Journal no. 47088. Indexed in Cross Ref, Index Copernicus (ICV 80.82), NASA, Ads, Researcher Id Thomson Reuters, DOAJ.

Esmeralda Hysenbelliu. "GPU Implementation over IPTV Software Defined Networking." International Journal of Engineering Research and Applications (IJERA), vol. 7, no. 8, 2017, pp. 41–45.