FILTROS DE INTERPOLAÇÃO DO DECODIFICADOR HEVC VISANDO MPSOCS HETEROGÊNEOS COM SUPORTE OPENCL

MATEUS SANTOS DE MELO; CARLOS MICHEL BETEMPS, MARCELO SCHIAVON PORTO; BRUNO ZATT

Universidade Federal de Pelotas
Grupo de Arquiteturas e Circuitos Integrados (GACI)
Video Technology Research Group (ViTech)
{msdmelo, cm.betemps, porto, zatt}@inf.ufpel.edu.br

1. INTRODUÇÃO

A disponibilidade de sistemas com múltiplas, usualmente heterogêneas, unidades de processamento, como *Multiprocessor Systems on Chip* (MPSoCs), contribuem para uma melhor utilização do sistema em função de diferentes aplicações. Como estes dispositivos são comumente usados em aplicações embarcadas, torna-se necessária a exploração eficiente de suas unidades de processamento de modo a gerenciar eficientemente requisitos como desempenho e energia. Com a popularização de MPSoCs capazes de capturar vídeos e plataformas/serviços de compartilhamento deste tipo de mídia, tarefas de (de)codificação são cada vez mais frequentes neste tipo de sistema. Os vídeos hospedados nestes serviços apresentam potencial de serem visualizados (decodificados) inúmeras vezes e, em grande parte, por MPSoCs. Em especial, os Filtros de Interpolação de amostras apresentam importante representatividade no esforço computacional em decodificadores de vídeo.

No fluxo de codificação do padrão HEVC (*High Efficiency Video Coding*) (SULLIVAN, 2012), atual padrão estado da arte, o vídeo a ser codificado é dividido em quadros, e cada quadro dividido em blocos de L x L amostras, tipicamente L = 64, chamadas de *Coding Tree Unit* (CTU). O padrão HEVC admite o particionamento recursivo de uma CTU de acordo com a estrutura de uma árvore quaternária de até quatro níveis de profundidade, sendo cada folha desta divisão também recursiva, chamada de *Coding Unit* (CU). Para as etapas de Predição, onde são utilizados os Filtros de Interpolação, as CUs podem ser particionadas em blocos menores, chamados *Prediction Unit* (PU).

A Compensação de Movimento tipicamente é realizada conforme apresentado na Figura 1. De maneira sequencial, para cada PU em uma CU, inicialmente são identificados os vetores de movimentos (processado durante a Estimação de Movimento), realizado o acesso a memória referente à posição indicada pelos vetores de movimento, e então é realizada a interpolação destas amostras e aplicada a predição ponderada sobre estas, caso necessário.

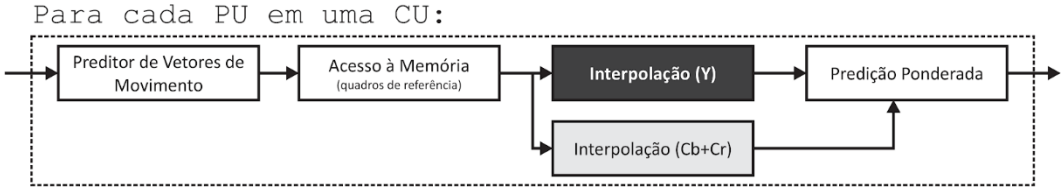

Figura 1: Fluxo de processamento da biblioteca Libde265.

Desse modo, este trabalho apresenta implementações de Filtros de Interpolação de amostras de luminância em OpenCL, presentes na etapa de Compensação de Movimento, com suporte ao processamento paralelo e heterogêneo, não explorado em soluções atuais dos padrões de codificação de vídeo. As implementações foram embarcadas na biblioteca de decodificação

Libde265, que segue o padrão HEVC e está presente em alguns *softwares* comerciais, os resultados foram extraídos da biblioteca executando sobre o MPSoC *Samsung Exynos* 5422 embarcado na plataforma Odroid-XU3.

2. METODOLOGIA

De modo a elevar o volume de dados a serem processados em paralelo pelas implementações com suporte OpenCL (STONE, 2010), foram realizadas modificações no fluxo de processamento da biblioteca, conforme apresentado na Figura 2. Originalmente, o fluxo de processamento da biblioteca Libde265 realiza a interpolação a nível de PU, entretanto com a metodologia proposta, a interpolação das amostra de luminância (Y) é realizada a nível de CU. Assim, para cada PU em uma CU: são identificados os vetores de movimento, realizado o acesso à memória das respectivas amostras, e então armazenadas em uma estrutura de dados. Após o levantamento de todas as amostras a serem interpoladas em uma CU, a interpolação das amostras de luminância é realizada pelo o *kernel* OpenCL, com exploração de paralelismo. Finalmente, após a interpolação é realizada predição ponderada destas amostras para cada PU, caso necessário.

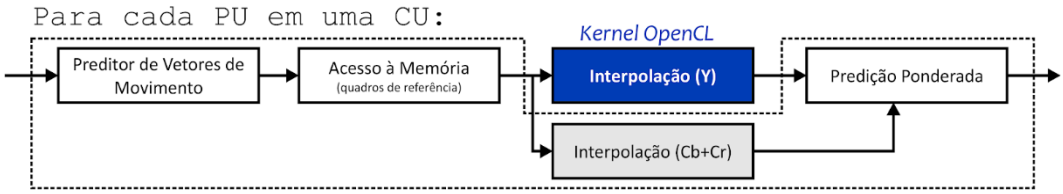

Figura 2: Fluxo de processamento da biblioteca Libde265 com suporte OpenCL.

Foram realizadas duas versões para exploração de processamento paralelo e heterogêneo utilizando o OpenCL. Estas versões diferenciam-se no que diz respeito a volume de dados transferidos e processados:

Estática: possui suporte ao processamento de todos os tamanhos de CUs possíveis. O volume de dados transferidos e processados são alocados de maneira estática considerando o maior volume de dados que pode ser processado, ou seja, uma CU 64x64. Entretanto, com esta política, os recursos alocados são superestimados em caso de CUs de tamanho menor que 64x64, assim, as amostras processadas que não referem-se a CU não serão utilizadas pelo decodificador. Este percentual de amostras não utilizadas pelo decodificador é de 75%, 93,75% e 98,44% em caso de CUs de tamanho 32x32, 16x16 e 8x8, respectivamente. Sendo este descarte de amostras o ponto negativo desta implementação, na tentativa de deixar o fluxo de processamento mais regular.

Parametrizável: Esta versão possui suporte ao processamento de todos os tamanhos de CUs, de maneira configurável. Além disso, os recursos alocados não são superestimados para as CUs, ou seja, todas as amostras processadas são utilizadas pelo decodificador. Para esta versão foi adotada a seguinte nomenclatura: XX-YY, na qual XX e YY indicam os tamanhos de CU máximo e mínimo a ser processado com o uso do OpenCL, os demais tamanhos de CU são processados de forma sequencial pelo fluxo de processamento padrão da biblioteca Libde265.

A exploração de paralelismo, na etapa de interpolação, é encontrada na literatura como em (ZATT, 2008; PAIM, 2016) utilizando blocos base de tamanho 4x4 e 8x8, referente ao grão de paralelismo. Entretanto, a utilização de blocos base de tamanho 8x8 exige o descarte de amostras processadas em caso de PUs 4x8 e 8x4, assim, comprometendo a regularidade do fluxo de processamento. Desse

modo, o foi utilizado bloco base de tamanho 4x4 como grão de paralelismo, de modo a evitar o descarte de amostras processadas neste caso.

Para obtenção dos resultados, foram utilizados cinco vídeos (*BasketballDrive*, *BQTerrace*, *Cactus*, *Kimono* e *ParkScene*) de resolução 1920x1080 presentes nas Condições Comuns de Teste (CCT) do padrão HEVC (BOSSEN, 2013). Os vídeos utilizados foram previamente codificados com o *software* HM 16.19 (*software* de referência do padrão HEVC) considerando os quatro Parâmetros de Quantização (QP) também definidos nas CCT, para a configuração de codificação temporal *Low-Delay*. Assim, para os vídeos previamente codificados, os 50 primeiros quadros foram utilizados para extrair resultados de decodificação utilizando a biblioteca Libde265, e as duas versões com suporte OpenCL.

Os resultados foram obtidos com o uso da plataforma Odroid-XU3, a qual é equipada com o MPSoC Samsung Exynos 5422 com suporte à tecnologia big.LITTLE HMP (*Heterogeneous Multi-Processing*) com ARM Cortex-A15@2,0GHz *quad core* e ARM Cortex-A7@1,4 GHz *quad core*, processadores estes com suporte ao OpenCL 1.2 com o uso do PoCL (*Portable Computing Language*) em sua versão 0.13. Além disso, o MPSoC dispõe de 2GByte de memória compartilhada LPDDR3 RAM@933MHz e uma GPGPU Mali-T628 MP6@600MHz, com suporte nativo ao OpenCL 1.1.

3. RESULTADOS E DISCUSSÃO

Abaixo, na Figura 3, são apresentados os resultados em termos de tempo de processamento obtidos na plataforma Odroid-XU3. Neste gráfico, o eixo Y gráfico refere-se ao tempo de decodificação referente a cada uma das configurações dispostas no eixo X. Estas configurações, dispostas no eixo X, referem-se a: versões da biblioteca Libde265 com ou sem o suporte ao OpenCL, unidade de processamento (CPU_s no caso de sequencial ou CPU / GPU em caso de processamento paralelo com o OpenCL), o QP utilizado; e por fim, o tamanho de bloco suportado pelas versões OpenCL, o qual é apresentado para tamanhos de CUs presentes nos intervalos 64-08, 64-16, 64-32 e 64-64 na versão OpenCL parametrizável, e 64-08 na estática.

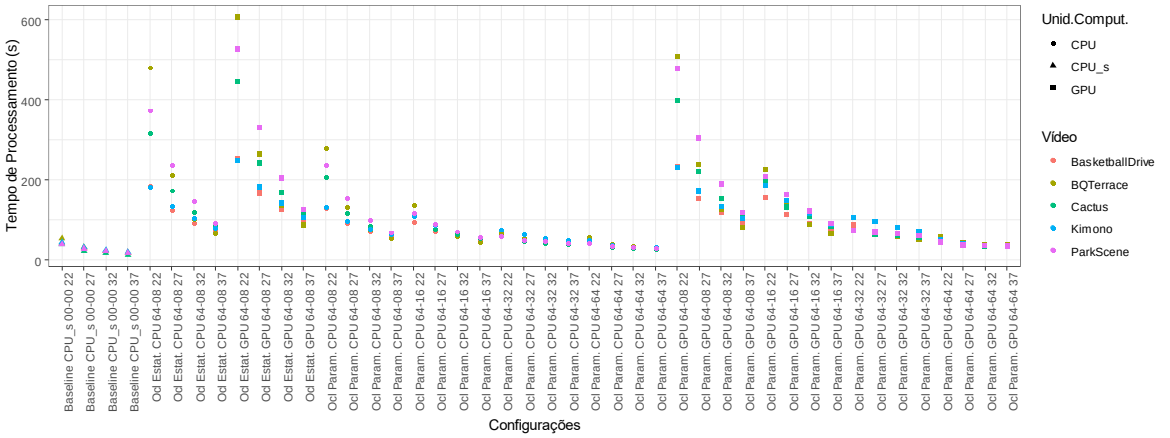

Figura 3: Resultados obtidos em termos de Desempenho.

Conforme pode ser observado na Figura 3, para um mesmo valor de QP (que influencia no tamanho da CU utilizada no processo de codificação; tipicamente maiores tamanhos para QPs maiores), os melhores resultados foram obtidos com a biblioteca Libde265 em sua vesão *baseline*, ou seja, sem o suporte ao processamento paralelo e heterogêneo com OpenCL. O acréscimo no tempo de processamento das implementações OpenCL pode estar relacionado ao gargalo de comunicação das versões OpenCL suportada pelo MPSoC, a qual estas não

oferecem suporte a tradução de espaço de memórias, como no OpenCL 2.0, sendo necessária a transferência de dados mesmo em um ambiente de memória compartilhada.

Para as configurações com suporte ao OpenCL, é possível observar a significativa diferença no tempo de processamento das versões estática e parametrizável para tamanho de CUs no intervalo 64-08, independente da unidade de processamento. Este comportamento pode ser justificada devido ao volume de dados processados e transferido, pois, na versão estática, este volume de dados é superestimado, considerando sempre o pior caso, o de uma CU 64x64, assim, também agravada pelo gargalo de comunicação da versão do OpenCL suportada pelo MPSoC.

Para as implementações OpenCL, para um mesmo QP, a versão parametrizável apresenta melhores resultados ao passo que não processa blocos de tamanhos menores. Demonstrando assim, a ineficiência do processamento paralelo e heterogênea de CUs menores nesta metodologia apresentada. Além disso, as versões com suporte ao OpenCL, para uma mesma configuração, apresentaram melhores resultados em CPU do que em GPU, dentre os motivos possíveis, pode ser apontado a baixa frequência de operação da GPU.

4. CONCLUSÕES

Este trabalho apresentou uma exploração de processamento paralelo e heterogêneo de Filtros de Interpolação de amostras da etapa de Compensação de Movimento de um decodificador de vídeo referente ao padrão HEVC. As implementações foram realizadas com OpenCL e embarcadas na biblioteca de decodificação Libde265. Assim, tendo como a inovação obtida com este trabalho a exploração de processamento paralelo e heterogêneo em um MPSoC, atualmente não explorada deste modo na literatura.

Entretanto, as implementações OpenCL apresentaram acréscimos no tempo de processamento da biblioteca Libde265 devido ao gargalo de comunicação inerentes às versões do OpenCL suportadas pelo MPSoC. Além disso, a baixa frequência de operação da GPU pode justificar melhores resultados com o OpenCL utilizando a CPU como unidade de processamento. Entretanto, ainda assim, o OpenCL mostrou-se como um bom modelo para o desenvolvimento de aplicações com suporte ao processamento heterogêneo, visto a sua fácil portabilidade entre as distintas unidades de processamento.

5. REFERÊNCIAS BIBLIOGRÁFICAS

BOSSEN, F. "Common test conditions and software reference configurations." JCTVC-L1100 12 (2013).

PAIM, G., et al. "*High-throughput and memory-aware hardware of a sub-pixel interpolator for multiple video coding standards*." IEEE *International Conference on Image Proces.* (ICIP). 2016.

STONE, J. E., et al. "OpenCL: A parallel programming standard for heterogeneous computing systems." Computing in science & engineering (2010)

SULLIVAN, G. J., et al. "Overview of the high efficiency video coding (HEVC) standard." IEEE Trans. Circuits Syst. Video Technol. (2012): 1649-1668.

ZATT, B., et al. "High throughput architecture for H. 264/AVC motion compensation sample interpolator for HDTV." Proceedings of the 21st annual symposium on Integrated circuits and system design. ACM, 2008.