UM CIRCUITO DEDICADO PARA A IME DO HEVC EXPLORANDO O REUSO DE DADOS E DE OPERAÇÕES

MURILO PERLEBERG[1]; VLADIMIR AFONSO[2];
LUCIANO AGOSTINI[3]; BRUNO ZATT[4]; MARCELO PORTO[5]

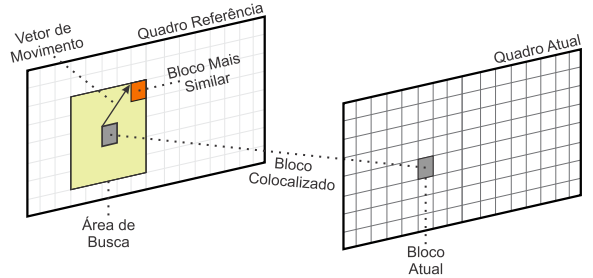
[1]*Universidade Federal de Pelotas – mrperleberg@inf.ufpel.edu.br*
[2]*Instituto Federal Sul Rio-Grandense – vafonso@inf.ufpel.edu.br*
[3]*Universidade Federal de Pelotas – agostini@inf.ufpel.edu.br*
[4]*Universidade Federal de Pelotas – zatt@inf.ufpel.edu.br*
[5]*Universidade Federal de Pelotas – porto@inf.ufpel.edu.br*

1. INTRODUÇÃO

Vídeos digitais estão cada vez mais presentes no cotidiano das pessoas, tanto pela facilidade em encontrar vídeos de entretenimento (em serviços de *streaming* como *Youtube* e *Netflix*), como também pela capacidade que os dispositivos móveis atuais possuem de capturar e compartilhar momentos e recordações. Buscando melhorar a eficiência do armazenamento e da transmissão dos vídeos digitais, diversas técnicas de compressão podem ser aplicadas sobre os vídeos digitais. Sendo assim, o padrão *High Efficiency Video Coding* (HEVC) (SULLIVAN, 2012) é um dos padrões de codificação mais importantes atualmente (BITMOVIN, 2021), e engloba diversas técnicas de compressão que permitem obter uma alta eficiência de compressão.

Para aplicar as ferramentas de codificação, cada quadro do vídeo (imagem em um único instante de tempo) é dividido em várias Unidades de Codificação em Árvore (CTUs), de tamanho 64×64, e cada CTU pode ainda ser dividida em um ou mais blocos pequenos, nos quais são aplicadas as ferramentas de codificação. A Estimação de Movimento Inteira (IME) é a ferramenta mais complexa presente no codificador HEVC, porém é uma das etapas mais importantes e indispensável em qualquer padrão de codificação de vídeo.

O diagrama da IME está representado na Figura 1-(a). A IME busca representar o bloco atual utilizando informações de um bloco pertencente ao quadro de referência, que já foi codificado anteriormente. Para encontrar o bloco a ser utilizado, a IME realiza uma busca por blocos candidatos em uma área de busca no quadro de referência, em busca do bloco candidato mais similar ao bloco atual. Como critério de similaridade entre o bloco candidato e o bloco atual, a métrica SAD, representada na Figura 1-(b), é aplicada. Assim, o resultado da IME é o vetor de movimento relativo ao bloco candidato mais similar (candidato com menor valor de SAD) que foi avaliado, além do resíduo (diferença entre o bloco atual e o candidato).



(a)

$$SAD = \sum_{(x,y)\in B} \left| C_{(x,y)} - R_{(x,y)} \right|$$

(b)

Figura 1. (a) Processamento da IME; (b) Formula do SAD.

A grande complexidade da IME se dá pelo fato de que existem muitos blocos candidatos para serem avaliados, além de que a CTU pode ser dividida em até 24 diferentes tamanhos de bloco para a aplicação da IME. Além disso, os algoritmos permitem que a IME executada para cada bloco de diferentes tamanhos possa avaliar blocos candidatos completamente distintos, o que inviabiliza uma avaliação eficiente dos blocos candidatos através do reuso de dados e de operações.

Assim, para obter o processamento de vídeos em ultra alta resolução, os trabalhos da literatura empregam circuitos dedicados para a implementação da IME. Contudo, as soluções destes trabalhos da literatura realizam otimizações neste circuito dedicado que permitem obter uma pequena redução no número de acessos á memória, porém estes trabalhos não conseguem explorar um completo reuso de dados para todos os blocos candidatos que são avaliados pelas suas arquiteturas. Sendo assim, este trabalho apresenta um circuito dedicado para a IME empregando um reuso completo de dados e de operações.

2. METODOLOGIA

Inicialmente, foi desenvolvido um algoritmo que permite um completo reuso de dados em todos os blocos candidatos que são avaliados. Neste algoritmo o processamento de todos blocos de diferentes tamanhos foram sincronizados, de forma que todos os blocos de uma CTU avaliam os mesmos candidatos. Assim, um algoritmo é aplicado sobre os valores de SAD obtidos durante o processamento dos blocos 64×64 para definir quais os blocos candidatos que devem ser avaliados. Outras heurísticas também foram aplicadas de forma a reduzir a demanda por dados de blocos vizinhos, e também reduzir o número de blocos candidatos que são requisitados para avaliação. Assim, este algoritmo proposto foi avaliado quanto ao seu impacto na eficiência de compressão de vídeos, considerando todas as Condições Comuns de Teste do HEVC (SHARMAN, 2018), e foi obtido um aumento médio na métrica BD-rate (BJONTEGAARD, 2018) de 1,14%, o que representa uma pequena redução na eficiência de codificação.

Na sequência, o algoritmo desenvolvido foi implementado em um circuito dedicado, o qual está representado na Figura 2-(a). Este circuito possui três unidades principais. Um controle, que implementa o algoritmo proposto para definir os blocos candidatos a serem avaliados, além de duas unidades de processamento.

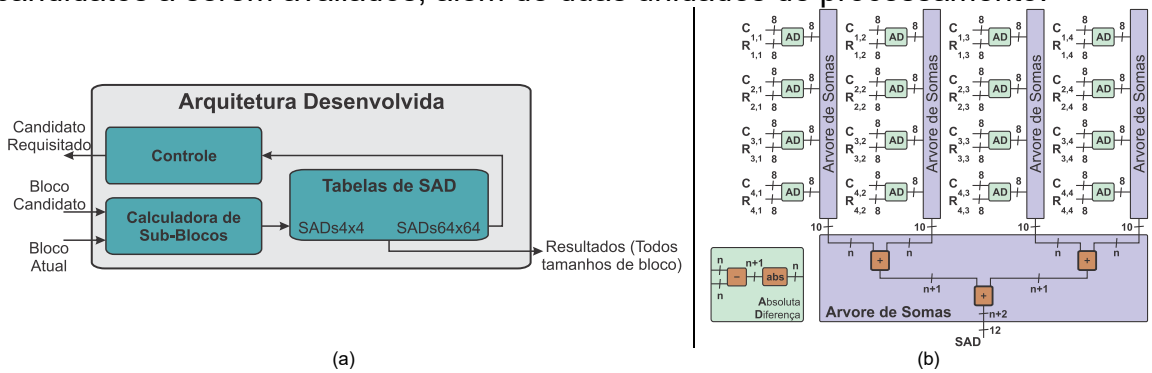


Figura 2. Arquitetura Desenvolvida: (a) Arquitetura Topo; (b) Unidade Calculadora de Sub-Blocos.

As amostras do bloco candidato e do bloco inicial passa inicialmente pela unidade Calculadora de Sub-Blocos. Esta unidade divide estas amostras em sub-blocos 4×4, e então aplica a arquitetura representada na Figura 2-(b) para calcular o valor do SAD de cada sub-bloco 4×4 no qual o bloco candidato pode ser dividido. Como pode ser visto na Figura 2-(b), o calculo do SAD de cada sub-bloco 4×4

requer 16 unidades de Absoluta Diferença (AD), além de cinco Árvore de Somas. Na sequência, o valor do SAD de cada um dos sub-blocos 4×4 é passado para a unidade Tabelas de SAD. Esta unidade é a responsável por acumular o SAD dos sub-blocos 4×4 vizinhos, de forma a obter o valor do SAD de cada um dos 24 tamanhos de bloco suportados pelo padrão HEVC. Além disso, o SAD calculado para o bloco de tamanho 64×64 é utilizado pelo controle da arquitetura para definir quais os próximos blocos candidatos a serem avaliados.

Duas versões dessa arquitetura foram desenvolvidas, utilizando diferentes níveis de paralelismo. Em uma das versões, a arquitetura é capaz de processar 4 linhas de cada bloco candidato a cada ciclo de *clock*, enquanto na outra versão a arquitetura processa 8 linhas por ciclo de *clock*. Consideramos que o algoritmo define até 326 blocos candidatos para avaliação, o que foi suficiente para cobrir o pior caso considerando as sequências de vídeo das Condições Comuns de Teste (SHARMAN, 2018). Assim, para o processamento destes 326 blocos candidatos a versão da arquitetura que processa 4 linhas por ciclo requer 5229 ciclos de *clock*, enquanto a versão que processa 8 linhas por ciclo requer 2742 ciclos de *clock*, resultando assim nas frequências de operação de 324,3MHz e 167,8MHz para as versões de 4 e 8 linhas por ciclo, respectivamente.

3. RESULTADOS E DISCUSSÃO

Ambas as versões da arquitetura foram descritas em VHDL e sintetizadas para tecnologia ASIC, utilizando a ferramenta *Cadence RTL Compiler*, e a biblioteca de células de 40nm da TSMC. Estas arquiteturas foram sintetizadas para a frequência necessária para processamento de vídeos 2160p@30fps, isto é, 324,3MHz e 167,8MHz para as versões que processam 4 e 8 linhas por ciclo, respectivamente. Os resultados e características são apresentados na Tabela I, juntamente com os resultados de alguns trabalhos relacionados da literatura.

Tabela I – Resultados comparativos da arquitetura proposta

	FAN (2018)	**GU (2019)**	**KIM (2020)**	**Este Trabalho**	
				4 linhas por ciclo	**8 linhas por ciclo**
Algoritmo	Diamond	Diamond	TZS	TZS Sincronizado	
Área de Busca	192×192	96×96	192×192	192×192	
Tamanhos de Bloco	Todos	Quadrados	Todos	Todos	
BD-Rate (%)	-0,5	0,55	0,17	1,14	
Tecnologia ASIC	65nm	TSMC 65nm	65nm	TSMC 40nm	
Taxa máxima de processamento	2160p@ 30fps	2160p@ 139fps	2160p@ 120fps	2160p@120fps	
Área (Portas NAND2)	489,4k	225,7k	439,8k	299,7k	330,4k
Potência (mW) (2160p@30fps)	128,5	N.A.	145,7	60,8	48,0

Os trabalhos da literatura propõem diferentes técnicas para reduzir a complexidade da IME. No trabalho de FAN (2018) foi proposto uma arquitetura em diamante que avalia muito mais blocos candidatos do que o algoritmo padrão do software de referência do HEVC, porém com uma menor quantidade de decisões do que este algoritmo, logo conseguindo um pequeno ganho na eficiência de compressão. Além disso, a quantidade de recursos do seu circuito e a potência dissipada

são maiores do que a do nosso circuito. Já a arquitetura de GU (2019) também adota um algoritmo em diamante, porém ele adota uma área de busca reduzida e suporta apenas o processamento de blocos quadrados para reduzir a sua complexidade e, logo, a arquitetura de GU (2019) requer uma menor área do que a do trabalho proposto. Porém, não foi apresentada uma análise da dissipação de potência do trabalho de GU (2019). Já no trabalho de KIM (2020) foi proposto uma arquitetura que permite aplicar o reuso de operações durante o processamento de alguns blocos vizinhos. Embora KIM (2020) atinja um menor impacto na métrica BD-Rate, ele requer uma maior área do que ambas as versões do nosso circuito, além de apresentar uma maior dissipação de potência do que o trabalho proposto.

A arquitetura proposta possibilita uma grande redução nos acessos a memória, visto que os dados de cada bloco candidato são requisitados apenas uma vez e todo o processamento que demanda este bloco candidato é realizado em paralelo. Além disso, podemos ver pela Tabela I que ao duplicar o nível de paralelismo da arquitetura, temos um pequeno aumento na área resultante do circuito, visto que apenas a Unidade Calculadora de Sub-Blocos precisa ser completamente duplicada. Além disso, podemos ver uma pequena redução na potência dissipada com o aumento do nível de paralelismo da solução proposta, obtida devido à redução na frequência de operação para realizar o processamento de vídeos 2160p@30fps.

4. CONCLUSÕES

Esse trabalho propõe uma arquitetura para a ferramenta IME do HEVC, capaz de processar todos os tamanhos de bloco através do sincronismo de todos os blocos de cada CTU. Esse sincronismo possibilita o reuso de dados e de operações durante o processamento, além da redução nos acessos realizados a memória. No trabalho completo, avaliamos diferentes memórias para este sistema IME, sendo este trabalho completo submetido para a revista *IEEE Transactions on Circuits and Systems for Video Technology*, e estando atualmente em processo de revisão.

5. REFERÊNCIAS BIBLIOGRÁFICAS

BITMOVIN. **Bitmovin video developer report 2021**. 2021. Acessado em 18 ago. 2022. Online. Disponível em https://go.bitmovin.com/video-developer-report

BJONTEGAARD, G., Improvements of the BD-PSNR model, **VCEG-AI11**, 2018.

FAN, Y. et al. A Hardware-Oriented IME Algorithm for HEVC and its Hardware Implementation, **IEEE Transactions on Circuits and Systems for Video Technology**, v. 28, n. 8, p. 2048-2057, 2018.

GU, C. et al. A Micro-Code-Based Hardware Architecture of Integer Motion Estimation for HEVC, **IEEE International Conference on Very Large Scale Integration**, Peru, 2019.

KIM, T. et al. Fast Hardware-Based IME With an Idle Cycle and Computational Redundancy Reduction, **IEEE Transactions on Circuits and Systems for Video Technology**, v. 30, n. 6, p. 1732-1744, 2020.

SHARMAN, K. et al. Common Test Conditions. **JCTVC-AF1100**, Ljubljana, 2018.

SULLIVAN, G. J. et al. Overview of the High Efficiency Video Coding (HEVC) Standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1649-1668, 2012.