Arquiteturas Energeticamente Eficientes para SATD com Tamanho de Bloco Variável no HEVC

André Beims Bräscher

Departamento de Informatica e Estatística – Universidade Federal de Santa Catarina
Florianópolis – SC – Brazil
andre.brascher@grad.ufsc.br

Abstract. *The inter-frame prediction is the video coding step that provides the greatest compression. For such step the calculation of a block similarity metric is necessary. A good alternative for such calculation is the SATD, although it presents quadratic energy consumption in relation to the number of processed pixels. A possible alternative is to substitute the usual Transpose Buffer (TB) structure by a Linear Buffer (LB), re-calculating partial results. In this work, two alternatives for SATD calculation using LB are presented and compared to an architecture that uses TB. The proposed architectures achieved great area reduction.*

Resumo. *A predição inter-quadros é a etapa da codificação de vídeos responsável pela maior compressão. Para tanto, requer-se o cálculo de uma métrica de similaridade entre blocos. Uma métrica utilizada é a SATD porém a energia necessária para tal métrica tende a aumentar quadraticamente ao número de píxeis processados. Uma possível alternativa é a substituição da usual estrutura Transpose Buffer (TB) por Linear Buffer (LB), realizando recálculo de parcelas. Este trabalho apresenta duas alternativas de arquiteturas para o cálculo de SATD usando LB, comparadas a uma arquitetura com TB. As arquiteturas propostas apresentaram grande redução de área.*

1. Introdução

Um vídeo, em formato *raw* (ou seja, sem nenhum tipo de compressão), necessita de um grande volume de dados [RICHARDSON, 2003]. Portanto, a compressão de vídeo torna-se fundamental, em especial para as resoluções atualmente utilizadas (*e.g. Full-HD*, *Quad-HD*). Apesar de não possuir complexidade assintótica alta, a compressão de vídeo é uma atividade computacionalmente intensiva [SEIDEL, 2014]. Além disso, o surgimento do padrão de Codificação de Vídeo de Alta Eficiência - *High Efficiency Video Coding* (HEVC) [SULLIVAN et al., 2012] como sucessor do H.264 [ITU-T, 2003], [SULLIVAN, 2005], resultou em melhorias de aproximadamente 50% em eficiência de codificação [SULLIVAN et al., 2012]. Porém tal melhoria vem a um custo: o Modelo do HEVC - *HEVC Model* (HM) [JCT-VC, 2013] é pelo menos 4 vezes mais

lento em comparação ao código de referência do H.264, Modelo Conjunto - *Joint Model* (JM) [JVT, 2011], em configurações similares [BOSSEN et al., 2012]. Segundo Delagi (2010), tal complexidade tende a aumentar cerca de 10 vezes entre 2012 e 2020.

Quando feita em Dispositivos Móveis Portáteis - *Portable Mobile Devices* (PMDs) a compressão deve ocorrer em tempo real, tendo em vista as restrições na quantidade de quadros que podem ser armazenados sem compressão. Muitas vezes tal restrição de desempenho pode comprometer o consumo de energia. Logo, a eficiência energética também deve ser considerada, em conjunto com o desempenho. Uma possível abordagem para melhorar o desempenho da codificação e garantir eficiência energética é realizar tarefas de computação intensiva em *hardware* dedicado [SEIDEL, 2014].

Enquanto responsável por parte significativa da compressão, a Estimação de Movimento – *Motion Estimation* (ME) também consome parte significativa do tempo de codificação [BOSSEN et al., 2012]. Tal etapa se tornou ainda mais crítica considerando o HEVC, no qual a ME é responsável por cerca de 40% do tempo de codificação, mesmo usando o algoritmo de busca rápida [BOSSEN et al., 2012]. A ME consiste, basicamente, em comparar a similaridade entre o bloco sendo codificado, chamado de original, e diversos blocos candidatos para substituí-lo. Para isso é necessário o uso de uma métrica de similaridade, tais quais a Soma das Diferenças Absolutas - *Sum of Absolute Differences* (SAD), a Soma das Diferenças Quadráticas - *Sum of Squared Differences* (SSD) e a Soma das Diferenças Transformadas Absolutas - *Sum of Absolute Transformed Differences* (SATD) [RICHARDSON, 2003].

A SAD é a métrica mais simples dentre as mencionadas anteriormente, por isso tal métrica é comumente utilizada, especialmente para sistemas embarcados com restrições no consumo de energia. A Equação 1 define o cálculo da SAD, onde $d_{i,j}$ é o valor na posição *i, j da* matriz *D* calculada de acordo com a Equação 2. Nota-se que esta métrica consiste apenas em somar todas as diferenças absolutas entre píxeis candidatos e originais.

$$SAD_{N\times M} = \sum_{i=1}^{N}\sum_{j=1}^{M} |d_{i,j}| \quad (1)$$

$$D_{N\times M} = C_{N\times M} - O_{N\times M} \quad (2)$$

A SATD pode ser vista como a SAD, com as diferenças transformadas para o domínio de frequências antes de realizar a operação de absoluto e acumular os valores. Usualmente a transformada escolhida é a Transformada Hadamard – *Hadamard Transform* (HT), devido ao fato de que é uma boa aproximação à Transformada Discreta dos Cosenos - *Discrete Cosine Transform* (DCT) (transformada usada na etapa de transformação), porém mais simples de se calcular. Segundo Zhu e Xiong (2009), a SATD baseada na HT obtém resultados melhores em comparação com a SAD.

O cálculo da SATD baseada na HT (doravante referenciada apenas como SATD) é apresentado na Equação 3, onde *c* é uma constante de dimensionamento, $td_{i,j} \in TD$ é

um elemento da matriz de diferenças transformadas. Tal matriz é obtida como mostrado na Equação 4, onde $T_{N\times N}$ representa a matriz Hadamard (Equação 5 mostra uma Hadamard 4×4).

$$SATD_{N\times N} = c \times \sum_{i=1}^{N}\sum_{j=1}^{N} |td_{i,j}| \quad (3)$$

$$TD_{N\times N} = T_{N\times N} \times D_{N\times N} \times T^{-1}_{N\times N} \quad (4)$$

$$H_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \quad (5)$$

Pode-se perceber, pela definição da SATD, o custo extra decorrente de calcular a transformada. Por este motivo, a SATD tipicamente é usada apenas para poucos casos como uma maneira de realizar ajustes finos, como na Estimação de Movimentos Fracionária – *Fractional Motion Estimation* (FME) e na predição intra.

É importante também considerar que a fase de transformada pode ser dividida em duas partes [PORTO et al., 2005], devido à propriedade da separabilidade. Tais partes são responsáveis pelos resultados de, respectivamente, a primeira e segunda multiplicação de matrizes da Equação 4. Neste cenário de divisão das operações de transformada, as partes são chamadas de transformadas-1D que formam uma transformada-2D.

Padrões de codificação de vídeo do estado da arte, tais como H.264 e HEVC, usam diferentes tamanhos de blocos. Por isso é preciso levar em consideração como a combinação de arquitetura e métrica vai se comportar ao variar o tamanho dos blocos processados. No caso da SATD, a estrutura necessária para seu cálculo aumenta significativamente ao aumentar os tamanhos dos blocos. Este fenômeno se deve à natureza do cálculo da SATD, envolvendo operações com matrizes.

Outro aspecto significativo a respeito da SATD, é o fato de que a operação não é bem definida para matrizes assimétricas. Assim, no caso do HEVC, o HM faz o cálculo desses blocos somando os valores de SATD (compondo o valor desses blocos). Além disso, o HM implementa tamanhos maiores que 8×8 como composição de blocos menores. Blocos de tamanhos não quadrados ($TD_{N\times M}$, sendo $N\neq M$) ou de tamanhos que não são potências de 2 também precisam de composição, uma vez que não existe HT para tais tamanhos. Porém, segundo He et al. (2015), calcular a SATD com blocos maiores, em vez de compor blocos, resulta em ganhos médios de \$0,05 dB\$.

Este trabalho explora alternativas de arquiteturas, buscando maior eficiência energética para o cálculo da SATD na compressão de vídeos de alta resolução (*e.g.*, *Full-HD*). Para isso foi desenvolvida uma arquitetura apresentando duas características principais. São estas:

- O uso e um *Buffer* Linear – *Linear Buffer* (LB)
- A possibilidade de se usar uma técnica de Eliminação Parcial das Distorções - *Partial Distortion Elimination* (PDE). Seidel, Bräscher e Güntzel (2015) apresentam uma arquitetura de SAD que usa tal técnica.

Tal arquitetura foi comparada diretamente à arquitetura com *Buffer* de Transposição – *Transpose Buffer* (TB) de Cancellier et al. (2014).

Na Seção 2 são apresentadas as arquiteturas propostas. A Seção 3 apresenta os resultados para as sínteses das arquiteturas propostas em comparação com uma arquitetura com TB. As conclusões provenientes deste trabalho são apresentadas na Seção 4.

2. Arquitetura Proposta

A Figura 1 mostra a parte comum aos blocos operativos das duas abordagens (TB e LB). Primeiramente as entradas *O* e *C* são armazenadas em registradores e em seguida é feita a diferença de tais entradas. Então, é realizada uma etapa de primeira transformada-1D FHT das diferenças e os resultados são armazenados em um *buffer*. As duas últimas etapas são apresentadas na Figura 1 como "*First Transform and Buffer Specific Designs*". Os resultados parciais passam então por uma segunda etapa de transformada-1D FHT em *"N-Inputs 1-D H Transform"*. Em seguida faz-se o absoluto dos valores obtidos da segunda transformada-1D. Os valores absolutos são somados em um esquema de árvore de soma e acumulados em *PSATD*. Ao final, os valores são divididos por dois usando um deslocamento de bits à direita e tem-se o resultado da SATD. No caso da LB-SATD, não é necessário o registrador após o acumulador devido a um estado de *"done"*.

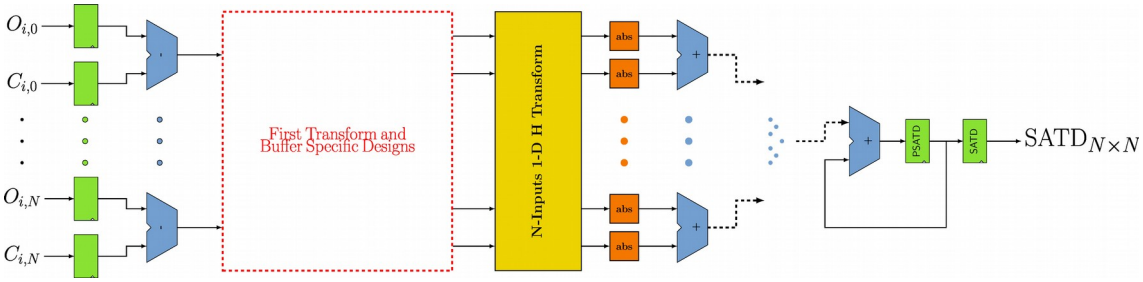

Figura 1. Bloco operativo de ambas as arquiteturas (SATD-LB e SATD-TB). O bloco tracejado pode ser substituído por um dentre dois modelos específicos para o cálculo da SATD, o TB ou o LB.

Na Figura 2 é apresentada a estrutura específica à arquitetura TB-SATD, contendo a estrutura para a realização da primeira transformada-1D, o TB e multiplexadores. A primeira transformada é realizada de maneira a preencher o *buffer* alternando entre preenchimento por linhas e por colunas. O *buffer* é composto por registradores com multiplexadores nas suas entradas para determinar se a escrita será feita por linha ou por coluna. A cada passo de preenchimento de tal estrutura, os dados são deslocados por linha ou por coluna (no mesmo sentido em que a escrita está sendo

realizada). Da mesma forma, as saídas de tal estrutura contém multiplexadores para ler de maneira transposta os dados à medida que eles chegam nas últimas posições. A leitura transposta ocorre pois os dados da primeira execução são lidos durante a segunda execução. Ou seja, se na primeira execução os resultados foram escritos por colunas, na segunda os dados são escritos por linhas e lidos da última linha. Em tal exemplo, na terceira execução o sentido de operação volta a ser por linhas. Assim é mantido um fluxo contínuo após um período inicial, que é interrompido no final da execução para esvaziar o TB.

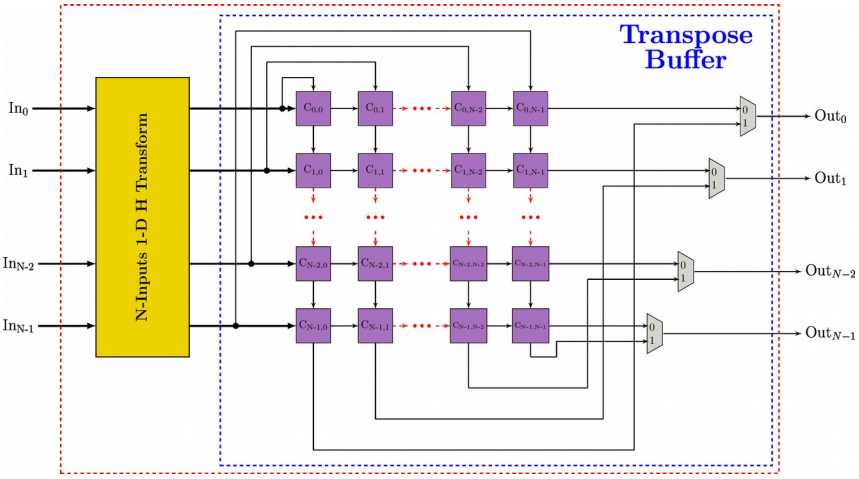

Figura 2. Arquitetura específica à TB-SATD.

O problema de se usar um LB está no fato de que ao realizar a transposição, apenas parte dos resultados parciais ficam armazenados. Com isso, a cada ciclo de cálculos da primeira transformada-1D, $\frac{N-1}{N}$ resultados deixam de ser usados. Assim é preciso repetir a transformada-1D para cada posição da transposição, selecionando o resultado que será usado através de multiplexadores. A estrutura usada para tal esquema de operação é apresentada na Figura 3. Nesta figura, os blocos *"butterfly"* contém a estrutura apresentada anteriormente, na Figura \ref{fig.butterflyStruct}. Percebe-se que a arquitetura apresentada na Figura 3 é significativamente menor que a versão da Figura 2. A redução de área vem do uso de uma transformada-1D reduzida para calcular apenas um resultado por ciclo e um *buffer* com apenas uma linha do TB. Além disso o LB não apresenta conexões entre registradores, reduzindo a quantidade de fios. Tal característica é especialmente importante à medida que se reduz os tamanhos das tecnologias.

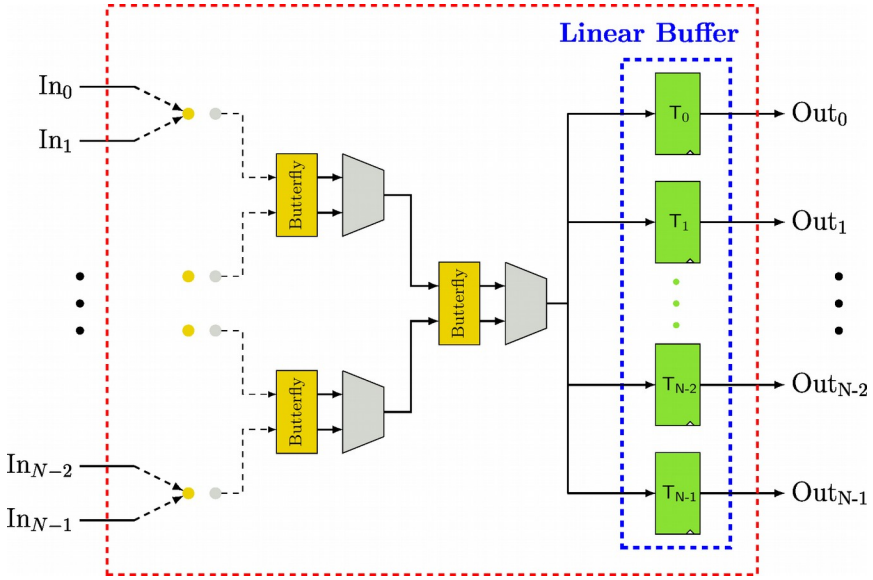

Figura 3. Bloco operativo de ambas as arquiteturas (SATD-LB e SATD-TB). O bloco tracejado pode ser substituído por um dentre dois modelos específicos para o cálculo da SATD, o TB ou o LB.

Para o controle das arquiteturas com TB e LB foram usadas, respectivamente as FSMs nas Figuras 4 e 5. Ambas as FSMs consideram que a cada ciclo de cálculo os vetores de dados adequados estarão estáveis em suas entradas. A FSM da LB-SATD tem muito menos estados pois foram colocados contadores no bloco operativo, responsáveis por contar colunas e linhas processadas. Em contrapartida, as contagens necessárias à TB-SATD são realizadas através dos estados das FSMs. Além disso, as FSMs para TB-SATD modelam a lógica de carregamento e descarregamento do TB, enquanto que tais funções são realizadas no decorrer da operação da LB-SATD (carrega na primeira etapa de transformada e descarrega enquanto soma as parcelas).

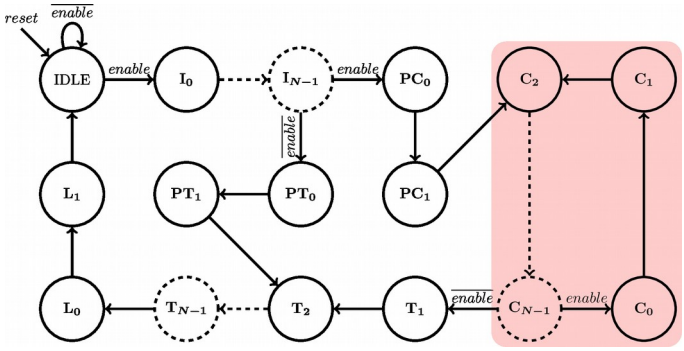

Figura 4. Máquina de Estados Finitos para o controle da arquitetura TB-SATD.

Apesar da LB-SATD ter menos estados, são necessários N^2+N+2 ciclos para se obter uma $SATD_{N\times N}$. Por outro lado apenas N ciclos são necessários para a TB-SATD. Isso ocorre pois a LB-SATD precisa calcular as N colunas (da primeira transformada-1D), para cada uma das N linhas a serem processadas pela segunda transformada-1D. A Tabela 1, mostra com mais detalhes os ciclos necessários para obter uma SATD, considerando diferentes tamanhos para as arquiteturas consideradas. Assim fica aparente a troca realizada entre tamanho do *buffer* (N^2 *vs.* N) por ciclos

necessários para o cálculo (*N vs.* N^2+N+2), respectivamente para TB-SATD e LB-SATD.

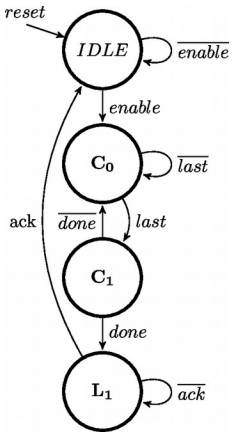

Figura 5. Máquina de Estados Finitos para o controle das arquiteturas LB-SATD e LB-SATD PDE.

Tabela 1. Ciclos/SATD, considerando diferentes tamanhos de blocos, para as arquiteturas TB-SATD e LB-SATD.

	4×4	8×8	16×16	32×32	$N \times N$
TB	4	8	16	32	N
LB	22	74	274	1058	$N^2 + N + 2$

3. Resultados

As arquiteturas foram sintetizadas com a ferramenta *Synopsys Design Compiler* [SYNOPSYS, 2011]. As sínteses usaram uma biblioteca de célula padrão - *standard cell* industrial de 45nm da TSMC, versões Nominal e *Low-Vdd/High-Vt* (LH). LH, consiste em reduzir a tensão de alimentação dos transistores e aumentar a tensão de limiar - *threshold*. Isso tem o objetivo de reduzir o consumo de energia dos transistores, porém os torna mais lentos. Devido aos maiores atrasos de transistores LH, eles só são recomendáveis quando os requisitos de *timing* não são muito restritos. Usar LH com requisitos muito restritos pode obrigar o uso de portas muito grandes (piorando o consumo de energia) ou causar violações de *timing*. Foram usadas as mesmas configurações usadas por Seidel, Bräscher e Güntzel (2016). Tais configurações são: atrasos de entrada e saída de 60% do período de relógio e máxima capacitância primária de entrada ajustada para 10 vezes a capacitância de uma porta *And* de duas entradas. Além disso, o *Synopsys Design Compiler* foi executado no modo *Topographical* para estimar capacitâncias parasitas de roteamento [SYNOPSYS, 2009]. As arquiteturas foram sintetizadas a fim de manter a mesma quantidade de amostras por unidade de tempo (*throughput*) de trabalhos anteriores, 16 milhões de blocos $4\times4/s$ [WALTER; DINIZ; BAMPI, 2011].

Assim foram obtidos os resultados de área apresentados na Figura 6 e de energia mostrados na Figura 7.

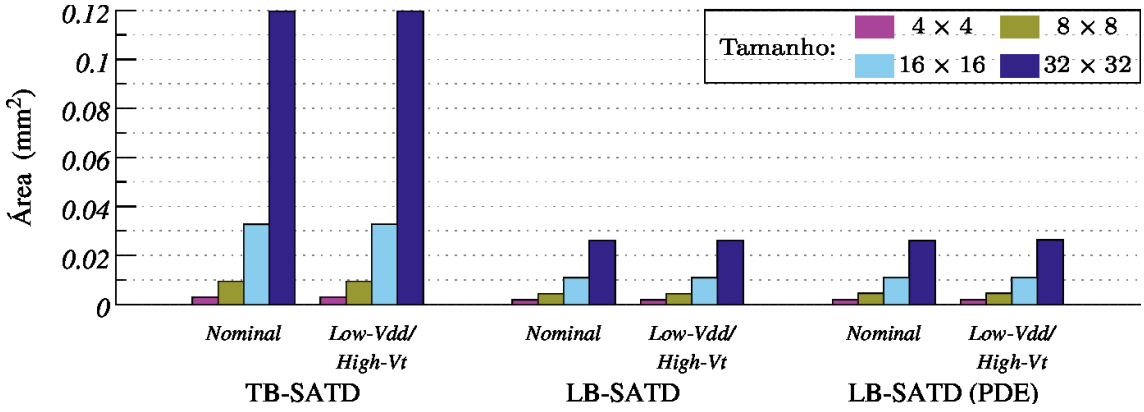

Figura 6. Estimativas de área.

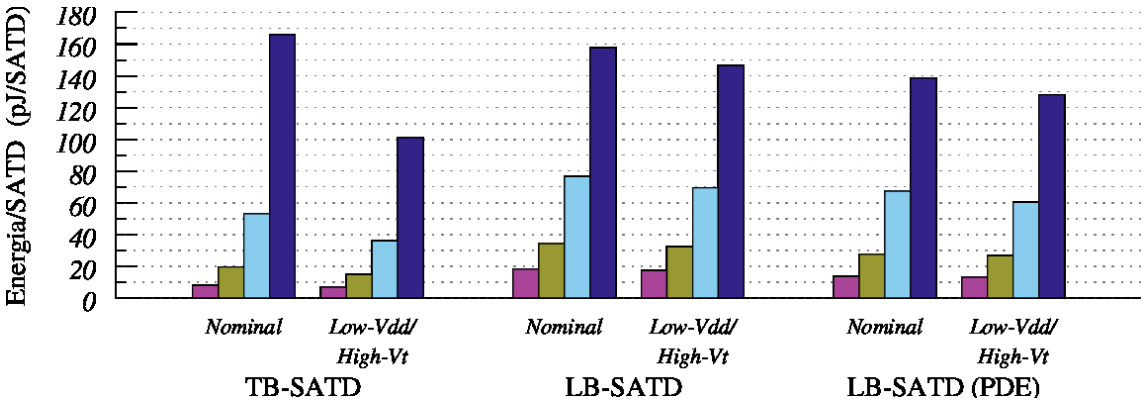

Figura 7. Estimativas de energia por 16 píxeis de SATD.

4. Conclusão

O objetivo deste trabalho foi investigar alternativas que pudessem reduzir o consumo energético do cálculo de SATD. Para isto, foi proposta uma alternativa arquitetural para cálculo de SATD (LB-SATD) para vários tamanhos de blocos de píxeis. Tal arquitetura apresenta uma estrutura do tipo LB, para reduzir o problema da escalabilidade de arquiteturas como a TB-SATD. Os resultados obtidos das arquiteturas LB-SATD foram apresentados por Seidel, Bräscher e Güntzel (2016). Além disso, foram implementadas versões das arquiteturas LB-SATD com PDE, a fim de reduzir o grande número de ciclos necessários para se obter uma SATD. A fim de mensurar o impacto do uso de PDE, foram feitas simulações na HM conforme as CTC, para obter informações estatísticas de quantos ciclos podem ser poupados ao usar PDE. Considerando as informações de média de ciclos necessários para SATD com PDE, obteve-se melhor consumo de energia, porém não foi possível superar a TB-SATD considerando síntese LH. Notou-se alta taxa de eliminação de candidatos gerados computacionalmente, assim a arquitetura LB-SATD PDE é uma boa alternativa para este tipo de aplicação. Por fim, tem-se como possível trabalho futuro a realização de testes com *clock gating* para as arquiteturas avaliadas durante este trabalho. Tal técnica pode ser especialmente benéfica para a LB-SATD, devido à alta frequência de operação aliada ao fato de que a estrutura após o LB fica sub-utilizada durante parte significativa da operação da arquitetura.

References

BOSSEN, F. et al. HEVC complexity and implementation analysis. IEEE Transactions on Circuits and Systems for Video Technology, v. 22, n. 12, p. 1685–1696, Dec 2012. ISSN 1051-8215.

CANCELLIER, L. H. et al. Energy-efficient Hadamard-based SATD architectures. In: Proceedings of the 27th Symposium on Integrated Circuits and Systems Design. New York, NY, USA: ACM, 2014. (SBCCI '14), p. 36:1–36:6. ISBN 978-1-4503-3156-2.

DELAGI, G. Harnessing technology to advance the next-generation mobile user-experience. In: 2010 IEEE International Solid-State Circuits Conference - (ISSCC). [S.l.: s.n.], 2010. p. 18–24. ISSN 0193-6530.

HE, G. et al. High-throughput power-efficient vlsi architecture of fractional motion estimation for ultra-hd HEVC video encoding. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, v. 23, n. 12, p. 3138–3142, Dec 2015. ISSN 1063-8210.

ITU-T. H.264 : Advanced video coding for generic audiovisual services. Geneva, May 2003.

JCT-VC. HEVC Test Model. 2013. Dispon´ ıvel em: <http://hevc.hhi.fraunhofer.de/>.

JVT. JM JOINT VIDEO TEAM Reference Software. 2011. Disponível em: <http://iphome.hhi.de/suehring/tml/>.

MAICH, H. et al. HEVC fractional motion estimation complexity reduction for real-time applications. In: Circuits and Systems (LASCAS), 2014 IEEE 5th Latin American Symposium on. [S.l.: s.n.], 2014. p. 1–4.

NDILI, O.; OGUNFUNMI, T. Efficient sub-pixel interpolation and low power vlsi architecture for fractional motion estimation in H.264/AVC. In: Signal Processing and Communication Systems (ICSPCS), 2010 4th International Conference on. [S.l.: s.n.], 2010. p. 1–10.

PEREIRA, F. et al. H.264 8x8 inverse transform architecture optimization. In: Proceedings of the 24th Edition of the Great Lakes Symposium on VLSI. New York, NY, USA: ACM, 2014. (GLSVLSI '14), p. 83–84. ISBN 978-1-4503-2816-6. Disponível em: <http://doi.acm.org/10.1145/2591513.2591564>.

PORTO, M. et al. Design space exploration on the H.264 4 × 4 Hadamard transform. In: NORCHIP Conference, 2005. 23rd. [S.l.: s.n.], 2005. p. 188–191.

RICHARDSON, I. E. G. H. 264 and MPEG-4 video compression: video coding for next-generation multimedia. [S.l.]: John Wiley & Sons Inc, 2003.

SEIDEL, I. Dissertação (mestrado), Análise do impacto de pel decimation na codificação de vídeos de alta resolução. Florianópolis-SC: [s.n.], 2014.

SEIDEL, I. Redução de Complexidade e Energia em Codificadores de Vídeo Digital Preservando a Eficiência de Codificação: Exploração de Propriedades das Métricas de Distorção Aplicadas no Casamento de Blocos. Tese (Seminário de Andamento (doutorado)) — UFSC, Florianópolis-SC, 2015.

SEIDEL, I.; BRÄSCHER, A. B.; GÜNTZEL, J. L. Combining pel decimation with partial distortion elimination to increase sad energy efficiency. In: . [S.l.: s.n.], 2015. No prelo.

SEIDEL, I.; BRÄSCHER, A. B.; GÜNTZEL, J. L. Energy-efficient SATD for beyond HEVC. In: Proceedings of the 2016 IEEE International Symposium on Circuits and Systems. [S.l.: s.n.], 2016.

SINANGIL, M. E. et al. Cost and coding efficient motion estimation design considerations for high efficiency video coding (HEVC) standard. IEEE Journal of Selected Topics in Signal Processing, v. 7, n. 6, p. 1017–1028, Dec 2013. ISSN 1932-4553.

SULLIVAN, G. J. Overview of international video coding standards (preceding H.264/AVC). In: . [S.l.]: Presented at Workshop on Video and Image Coding and Applications (VICA), 2005.

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, Dec 2012. ISSN 1051-8215.

SYNOPSYS. Synopsys's Design Compiler User Guide, Version C-2009.06. 2009.

SYNOPSYS. Synopsys Design Compiler, Version F-2011.09-SP5-2. 2011.

TANG, X. l.; DAI, S. k.; CAI, C. h. An analysis of tzsearch algorithm in jmvc. In: Green Circuits and Systems (ICGCS), 2010 International Conference on. [S.l.: s.n.], 2010. p. 516–520.

WALTER, F. L.; DINIZ, C. M.; BAMPI, S. Synthesis and comparison of low-power high-throughput architectures for SAD calculation. In: 2011 IEEE Second Latin American Symposium on Circuits and Systems (LASCAS). [S.l.]: IEEE, 2011. p. 1–4. ISBN 978-1-4244-9484-2.

ZHU, C.; XIONG, B. Transform-exempted calculation of Sum of Absolute Hadamard Transformed Differences. Circuits and Systems for Video Technology, IEEE Transactions on, v. 19, n. 8, p. 1183–1188, Aug 2009. ISSN 1051-8215.

ZHU, J. et al. Fast prediction mode decision with Hadamard transform based rate-distortion cost estimation for HEVC intra coding. In: 2013 IEEE International Conference on Image Processing. [S.l.: s.n.], 2013. p. 1977–1981. ISSN 1522-4880.