

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

DESENVOLVIMENTO E IMPLEMENTAÇÃO
EM FPGA DE UM COMPRESSOR SEM
PERDAS DE BAIXA COMPLEXIDADE PARA
IMAGENS DE SATÉLITE

YURI GONZAGA GONÇALVES DA COSTA

JOÃO PESSOA-PB
Julho-2012

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

**DESENVOLVIMENTO E IMPLEMENTAÇÃO EM FPGA
DE UM COMPRESSOR SEM PERDAS DE BAIXA
COMPLEXIDADE PARA IMAGENS DE SATÉLITE**

YURI GONZAGA GONÇALVES DA COSTA

JOÃO PESSOA-PB
Julho-2012

YURI GONZAGA GONÇALVES DA COSTA

**DESENVOLVIMENTO E IMPLEMENTAÇÃO EM FPGA
DE UM COMPRESSOR SEM PERDAS DE BAIXA
COMPLEXIDADE PARA IMAGENS DE SATÉLITE**

DISSERTAÇÃO APRESENTADA AO CENTRO DE INFORMÁTICA DA
UNIVERSIDADE FEDERAL DA PARAÍBA, COMO REQUISITO PARCIAL
PARA OBTENÇÃO DO TÍTULO DE MESTRE EM INFORMÁTICA
(SISTEMAS DE COMPUTAÇÃO).

Orientador: Prof. Dr. José Antônio Gomes de Lima

JOÃO PESSOA-PB
Julho-2012

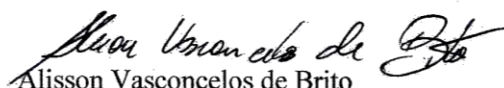
C837d Costa, Yuri Gonzaga Gonçalves da.
Desenvolvimento e implementação em FPGA de um
compressor sem perdas de baixa complexidade para imagens
de satélite / Yuri Gonzaga Gonçalves da Costa.-- João Pessoa,
2012.
94f. : il.
Orientador: José Antônio Gomes de Lima
Dissertação (Mestrado) – UFPB/CCEN
1. Informática. 2. Satélites – imagens. 3. Sensoriamento
remoto. 4. Compressão de dados. 5. Codificação de Golomb-
Rice. 6. Verificação Funcional. 7. FPGA.

UFPB/BC

CDU: 004(043)

Ata da Sessão Pública de Defesa de Dissertação de
Mestrado de **YURI GONZAGA GONÇALVES
DA COSTA**, candidato ao Título de Mestre em
Informática na Área de Sistemas de Computação,
realizada em 31 de julho de 2012.

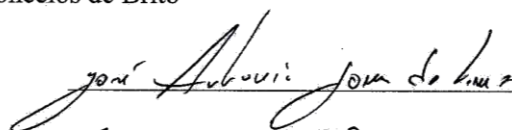
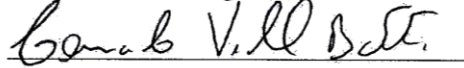
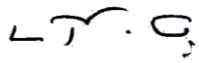
Ao trigésimo primeiro dia do mês de julho do ano dois mil e doze, às quatorze horas, na sala do REUNI - da Universidade Federal da Paraíba, reuniram-se os membros da Banca Examinadora constituída para examinar o candidato ao grau de Mestre em Informática, na área de “Sistemas de Computação”, na linha de pesquisa “Sinais, Sistemas”, o Sr. **YURI GONZAGA GONÇALVES DA COSTA**. A comissão examinadora foi composta pelos professores doutores: JOSÉ ANTÔNIO GOMES DE LIMA (PPGI-UFPB), Orientador e Presidente da Banca, LEONARDO VIDAL BATISTA (PPGI-UFPB), examinador interno e LEILA MARIA GARCIA FONSECA (INPE) como examinadora externa. Dando início aos trabalhos, o professor JOSÉ ANTÔNIO GOMES DE LIMA, cumprimentou os presentes, comunicou aos mesmos a finalidade da reunião e passou a palavra ao candidato para que o mesmo fizesse, oralmente, a exposição do trabalho de dissertação intitulado “Desenvolvimento e Implementação em FPGA de um Compressor sem Perdas de Baixa Complexidade para Imagens de Satélite”. Concluída a exposição, o candidato foi arguido pela Banca Examinadora que emitiu o seguinte parecer: “aprovado”. Assim sendo, deve a Universidade Federal da Paraíba expedir o respectivo diploma de Mestre em Informática na forma da lei e, para constar, eu, professor Alisson Vasconcelos de Brito, coordenador deste programa, servindo de secretário, lavrei a presente ata que vai assinada por mim mesmo e pelos membros da Banca Examinadora. João Pessoa, 31 de julho de 2012.


Alisson Vasconcelos de Brito

Prof. Dr. José Antônio Gomes de Lima
Orientador (PPGI-UFPB)

Prof. Dr. Leonardo Vidal Batista
Examinador Interno (PPGI-UFPB)

Profa. Dra. Leila Maria Garcia Fonseca
Examinadora Externa (INPE)

Aos Gonzagas.

Agradecimentos

Em primeiro lugar, agradeço a Deus por ter me escolhido para ser quem sou, para viver o que eu vivi e conhecer as pessoas que conheci.

Depois, quero agradecer aos meus pais pelo exemplo e por me apoiarem desde o princípio, me ajudando a tomar as decisões necessárias ao longo desses 25 anos de vida. A educação e o convívio que obtive dentro de minha casa foram fundamentais para entender como cheguei até aqui.

Quero enaltecer as figuras da minha irmã e madrinha Luízi, dos meus irmãos Guinha e Ygor, e do meu primo-irmão Daniel. Sempre estiveram ao meu lado e continuam me acompanhando, mesmo quando eu escolho fazer coisas diferentes. Muito obrigado pelos momentos de alegria e cumplicidade que passamos juntos.

Agradeço em especial a Camila por todo o companheirismo nos momentos de desestímulo e dedicação incondicional. Sem você, Pretinha, sou como uma imagem de satélite em que ocorreu uma falha devido à radiação espacial.

Ainda agradeço a todos do LASID: ao meu orientador, professor José Antônio, e professores Leonardo e Hamilton, além dos amigos Daniel, Bruno, Igor, Lucas, Janduy e Guilherme. Todos os momentos de convivência e aprendizado vividos desde a graduação foram muito importantes não só para o mestrado mas para a vida toda.

Também quero registrar um agradecimento à Divisão de Processamento de Imagens do Instituto Nacional de Pesquisas Espaciais, principalmente nas figuras da Professora Leila Fonseca, Lílian Nogueira e Antônio Lopes, pelo fornecimento de imagens e pelo intercâmbio acadêmico.

Por fim, muito obrigado a todos os demais amigos, colegas, professores e funcionários que encontrei pela UFPB ou fora dela e que direta ou indiretamente contribuíram para que eu alcançasse esse objetivo. Torço muito pelo sucesso cada vez maior do Centro de Informática, pois sei do potencial pelas pessoas que o compõe.

“A melhor maneira de prever o futuro é inventá-lo.” (Peter Drucker)

Resumo

A quantidade de dados gerados e transmitidos pelos satélites para as estações na Terra é cada vez maiores. Com o passar do tempo e avanço da tecnologia, os sistemas de imageamento espaciais, particularmente as missões de observação da Terra, tem utilizado equipamentos com resoluções cada vez maiores. Por esse motivo, se faz necessário garantir que os dados cheguem ao destino de maneira confiável. Dentre algumas técnicas envolvidas, a compressão de dados é o meio mais viável de alcançar esse requisito. Um sistema de compressão de dados para esse fim deve obedecer algumas condições, principalmente quanto ao desempenho. Nesse contexto, implementações em hardware baseadas em predição e codificação de Golomb-Rice têm obtido excelentes resultados considerando desempenho do hardware e da compressão, tanto nos casos sem perdas como nos com perdas. O presente trabalho apresenta uma proposta de hardware digital de um compressor sem perdas para imagens de satélite baseado em predição e codificação Golomb-Rice que busca um balanceamento entre os requisitos de desempenho e a propagação de erros, um problema comum no âmbito de sistemas espaciais e que é potencializado no caso dos compressores de dados. Para validação e análise do compressor, é seguida uma metodologia de verificação funcional de hardware digital e o desenvolvimento de um protótipo em FPGA. Dado um conjunto de imagens do satélite CBERS-2B disponibilizadas pelo Instituto Nacional de Pesquisas Espaciais, os resultados obtidos em FPGA mostram que esse compressor alcança razão de compressão média de 3,4, valor comparável a trabalhos correlatos, e velocidade de 28 MPixel/s (224 Mbit/s). Considerando a natureza das imagens, a compressão pode ser paralelizada por meio de simultâneos núcleos compressores em uma abordagem *multicore*. Por exemplo, usando 5 núcleos, o sistema proposto é capaz de comprimir essas imagens em uma velocidade de 142 MPixel/s (1.1 Gbit/s). Todas essas características tornam-no útil e efetivo para a aplicação em um sistema moderno de imageamento para sensoriamento remoto.

Palavras-chave: Imagens de Satélite, Sensoriamento Remoto, Compressão de Dados, Predição, Codificação de Golomb-Rice, Verificação Funcional, FPGA.

Abstract

The amount of data generated and transmitted by satellites to ground stations is always growing. As the technology advances, space imaging systems, especially those present in Earth observing missions, use equipment of increasing resolutions. Hence, it is necessary to ensure that this great quantity of data arrives at their destination reliably. Among some techniques involved, data compression plays an important role to accomplish this requirement. A data compression system for this purpose must comply with some conditions, particularly regarding performance. In this context, hardware implementations based on prediction and Golomb-Rice coding has achieved excellent results considering hardware and compression performance in both lossless and lossy cases. This work proposes a digital hardware approach of a low complexity satellite image lossless compressor based on prediction and Golomb-Rice coding that is attuned to the balance between performance requirements and error propagation, a common issue in space systems environment that is enhanced by data compression. In order to validate and analyze the compressor, a functional verification and FPGA prototyping methodology were followed. Given an image set from Brazilian's National Institute for Space Research (INPE, in the Portuguese acronym), CBERS-2B satellite, its results in FPGA show that this compressor achieves average compression ratio of 3.4, comparable value to related works in this area, and throughput of 28 MPixel/s (224 Mbit/s). Taking advantage of images nature, its compression can be parallelized through simultaneous multi-cores compressors. For example, using 5 cores, this work is able to compress those images in a rate of 142 MPixel/s (1.1 Gbit/s). All these features make it useful and effective in a current remote sensing imaging system.

Keywords: Satellite Images, Remote Sensing, Data Compression, Golomb-Rice Coding, Prediction, Functional Verification, FPGA.

Sumário

Capítulo I: Introdução.....	14
1.1. Motivação.....	14
1.2. Objetivos	16
1.3. Organização.....	17
Capítulo II: Fundamentação Teórica.....	18
2.1. Sensoriamento Remoto.....	18
2.1.1. Radiação e o espectro eletromagnético	19
2.1.2. Características das Imagens de Sensoriamento Remoto.....	21
2.2. Compressão de Dados.....	22
2.2.1. Autocorrelação, Descorrelação e Modelo de Compressão Genérico.....	25
2.2.1.1. Predição	26
2.2.1.2. Codificação de Golomb-Rice	27
2.2.1.2.1. Cálculo do parâmetro k	28
2.3. Compressão de Dados, Propagação e Resistência a Erros	29
2.4. Trabalhos Correlatos.....	30
Capítulo III: Sistema Proposto e Metodologia	34
3.1. Escolha do Algoritmo.....	35
3.2. Implementação do Modelo de Referência	37
3.3. Teste com bancos de imagens.....	39
3.3.1. Resistência a erros	41
3.4. Implementação em HDL.....	41
3.4.1. Protocolo de sinais.....	42
3.4.2. Preditor	42
3.4.3. Mapeamento	43
3.4.4. Cálculo de k	43
3.4.5. Codificação Unária.....	44
3.4.6. Codificação Binária	44
3.4.7. Visão em blocos.....	44
3.5. Verificação Funcional	46
3.6. Prototipação em FPGA.....	47
3.6.1. Abordagem <i>Multicore</i>	49
3.7. Análise de Desempenho	49
Capítulo IV: Resultados	51

4.1. Atualização do parâmetro da codificação de Golomb-Rice.....	51
4.2. Teste com bancos de imagens.....	52
4.2.1. Fragmentação em blocos (Resistência a erros)	55
4.3. Síntese em FPGA	55
4.4. <i>Throughput</i>	58
4.5. Discussão dos resultados.....	59
Capítulo V: Conclusão	63
5.1. Trabalhos Futuros	64
Referências Bibliográficas	66
Apêndice I: Resultado detalhado de compressão para o banco de imagens A	70
Apêndice II: Resultado detalhado de compressão para o banco de imagens B	80
Apêndice III: Artigo “ <i>A Low Complexity Lossless Data Compressor IP-Core for Satellite Images</i> ”.....	82
Glossário	91

Lista de Equações

Equação 1 – Taxa de compressão.....	24
Equação 2 – Razão de compressão	25
Equação 3 – Método de predição bidimensional.....	26
Equação 4 – Quociente da divisão de Golomb	27
Equação 5 – Resto da divisão de Golomb	27
Equação 6 – Mapeamento do erro de predição	28
Equação 7 – Inequação para o cálculo de k	29
Equação 8 – Modificação da média usada no cálculo de k	38
Equação 9 – <i>Throughput</i> do protótipo	50

Lista de Figuras

Figura 1: Elementos envolvidos em um sistema de imageamento.....	19
Figura 2: O espectro eletromagnético	20
Figura 3: Representação digital de uma imagem de sensoriamento remoto	21
Figura 4: Modelo de compressão genérico.	26
Figura 5: a) exemplo de borda vertical detectada; b) exemplo de borda horizontal detectada.....	26
Figura 6: Fluxo de atividades empregado no trabalho.....	35
Figura 7: Composição do método proposto baseado em Golomb-Rice.	36
Figura 8: Diagrama de classes do modelo de referência.	38
Figura 9: Amostras das imagens do banco A (a) e banco B (b).	40
Figura 10: Esquematização dos testes com bancos de imagens.	41
Figura 11: Transferências de dados usando os sinais do protocolo.	42
Figura 12: Diagrama de bloco do compressor.....	44
Figura 13: Visão em blocos do (a) compressor e (b) codificador.....	45
Figura 14: Visão geral de um testbench.....	46
Figura 15: Protótipo esquematizado.	48
Figura 16: Comparação de RC média para todas as imagens de todas as bandas do banco A.	53
Figura 17: RCs médias obtidas para imagens de 8 bits do banco B.	53
Figura 18: RCs médias obtidas para imagens de 10 bits do banco B.	54
Figura 19: RCs médias obtidas para imagens de 12 bits do banco B.	54
Figura 20: Resultado da compressão fragmentada em blocos menores para as imagens do banco A.	55
Figura 21: Quantidade de elementos lógicos utilizados devido ao aumento de núcleos de compressão.	56
Figura 22: Quantidade de memória sintetizada devido ao aumento de núcleos de compressão.....	57

Figura 23: Frequência máxima alcançada em função do aumento de núcleos de compressão.....	57
Figura 24: Variação da quantidade de ciclos de relógio por núcleos replicados.....	58
Figura 25: Influência do aumento de núcleos no Throughput do sistema.....	59

Lista de Tabelas

Tabela 1: Exemplos de códigos de Golomb.....	27
Tabela 2: RCs obtidas para comparação das abordagens de atualização do parâmetro da codificação de Golomb-Rice	52
Tabela 3: Resultados de síntese em hardware e suas comparações.....	56
Tabela 4: Comparativo de frequência e <i>throughput</i>	59
Tabela 5: Resultados de RC do trabalho proposto para o banco A.....	70
Tabela 6: Resultados de RC utilizando abordagem original de atualização do parâmetro da codificação de Golomb-Rice.	71
Tabela 7: Resultados de RC do SPIHT para o banco A.....	71
Tabela 8: Resultados de RC do JPEG-2000 para o banco A.....	72
Tabela 9: Resultados de RC do JPEG-LS para o banco A.....	72
Tabela 10: RCs obtidas da compressão em blocos de 2906x2906 do trabalho proposto.....	73
Tabela 11: RCs obtidas da compressão em blocos de 1453x1453 do trabalho proposto.....	74
Tabela 12: RCs obtidas da compressão em blocos de 726x726 do trabalho proposto.....	74
Tabela 13: RCs obtidas da compressão em blocos de 363x363 do trabalho proposto.....	75
Tabela 14: RCs obtidas da compressão em blocos de 181x181 do trabalho proposto.....	75
Tabela 15: RCs obtidas da compressão em blocos de 90x90 do trabalho proposto....	76
Tabela 16: RCs obtidas da compressão em blocos de 45x45 do trabalho proposto....	77
Tabela 17: RCs obtidas da compressão em blocos de 23x23 do trabalho proposto....	77
Tabela 18: RCs obtidas da compressão em blocos de 11x11 do trabalho proposto....	78
Tabela 19: RCs obtidas da compressão em blocos de 5x5 do trabalho proposto.	78
Tabela 20: Quadro comparativo entre algoritmos de compressão sem perdas no banco de imagens B.....	80

Capítulo I: Introdução

1.1. Motivação

O principal objetivo do sensoriamento remoto é expandir a percepção sensorial do ser humano, seja através da visão panorâmica dada pela visão aérea ou espacial, ou através da possibilidade de se obter informação do espectro eletromagnético, inacessível à visão humana. Nesse contexto, o satélite atua hoje de forma importante na coleta de dados através da captação de imagens, fazendo uso dos mais diversos tipos de sensores para esse fim. Tais satélites têm evoluído de maneira a produzirem imagens de resoluções espaciais, radiométricas e espectrais cada vez maiores (LOPES FILHO e D'AMORE, 2010).

Sendo assim, sistemas espaciais tem que lidar com um grande volume de dados, particularmente quando se trata de missões espaciais com objetivo de imageamento, ou seja, obtenção de imagens, o que representa uma maior demanda computacional, especialmente em termos de armazenamento e transmissão (YU, VLADIMIROVA e SWEETING, 2009). O aumento dos recursos de armazenamento influencia, por exemplo, no tamanho e no peso do satélite e no seu consumo de energia, os quais são pontos críticos no projeto e lançamento de um satélite no espaço.

Já em relação às transmissões entre um sistema espacial e a Terra, a disponibilidade de largura de banda para realiza-las é determinada pela combinação de vários fatores, tais como o tamanho da antena utilizada, a potência do transmissor empregado e o número de estações alocadas no solo para esse fim. Além disso, é notável o aumento no número de países participantes de programas espaciais e na quantidade de missões espaciais para os mais diversos fins, o que contribui para o constante crescimento do número de satélites no espaço, o que leva a uma concorrência, um congestionamento cada vez maior dentre os satélites em órbita da Terra. Tudo isso leva a constatação de que largura de banda tem se tornado um recurso cada vez mais escasso nesse ambiente.

Algumas soluções de problemas relacionados com armazenamento e transmissão podem ser inviáveis. Por exemplo, aumentar a quantidade de dispositivos de armazenamentos para que possa lidar com o número de imagens produzidas pelo satélite pode comprometer o seu correto funcionamento. Mais dispositivos ligados ao satélite significam um maior consumo de energia, além de tamanho e peso também maiores, indo de encontro aos limites de energia disponível e dimensões permitidas,

restrições que podem ser impostas por outras partes do satélite, como fontes de alimentação e sistemas de navegação.

Sendo assim, a redução na quantidade de dados gerados na fonte minimiza esses problemas. Por esse motivo, compressão de dados é uma solução viável e direta para tratar essas dificuldades (EUROPEAN SPACE AGENCY, 2011). Isso significa que a utilização de compressão de dados é capaz de reduzir a quantidade de dados a ser armazenada e transmitida pelos satélites, um benefício que justifica a inclusão de algoritmos, tanto em software como em hardware, com esse intuito no projeto de um satélite.

Logo, hoje em dia, sistemas de compressão de dados a bordo, ou seja, usados a bordo do satélite, são cruciais para sistemas espaciais que envolvem uma grande quantidade de dados, como no sensoriamento remoto. Por esse motivo, estudos nessa área possuem grande importância, particularmente no Brasil onde a compressão de dados espaciais é considerada um ponto crítico devido à imperícia dos órgãos espaciais brasileiros nesse assunto e a consequente dependência de tecnologia estrangeira.

Além disso, algumas missões espaciais requerem transmissão de dados à Terra em tempo real, o que demanda dos compressores de dados a capacidade de operarem em altas taxas de desempenho, isto é, os compressores devem ser capazes de comprimir dados na mesma velocidade que são produzidos pelos imageadores.

Tudo isso mostra que a escolha de uma solução de compressão de dados é guiada pelos requisitos da missão espacial em questão.

Cita-se, por exemplo, o caso brasileiro do satélite CBERS-2B, lançado em 2007 e cuja operação foi encerrada no ano de 2010, do Programa Sino-Brasileiro de Recursos Terrestres (CBERS - *China Brazil Earth Resources Satellite*) desenvolvido pelo INPE. Ele era equipado com uma câmera imageadora de Amplo Campo de Visada (WFI – *Wide Field Imager*), uma de Alta Resolução CCD (*Charge-Coupled Device*) e uma Pancromática de Alta Resolução HRC (*High Resolution Camera*). A primeira produz dados em uma taxa de 1,1 Mbit/s, a segunda em 106 Mbit/s e a terceira em 432 Mbit/s (INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS, 2007). Dentre as três câmeras, apenas a HRC utiliza algum esquema de compressão de dados, reduzindo a quantidade de dados gerados para uma vazão de saída de 58 Mbit/s.

Existem duas classes de compressores: sem e com perdas. Compressão sem perdas é aquela em que a informação original pode ser reconstruída exatamente sem

nenhuma perda, enquanto na com perdas a imagem reconstruída é diferente da original (EUROPEAN SPACE AGENCY, 2011).

Em geral, a compressão com perdas possui maior capacidade de reduzir a quantidade de dados, tendo em vista que descarta informação. Por isso, muitas vezes os requisitos de transmissão das missões espaciais obrigam a utilização de métodos de compressão com perdas. Mesmo assim, os compressores sem perdas são bastante utilizados, principalmente quando há necessidade de maior qualidade nas imagens ou no caso de transmissão de outros dados que não sejam imagens e que não permitem a introdução de perdas (YU, VLADIMIROVA e SWEETING, 2009).

Outro ponto a ser considerado nesse contexto é que o ambiente espacial está exposto a vários tipos de radiação, fator de preocupação para sistemas eletrônicos, pois favorece a ocorrência de falhas que alteram o comportamento funcional do sistema. Ao se utilizar técnicas de compressão de dados, possíveis problemas decorrentes dessas falhas são potencializados, pois os algoritmos desenvolvidos para esse fim normalmente criam dependência entre os dados (RICE, 1974). Assim, a ocorrência de uma falha qualquer leva o processo de compressão/descompressão à obtenção de dados corrompidos. Para minimizar problemas desse tipo, existem técnicas algorítmicas, chamadas de técnicas de resistência a erros, capazes de oferecer alternativas que viabilizam a utilização de compressão de dados em missões espaciais (ABRARDO, BARNI, *et al.*, 2010).

1.2. Objetivos

O objetivo geral deste trabalho consiste em desenvolver um compressor de dados sem perdas para imagens de satélite que possua um desempenho aceitável de acordo com os padrões atuais de câmeras imageadoras e que seja resistente a erros. Sendo um compressor voltado à área espacial e tendo em vista os requisitos de desempenho a serem alcançados, sua implementação será feita em hardware digital, particularmente dispositivo FPGA (*Field Programmable Gate Array*), por serem plataformas mais fáceis de projetar e prototipar, utilizando linguagens de descrição de hardware (HDLs - *Hardware Description Languages*).

Para tanto, é possível subdividir o trabalho nos seguintes objetivos específicos:

- Análise de algoritmos de compressão sem perdas para imagens de satélite, considerando seus requisitos, particularmente quanto ao desempenho e à resistência a erros;
- Implementação em alto nível do algoritmo escolhido/definido, servindo como modelo de referência;

- Realização de testes com banco de imagens de satélite com o modelo de referência para validação do modelo implementado e análise da compressão empregada;
- Implementação em HDL do algoritmo escolhido com intuito de implantá-lo em hardware;
- Utilização da metodologia BVM (*Brazil-IP Verification Methodology*) (ARAÚJO, 2010) para verificação funcional do hardware desenvolvido;
- Prototipação em FPGA do compressor proposto;
- Investigação e análise do desempenho do protótipo.

1.3. Organização

Este trabalho está organizado da seguinte forma. No Capítulo II é realizada a fundamentação teórica para introduzir os principais conceitos que são utilizados neste trabalho e servem como base para entendimento dessa dissertação, além de trabalhos correlatos que também o embasam. O Capítulo III expõe a metodologia de trabalho utilizada para fins de elaboração, implementação, verificação e prototipagem do sistema proposto. Os resultados obtidos através do modelo de referência, implementação em HDL e protótipo em FPGA são apresentados no Capítulo IV, bem como suas análises e comparações com outros trabalhos. Por fim, o Capítulo V contém a conclusão de tudo que foi apresentado. Além dos capítulos já relacionados, o texto possui nos Apêndices I e II uma série de tabelas com todas as razões de compressão obtidas para os testes realizados. Já o Apêndice III contém um artigo intitulado “*A Low Complexity Lossless Data Compressor IP-Core for Satellite Images*” aceito para apresentação no *25th Symposium on Integrated Circuits and Systems Design* (SBCCI 2012), indicativo da relevância do presente trabalho. Por fim, é possível encontrar um glossário contendo os principais termos e siglas utilizadas.

Capítulo II: Fundamentação Teórica

Este capítulo define e explica os conceitos principais envolvidos no presente trabalho. Em primeiro lugar, um tema que está relacionado com o mesmo é o sensoriamento remoto, uma vez que o trabalho trata da compressão de imagens de satélite. Sendo assim, esse tópico é abordado através da apresentação dos seus conceitos básicos e da contextualização do imageamento de satélite em um sistema de sensoriamento remoto, o que inclui o efeito da radiação e a definição do espectro eletromagnético, relacionando-os ao conceito de imagem digital, objeto de estudo desse trabalho.

Depois, é importante introduzir os conceitos da compressão de dados, tema central do trabalho em questão, e a partir deles tratar de seus tipos, aplicações e das formas de mensurar os resultados da compressão.

A compressão de dados empregada utiliza um modelo teórico arquitetado em função dos conceitos de autocorrelação e descorrelação. Tal modelo é explicado em mais detalhes, incluindo apresentação de técnica de predição adequada a imagens, conceituação da codificação de Golomb-Rice e a atualização de seu parâmetro de codificação visando melhoria do desempenho de compressão.

A união entre compressão de dados e o ambiente espacial presente no sensoriamento remoto de missões de observação da Terra possui características próprias, particularmente no que tange a propensão a erros existente nesse tipo de ambiente. Sendo assim, é importante explorar as técnicas disponíveis para auxiliar os compressores a contornar ao máximo tais situações adversas.

Por fim, se faz necessário apresentar trabalhos correlatos que de diversas formas se relacionam à proposta aqui apresentada, o que permite inserir o trabalho proposto dentro de um contexto maior. São apresentados trabalhos que contribuíram para a enunciação do presente e para a comparação de resultados, perante o estado-da-arte da bibliografia relacionada.

2.1. Sensoriamento Remoto

Sensoriamento remoto refere-se à obtenção de informação sobre um objeto sem estar em contato com ele. A informação é obtida da captura e medição da energia refletida ou emitida pelo objeto, a qual é usada posteriormente para processamento, análise e/ou aplicação (REES, 1999).

Geralmente, o processo de obtenção da informação envolve a interação entre radiação incidente e o alvo de interesse. Os sistemas de imageamento são compostos pelos seguintes elementos, ilustrados na Figura 1.

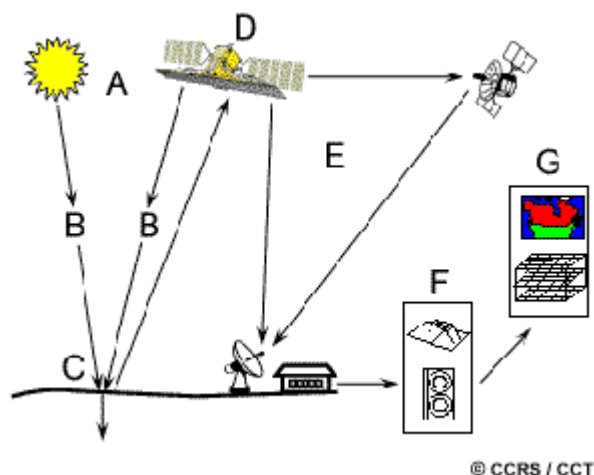


Figura 1: Elementos envolvidos em um sistema de imageamento (NATURAL RESOURCES CANADA, 2008).

- *Fonte de energia (A)* - responsável por iluminar ou fornecer energia eletromagnética ao alvo de interesse;
- *Radiação e a atmosfera (B)* - a energia interage com a atmosfera no caminho entre a fonte e o alvo e entre o alvo e o sensor;
- *Interação com o alvo (C)* - a interação entre a energia e o alvo depende das propriedades do alvo e da radiação;
- *Obtenção da energia pelo sensor (D)* - conforme a energia é emitida ou refletida pelo alvo, um sensor remoto faz a captura e a transforma em números digitais;
- *Transmissão, recepção e processamento (E)* – o sinal digital é transmitido a uma estação de recepção e processamento, onde os dados são armazenados e processados;
- *Interpretação e análise (F)* - a imagem processada é interpretada, seja visualmente por um ser humano ou automaticamente por computador, para extração de informações sobre o alvo;
- *Aplicação (G)* - a informação extraída é utilizada para melhor entendimento do objeto alvo, revelação de novas informações ou ajudar na resolução de um problema em particular.

O presente trabalho trata da compressão dos dados obtidos pelos sensores de imageamento antes de serem transmitidos à estação receptora. Logo, trata-se de uma tarefa existente entre os elementos D e E.

2.1.1. Radiação e o espectro eletromagnético

O primeiro requisito, como visto acima, é a existência de uma fonte de energia, que se apresenta na forma de radiação eletromagnética. Toda radiação

eletromagnética possui propriedades físicas fundamentais e se comporta como uma onda. Sabendo disso, duas das suas características são particularmente importantes para o sensoriamento remoto: o comprimento de onda e a frequência.

O comprimento de onda é o tamanho de um ciclo de onda, o qual pode ser medido como a distância entre dois picos sucessivos. Ele é representado pela letra grega lambda (λ) e sua unidade é o metro (m). A frequência é o número de ciclos de uma onda passando em um ponto fixo por unidade de tempo. Sua unidade é o Hertz (Hz), que equivale a um ciclo por segundo.

Comprimento de onda e frequência estão relacionados pela fórmula $c = \lambda f$, onde c é a velocidade da luz (3×10^8 m/s). Logo, uma é inversamente proporcional à outra.

Os tipos de radiação de acordo com seu comprimento de onda são organizados no que é conhecido por espectro eletromagnético. Ele varia dos menores comprimentos de onda, incluindo os raios gama e raios-x, para os maiores comprimentos de onda, incluindo micro-ondas e ondas de rádio. Todo o espectro é ilustrado na Figura 2.

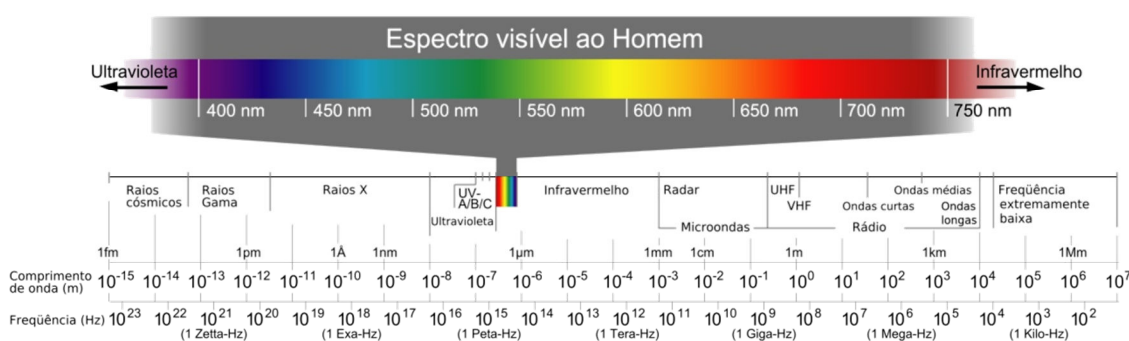


Figura 2: O espectro eletromagnético. [fonte: <http://pion.sbfisica.org.br>]

A porção ultravioleta (UV) do espectro possui os menores comprimentos de onda geralmente usados no sensoriamento remoto. Alguns materiais da superfície terrestre, principalmente rochas e minérios, brilham ou emitem luz visível quando iluminados por radiação UV.

Espectro visível refere-se à parte perceptível pelo olho humano, o que constitui apenas uma pequena porção do espectro completo. Assim, existe uma ampla faixa do espectro que não pode ser percebida pelos sentidos humanos, ou seja, é invisível. Contudo, podem ser detectadas por instrumentos de sensoriamento remoto e usadas em vantagem dos seres humanos (NATURAL RESOURCES CANADA, 2008). O espectro visível cobre uma faixa que vai aproximadamente do comprimento de onda de 0,4 μ m até 0,7 μ m. É nessa porção do espectro que se associam os conceitos de cores. O maior comprimento de onda visível é o vermelho, enquanto o menor

comprimento é o violeta. Por exemplo, o azul possui comprimento de onda entre 0,446 μm e 0,5 μm .

Depois, destaca-se a parte conhecida como infravermelho, que cobre a região de comprimento de onda aproximadamente entre 0,7 μm e 100 μm .

2.1.2. Características das Imagens de Sensoriamento Remoto

Energia eletromagnética pode ser detectada eletronicamente pelos sensores, produzindo um arranjo de valores numéricos, os quais são organizados em uma imagem. A imagem pode ser representada e visualizada em um formato digital pela subdivisão da imagem em pequenas áreas de tamanhos iguais chamadas de *pixels*, que possuem valores numéricos, captados pelo sensor, representando seu brilho. A Figura 3 apresenta um esquema da representação digital de uma dessas imagens.

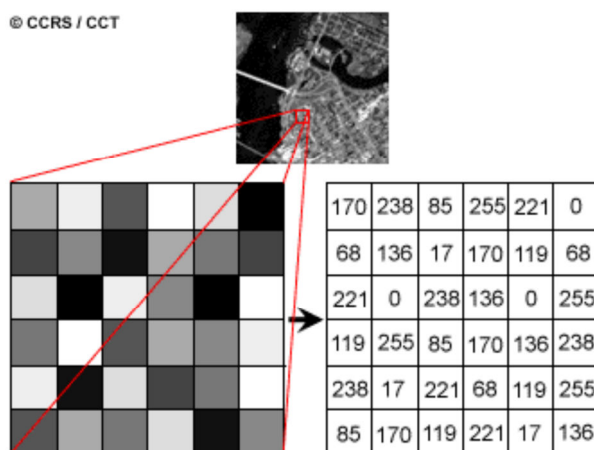


Figura 3: Representação digital de uma imagem de sensoriamento remoto (NATURAL RESOURCES CANADA, 2008).

Muitos sensores operam apenas em uma pequena faixa do espectro eletromagnético, normalmente chamada de banda ou canal. Dessa maneira, é possível combinar os dados obtidos por sensores de bandas diferentes (normalmente, as três cores primárias: vermelho, verde e azul) para formar imagens coloridas. Cada banda separadamente pode ser visualizada como uma imagem de níveis de cinza.

A distância entre o alvo a ser imageado e o instrumento imageador, juntamente com o tamanho do detector, determina o detalhamento da informação obtida e a área total imageada pelo sensor. Sensores mais distantes captam áreas maiores, porém registram menor quantidade de detalhes. Esse compromisso depende de uma característica chamada de resolução espacial, que se refere ao tamanho do menor elemento passível de ser detectado.

Enquanto o arranjo de pixels descreve a estrutura espacial de uma imagem, as características radiométricas descrevem o conteúdo de interesse real em uma imagem. Para toda aquisição de uma imagem, seja ela feita por um filme ou por um

sensor, a sensibilidade à magnitude da energia eletromagnética determina a resolução radiométrica, a qual descreve a habilidade de um sistema de imageamento de discriminar as variações na energia. Quanto maior for a resolução radiométrica de um sensor mais sensível ele é na detecção de diferenças na energia refletida ou emitida.

Dados de imageamento são representados por números digitais positivos, os quais variam entre zero e um certo valor máximo. Esse intervalo corresponde ao número de bits usados para codificar os números no formato binário. O número máximo usado na representação dos níveis de brilho depende de quantos bits são usados na representação da energia captada.

Existem também os conceitos de resolução espectral e resolução temporal. A primeira é definida pela faixa do espectro eletromagnético que é usada na captura de uma banda de uma imagem. Já a segunda refere-se ao intervalo de tempo de aquisição de uma mesma área de um alvo na superfície terrestre.

2.2. Compressão de Dados

Compressão de dados é o processo de converter um bloco de dados de entrada em um bloco de dados de saída de menor tamanho (SALOMON, 2004). O bloco de dados pode ser um arquivo qualquer, uma imagem, um vídeo, um áudio ou um fluxo de dados enviado pela Internet. E em cada caso pode existir uma técnica específica para compressão dos dados.

Existem duas razões principais para a aplicação da compressão de dados. A primeira está ligada ao armazenamento de dados. A redução da quantidade de dados permite aumentar a capacidade de dispositivos de armazenamento, tais como HD, *pendrive*, etc. Em segundo lugar, ela favorece a redução nos tempos de transferência de dados, como, por exemplo, no carregamento de uma página da Web ou durante o *download* de um arquivo da Internet.

Dados são empregados com o intuito de representar alguma informação, a qual pode ser organizada na forma de símbolos. Dependendo do contexto, um símbolo pode ser uma letra, uma frase inteira, um bit, um pixel, etc. Dessa forma, os símbolos servem para codificar as informações em dados. Os vários métodos de compressão de dados utilizam-se de cada um desses casos para explorá-los com o objetivo de reduzir a quantidade de dados. Isso se dá devido ao fato de que geralmente existe redundância na codificação de uma informação em dados.

Por exemplo, em textos da língua portuguesa, comumente a letra A é bem mais frequente que a letra Z. Esse comportamento é chamado de redundância alfabética. Outro tipo é a redundância contextual. Ela acontece, por exemplo, quando ocorre uma letra Q em um texto da língua portuguesa, já que quase sempre ela será seguida por

uma letra U. Da mesma maneira, existem diversos outros casos de redundâncias em outras aplicações, como imagens, áudios, etc.

A ideia de compressão por redução das redundâncias sugere a utilização de codificação menor (códigos de comprimento menor) para símbolos que ocorram com maior frequência e códigos maiores para símbolos que ocorram raramente. Uma análise dos vários métodos de compressão mostra que todos eles estão relacionados com essa proposição (SALOMON, 2004).

A maioria dos métodos de compressão é classificada em quatro categorias principais:

- Codificação *Run-Length* (RLE): Uma das formas mais simples de compressão de dados aproveita a repetição de símbolos como fonte de redundância. Não é uma técnica poderosa já que nem sempre é possível encontrar muita repetição de símbolos em um arquivo. Contudo, é bastante utilizada como pré-processamento para outros algoritmos de compressão (AL-LAHAM e EMARY, 2007);
- Métodos Estatísticos: Na tentativa de codificar eficientemente as redundâncias, tais métodos utilizam a estatística de ocorrência dos símbolos, que pode ser estática ou dinâmica. Dentre as técnicas pertencentes a essa categoria, cita-se, por exemplo, a codificação de Huffman (HUFFMAN, 1952);
- Métodos Baseados em Dicionário: Ao ler dados de entrada, um método baseado em dicionário procura por símbolos agrupados que façam parte de um dicionário. Cada ocorrência do dicionário no arquivo de entrada é representada no arquivo de saída pelo índice ou apontador daquele grupo dentro do dicionário ao invés de um código. Quanto maior for o grupo melhor será o desempenho da compressão. Similarmente às probabilidades dos métodos estatísticos, um dicionário pode ser estático ou dinâmico. Um exemplo de método baseado em dicionário é o LZW (WELCH, 1984);
- Métodos Baseados em Transformadas: A ideia por trás dos métodos baseados em transformadas é submeter os dados a uma transformação matemática e, assim, reestruturá-los de tal maneira que eles fiquem mais adequados à compressão propriamente dita por parte de um segundo estágio do método como um todo. Por se tratarem de processos em duas etapas, tendem a ser métodos mais lentos ao passo em que alcançam melhores resultados de compressão. Alguns exemplos são explorados em mais detalhes neste capítulo, principalmente quanto ao uso da transformada *wavelet* (SAID e

PEARLMAN, 1996) (KIELY e KLIMESH, 2003) (ISO/IEC, 2004) (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007);

Os métodos ainda podem ser divididos em não adaptativos, semi-adaptativos e adaptativos. Um método não adaptativo é rígido, ou seja, não modifica suas operações, seus parâmetros ou suas tabelas de acordo com os dados particulares que estão sendo comprimidos em um determinado momento. Os semi-adaptativos são aqueles que antes de modificarem seus parâmetros, adaptando-se aos dados de entrada, leem todos os dados que serão comprimidos. E, por fim, os adaptativos são capazes de, em tempo de execução, modificarem parâmetros ajustando-os aos dados de entrada em busca de melhores resultados.

Existe também a separação entre métodos de compressão com perdas e sem perdas. A introdução de perdas aceitáveis no processo de compressão é muitas vezes utilizada para alcançar maiores níveis de compressão. Normalmente, perdas estão associadas à compressão de imagens, vídeos e áudios. Isso acontece porque as perdas introduzidas muitas vezes são imperceptíveis pelos seres humanos. Por exemplo, cita-se o MP3, técnica que revolucionou o mercado mundial musical, que remove frequências de áudio quase imperceptíveis ao sistema auditivo humano.

Contudo, para certos casos, as perdas em compressão de dados podem ser inaceitáveis. É o caso de um arquivo executável, por exemplo, uma vez que a decodificação de um desses arquivos diferentemente do original levará a produção de um arquivo corrompido e que não terá mais utilidade. A compressão sem perdas é também importante na área biológica, particularmente quando relacionada a diagnósticos médicos (OGIELA e TADEUSIEWICZ, 2008), e espacial, pois a introdução de perdas em algumas missões espaciais não é desejável, como, por exemplo, em dados científicos e imagens de alta precisão (YU, VLADIMIROVA e SWEETING, 2009) (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006).

As principais formas de se mensurar os resultados de compressão de dados são a taxa de compressão (TC) e a razão de compressão (RC). A taxa de compressão é definida como

$$TC = \frac{\text{número de bits dos dados comprimidos}}{\text{número de bits dos dados originais}} \quad \text{Equação 1}$$

Valores maiores que um indicam que não ocorreu compressão, mas sim expansão. A taxa de compressão pode ser apresentada na forma de porcentagem. Por exemplo, uma taxa de compressão de 0,6 significa que os dados após a

compressão ocupam 60% do tamanho dos dados originais. A RC é o inverso da taxa de compressão,

$$RC = \frac{\text{número de bits dos dados originais}}{\text{número de bits dos dados comprimidos}} \quad \text{Equação 2}$$

Nesse caso, para um valor menor que um, ocorreu expansão ao invés de compressão (SALOMON, 2004).

2.2.1. Autocorrelação, Descorrelação e Modelo de Compressão Genérico

Existe uma medida de similaridade que é capaz de avaliar o grau de redundância em um conjunto de símbolos usados para representar informação: a autocorrelação (RUSS, 2011). Assim, usando o exemplo de uma imagem qualquer e sabendo que comumente existe redundância espacial, já que pixels vizinhos geralmente possuem certo grau de similaridade, diz-se que os pixels estão autocorrelacionados. Conclui-se que um alto grau de autocorrelação indica similaridade entre o grupo de símbolos de uma representação e, conseqüentemente, a existência de redundância, o que favorece a compressão de dados (SALOMON, 2000).

Sendo assim, compressão de dados pode ser alcançada através da tentativa de remover ou reduzir a correlação existente em um conjunto de dados, ao que se dá o nome de descorrelação (SALOMON, 2001). Após a etapa de descorrelação, os algoritmos utilizam algum tipo de codificação que melhor explore a característica da informação residual no intuito de reduzir a quantidade de bits usados no arquivo de saída.

Considerando o caso específico de imagens, já que o presente trabalho tem esse foco, é possível visualizar na **Erro! Autoreferência de indicador não válida.** um modelo de compressão genérico baseado no princípio da descorrelação aqui exposto. Considere a imagem original x_0 . Ela é submetida a uma etapa de descorrelação com o objetivo de reduzir a correlação entre os pixels, a redundância entre eles, produzindo uma imagem descorrelacionada x_d . Essa, por sua vez, é utilizada na etapa de codificação, que tenta reduzir a quantidade de bits empregada na representação dos símbolos, principalmente naqueles mais frequentes como sugerido anteriormente, resultando na imagem final comprimida x_c representada por uma sequência de bits (MEIRA, LIMA e BATISTA, 2005).

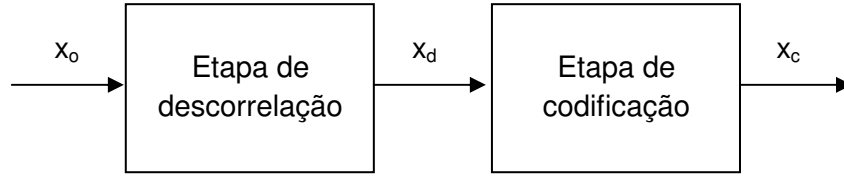


Figura 4: Modelo de compressão genérico.

Codificadores desse tipo dependem de um modelo estatístico que descreva a sequência a ser codificada. Quando os pixels são fortemente correlacionados, altos ganhos de compressão podem ser alcançados se o codificador for baseado em um modelo contextual que capture corretamente essa correlação.

2.2.1.1. Predição

Uma maneira muito comum de se realizar descorrelação é através de técnicas de predição por oferecer um bom compromisso entre simplicidade e ganho de compressão (ZHANG, LIU e YU, 2001). Quando existe uma alta correlação entre um grupo de pixels e, por conseguinte, uma grande redundância entre eles, é razoável que o valor de certo pixel desse grupo seja predito a partir dos valores dos seus vizinhos (PECHYONYJ e BOGOMJAKOV, 1998). Sendo assim, pode-se definir uma técnica de predição que estima o valor do pixel $x_{i,j}$, na linha i e coluna j da imagem, como o valor $\widetilde{x}_{i,j}$, baseada em sua vizinhança.

A seguinte técnica de predição é apresentada por ser a empregada no trabalho atual e é baseada no que propõe o padrão JPEG-LS (ISO/IEC, 1999). Trata-se de uma predição bidimensional baseada na detecção de bordas e que pode ser formalizada pela fórmula a seguir.

$$\widetilde{x}_{i,j} = \begin{cases} \min(x_{i,j-1}, x_{i-1,j}) & \text{se } x_{i-1,j-1} \geq \max(x_{i,j-1}, x_{i-1,j}), \\ \max(x_{i,j-1}, x_{i-1,j}) & \text{se } x_{i-1,j-1} \leq \min(x_{i,j-1}, x_{i-1,j}), \\ x_{i,j-1} + x_{i-1,j} - x_{i-1,j-1} & \text{caso contrário.} \end{cases} \quad \text{Equação 3}$$

Por exemplo, na Figura 5.a, $\widetilde{x}_{i,j} = 100$ já que uma borda vertical foi detectada. Já na Figura 5.b, $\widetilde{x}_{i,j} = 102$ pois uma borda horizontal foi detectada.

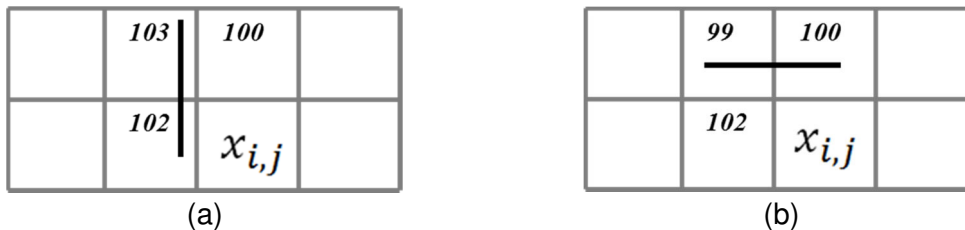


Figura 5: a) exemplo de borda vertical detectada; b) exemplo de borda horizontal detectada.

Por fim, é realizado o cálculo do erro de predição $e_{i,j} = x_{i,j} - \widetilde{x}_{i,j}$, que corresponde a remoção da redundância encontrada pelo esquema de predição. Os

valores de erro de predição para cada pixel servem à próxima etapa como informação residual para que sejam codificados.

2.2.1.2. Codificação de Golomb-Rice

Como etapa de codificação, o presente trabalho trata da codificação de Golomb-Rice. Inicialmente proposto na década de 60 por Solomon W. Golomb (GOLOMB, 1966), os códigos de Golomb correspondem a uma família de codificações de código de prefixo com comprimento ótimo para determinadas distribuições de probabilidade dos símbolos a serem codificados (SEROUSSI e WEINBERGER, 1997).

O esquema de codificação de Golomb pode ser explicado da seguinte maneira. Considere o número inteiro não negativo n a ser codificado. Primeiro, é computado o quociente da seguinte divisão, onde m é um parâmetro inteiro maior que zero:

$$q = \left\lfloor \frac{n}{m} \right\rfloor \quad \text{Equação 4}$$

Depois, calcula-se o resto dessa divisão:

$$r = n - qm \quad \text{Equação 5}$$

O código é composto pela concatenação de duas partes. A primeira é a codificação unária de q , ou seja, corresponde a repetição de q bits '1' e um bit '0', o qual é utilizado como delimitador entre as duas partes.

A segunda parte depende de r e $b = \lceil \log_2 m \rceil$. Se $0 \leq r < 2^b - m$, ela é formada pelos $b - 1$ bits menos significativos de r . Por outro lado, se $2^b - m \leq r < m$, ela é formada pelos b bits menos significativos de r .

Escolhendo $m = 3$, por exemplo, temos $b = 2$ e r pode assumir os valores 0, 1 e 2. Para $0 \leq r < 1$, temos o código '0' quando $r = 0$. Caso contrário, $1 \leq r < 3$, temos os códigos '01' e '10' quando $r = 1$ e $r = 2$, respectivamente. A Tabela 1 mostra alguns exemplos de códigos de Golomb para $m = 3$ e $m = 5$.

Tabela 1: Exemplos de códigos de Golomb.

n	0	1	2	3	4	5	6	7	8
$m = 3$	00	001	010	100	1001	1010	1100	11001	11010
$m = 5$	000	001	010	0011	0100	1000	1001	1010	10011

Pesquisas mostram que a utilização de esquemas de predição juntamente com codificação de Golomb para compressão sem perdas, especialmente para imagens, alcançam taxas próximas a outros métodos mais complexos como, por exemplo, a codificação aritmética (HOWARD e VITTER, 1994) (WEINBERGER, SEROUSSI e SAPIRO, 1996). Esquemas de predição têm sido amplamente utilizados por capturar bem a relação entre pixels e ser de baixa complexidade.

Um detalhe importante é que o erro de predição pode assumir valores negativos. Porém, como foi visto, a codificação de Golomb trabalha apenas com números não negativos. Para suplantar esse problema, é necessário realizar uma etapa intermediária entre a predição e a codificação.

Os valores negativos podem ser mapeados como valores positivos, intercalando-os com os “reais” valores positivos, diferenciando-os entre ímpares e pares, respectivamente. Mais formalmente, o erro de predição e é mapeado de acordo com:

$$M(e) = \begin{cases} 2e & \text{se } e \geq 0, \\ 2|e| - 1 & \text{se } e < 0. \end{cases} \quad \text{Equação 6}$$

Foi observado que um bom modelo probabilístico para modelar erros de predição é dado por uma distribuição geométrica bilateral com centro em zero (O'NEAL JR., 1966). Depois do mapeamento, essa distribuição corresponde a uma aproximação de uma distribuição geométrica unilateral (SEROUSSI e WEINBERGER, 1997). Isso significa que existe maior chance de ser codificado um valor mais próximo de zero, o que produz menos bits, como resultado da codificação.

Posteriormente, Robert F. Rice estudou o subconjunto dos códigos de Golomb onde $m = 2^k$, que passou a ser conhecido por codificação de Golomb-Rice (RICE, 1979). Nesse caso, como o m sempre é uma potência de dois, obtém-se duas vantagens principais em termos de implementação:

- q pode ser calculado através da operação de deslocamento de bits para a direita ($q = n \gg k$);
- r corresponde aos k bits menos significativos de n , pois o resto da divisão é justamente os bits que foram perdidos durante a operação de deslocamento a direita.

Por outro lado, a redução do número de possibilidades para o parâmetro m diminui a proximidade da otimalidade dos comprimentos dos códigos de Golomb. Mesmo assim, o comprimento médio dos códigos, dependendo da escolha do parâmetro, ainda é razoavelmente próximo às abordagens mais amplas (SEROUSSI e WEINBERGER, 1997).

Sendo assim, é importante que o compressor mantenha o parâmetro m ou k muito bem ajustado à produção dos menores comprimentos dos códigos.

2.2.1.2.1. Cálculo do parâmetro k

Quanto menor for o parâmetro k menor será o menor comprimento alcançável por um código. Porém, ao mesmo tempo, menor será o número de códigos possíveis

com comprimento baixo. Assim, valores baixos de k produzem comprimentos de códigos pequenos para valores pequenos a serem codificados. Porém, se codificarmos valores grandes com um parâmetro k baixo, ele irá produzir mais bits que se ajustássemos k para um valor maior mais adequado. Por exemplo, para $k = 0$, que corresponde apenas à codificação unária, o valor zero é codificado como '0' e o valor três como '1110'. Porém, se mudarmos para $k = 1$, zero é codificado como '00' (um bit a mais) e três como '101' (um bit a menos).

Isso sugere que o parâmetro seja ajustado em tempo de execução dependendo dos valores de entrada. Dado uma sequência de símbolos de entrada que seguem uma distribuição Laplaciana, em (MEIRA, LIMA e BATISTA, 2005) calcula-se o valor de k como $\lceil \log_2 a' \rceil$, onde a' é a média aritmética dos valores absolutos da sequência de erros de predição. Assim, a cada novo valor de entrada, a' é atualizado e k é recalculado. Para facilitar a implementação em hardware, k é calculado por meio de estruturas condicionais a partir da inequação abaixo.

$$t \cdot 2^k \geq a, \quad \text{Equação 7}$$

onde t é a quantidade de erros de predição codificados até então e a a soma dos valores absolutos desses erros. Dessa forma, evita-se o cálculo do logaritmo e da divisão $a' = \frac{a}{t}$.

2.3. Compressão de Dados, Propagação e Resistência a Erros

Uma preocupação notável é quanto à exposição à radiação presente no ambiente espacial e que é capaz de alterar o comportamento de componentes eletrônicos pela ocorrência de diversos tipos de falhas. Além disso, a transmissão de dados no espaço é bem mais propensa a erros do que em situações ordinárias.

Quando do emprego de um sistema que não utilize compressão de dados, a ocorrência de falhas geralmente afeta os dados recebidos na estação destino apenas nos símbolos envolvidos com a parte corrompida dos dados, o que por si só já é um fato indesejado. Contudo, sistemas que adotam algum tipo de compressão de dados podem ter os efeitos dos erros potencializados (RICE, 1974). Isso ocorre principalmente em algoritmos de compressão que se utilizam de algum tipo de dependência entre símbolos no decorrer do processo de compressão. Por exemplo, na técnica de predição apresentada anteriormente, ocorre a atualização de um parâmetro de codificação. A existência de uma falha em um único bit conduzirá à atualização errada de tal parâmetro, que por sua vez influenciará negativamente todas as atualizações consequentes ao propagar tal erro.

É possível fazer uso de componentes eletrônicos especialmente fabricados para resistirem aos efeitos da radiação. Contudo, em se tratando de processos específicos, caros e que limitam a capacidade performática do circuito produzido, tem sido cada vez mais incentivada à utilização de técnicas de resistência a erros no nível algorítmico, permitindo o barateamento e aumento de desempenho dos sistemas espaciais ao utilizarem componentes comuns (ABRARDO, BARNI, *et al.*, 2010).

Como aponta (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006), a forma mais simples e direta de se alcançar tal resistência é limitar a propagação dos erros através da fragmentação dos dados em blocos menores, os quais são comprimidos independentemente uns dos outros. Estratégias equivalentes são utilizadas em diversos outros trabalhos, como em (LOPES FILHO e D'AMORE, 2010), (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007), (KIELY e KLIMESH, 2003) e (ABRARDO, BARNI, *et al.*, 2010). A redução nos efeitos da propagação de um erro está presente no fato de que a descompressão errada é limitada apenas ao bloco afetado pelo erro. (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006) e (RICE, 1974) tratam ainda da importância dos sistemas aplicarem esquemas de codificação robustos a nível de canal de comunicação, como os códigos de Reed-Solomon, para reduzir a probabilidade de ocorrência de erros de comunicação.

Contudo, há um custo ao se fazer uso desse tipo de técnica. A limitação da compressão a uma quantidade menor de símbolos, uma vez que os dados são fragmentados em blocos menores, conduz a uma menor eficiência em termos de compressão, particularmente para compressores adaptativos e contextuais que normalmente obtém melhores resultados em função do tamanho do contexto envolvido.

2.4. Trabalhos Correlatos

O entendimento de técnicas de compressão utilizadas para fins espaciais é necessário para realizar um levantamento das vantagens e desvantagens dos métodos de compressão sem perdas de imagens de satélite.

(MEIRA, LIMA e BATISTA, 2005) trata da compressão sem perdas em FPGA de sinais de eletrocardiogramas utilizando predição linear e codificação de Golomb-Rice. Os autores demonstram a simplicidade da codificação utilizada, porém, por manusear sinais unidimensionais, a técnica de predição linear não é apropriada para imagens, que por serem sinais bidimensionais, são mais bem exploradas pelos compressores através de predição bidimensional. Além disso, eles apresentam o embasamento matemático da codificação de Golomb-Rice e detalhes importantes

quanto ao mapeamento dos erros de predição e atualização do valor do parâmetro k . Complementarmente, (MARQUES, GAMBARRA, *et al.*, 2011) apresentam uma proposta de compressão sem perdas, também em FPGA, que emprega uma técnica baseada em codificação Gray, modelagem PPM (*Prediction by Partial Matching*) e codificação aritmética. Embora o compressor seja de propósito geral, ele é aplicado em sinais biológicos e imagens médicas, obtendo bons resultados. Um ponto forte desse trabalho é que ele baseia-se em uma metodologia robusta ligada ao programa Brazil-IP, principalmente quanto à verificação funcional, e que permitiu que o compressor fosse devidamente prototipado em FPGA, projetado e fabricado em um chip ASIC (*Application Specific Integrated Circuit*). Por outro lado, sua utilização considerando os requisitos espaciais foi descartada por não ter foco no desempenho necessário para tal aplicação.

(LOPES FILHO e D'AMORE, 2010) implementaram em FPGA o padrão JPEG-LS (ISO/IEC, 1999), o qual define um conjunto de métodos de compressão sem perdas e quase sem perdas (*near-lossless*) para imagens baseado no algoritmo LOCO-I (*Low Complexity Lossless Compression for Images*). Ele possui um parâmetro de controle que define a quantidade de perdas aceitas na compressão. O valor zero significa que a compressão é sem perdas enquanto que o aumento desse valor indica que é possível alcançar maiores razões de compressão ao custo de ocorrer maior degradação na imagem descomprimida resultante. A modelagem e codificação variam contextualmente. O algoritmo seleciona modos diferentes de operação mais adequados a cada caso dependendo de uma região de vizinhança do pixel a ser codificado. Além disso, ele utiliza predição bidimensional e codificação de Golomb. Esse trabalho empregou um conjunto de imagens do satélite CBERS-2B e obteve, para o caso sem perdas, no pior caso, uma RC de 2,3. Adicionalmente, o trabalho também se preocupa com o problema da propagação de erros. No intuito de combatê-la, realizou a compressão das mesmas imagens reiniciando o processo após algumas linhas e verificou a influência do reinício na razão de compressão, em comparação com o processo anterior que comprimiu a imagem inteira sem reinício. Por exemplo, ao reiniciar o processo de compressão a cada cinco linhas, a RC caiu para 2,1. A arquitetura de hardware implementada considera os casos sem perdas e o quase sem perdas para o parâmetro igual a um, e usa memória RAM para armazenar os pixels de uma linha da imagem, necessários à modelagem do contexto. Seu desempenho foi analisado para um protótipo em FPGA.

Existe um comitê composto pelas principais agências espaciais do mundo, dentre elas a NASA (*National Aeronautics and Space Administration*) e o INPE, criado

com o intuito de desenvolver padrões para a comunicação e sistemas de dados espaciais. Um dos documentos produzidos pelo comitê é um relatório que trata da compressão de imagens (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007) e define um algoritmo em particular para esse propósito baseado na Transformada Wavelet Discreta (DWT – *Discrete Wavelet Transform*) e também na codificação de Golomb-Rice, que suporta tanto compressão com perdas como sem perdas. Essa proposta também utiliza uma abordagem de fragmentação da imagem em blocos com o intuito de limitar os efeitos dos erros inerentes ao ambiente espacial. Para avaliar o desempenho do compressor em termos de razão de compressão, o documento compara os resultados obtidos para um conjunto de imagens com os seguintes outros algoritmos: JPEG-2000, JPEG-LS, CCSDS/Rice, SPIHT e ICER.

O JPEG-2000 também é um algoritmo baseado em codificação de dados da transformada wavelet, podendo trabalhar com números inteiros ou de ponto flutuante para obter compressão com e sem perdas (ISO/IEC, 2004). Comparando com a recomendação do comitê CCSDS (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007), o JPEG-2000 tende a realizar melhor compressão ao custo de possuir maior complexidade.

CCSDS/Rice refere-se à outra recomendação do comitê (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006), voltada para a compressão sem perdas de dados espaciais de uma maneira geral, não apenas imagens. É um algoritmo baseado em predição linear e simultâneas variações da codificação de Rice.

SPIHT (*Set Partitioning In Hierarchical Trees*) é um método de compressão progressivo para imagens, que explora as características da transformada *wavelet* para tornar a codificação mais eficiente ao usar uma abordagem simples através da quantização escalar uniforme (SAID e PEARLMAN, 1996). A codificação é aritmética e utiliza uma técnica adaptativa e baseada em contexto. Assim, ele promete ser um compressor rápido e eficaz.

Outro adepto da compressão progressiva de imagens baseado em transformada *wavelet* é o algoritmo ICER, que foi o método utilizado na missão de veículos exploradores de Marte (*Mars Exploration Rovers*) para a compressão com e sem perdas de imagens capturadas pelas câmeras a bordo de tais veículos (KIELY e KLIMESH, 2003). Além disso, os dados provenientes da transformada são processados por um codificador que opera a partir de um modelo estatístico e contextual. Outro ponto importante é a fragmentação da imagem em blocos

independentes como forma de reduzir os efeitos de possíveis erros provenientes do ambiente espacial.

(WANG, ZENG, *et al.*, 2010) apresentam uma implementação em FPGA do padrão JPEG-LS que tenta alcançar menor complexidade e maior *throughput* ao modificar o modelo de predição e remover a realimentação do ciclo de atualização dos parâmetros de contexto. Ao alcançar taxas altas de desempenho, propõe que tal implementação seja utilizada com imagens de satélite.

Outro exemplo de abordagem de compressão em hardware é encontrada em (CHEN, CANAGARAJAH, *et al.*, 2007), que trata de um compressor sem perdas baseado em predição, contexto e modelagem estatística. Ele define uma arquitetura dinamicamente reconfigurável para esse fim ao mesmo tempo em que visa à baixa complexidade e *throughput* alto ao utilizar um esquema de *pipeline*. Tal abordagem foi implementada e avaliada para um FPGA da empresa Xilinx.

É possível ainda encontrar algumas soluções comerciais relacionadas à compressão de imagens em hardware. Por exemplo, (CAST, 2007) oferece um IP-Core codificador JPEG-LS desenvolvido para operar em sistemas de alto desempenho, como em HDTV, oferecendo taxas de até 161 MHz para vários fabricantes de FPGA e diversas interfaces de comunicação para entrada e saída de dados.

Capítulo III: Sistema Proposto e Metodologia

No intuito de propor uma solução de compressão de imagens de satélite em hardware que possua vantagens em relação a outros métodos existentes e, conseqüentemente, relevância para o seu contexto técnico-científico, é importante que se sigam determinadas atividades, tanto para que os objetivos traçados no Capítulo I sejam alcançados como para que o trabalho realizado esteja dentro de um nível de qualidade e confiabilidade desejável.

Nesse intento, foi definida uma sequência lógica de passos que trata da escolha do algoritmo, sua análise dentro da conjuntura do trabalho, implementação e avaliação do compressor em hardware, culminando por fim nos resultados obtidos, baseados em todo o trabalho realizado, e suas devidas conclusões.

O ponto de partida para o prosseguimento do trabalho corrente incide sobre a escolha do algoritmo, denotando sua importância central dentro dessa proposta. Tal escolha passa por questões fundamentais aqui levantadas, como implementação em hardware e aplicabilidade a sistemas espaciais, levando em consideração seus requisitos inerentes.

Mesmo em se tratando de um trabalho de desenvolvimento de um compressor em hardware, primeiramente se fez necessário conceber sua devida implementação em alto nível, o que constituiu mais um passo necessário para o método aqui proposto. O modelo em alto nível, também chamado de modelo de referência, é particularmente importante para que testes com imagens de satélites sejam conduzidos, visando à avaliação do compressor proposto.

Posteriormente, também foi concebido o modelo de hardware propriamente dito, o que foi feito através de uma linguagem de descrição de hardware ou HDL (*Hardware Description Language*). Ele foi, então, submetido à validação de suas funções e de seu desempenho, principalmente via prototipagem em FPGA. Tudo isso favoreceu o estabelecimento de uma análise final do compressor proposto e todo o método usado para se chegar até ele.

Diante do exposto, as atividades aqui descritas são relacionadas, para fins de organização, em sete pontos bem definidos: escolha do algoritmo, implementação do modelo de referência, teste com bancos de imagens, implementação em HDL, verificação funcional, prototipação em FPGA e análise de desempenho. Graficamente, a Figura 6 apresenta a sequência de atividades definida.

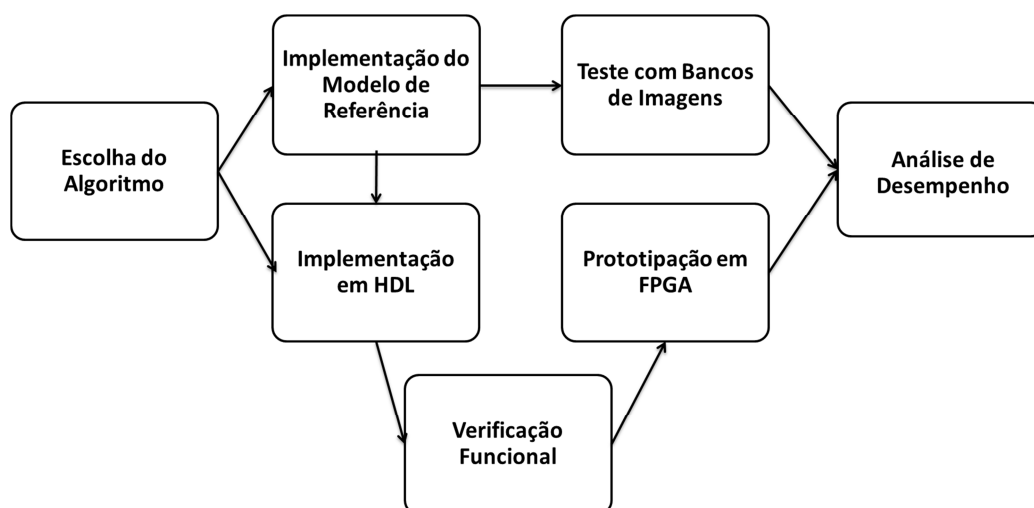


Figura 6: Fluxo de atividades empregado no trabalho.

3.1. Escolha do Algoritmo

Segundo (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006), para se escolher um algoritmo de compressão para sistemas espaciais devem-se observar os seguintes requisitos:

- Estar preparado para se adaptar às mudanças na estatística dos dados para maximizar seu desempenho de compressão;
- Possuir reduzido número de passos de processamento, quantidade de memória necessária e energia consumida;
- Ser um procedimento que não necessite operar sobre a mesma imagem mais de uma vez;
- Favorecer a montagem de pacotes independentes pelo sistema de telemetria, responsável por transmitir os dados para as estações de recebimento no solo;
- Limitar os efeitos dos erros do canal de comunicação;

Analisando os pontos acima, percebe-se que o algoritmo em questão deve ser adaptativo, de baixa complexidade, de tempo real e resistente a erros.

Dentre várias técnicas de compressão sem perdas, algumas das quais citadas no Capítulo II, a codificação de Golomb-Rice tem sido amplamente usada em sistemas espaciais, principalmente devido à compatibilidade com os requisitos apresentados anteriormente (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006), mesmo que as razões de compressão sejam menores que em outros métodos encontrados na literatura.

Sendo assim, uma abordagem baseada em predição e codificação de Golomb-Rice, a partir do modelo de compressão genérico apresentado no Capítulo II, foi escolhida por possuir simplicidade nas operações, o que favorece a baixa

complexidade do sistema, implementação em hardware e facilita a obtenção de altas taxas de dados. Além disso, paralelamente, a técnica de predição juntamente com a atualização do parâmetro de codificação representam uma maneira de adaptar-se à natureza dos dados em busca de bons resultados de compressão. Por fim, esse modelo permitiu a utilização da técnica de fragmentação da imagem em blocos menores independentes, favorecendo a inclusão de resistência a erros ao sistema ao mesmo tempo em que admite a divisão dos dados comprimidos em pacotes pelo sistema de telemetria para fins de transmissão. Essas características estão de acordo com os requisitos apresentados para sistemas espaciais.

Mesmo com algumas aplicações do Golomb-Rice já difundidas nesse contexto, como em (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2006) e no padrão JPEG-LS (WEINBERGER, SEROUSSI e SAPIRO, 1996) (ISO/IEC, 1999) (LOPES FILHO e D'AMORE, 2010), o presente trabalho propõe uma nova abordagem baseada em (MEIRA, LIMA e BATISTA, 2005), por possuir menor complexidade, o que favorece a obtenção de melhor vazão de dados, menor uso de recursos computacionais e memórias, e menor consumo de energia.

O compressor é composto por um preditor, como etapa de decorrelação, e um codificador, conforme o modelo de compressão genérico. A Figura 7 sumariza tal composição.

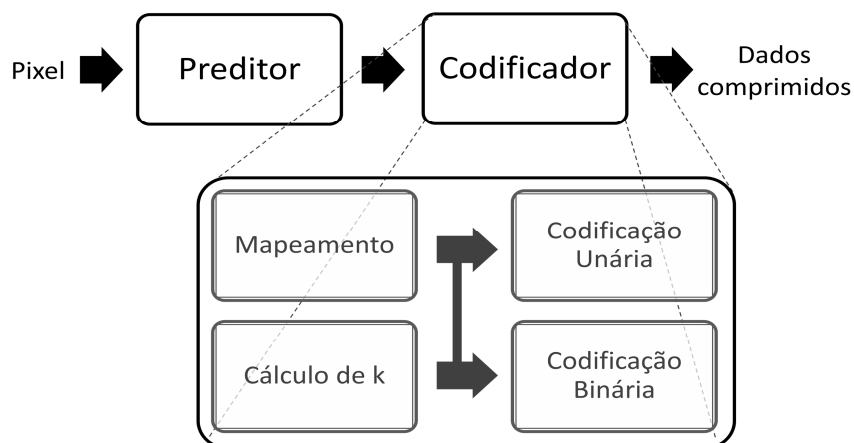


Figura 7: Composição do método proposto baseado em Golomb-Rice.

A técnica de predição empregada é aquela baseada em detecção de bordas apresentada na seção 2.2.1.1. Sua escolha é justificável pela simplicidade em termos de operações ao mesmo tempo em que alcança bons resultados por empregar uma abordagem direta de detecção de bordas, evitando que a predição seja feita baseada em vizinhos com uma variação grande de valor com relação ao pixel atual.

Como etapa de codificação, o compressor baseia-se na codificação de Golomb-Rice (codificações unária e binária), que utiliza o mapeamento dos valores de

erro dentro do intervalo positivo e, adaptativamente, calcula o parâmetro k para cada novo pixel de entrada.

3.2. Implementação do Modelo de Referência

Com uma base teórica para o modelo de compressão já estabelecida a partir da escolha do algoritmo, é possível e desejável fazer sua implementação de referência utilizando uma linguagem de alto nível moderna. Essa etapa é importante devido a três motivos principais. Primeiramente, facilita não só o entendimento mas também a experimentação de detalhes de implementação visando o modelo de hardware, tais como lógicas de controle e estruturas de dados usadas. Em segundo lugar, por meio de uma versão estável do modelo de alto nível, é possível realizar testes de desempenho de compressão utilizando imagens reais de satélite e avaliar como modificações na implementação interferem no desempenho. Por fim, o modelo de referência é um componente fundamental na etapa de verificação funcional.

De acordo com os dois primeiros motivos acima, Java, enquanto linguagem orientada a objetos de propósito geral, foi escolhida por ser uma linguagem fácil, versátil e que possui uma vasta biblioteca de classes, o que favoreceu o processo de aprendizagem e experimentação do algoritmo bem como o teste com imagens de satélite. Já para a etapa de verificação funcional, se fez necessário a utilização de uma linguagem específica para esse fim, o SystemVerilog, que foi escolhido principalmente por possuir suporte à orientação a objetos, o que permitiu a adaptação fácil e direta do código Java.

Uma das ideias principais por trás do modelo de referência é implementar tanto a funcionalidade de compressão como a de descompressão, para que seja possível comprovar seu correto funcionamento através da comparação da imagem inicial original com a imagem final descomprimida. Como se trata de compressão sem perdas, as imagens devem ser idênticas.

Em alto nível, o compressor e o descompressor podem ser visualizados da seguinte forma. Uma imagem original é usada pixel a pixel, os quais servem para a realização da predição. Subtrai-se tal resultado do pixel original para se obter o erro de predição, que é utilizado para a codificação, salvando-o devidamente em um arquivo comprimido. Na codificação, cada valor de erro é mapeado antes de ser codificado unariamente e binariamente, baseado no valor de k , que é atualizado a cada passo.

Similarmente, o descompressor produz uma imagem final descomprimida. Para isso, utiliza-se de um decodificador que, a partir do arquivo comprimido, executa as decodificações unárias e binárias, das quais se extraem os valores mapeados de erros de predição, baseados também em k , atualizado a cada passo. Realize-se, então, o

mapeamento reverso para se descobrir os erros. Tais valores somados ao resultado da predição produzem os valores dos pixels originais que são atribuídos à imagem descomprimida.

Na compressão, o primeiro pixel é salvo sem compressão no arquivo de saída para que, na descompressão, seja possível realizar a primeira predição, a qual desencadeia todas as demais. Além disso, na primeira linha e nos primeiros pixels de cada linha, onde não é possível realizar a predição bidimensional, prediz-se o pixel atual linearmente: na primeira linha, cada pixel é predito como o vizinho da esquerda enquanto que nos primeiros pixels de cada linha a predição é feita a partir do vizinho superior.

O comportamento do compressor e do descompressor foi sintetizado no diagrama da Figura 8, reunindo os tipos de dados necessários (imagens, parâmetros,...), os quais são manipulados através de operações bem definidas (comprimir, codificar, predizer,...) e assim compor as entidades envolvidas nos processos de compressão e descompressão, conforme a descrição dos parágrafos acima.

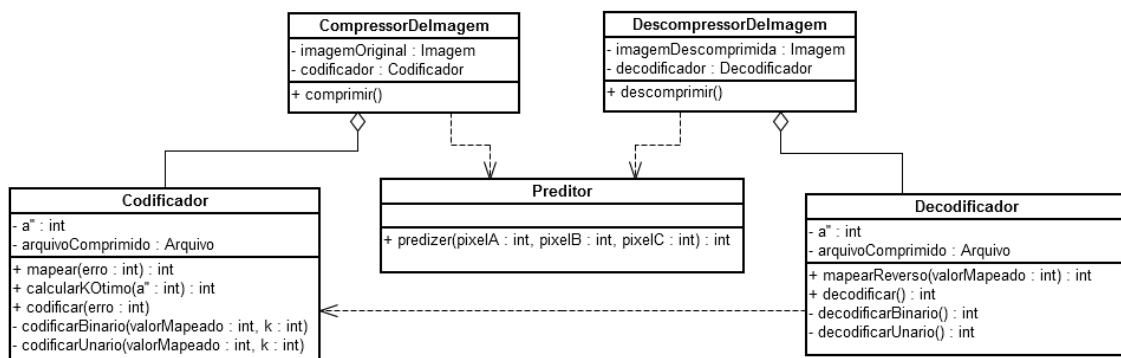


Figura 8: Diagrama de classes do modelo de referência.

Quanto à atualização do parâmetro k , visando sua implementação em hardware, em comparação ao método apresentado na seção 2.2.1.2.1, o presente trabalho propõe uma simplificação com o intuito de perseguir o requisito de baixa complexidade. O valor de a' (página 29) é substituído por:

$$a'' = \frac{\sum_{i=1}^t 2^{i-1} |e_i|}{2^t} \quad \text{Equação 8}$$

Isso significa que a'' é calculado por meio de uma média local, ou seja, cada valor absoluto de erro possui um peso igual à soma de todos os anteriores. A simplicidade está na substituição de uma divisão inteira comum por uma operação de deslocamento de bits, já que são executadas apenas divisões por potências de dois. Em termos de implementação, isso pode ser exemplificado conforme o código a seguir:

```

 $a'' = 0;$ 
(...)
for( $t = 0; t < T; t++$ ) {
    (...)
     $a'' = (a'' + |e_t|) \gg 1;$ 
}

```

Dessa maneira, não é necessário manter os valores de t e a . Nesse caso, a'' está sempre no intervalo $[0: 2^R - 2]$, onde R é a resolução radiométrica da imagem, ou seja, o número de bits usados nos pixels da imagem.

Apesar de aparentar ser uma simplificação grosseira, os resultados de compressão demonstraram que as duas abordagens possuem eficiência equivalente. Tais resultados são apresentados no Capítulo IV.

3.3. Teste com bancos de imagens

Tanto para a validação funcional do modelo de referência implementado como para a avaliação do desempenho de compressão do algoritmo em sua versão estável, foram realizados testes com imagens, particularmente imagens de sensoriamento remoto como principal foco do trabalho.

Assim, foram definidos dois banco de testes:

- A. Imagens selecionadas do Satélite CBERS-2B: 20 imagens brutas da câmera CCD, de cinco bandas cada (pancromática, azul, verde, vermelho e infravermelho próximo), com resolução radiométrica de 8 bits e tamanho de 5812 x 5812 pixels. São as mesmas imagens usadas por (FARIA, FONSECA e COSTA, 2012). Elas foram fornecidas pelo INPE sem nenhum tipo de correção, sendo, assim, mais adequadas para experimentos de compressão de dados, já que correspondem exatamente às que foram obtidas pelos sensores no satélite;
- B. Imagens selecionadas do conjunto referência do documento (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007): dentre variadas fontes de imageamento espacial, foram escolhidas 21 imagens brutas de sensoriamento remoto em níveis de cinza, sendo 11 imagens de resolução radiométrica de 8 bits, 8 de 10 bits e 2 de 12 bits, capturadas em tamanhos variados;

Algumas amostras de imagens de ambos os bancos são apresentadas na Figura 9.

A importância de se utilizar as imagens do banco A se dá pelo fato de inserir o presente trabalho dentro de um contexto nacional, uma vez que grande parte das

imagens capturadas pelos sensores do satélite CBERS-2B se deram em território brasileiro. O programa CBERS (*China-Brazil Earth Resources Satellite* ou Satélite Sino-Brasileiro de Recursos Terrestre) contempla a construção de satélites em cooperação com a China, particularmente em missões de sensoriamento remoto. O satélite CBERS-2B foi lançado em 2007 e esteve em operação até 2010 (INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS, 2007). Os resultados obtidos com este banco não só servem para comparação com (LOPES FILHO e D'AMORE, 2010), mas também para comprovar a aplicabilidade do sistema de compressão desenvolvido neste trabalho em missões espaciais brasileiras, como a do CBERS.

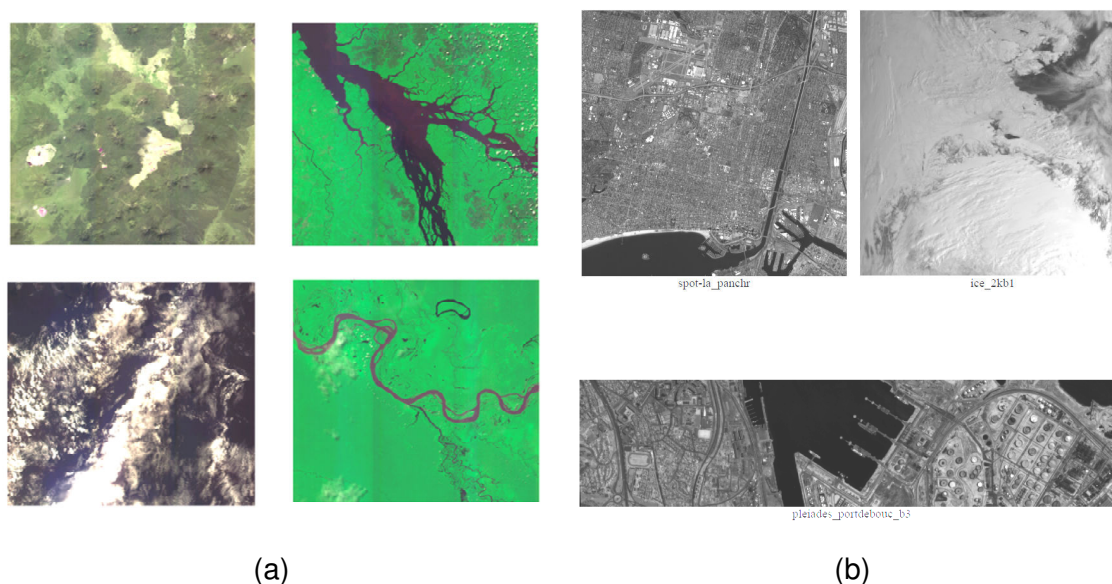


Figura 9: Amostras das imagens do banco A (a) e banco B (b).

Por sua vez, o banco B é importante por fazer parte de um documento amplamente discutido no âmbito da compressão de imagens espaciais (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007). Ele permite a comparação com diversos outros métodos de compressão explorados em aplicações espaciais, já que o próprio documento fornece esses dados. Contudo, a menor quantidade de imagens juntamente com seus tamanhos reduzidos, em comparação às imagens do banco A, sugerem que os testes com tais imagens não explorem o compressor/descompressor tão amplamente como o outro banco.

Como esquematizado na Figura 10, os testes realizados, de uma maneira geral, consistiram na comparação entre a imagem original e a descomprimida obtida ao fim do processo compressão-descompressão. Tais imagens devem ser idênticas, já que se trata de um método de compressão sem perdas, ao mesmo tempo em que também valida a implementação do par compressor/descompressor. Além disso, a partir dos arquivos de entrada e dos arquivos comprimidos é possível calcular as

razões de compressão em cada caso para fins de avaliação e comparação do desempenho de compressão perante outros trabalhos.

Os testes foram conduzidos com sucesso para todas as imagens selecionadas dos bancos A e B, validando, então, a implementação de alto nível e fornecendo os resultados, os quais são apresentados oportunamente no Capítulo IV.

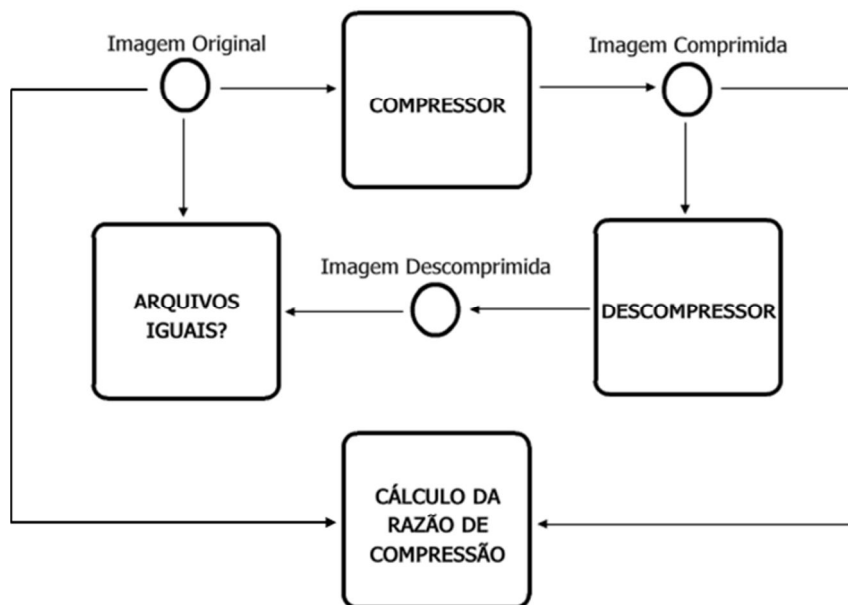


Figura 10: Esquematização dos testes com bancos de imagens.

3.3.1. Resistência a erros

As imagens do banco A também foram utilizadas para testar a técnica de resistência a erros, que consistiu na fragmentação de cada imagem em blocos menores, os quais foram comprimidos independentemente como se fossem imagens separadas. Todos os tamanhos dos arquivos de cada bloco comprimido foram somados com o intuito de calcular a RC da imagem completa. As dimensões selecionadas para os blocos foram de 5812x5812 (imagem completa), 2096x2096, 1453x1453, 726x726, 363x363, 181x181, 90x90, 45x45, 23x23, 11x11 e 5x5.

Dessa maneira, diversas configurações foram testadas com o intuito de observar o impacto do tamanho do bloco no desempenho de compressão.

3.4. Implementação em HDL

Dentre as formas possíveis de se projetar um hardware digital, a utilização de uma linguagem de descrição de hardware (HDL) é destacada pela facilidade de se abstrair complexidade. Foi escolhida, então, a linguagem SystemVerilog que, além de fornecer recursos de verificação, também permite a descrição de hardware de maneira concisa e robusta. Essa combinação facilitou a integração de elementos de verificação com descrições de hardware propriamente dito dentro do ambiente de simulação.

A implementação do compressor foi feita com a subdivisão do sistema em dois módulos principais, como mostrado na Figura 7 (Página 36): preditor e codificador. O codificador, por sua vez, é dividido em quatro módulos: mapeamento, cálculo de k, codificação unária e codificação binária.

3.4.1. Protocolo de sinais

Para facilitar o fluxo de dados entre os módulos, foi implementado um protocolo de sinais simples baseado nos sinais *ready* e *valid*. Trata-se de um protocolo síncrono composto por quatro sinais básico: *clk*, *data*, *valid* e *ready*. Além deles, ainda existem os sinais *reset_n* e *last*. Abaixo, segue a descrição de cada um deles.

- *clk*: clock global do sistema;
- *data*: dados a serem transferidos (variam em quantidade e largura de acordo com o módulo);
- *valid*: sinaliza a existência de dados válidos a serem transferidos;
- *ready*: indica que o módulo está pronto para realizar a transação;
- *reset_n*: reset síncrono do sistema, ativado em nível lógico baixo;
- *last*: usado para comunicar a transferência do último conjunto de dados. Para o compressor proposto, sinaliza a entrada do último pixel da imagem ou último bit na saída.

A regra fundamental do protocolo é: Dados são transferidos na borda de subida do *clock* quando *valid* e *ready* estão ambos em nível lógico alto. A Figura 11 mostra as transações envolvendo os sinais apresentados. As setas indicam os momentos exatos de cada transferência.

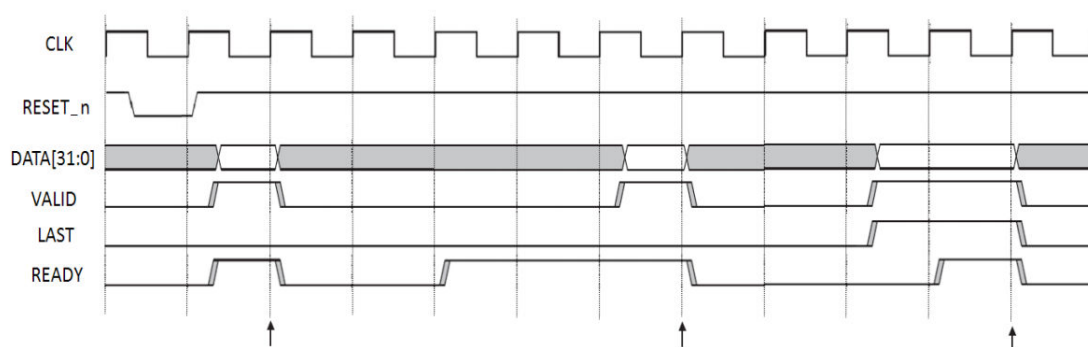


Figura 11: Transferências de dados usando os sinais do protocolo.

3.4.2. Preditor

O módulo preditor implementa a predição escolhida, incluindo as particularidades adotadas pelo modelo de referência e detalhadas a seguir. O primeiro pixel não é predito, pois é usado como base para iniciar a descompressão. Assim, ele

é serializado bit-a-bit na saída do compressor. A predição para os demais pixels da primeira linha é realizada linearmente a partir do vizinho a esquerda, já que ainda não é possível realizar a predição bidimensional. Da mesma maneira, a predição para o primeiro pixel de cada linha a partir da segunda é feita com o vizinho superior.

Para que seja feita a predição bidimensional faz-se necessário manter armazenados os últimos $w + 1$ pixels, onde w é a largura da imagem, considerando que a imagem está sendo percorrida, para fins de realizar a compressão, de cima para baixo e da esquerda para a direita. Por exemplo, para as imagens do banco A, $w = 5812$, tem-se um total de armazenamento de 5813 pixels de 8 bits cada, ou seja, 46504 bits ou 5,6 KB. Essa quantidade de bits de armazenamento sugere a utilização de uma memória RAM separada para esse fim ao invés de elementos lógicos convencionais.

Sendo assim, o preditor possui um submódulo de memória RAM de $w + 1$ posições de R bits cada, sendo R a resolução radiométrica da imagem, e um barramento de endereço de $\lceil \log_2(w + 1) \rceil$ bits. Foi selecionada uma memória síncrona *dual port*, que permite a leitura e escrita de 2 pixels de uma só vez, para agilizar a transferência dos dados.

Para cada pixel de entrada (com exceção do primeiro pixel), com resolução radiométrica de R bits, é produzido um erro de largura de $R + 1$ bits na saída.

3.4.3. Mapeamento

Para implementar a função de mapeamento já apresentada, tem-se um módulo simples que seleciona e executa um dos dois casos de mapeamento:

- Valores positivos de erro são mapeados como pares. Para isso, é realizado um deslocamento para esquerda;
- Valores negativos de erro são mapeados como ímpares. Primeiro, é obtido o valor absoluto, que por sua vez é deslocado para esquerda e incrementado em uma unidade.

O valor mapeado também possui $R + 1$ bits.

3.4.4. Cálculo de k

Para o cálculo de k , o módulo seleciona o valor resultante a partir do valor corrente de a'' , mantido e atualizado de acordo com o valor do erro, conforme observado na implementação do modelo de referência.

Dessa maneira, k pode assumir valores na faixa $[0: R]$, necessitando assim de $\lceil \log_2 R \rceil + 1$ bits para representá-lo.

3.4.5. Codificação Unária

A partir dos valores de entrada, o módulo calcula o quociente da divisão pelo deslocamento k vezes para direita do valor mapeado. Depois disso, são disponibilizados serialmente na saída uma quantidade de bits '1' igual ao valor do quociente. Por fim, disponibiliza na saída um bit '0'.

3.4.6. Codificação Binária

A codificação binária é executada depois que a unária é completada. Os k bits menos significativos do valor mapeado são disponibilizados na saída um a um, do mais significativo para o menos significativo.

3.4.7. Visão em blocos

Diante dos módulos já apresentados, é possível ilustrar a sua disposição, conexão e sinais envolvidos em três níveis (considere $R = 8$):

- Diagrama de bloco do compressor (Figura 12);
- Preditor e codificador (Figura 13.a);
- Mapeamento, Cálculo de k , codificações unária e binária (Figura 13.b).

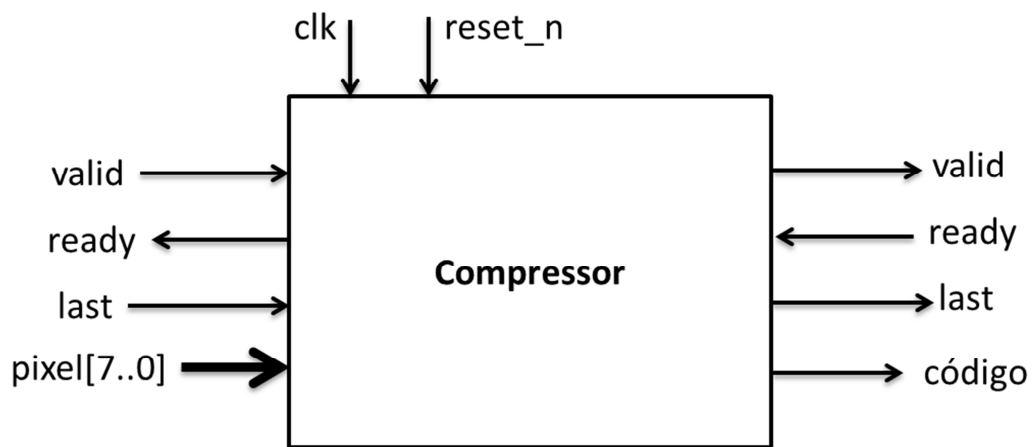


Figura 12: Diagrama de bloco do compressor.

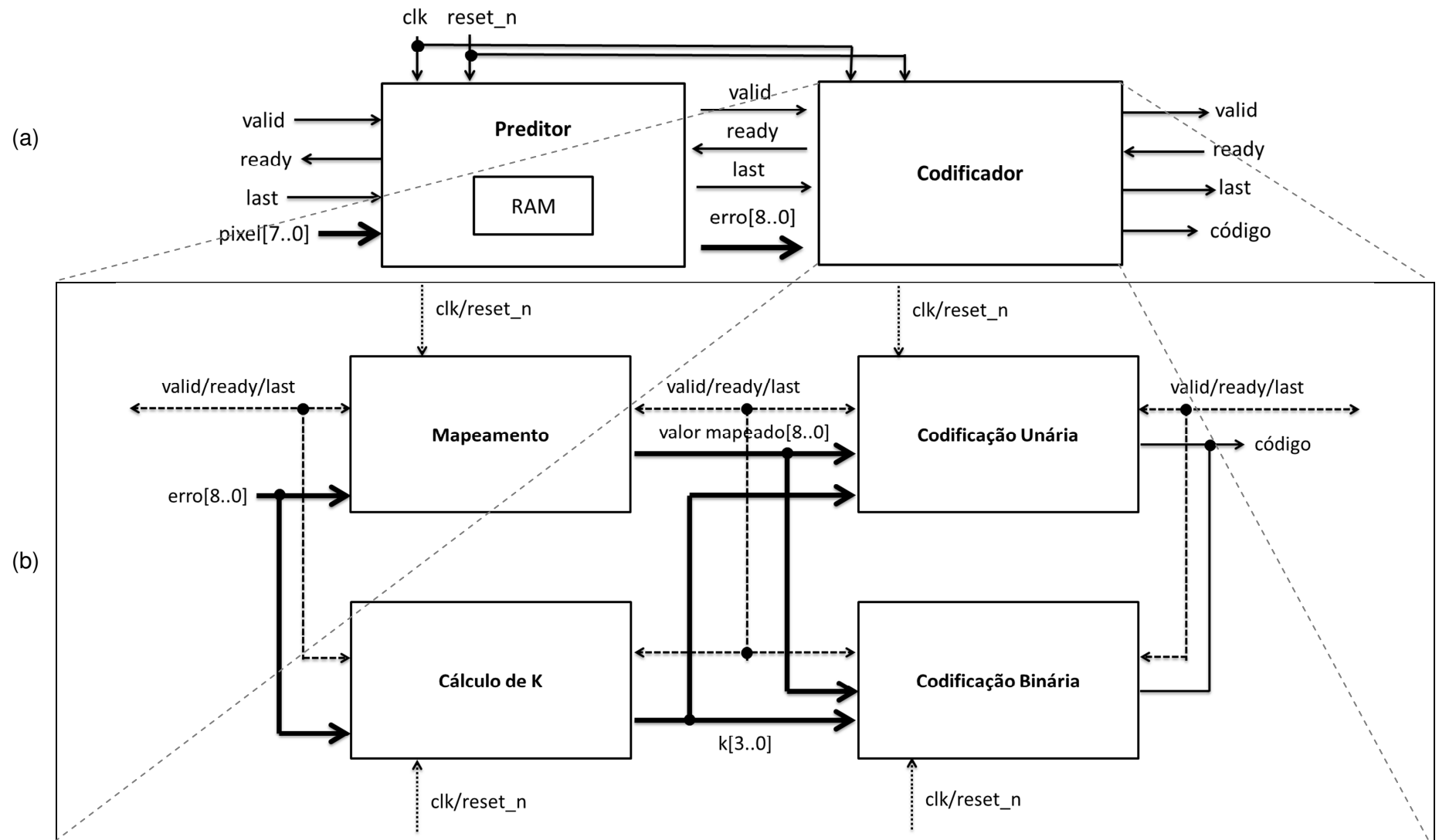


Figura 13: Visão em blocos do (a) compressor e (b) codificador.

3.5. Verificação Funcional

O objetivo da verificação é validar o modelo HDL do compressor. Na verificação funcional, estímulos de entrada são inseridos em uma implementação de alto nível (modelo de referência) e no módulo de hardware a ser verificado (DUV – *Design Under Verification*). As saídas de cada um deles são confrontadas e certos valores de saída são monitorados para que determinada condição seja atingida e a verificação seja finalizada.

Como já foi mencionada, a verificação funcional aqui apresentada baseia-se na metodologia BVM (*Brazil-IP Verification Methodology*) (ARAÚJO, 2010). Nela, os *testbenches*, que nada mais são que os ambientes virtuais onde ocorre todo o processo da verificação, são implementados em nível de transação em contraste com o DUV que é implementado em nível de sinais. Transações são operações finitas formadas por um conjunto de instruções e/ou dados para realizar determinada operação. A visão geral de um *testbench* é mostrada na Figura 14.

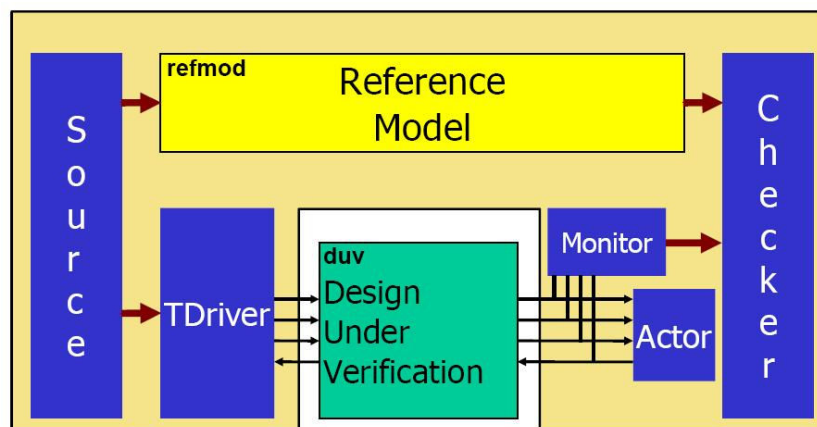


Figura 14: Visão geral de um testbench.

O *Source* é responsável por enviar a mesma transação de entrada para o *TDriver* e para o modelo de referência. O modelo de referência processa a transação de entrada e produz a respectiva transação de saída. Enquanto isso, o *TDriver* converte a transação de entrada em sinais que estimulam o DUV a produzir sinais de saída (através da interação com o *Actor*), os quais são observados pelo *Monitor* para a produção de transações equivalentes. Tais transações chegam ao *Checker* juntamente com as transações do modelo de referência a fim de serem comparadas.

As transações que fluem do *Source* são geradas aleatoriamente de acordo com uma distribuição de probabilidade direcionada. Tanto as transações aleatórias como as transações resultantes que chegam ao *Checker* são observadas com o intuito de determinar o momento de parar o processo de verificação. Para tal condição de parada se dá o nome de cobertura.

Dentre algumas opções de cobertura, cita-se a principal delas como a cobertura funcional, a qual observa sinais e transações e conta o número de ocorrência de determinadas situações especificadas.

Para a realização da verificação funcional, este trabalho utilizou a linguagem SystemVerilog, que permitiu a implementação do DUV, modelo de referência e os *testbenches* propriamente ditos. Vale aqui ressaltar a sua praticidade ao permitir integrar, em ambiente único, entidades de alto nível, transações, DUV, sinais e componentes do *testbench*, através dos recursos oferecidos pela linguagem para esses diversos fins. Os *testbenches* codificados em SystemVerilog foram executados na ferramenta QuestaSim versão 10.0 da Mentor Graphics.

Para a verificação do compressor, que é o DUV no presente trabalho, as transações de entrada são os pixels e as de saída são bits isolados. Também foi incluído o sinal *last* dentro das transações. O objetivo foi simular aleatoriamente a condição de último pixel da imagem e, em seguida, prosseguir com o primeiro pixel de uma imagem nova. Os valores de pixel foram considerados equiprováveis enquanto que o sinal *last* ocorreu com uma probabilidade de um a cada 10.000 (imagem 100 x 100), valor suficiente para estimular todas as funcionalidades desejadas.

Quanto ao cenário de cobertura configurado, esperou-se a ocorrência de todos os valores possíveis de pixels pelo menos três vezes, considerando uma resolução radiométrica de 8 bits por pixel. O sinal *last*, por sua vez, ocorreu pelo menos 2 vezes. Os dois valores possíveis do bit de saída, '0' e '1', tiveram que ocorrer pelo menos 10 vezes cada um, enquanto que o *last* na saída ocorreu obrigatoriamente na mesma quantidade de vezes que foi gerada na entrada. Todos esses parâmetros unidos foram fundamentais para a validação da execução das diversas funcionalidades envolvidas com os módulos de hardware implementados em HDL.

Com o fim das simulações, os componentes da verificação permitiram avaliar e corrigir quando necessário o comportamento do compressor quanto a sua funcionalidade tanto nos diversos submódulos como no sistema como um todo, produzindo um núcleo de compressão em hardware validado.

3.6. Prototipação em FPGA

Para fins de análise de desempenho do hardware proposto, validação e demonstração de seu funcionamento, foi criado um protótipo em FPGA a partir do hardware codificado em HDL e verificado funcionalmente. O protótipo foi desenvolvido a partir do kit de prototipação em FPGA da Altera, o *Nios II Development Kit – Stratix Edition* (ALTERA CORPORATION, 2004), o qual provê dispositivo FPGA EP1S10 da família Stratix, memórias flash, RAM estática e dinâmica, interfaces Ethernet,

CompactFlash, RS-232, JTAG, oscilador de 50 MHz, LEDs, displays de sete segmentos, etc.

O compressor foi sintetizado para o FPGA do kit juntamente com um processador embarcado no FPGA, o Nios II, processador *soft-core* da Altera, responsável por gerenciar os vários fluxos de dados envolvidos no recebimento e compressão de uma imagem e envio de volta do resultado da compressão. Além disso, o Nios II permite acionar contadores de ciclos de relógio com o objetivo de mensurar a performance do compressor.

Como interface de comunicação, foi escolhida a Ethernet para facilitar a implementação dos softwares, no host e no sistema embarcado, responsáveis por transmitir dados, fazendo uso de programação em socket, entre os dois agentes: o host e o kit.

O software do host, o qual, por questões de facilidade, foi implementado em Java, transmite as imagens do banco A pelo enlace de comunicação para que sejam comprimidas no kit, que, por sua vez, possui um software desenvolvido em C para o processador Nios II. Ele recebe essas imagens e as fornece ao compressor, que se comunica com o processador pelo seu barramento interno. Durante cada momento de compressão, são acionados os contadores de ciclos de relógio a fim de registrar a quantidade de tempo gasto em cada um desses momentos. Tanto o bloco comprimido de dados como as informações de tempo são enviadas de volta para o host. De posse desses dados, o software host executa a descompressão para verificar se a compressão foi realizada corretamente e armazena os valores de tempo de compressão, os quais são usados na análise de desempenho do compressor. O esquema para esse ambiente de prototipação é mostrado na Figura 15.

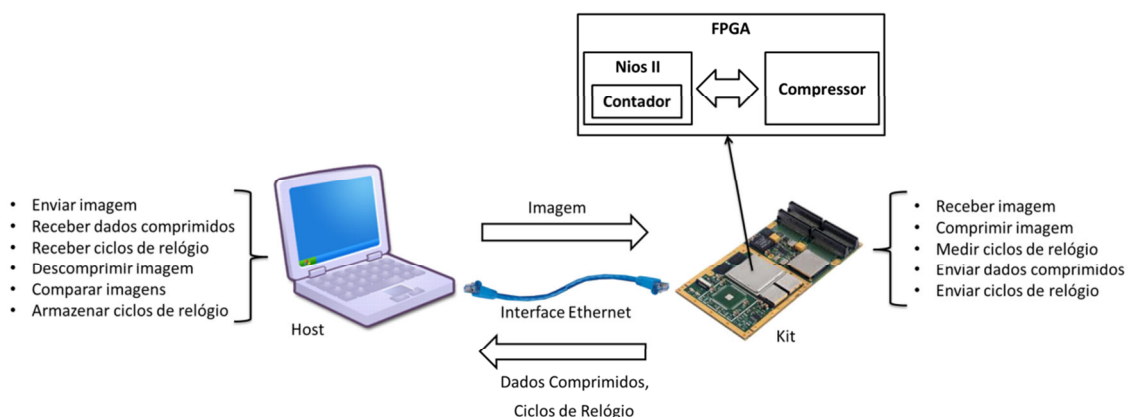


Figura 15: Protótipo esquematizado.

O protótipo funcionou corretamente para todas as imagens do banco A, considerando a frequência de relógio do kit de 50 MHz, demonstrando mais uma vez o

funcionamento esperado do compressor, além da obtenção dos resultados de tempo, que são apresentados e discutidos no Capítulo IV.

3.6.1. Abordagem *Multicore*

A prototipação também contemplou a abordagem de compressão baseada em vários núcleos de compressão, com o intuito de observar o ganho de desempenho em função da instanciação de dois a cinco núcleos de compressão em paralelo. Utilizou-se um cenário muito parecido com o descrito para apenas um núcleo, exceto pelo fato de não se preocupar mais com a descompressão, considerando que o protótipo de um núcleo já havia comprovado sua funcionalidade. Essa abordagem foi adotada para que o protótipo não tivesse que tratar a organização dos dados comprimidos oriundos de cada núcleo em particular, por se julgar ser um ponto fora do escopo desse trabalho.

Ao mesmo tempo, a imagem recebida para compressão é dividida entre os núcleos paralelamente de tal forma que cada um deles fica responsável por comprimir uma parte diferente da imagem.

Sendo assim, o protótipo *multicore* conseguiu medir a compressão de todas as imagens do banco A utilizando de dois a cinco núcleos em paralelo. Os valores de ciclos de relógio em cada caso foram armazenados para fins de análise de desempenho.

3.7. Análise de Desempenho

Estabelecido como forte requisito para o imageamento espacial, o desempenho em termos de *throughput* é um fator muito importante a ser analisado no compressor em hardware. Por isso, foram feitas as medidas de quantidade de ciclos de relógio gastos para a compressão de cada imagem do banco A considerando os casos desde um único núcleo de compressão até cinco deles em paralelo, por meio do protótipo em FPGA implementado.

Considerando a frequência máxima do kit de 50 MHz, é possível calcular o tempo exato gasto para se comprimir cada um dos casos descritos a partir do valor de ciclos de relógio obtido, uma vez que cada 1 segundo transcorrido corresponde a 50 ciclos.

Além disso, também é sabido que cada imagem do banco A possui 5812x5812 pixels, o que corresponde aproximadamente a $3,3 \times 10^9$ pixels. Assim, de posse desses dados, é possível calcular a vazão do compressor como a razão entre a quantidade de pixels comprimidos pelo tempo gasto para comprimi-los.

Obviamente, esse cálculo pode ser generalizado para qualquer frequência ou período de relógio do sistema, o que é importante já que a síntese do hardware

alcançou frequências maiores que a do kit. Tal generalização compõe a Equação 9, apresentada a seguir, onde T é o *throughput*, C a quantidade de ciclos de relógio, P o período e f a frequência.

$$T = \frac{3,3 \cdot 10^9}{C \cdot P} \text{ ou } T = 3,3 \cdot 10^9 \frac{f}{C} \quad \textbf{Equação 9}$$

Para a abordagem *multicore*, o valor de C considerado para o cálculo foi o número de ciclos do núcleo que obteve a maior contagem dentre os núcleos atuando em paralelo. Isso é necessário pois cada parte da imagem atribuída à seu respectivo núcleo leva um tempo diferente para ser comprimida em comparação com as demais, já que as partes são normalmente diferentes entre si. Então, a parte que toma mais tempo para ser comprimida dita o desempenho naquele caso perante os outros núcleos.

Capítulo IV: Resultados

Neste capítulo são apresentados os resultados obtidos de acordo com o que foi proposto, incluindo os testes com imagens, propriedades e desempenho do hardware.

Em primeiro lugar, a comparação entre as abordagens de atualização do parâmetro da codificação de Golomb-Rice, detalhadas nos Capítulos II e III, é suportada pelos resultados de compressão obtidos a partir das duas abordagens, os quais são apresentados neste capítulo.

Depois, os resultados de compressão com as imagens do banco A e B são apresentados na forma de médias de razão de compressão para cada conjunto de imagens. Esses valores são de grande importância para a comparação com outros métodos relacionados conforme a especificidade de cada banco. Há o caso especial da compressão fragmentada em blocos, feita com o banco A, com o intuito de perceber como a aplicação dessa técnica influencia nas razões de compressão em contraste ao método inicial de utilizar a imagem completa.

Em outra linha de raciocínio, é explicitada e avaliada a síntese do hardware proposto para apresentar e discutir variáveis como lógica, memória e frequência sintetizada nos dispositivos FPGA escolhidos. Esse resultado também merece ser comparado a outros trabalhos para ser percebido dentro de um contexto mais amplo.

A capacidade de comprimir dados por unidade de tempo é tratada como uma característica importante em análise por esse trabalho. Os resultados aqui elucidados vêm das medidas obtidas conforme prototipação em FPGA realizada com o intuito de enumerar tal capacidade para o hardware proposto. Esses dados de performance mudam conforme algumas variáveis envolvidas e são comparadas, dentro desse ambiente, com trabalhos correlatos.

Por fim, todas as características do presente trabalho aqui apresentadas favorecem uma discussão acerca de sua aplicabilidade e efetividade considerando os demais trabalhos usados nas diversas comparações.

4.1. Atualização do parâmetro da codificação de Golomb-Rice

Como exposto no Capítulo III, foi proposta uma nova abordagem de atualização do parâmetro da codificação de Golomb-Rice. Tal abordagem é mais simples e, conseqüentemente, mais adequada para o presente trabalho ao utilizar operações de deslocamento de bits ao invés de divisões inteiras.

Para validar tal proposta se faz necessário comparar os resultados de compressão de ambas as abordagens e assim avaliar o compromisso entre

simplicidade e desempenho de compressão. Para tanto, ambas as abordagens foram submetidas à compressão de todas as imagens do banco A.

A abordagem original alcançou uma RC média global de 3,3 ao passo que a proposta obteve média global de 3,4 de razão de compressão. A Tabela 2 lista as RC médias de cada banda obtidas nas diferentes abordagens. Além disso, no Apêndice I é possível encontrar tabelas com todas as RCs obtidas de todas as imagens para as duas abordagens.

Tabela 2: RCs obtidas para comparação das abordagens de atualização do parâmetro da codificação de Golomb-Rice

Abordagem	B1	B2	B3	B4	B5
Original	3,27	3,54	3,35	2,91	3,44
Proposta	3,38	3,63	3,47	2,96	3,54

4.2. Teste com bancos de imagens

Ao contrário do banco B, que já acompanha resultados comparativos na própria literatura, para a avaliação de resultados através da comparação com outros métodos usando o banco A, foi realizada a compressão das mesmas imagens utilizando implementações gratuitas acessíveis na internet. Os softwares selecionados foram:

- SPIHT: Programa de avaliação fornecido pelo *Center for Image Processing Research* do *Rensselaer Polytechnic Institute* (<http://www.cipr.rpi.edu/research/SPIHT/>);
- JPEG-2000: Foi utilizada a implementação do *JasPer Project* (<http://www.ece.uvic.ca/~frodo/jasper/>). Trata-se de uma iniciativa de código aberto para fornecimento gratuito de um software de referência para a especificação do padrão JPEG-2000 Part-I;
- JPEG-LS: Software de referência disponibilizado pela *HP Labs* (<http://www.hpl.hp.com/loco/>) para o padrão JPEG-LS, algoritmo LOCO-I.

Além disso, foi levado em consideração o trabalho de (LOPES FILHO e D'AMORE, 2010) por utilizar o mesmo conjunto de imagens. Ele não faz uso da banda pancromática e apresenta apenas o pior caso de RC obtida para cada banda, que foi de 2,3 para todas elas.

Tais resultados são mostrados na íntegra no Apêndice I, tanto para o trabalho proposto como para os demais métodos. Abaixo, na Figura 16, é apresentada graficamente a comparação de razão de compressão entre o sistema proposto e os algoritmos acima relacionados através do cálculo da média de todas as RCs obtidas para todas as bandas de todas as imagens do banco A.

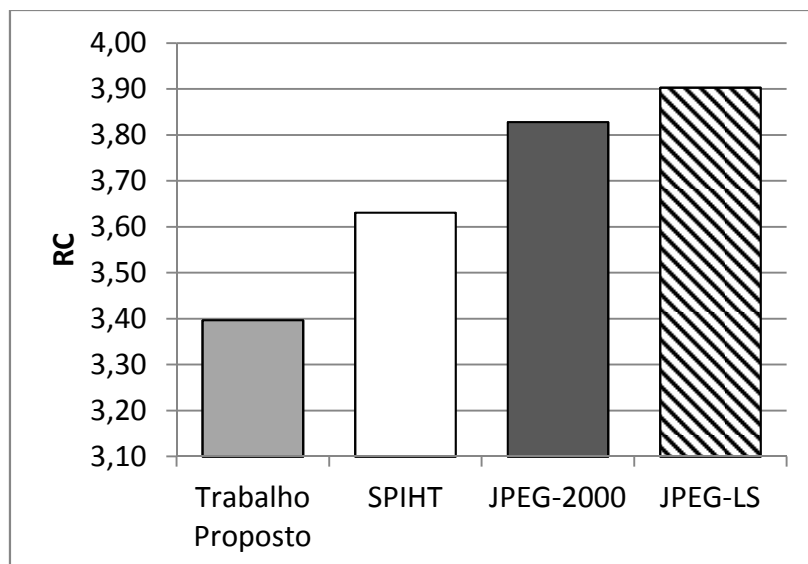


Figura 16: Comparação de RC média para todas as imagens de todas as bandas do banco A.

Já em relação ao banco B, os resultados apresentados por (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007) para diversos outros compressores sem perdas, cujas abordagens já foram explicadas anteriormente, são tomados como base comparativa.

Os gráficos da Figura 17 e da Figura 18, separados de acordo com as imagens de uma mesma resolução radiométrica, comparam os diversos algoritmos e o trabalho proposto em termos de razão de compressão alcançada em cada caso. Esses mesmos resultados detalhados são apresentados no Apêndice II.

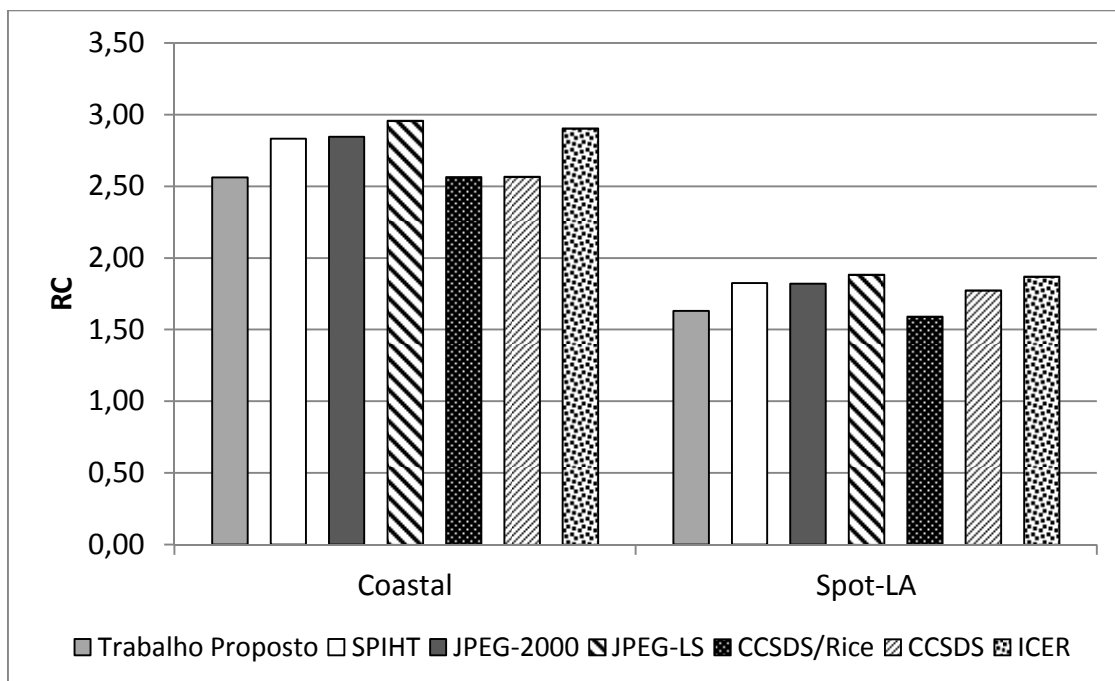


Figura 17: RCs médias obtidas para imagens de 8 bits do banco B.

Na Figura 17 são apresentadas as razões de compressão para todos os algoritmos listados em comparação ao trabalho proposto. Trata-se de uma média para todas as bandas disponíveis para as imagens *Coastal* e *Spot-LA*, as quais possuem 8 bits de resolução radiométrica.

Similarmente, são apresentadas as RCs médias obtidas entre os diversos métodos para as bandas disponíveis nas imagens de 10 bits (Figura 18), *Ice*, *India*, *Ocean*, *LandesV_G7* e *Marseille_G6*, e de 12 bits (Figura 19), *pleiades_portdebouc*.

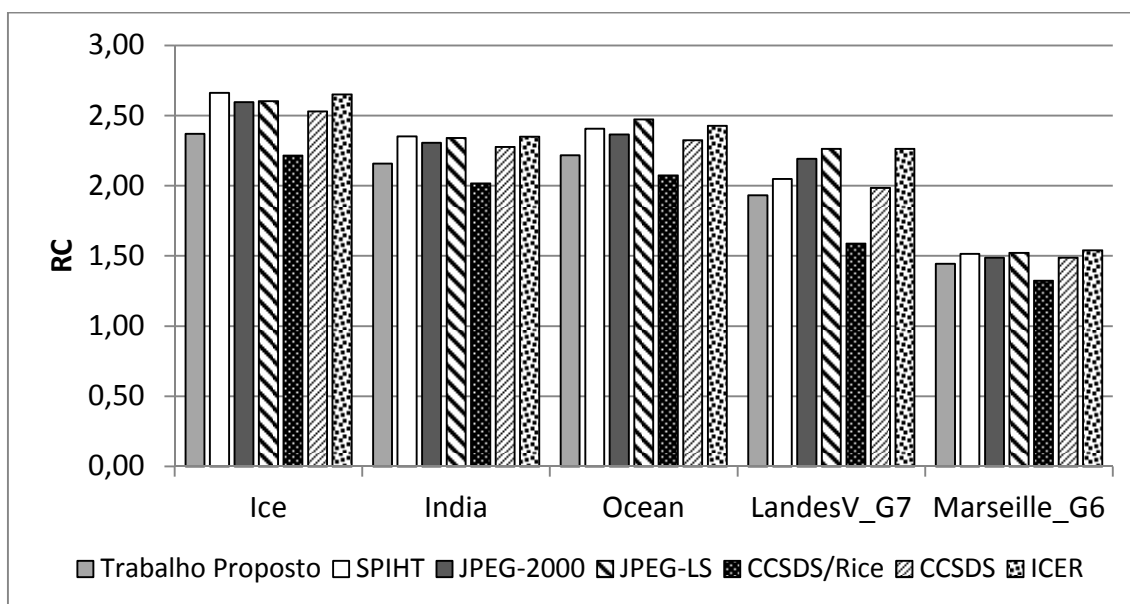


Figura 18: RCs médias obtidas para imagens de 10 bits do banco B.

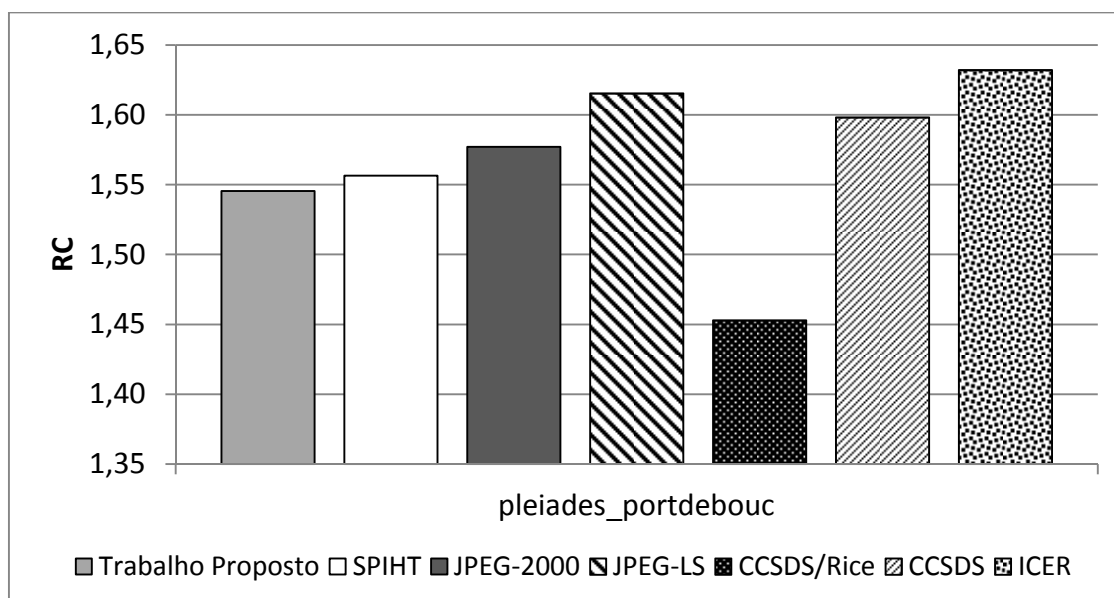


Figura 19: RCs médias obtidas para imagens de 12 bits do banco B.

4.2.1. Fragmentação em blocos (Resistência a erros)

O banco A também foi usado nos testes da abordagem de fragmentação da imagem em blocos com o intuito de verificar o desempenho da compressão nessa situação, mais adequada por oferecer maior resistência à propagação de erros, como já foi visto nos Capítulos II e III. O resultado aqui apresentado no gráfico da Figura 20 corresponde às RCs médias obtidas da fragmentação de todas as imagens do banco A em blocos, partindo inicialmente da imagem inteira (5812x5812) até a divisão em blocos de 5x5.

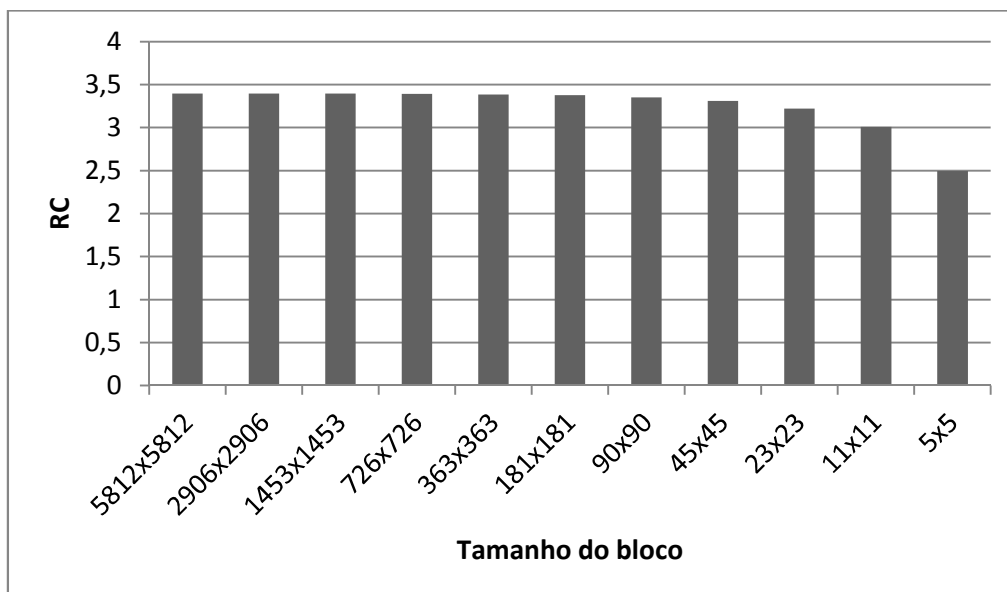


Figura 20: Resultado da compressão fragmentada em blocos menores para as imagens do banco A.

Os valores de RC para todas as imagens do banco A para a compressão fragmentada em blocos também podem ser encontrados no Apêndice I.

4.3. Síntese em FPGA

O compressor foi sintetizado para os dispositivos Stratix II EP2S60F484C3, Stratix EP1S10F780C6 e Cyclone III EP3C25E144C7, todos da Altera Corporation. Como já foi visto no Capítulo III, considerando as imagens do banco A, ele necessita de 46.504 bits de memória, o que representa uma pequena parte da quantidade total de memória disponível no FPGA. Em termos de elementos lógicos, o mesmo comportamento é observado, pois o compressor utiliza uma quantidade pequena de lógica presente no FPGA. Considerando em primeiro lugar apenas um núcleo do compressor, esses dados são comparados com características similares de trabalhos correlatos na Tabela 3. É elencada a tecnologia na qual cada trabalho foi implementado, incluindo diversos FPGAs diferentes e alguns outros em ASIC.

Dependendo do caso, a tecnologia define a unidade característica que compõe a área ocupada por cada circuito sintetizado na forma de elementos lógicos, *slices* e portas lógicas.

Tabela 3: Resultados de síntese em hardware e suas comparações.

Trabalho	Área	Memória (bits)	Tecnologia
Proposto	588 (< 1%) elementos lógicos	46.504 (2%)	Altera Stratix II EP2S60F484C3
	574 (5%) elementos lógicos	46,504 (5%)	Altera Stratix EP1S10F780C6
	564 (2%) elementos lógicos	46.504 (8%)	Altera Cyclone III EP3C25E144C7
(LOPES FILHO e D'AMORE, 2010)	2.375 (10%) elementos lógicos	13 x 9K (20%)	Altera Cyclone III EP3C25
(CAST, 2007)	8.867 (86%) elementos lógicos	18 x 9K (40%)	Altera Cyclone III EP3C10-6
	58.137 portas	102.076	ASIC UMC 0.18μm
	51.506 portas	102.076	ASIC TSMC 0.09μm
(WANG, ZENG, <i>et al.</i> , 2010)	2.650 (24%) <i>slices</i>	25 x 18K (44%)	Xilinx Virtex II XC2V2000FG676
(CHEN, CANAGARAJAH, <i>et al.</i> , 2007)	1.928 (13%) <i>slices</i>	63.078 (2%)	Xilinx Virtex 4 SX35

Além disso, também foi realizada a síntese em todos os três dispositivos utilizando de dois a cinco núcleos replicados do compressor com o intuito de observar mudanças em área, memória e frequência máxima devido ao aumento de núcleos. Os resultados são apresentados nas figuras a seguir. A Figura 21 mostra a quantidade de elementos lógicos utilizados devido ao aumento da quantidade de núcleos de compressão. Percebe-se que um aumento no número de núcleos provoca a utilização de uma maior quantidade de elementos lógicos. A diferença entre dispositivos é leve devida apenas às mudanças de estrutura interna para cada tecnologia em particular.

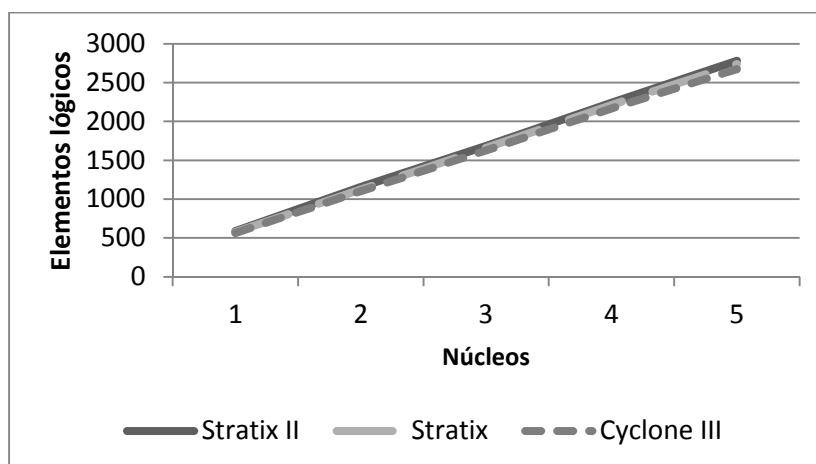


Figura 21: Quantidade de elementos lógicos utilizados devido ao aumento de núcleos de compressão.

Já a memória tende a permanecer em uma quantidade praticamente constante, apenas com um leve aumento devido à divisão da imagem entre os preditores de cada núcleo, como visto na Figura 22. Não há necessidade de apresentar o resultado para cada dispositivo, uma vez que não há mudanças entre eles nesse quesito. Isso acontece porque a imagem é compartilhada entre os núcleos, o que implica na divisão da memória entre eles.

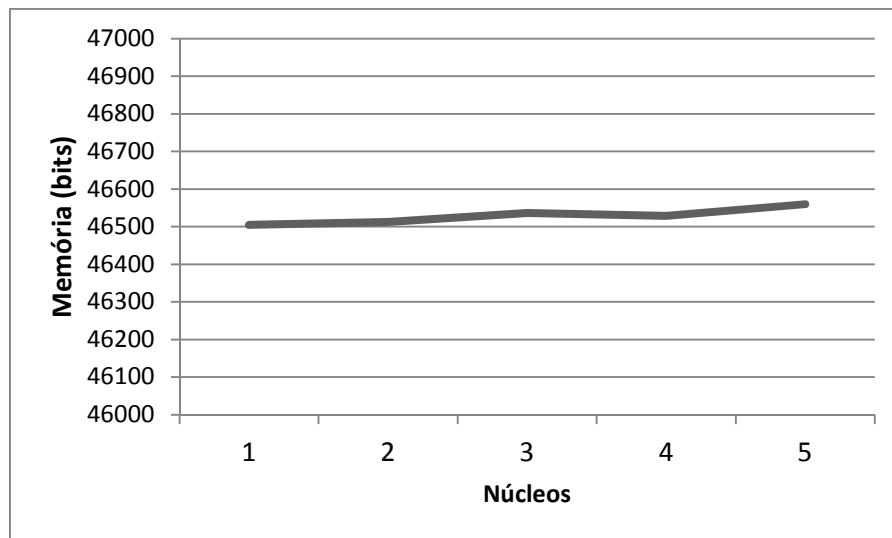


Figura 22: Quantidade de memória sintetizada devido ao aumento de núcleos de compressão.

Da Figura 23, observa-se que a frequência também permanece numa faixa de pequena variação, pois, sendo núcleos independentes em paralelo, não há muita interferência de caminhos entre eles. O dispositivo da família Stratix II é o que consegue atingir maiores frequências, seguido do Cyclone III e por último o Stratix.

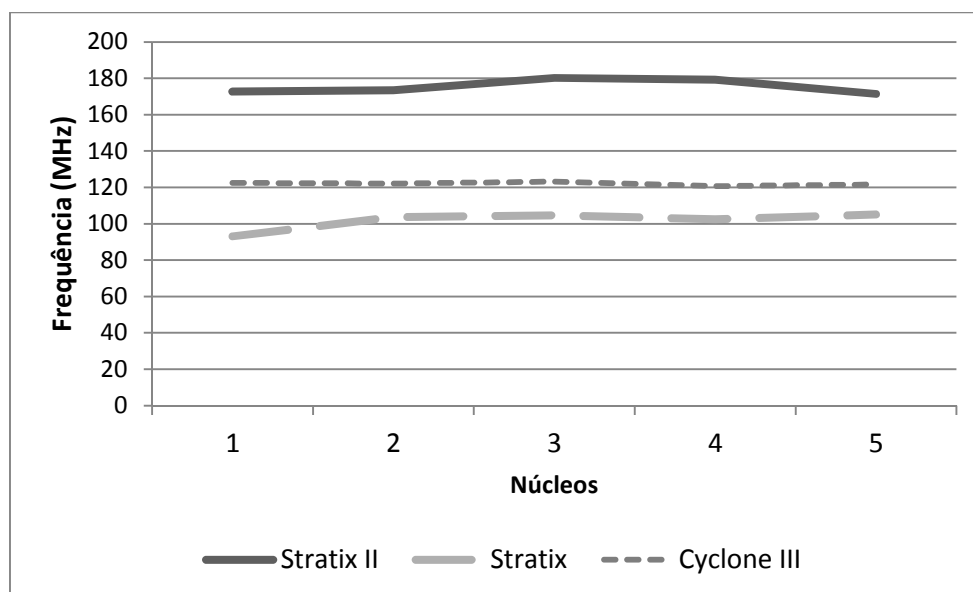


Figura 23: Frequência máxima alcançada em função do aumento de núcleos de compressão.

4.4. Throughput

É desejável mensurar a capacidade de processamento de pixels por segundo pelo compressor proposto a fim de analisar sua habilidade para interagir com um imageador que produz dados a serem comprimidos no decorrer do tempo. Além disso, é importante que tal desempenho seja comparado a outros trabalhos.

Medidas reais obtidas pelo protótipo em FPGA, conforme explicado no Capítulo III, permitiram definir quantidades de ciclos de relógio, utilizados no cálculo de desempenho do compressor proposto, como mostradas no gráfico da Figura 24, considerando o caso de um único núcleo de compressão e entre dois a cinco núcleos em paralelo.

De posse desses valores, é possível apresentar na Figura 25 as medidas de vazão de dados calculadas através da Equação 9, considerando as frequências máximas de operação obtidas, cujos valores compõem o gráfico da Figura 23.

Por exemplo, para a síntese em Stratix II, o *throughput* máximo chega a 28,62 Mpixel/s (ou 228,96 Mbit/s) para apenas um núcleo enquanto que cinco núcleos, utilizando o mesmo dispositivo, são capazes de comprimir em uma velocidade de 142,21 Mpixel/s (1,13 Gbit/s). A Tabela 4 apresenta esses resultados e os compara com os demais trabalhos na mesma ordem da Tabela 3. A conversão entre unidades de pixels para bits depende da resolução radiométrica das imagens utilizadas nos testes. Como nessa análise são usadas as imagens do banco A, com 8 bits por pixel, a conversão é realizada apenas por uma multiplicação por oito.

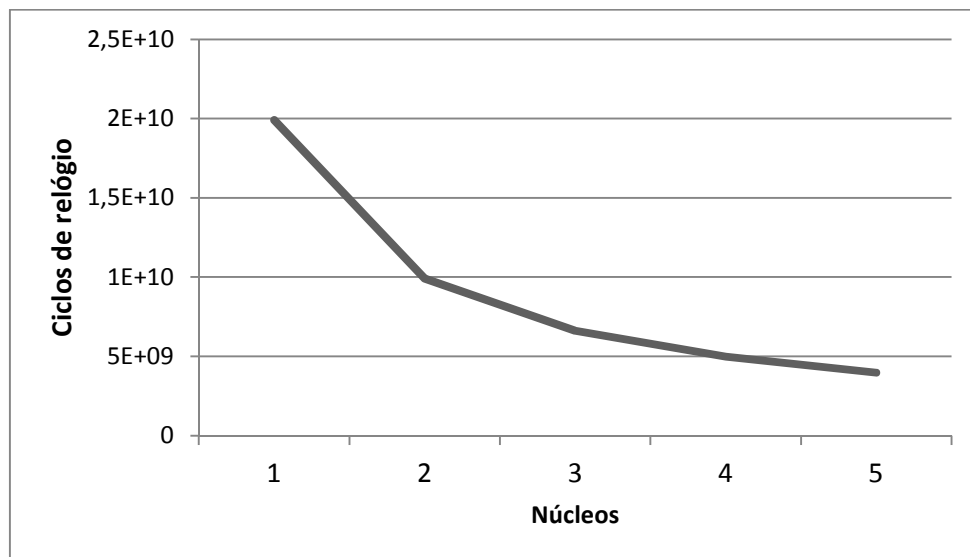


Figura 24: Variação da quantidade de ciclos de relógio por núcleos replicados.

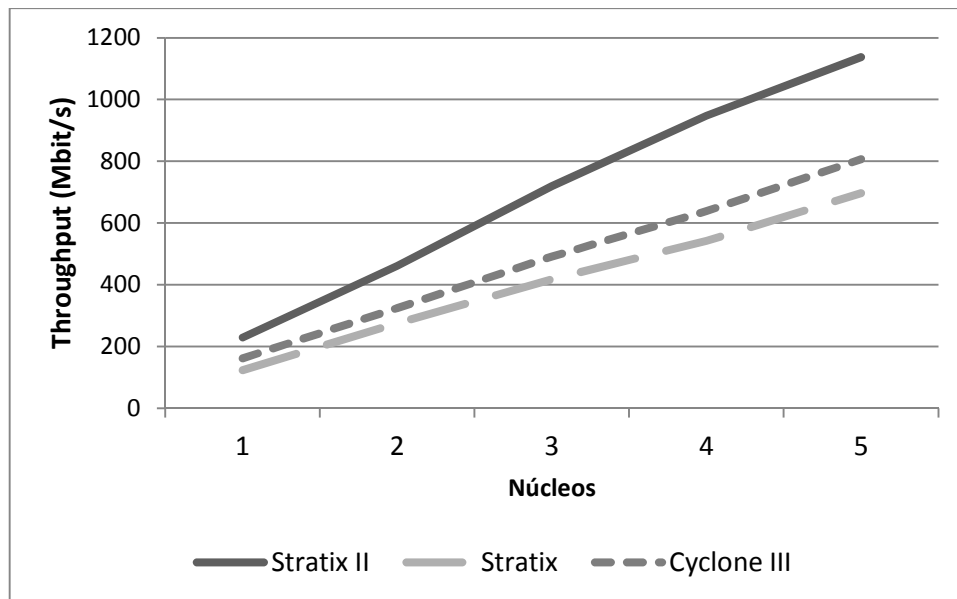


Figura 25: Influência do aumento de núcleos no Throughput do sistema.

Tabela 4: Comparativo de frequência e *throughput*.

Trabalho		Frequência (MHz)	Throughput	
			(Mpixel/s)	(Mbit/s)
Proposto (Stratix II)	1 núcleo	172,65	28,62	228,96
	5 núcleos	171,38	142,21	1137,68
Proposto (Stratix)	1 núcleo	93,06	15,43	123,44
	5 núcleos	104,95	87,09	696,72
Proposto (Cyclone III)	1 núcleo	122,41	20,29	162,32
	5 núcleos	121,52	100,84	806,72
(LOPES FILHO e D'AMORE, 2010)		50	12	96
(CAST, 2007)		100	100	800
		360	360	2880
		700	700	5600
(WANG, ZENG, <i>et al.</i> , 2010)		75	75	600
(CHEN, CANAGARAJAH, <i>et al.</i> , 2007)		123	15,3	122,4

4.5. Discussão dos resultados

Dentre as imagens do banco A, 20080305_L0_G1_P121_R31 se mostrou a mais difícil de ser comprimida enquanto 20080719_L0_G1_P151_R127 a mais fácil, para todas as bandas. Esse comportamento foi percebido em todos os testes. Já em relação as bandas das imagens do banco A, na média, a banda vermelha é a mais difícil de ser comprimida e a banda azul a mais fácil.

Como os resultados apontam, a utilização da abordagem mais simples de atualização do parâmetro da codificação de Golomb-Rice está amparada em um desempenho de compressão aceitável e até ligeiramente melhor que a abordagem

original. Dessa forma, o compressor proposto ganha em simplicidade enquanto mantém praticamente o mesmo desempenho de compressão.

Os resultados obtidos utilizando os bancos A e B mostram que o trabalho proposto alcança valores de razão de compressão compatíveis com os demais métodos de compressão relacionados nas comparações de acordo com o estado-da-arte na área. Para o banco A, a diferença média entre o trabalho proposto e o JPEG-LS, que alcançou as maiores RCs, é de 12,5%. As comparações utilizando o banco B mostram que o método proposto neste trabalho apresenta desempenho de compressão similar aos melhores métodos em cada caso, indicando que os resultados obtidos pelo compressor proposto são compatíveis com o que é possível encontrar hoje na literatura relacionada.

Similarmente aos demais métodos testados com o banco B, os resultados de compressão indicam que o aumento da resolução radiométrica implica em diminuição nas razões de compressão. Esse comportamento pode ser explicado pelo fato de maiores resoluções radiométricas implicarem em um aumento no número de níveis de quantização. Isso corresponde a um maior grau de detalhamento nas imagens obtidas pelo sensor, o que leva a uma menor correlação na imagem e consequentemente uma maior dificuldade de se realizar compressão.

Ao se fragmentar as imagens em blocos menores, era esperado que os valores de razão de compressão diminuíssem, já que a quantidade de informação prévia usada na compressão também diminui, influenciando na predição e na atualização do parâmetro de codificação. Contudo, os resultados mostram que a diminuição é muito pequena, justificando a utilidade da técnica com a finalidade de reduzir a propagação de erros. Na abordagem inicial em que se comprime a imagem de 5812x5812 (33.779.344 pixels) como um todo, considerando o pior caso em que o primeiro bit seja alterado por efeito de radiação espacial, por exemplo, toda a imagem seria descomprimida erroneamente. Por outro lado, caso a mesma imagem fosse dividida em blocos de 90x90 (8.100 pixels) para a compressão, um bit corrompido provocaria a perda apenas daquele bloco, representando aproximadamente 0,02% da mesma. Nessa abordagem, a RC média caiu apenas 1,3%. Logo, a compressão da imagem fragmentada em blocos pode ser vista como uma maneira simples e eficaz de tratar os problemas de erros inerentes às aplicações espaciais.

A diminuição da RC em função do tamanho do bloco também pode ser explicada pela importância da predição linear nos blocos muito pequenos. Isso significa dizer que em blocos muito pequenos existe uma quantidade grande de pixels nas bordas dos blocos em comparação com a quantidade total de pixels do bloco

como um todo. Logo, a predição linear, executada em todos os pixels das bordas, influi muito na RC final, o que leva a um menor desempenho de compressão, já que a predição bidimensional pode alcançar melhores resultados. Isso não acontece em um bloco maior, pois a quantidade de pixels nas bordas, em comparação com a quantidade total de pixels na imagem, é ínfima.

As comparações de hardware com outros trabalhos devem ser feitas com cuidado considerando variações algorítmicas e tecnológicas. Por exemplo, a quantidade de memória utilizada em cada caso varia de acordo com as dimensões das imagens usadas.

Mesmo sendo difícil estabelecer uma equivalência entre as várias tecnologias empregadas nos diversos trabalhos comparados, é perceptível que o trabalho proposto utiliza uma quantidade bem pequena de elementos lógicos e memórias disponíveis no FPGA, especialmente se compararmos com as outras implementações em tecnologia Altera, família Cyclone III.

Juntando a pequena área do circuito utilizado com o fato da imagem poder ser processada ao mesmo tempo em partes diferentes, é possível utilizar vários núcleos de compressão em paralelo. Com isso, a quantidade de lógica cresce proporcionalmente à quantidade de núcleos instanciados. A memória necessária permanece praticamente a mesma, pois ela é dividida entre os núcleos uma vez que a imagem também o é. A frequência máxima alcançada também se mantém praticamente inalterada.

A replicação de núcleos favorece também a vazão de dados do sistema. Um único núcleo trabalhando na frequência máxima obtida já apresenta um desempenho comparável a alguns dos outros trabalhos relacionados. Com o aumento de núcleos, o *throughput* cresce proporcionalmente à quantidade de compressores instanciados em paralelo, permitindo que essa taxa atinja centenas de Mbit/s, requisito fundamental para os satélites atuais, enquanto mantém os valores de área e memória em um nível aceitável.

Também é possível levantar a questão que a tecnologia empregada influencia no desempenho do hardware. Para o caso do dispositivo da família Stratix II, vê-se que o compressor alcança as maiores frequências e taxas de dados reportadas neste trabalho. Isso acontece porque esse dispositivo é o mais avançado em termos de tecnologias de fabricação de circuitos integrados dentre os selecionados para testes. Consequentemente, é o dispositivo mais caro. Percebe-se, então, como a escolha da tecnologia é um fator importante para se projetar um sistema digital de alta velocidade.

Os fatores aqui discutidos, tais como razão de compressão, síntese de hardware e desempenho, permitem que seja feita uma análise dos trabalhos pelo ponto de vista da complexidade. É possível perceber um ponto positivo ao se utilizar a abordagem de compressão proposta. Questões como facilidade de compreensão e implementação do método e especialmente os resultados obtidos em comparação com os outros trabalhos levam à interpretação que o trabalho proposto possui complexidade baixa, o que favorece não só sua implementação e otimização de uma maneira geral, mas também a sua utilização em um projeto de hardware e de missão espacial.

Capítulo V: Conclusão

Apesar da compressão de dados não ser um tema novo não só na computação de uma maneira geral mas também no âmbito espacial, ela é considerada muito importante, particularmente no que se refere à transmissão de grandes quantidades de dados do espaço para a Terra. É o caso das imagens de sensoriamento remoto, uma vez que existe a tendência de aumento de resoluções espacial, radiométrica e espectral dos diversos tipos de sensores. Isso revela a importância do sistema proposto, seus resultados e conclusões obtidas.

O presente trabalho apresenta uma solução de compressão sem perdas para imagens de satélite que explora a baixa complexidade como fator positivo para alcançar os requisitos inerentes às missões espaciais. Como mostram os resultados obtidos, o desempenho necessário é favorecido pela implementação em hardware, que provê vazão de dados em taxa suficiente para tal. Além disso, a simplicidade computacional sugere baixo consumo de energia, outro fator importante no projeto de um satélite. Todas essas características estão aliadas com ganhos de compressão compatíveis com outros métodos já utilizados em missões espaciais atuais.

Outra consequência da baixa complexidade é que o circuito demanda pequena quantidade de elementos lógicos e memória, o que favorece seu funcionamento em maiores frequências de operação, além de sugerir a inclusão de outras funcionalidades ao sistema, como técnicas de resistência a erros, redundância lógica, melhoramentos no método de compressão, etc. Uma das possibilidades foi abordada nesse trabalho e diz respeito à utilização de diversos núcleos de compressão operando em paralelo. Esse enfoque *multicore* é indicado para a obtenção de maiores taxas de *throughput* ao dividir o esforço de compressão entre núcleos trabalhando ao mesmo tempo. Ao utilizar um núcleo de hardware já validado, a expansão do sistema é facilitada pela simples replicação do mesmo.

Amplamente sugerida e aplicada na literatura, a atenção à propagação de erros nesse tipo de aplicação também é um requisito importante que foi abordado no compressor proposto. Sua inclusão no sistema é simples, de baixo impacto no ganho de compressão e de grande benefício.

Foi mostrado na comparação com outros trabalhos que o desenvolvimento de hardware para sistemas espaciais tem sido avaliado através de dispositivos FPGA convencionais. Porém, missões espaciais requerem materiais especiais adequados ao seu ambiente adverso. Tais materiais, que podem ser encontrados em FPGAs destinados a esse fim e na fabricação de ASICs específicos, são mais caros, de difícil acesso e seus processos de fabricação podem influir na performance dos dispositivos.

Mesmo assim, a validação, principalmente funcional, do compressor em FPGA convencional é útil, por exemplo, para o estudo e prova dos conceitos. O surgimento de técnicas mais robustas, particularmente no nível algorítmico, que permitam o funcionamento razoável de um FPGA convencional no ambiente espacial pode ser um grande aliado no barateamento e facilitação do projeto, integração e lançamento de satélites espaciais.

Para as missões espaciais, a manutenção de um sistema de hardware remoto é mais difícil, mesmo que empreguem dispositivos reconfiguráveis como FPGAs. Sendo assim, é fundamental que a metodologia de projeto seja robusta o suficiente para validar o hardware quanto a sua confiabilidade funcional em um nível aceitável. Dessa forma, esforços de verificação, testes e prototipação são cada vez mais necessários na implementação de sistemas espaciais em hardware. Isso demonstra outro ponto positivo do presente trabalho, que se baseia na metodologia BVM.

Com a utilização de imagens do satélite CBERS-2B disponibilizadas pelo INPE, o presente trabalho quis contribuir para o fortalecimento das pesquisas nacionais na área de compressão para incentivar e aproveitar as tecnologias brasileiras. Além disso, os resultados de compressão de imagens do CBERS obtidos neste trabalho podem contribuir efetivamente para o projeto de sistemas de compressão dos próximos satélites brasileiros.

Finalmente, é possível concluir que, dependendo dos requisitos de uma missão espacial que necessite de um hardware de compressão sem perdas, dentre as alternativas disponíveis, o método explorado no presente trabalho pode ser escolhido como método que possui um desempenho de compressão aceitável, aliado a sua implementação simples e desempenho similar aos outros métodos avaliados na literatura.

5.1. Trabalhos Futuros

O presente trabalho ainda pode ser explorado visando melhorar os problemas identificados e obter resultados mais precisos. Pode-se, por exemplo, modificar o sistema proposto para observar variações nas características de ganho de compressão, taxas de desempenho, síntese de hardware, etc. Neste sentido, tais melhorias seriam:

- Compressão alternativa para casos em que ocorra expansão ao invés de redução na quantidade de dados;
- Identificação e tratamento de variação abrupta nos dados de entrada, responsável por produzir códigos longos na saída;

- Otimizações na arquitetura de hardware, visando maior velocidade, particularmente pelo emprego de técnicas de pipeline;

Outro ponto a ser avaliado é quanto à abordagem *multicore*, pois seu emprego leva à definição de estratégias para a integração dos dados produzidos nas saídas de cada núcleo. Além disso, restrições impostas por interfaces de entrada e saída, e interação com memórias devem impactar na velocidade conceitual apresentada nos resultados correntes.

Um fator importante que não foi devidamente explorado no presente trabalho e que pode ser mais bem avaliado em trabalhos futuros é quanto ao consumo de energia. É importante realizar medidas de consumo para o sistema proposto, estabelecer comparações com outros trabalhos e observar possíveis mudanças na arquitetura e implementação visando alcançar esse requisito.

Referências Bibliográficas

- ABRARDO, A. et al. Error-resilient and low-complexity onboard lossless compression of hyperspectral images by means of distributed source coding. **IEEE Transactions on Geoscience and Remote Sensing**, 48, Abril 2010. 1892-1904.
- AL-LAHAM, M.; EMARY, I. M. M. E. Comparative Study Between Various Algorithms of Data Compression Techniques. **Proceedings of the World Congress on Engineering and Computer Science**, São Francisco, EUA, Outubro 2007. 326-336.
- ALTERA CORPORATION. **Nios Development Board - Reference Manual, Stratix Edition**. Altera. San Jose, CA, EUA. 2004.
- ARAÚJO, H. F. D. O. **BVM**: Reformulação da metodologia de verificação funcional VeriSC. Campina Grande: Universidade Federal de Campina Grande, 2010. Dissertação de Mestrado em Ciência da Computação.
- CARVALHO, E. G. et al. The Brazilian Multispectral Camera (MUX) for the China-Brazil Earth Resources Satellite - CBERS 3 & 4. **Proceedings of International Conference on Space Optics**, Rhodes, Greece, 4-8 Outubro 2010.
- CAST. JPEG-LS Compression Encoder Core. **CAST, Inc.**, 2007. Disponível em: <<http://www.cast-inc.com/ip-cores/images/jpegls-e/>>. Acesso em: 10 Agosto 2011.
- CHEN, X. et al. Hardware architecture for lossless image compression based on context-based modeling and arithmetic coding. **Proceedings of IEEE International SOC Conference**, Hsinchu, 24-27 Setembro 2007. 251-254.
- EUROPEAN SPACE AGENCY. Compression Stops Mission from Drowning in Data. **European Space Agency**, 2011. Disponível em: <http://www.esa.int/esaMI/Space_Engineering/SEM74PSRJHG_0.html>. Acesso em: 30 Junho 2011.
- FARIA, L. N.; FONSECA, L. M. G.; COSTA, M. H. M. Performance evaluation of data compression systems applied to satellite imagery. **JECE - Journal of Electrical and Computer Engineering**, New York, NY, United States, v. 2012, n. 18, p. 1-15, Janeiro 2012. ISSN 2090-0147.
- GOLOMB, S. W. Run-Length Encodings. **IEEE Transactions on Information Theory**, v. 12, p. 399-401, Setembro 1966.

HOWARD, G. P.; VITTER, S. J. **Fast and Efficient Lossless Image Compression**. Duke University. Durham. 1994.

HUFFMAN, D. A. A Method for the Construction of Minimum-Redundancy Codes. **Proceedings of the Institute of Radio Engineers**, Cambridge, Massachusetts, EUA, 40, Setembro 1952. 1098-1102.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS. Câmeras Imageadoras do CBERS-1, 2 e 2B. **Programa CBERS**, 2007. Disponível em: <<http://www.cbers.inpe.br/?content=cameras1e2e2b>>. Acesso em: 5 Julho 2011.

ISO/IEC. **14495-1: Information technology - Lossless and near-lossless coding of continuous tone still images: Baseline. JPEG-LS standard part-1**. ISO/IEC. [S.l.]. 1999.

ISO/IEC. **Information Technology—JPEG 2000 Image Coding System: Core Coding System**. International Standard, 15444-1:2004. Geneva. 2004.

KIELY, A.; KLIMESH, M. The ICER Progressive Wavelet Image Compressor. **Interplanetary Network Progress Report**, Pasadena, California, EUA, 15 Novembro 2003. 1-46.

LOPES FILHO, A.; D'AMORE, R. A low complexity image compression solution for onboard space applications. **Proceedings of the 23rd symposium on Integrated circuits and system design (SBCCI '10)**, New York, NY, USA, 2010. 174-179.

MARQUES, D. S. et al. A Biological Signals and Medical Images Lossless Compressor IP-Core Design. **Proceedings of First Workshop on Circuits and Systems Design**, João Pessoa - PB, 2011.

MEIRA, M. M.; LIMA, J. A. G. D.; BATISTA, L. V. An FPGA Implementation of a Lossless Electrocardiogram Compressor Based on Prediction and Golomb-Rice Coding. **Proceeding of XI Workshop Iberchip**, Salvador, 2005.

NATURAL RESOURCES CANADA. **Fundamentals of Remote Sensing**. [S.l.]: Canada Centre for Remote Sensing, 2008. Disponível em: <http://ccrs.nrcan.gc.ca/resource/tutor/fundam/chapter1/01_e.php>. Acesso em: 12 Julho 2011.

OGIELA, M. R.; TADEUSIEWICZ, R. **Modern Computational Intelligence Methods for the Interpretation of Medical Images**. Kraków: Springer, 2008.

O'NEAL JR., J. B. Predictive quantizing (differential pulse-code modulation) for the transmission of television signals. **Bell Syst. Tech. J.**, v. 45, p. 689-722, 1966.

PECHYONYJ, D.; BOGOMJAKOV, A. Lossless Image Compression. **Center for Intelligent Systems - Computer Science Department - Israel Institute of Technology**, 1998. Disponível em:
<http://cis.cs.technion.ac.il/Done_Projects/Projects_done/VisionClasses/DIP_1998/Lossless_Compression/lossless.html>. Acesso em: 5 Julho 2011.

REES, G. **The Remote Sensing Data Book**. Cambridge: Cambridge University Press, 1999.

RICE, R. F. **Channel coding and data compression system considerations for efficient, communication of planetary imaging data**. Jet Propulsion Laboratory - NASA. Pasadena, CA. 1974. (Tech. Memo. 33-695).

RICE, R. F. **Some practical universal noiseless coding technique**. Jet Propulsion Laboratory. [S.l.]. 1979.

RUSS, J. C. **The Image Processing Handbook**. 6ª. ed. Boca Raton, FL, EUA: CRC Press, 2011.

SAID, A.; PEARLMAN, W. A. A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees. **IEEE Transactions on Circuits and Systems for Video Technology**, 6, Junho 1996. 243-250.

SALOMON, D. Data Compression: The Complete Reference. 2nd Edition. **Extra Material**: Correlation in Statistics and in Data Compression, 3 Novembro 2000. Disponível em: <<http://www.davidsalomon.name/DC2advertis/Corr.pdf>>. Acesso em: 3 Maio 2012.

SALOMON, D. Data Compression: The Complete Reference. 2nd Edition. **Extra Material**: Decorrelation in Statistics: The Mahalanobis Transformation, 2001. Disponível em: <<http://www.davidsalomon.name/DC2advertis/DeCorr.pdf>>. Acesso em: 3 Maio 2012.

SALOMON, D. **Data Compression**: The Complete Reference. Nova Iorque: Springer, 2004.

SALOMON, D. **Data Compression**: The Complete Reference. Nova Iorque: Springer, 2004.

SEROUSSI, G.; WEINBERGER, M. J. On Adaptive Strategies for an Extended Family of Golomb-Type Codes. **Proceedings of the Conference on Data Compression (DCC '97)**, Washington, 1997. 131.

THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS. **Lossless Data Compression: Informational Report**. CCSDS. Washington, p. 42. 2006.

THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS. **Image Data Compression: Informational Report**. CCSDS. Washington, p. 117. 2007.

WANG, Z. et al. A High Performance Fully Pipelined Architecture for Lossless Compression of Satellite Image. **Proceedings of International Conference on Multimedia Technology (ICMT)**, Ningbo, 29-31 Outubro 2010. 1-4.

WEINBERGER, M. J.; SEROUSSI, G.; SAPIRO, G. LOCO-I: a low complexity, context-based, lossless image compression algorithm. **Proceedings of the Conference on Data Compression (DCC '96)**, Washington, 1996. 140.

WELCH, T. A Technique for High-Performance Data Compression. **IEEE Computer**, 17, Junho 1984. 8-19.

YU, G.; VLADIMIROVA, T.; SWEETING, M. Image compression systems on board satellites. **Acta Astronautica**, v. 64, n. 9-10, p. 988-1005, Maio-Junho 2009. ISSN 00945765.

ZHANG, R.; LIU, Z.; YU, N. Decorrelation of multispectral images for lossless compression. **Proceedings of SPIE**, Wuhan, China, 23 Outubro 2001.

Apêndice I: Resultado detalhado de compressão para o banco de imagens A

A Tabela 5 lista as razões de compressão obtidas para todas as imagens do banco A (imagens do satélite CBERS-2B, fornecidas pelo INPE) comprimidas pelo modelo de referência do trabalho proposto. Em azul é marcado o pior caso de cada banda e em vermelho o melhor. As colunas B1, B2, B3, B4 e B5 referem-se respectivamente aos resultados nas bandas pancromática, azul, verde, vermelho e infravermelho próximo.

Tabela 5: Resultados de RC do trabalho proposto para o banco A.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,89	4,12	4,02	3,70	4,18
20080302_L0_G1_P154_R127	3,63	3,91	3,79	2,93	3,78
20080305_L0_G1_P153_R127	3,98	4,34	4,33	3,84	4,29
20080323_L0_G1_P121_R31	1,96	1,96	1,89	1,89	1,97
20080328_L0_G1_P154_R126	3,11	3,28	2,99	2,22	3,12
20080525_L0_G1_P152_R126	4,12	4,40	4,23	3,09	4,15
20080712_L0_G1_P162_R102	3,84	4,21	4,09	2,70	4,08
20080713_L0_G1_P179_R118	3,50	3,73	3,35	3,04	3,52
20080716_L0_G1_P178_R105	4,18	4,65	4,49	2,64	4,36
20080719_L0_G1_P151_R127	4,34	5,03	4,94	5,24	4,94
20080719_L0_G1_P177_R130	3,04	3,25	2,94	2,87	3,15
20080805_L0_G1_P180_R103	3,80	4,11	4,22	2,68	4,15
20080811_L0_G1_P178_R119	3,15	3,24	2,85	2,64	3,06
20080814_L0_G1_P177_R124	3,39	3,61	3,30	3,17	3,47
20080826_L0_G1_P173_R103	3,03	3,36	3,15	2,63	3,27
20080829_L0_G1_P172_R100	2,71	2,85	2,77	2,31	2,81
20081212_L0_G1_P189_R112	3,30	3,56	3,19	3,04	3,36
20090103_L0_G1_P173_R103	2,92	2,95	2,99	2,61	2,97
20090201_L0_G1_P146_R137	2,88	3,15	3,05	3,13	3,14
20090202_L0_G1_P189_R112	2,88	2,99	2,86	2,79	2,94
Média	3,38	3,63	3,47	2,96	3,54
Média Total	3,40				

A Tabela 6, por sua vez, segue o mesmo padrão da Tabela 5, porém utiliza a abordagem original de atualização do parâmetro da codificação de Golomb-Rice, como apresentada no Capítulo II, em contraste aos resultados da Tabela 5 que utiliza a abordagem que simplifica as operações envolvidas na atualização do parâmetro da codificação de Golomb-Rice, proposta no Capítulo III.

Tabela 6: Resultados de RC utilizando abordagem original de atualização do parâmetro da codificação de Golomb-Rice.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,74	4,02	3,85	3,47	4,07
20080302_L0_G1_P154_R127	3,46	3,81	3,58	2,80	3,67
20080305_L0_G1_P153_R127	3,91	4,29	4,24	3,68	4,24
20080323_L0_G1_P121_R31	2,04	2,05	1,87	1,94	2,01
20080328_L0_G1_P154_R126	2,99	3,21	2,91	2,39	3,02
20080525_L0_G1_P152_R126	4,10	4,38	4,21	2,91	4,13
20080712_L0_G1_P162_R102	3,76	4,16	4,00	2,70	4,04
20080713_L0_G1_P179_R118	3,37	3,63	3,23	2,95	3,43
20080716_L0_G1_P178_R105	4,18	4,64	4,48	2,78	4,35
20080719_L0_G1_P151_R127	4,33	5,02	4,94	5,24	4,94
20080719_L0_G1_P177_R130	2,81	3,02	2,82	2,82	2,95
20080805_L0_G1_P180_R103	3,76	4,09	4,20	2,73	4,14
20080811_L0_G1_P178_R119	3,09	3,20	2,88	2,76	3,01
20080814_L0_G1_P177_R124	3,30	3,56	3,20	3,07	3,41
20080826_L0_G1_P173_R103	2,87	3,23	2,92	2,67	3,10
20080829_L0_G1_P172_R100	2,40	2,64	2,44	2,31	2,57
20081212_L0_G1_P189_R112	3,23	3,52	3,09	2,95	3,31
20090103_L0_G1_P173_R103	2,53	2,66	2,55	2,46	2,62
20090201_L0_G1_P146_R137	2,76	2,96	2,83	2,90	2,93
20090202_L0_G1_P189_R112	2,67	2,81	2,67	2,66	2,77
Média	3,27	3,54	3,35	2,91	3,44
Média Total	3,30				

Similarmente, Tabela 7, Tabela 8 e Tabela 9 listam as RCs obtidas para as implementações de referência do SPIHT, JPEG-2000 e JPEG-LS, respectivamente.

Tabela 7: Resultados de RC do SPIHT para o banco A.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	4,18	4,13	4,16	3,98	4,53
20080302_L0_G1_P154_R127	3,68	4,02	3,99	3,23	4,06
20080305_L0_G1_P153_R127	4,03	4,51	4,56	4,05	4,58
20080323_L0_G1_P121_R31	2,32	2,32	2,25	2,25	2,34
20080328_L0_G1_P154_R126	3,26	3,45	3,26	2,60	3,38
20080525_L0_G1_P152_R126	4,19	4,48	4,41	3,52	4,44
20080712_L0_G1_P162_R102	3,83	4,00	4,03	2,97	4,25
20080713_L0_G1_P179_R118	3,62	3,80	3,58	3,34	3,75
20080716_L0_G1_P178_R105	4,16	4,44	4,46	2,93	4,65
20080719_L0_G1_P151_R127	4,45	5,15	5,02	5,53	5,28
20080719_L0_G1_P177_R130	3,19	3,41	3,25	3,19	3,39

20080805_L0_G1_P180_R103	3,84	4,11	4,47	2,96	4,43
20080811_L0_G1_P178_R119	3,32	3,45	3,18	2,99	3,31
20080814_L0_G1_P177_R124	3,50	3,75	3,57	3,45	3,66
20080826_L0_G1_P173_R103	3,27	3,62	3,48	2,96	3,59
20080829_L0_G1_P172_R100	2,94	3,18	3,10	2,68	3,17
20081212_L0_G1_P189_R112	3,50	3,80	3,49	3,38	3,63
20090103_L0_G1_P173_R103	3,18	3,31	3,31	2,99	3,27
20090201_L0_G1_P146_R137	3,21	3,52	3,41	3,51	3,47
20090202_L0_G1_P189_R112	3,16	3,37	3,24	3,19	3,31
Média	3,54	3,79	3,71	3,28	3,82
Média Total	3,63				

Tabela 8: Resultados de RC do JPEG-2000 para o banco A.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	4,43	4,79	4,82	4,63	4,85
20080302_L0_G1_P154_R127	3,86	4,27	4,16	3,33	4,19
20080305_L0_G1_P153_R127	4,27	4,92	4,87	4,40	4,90
20080323_L0_G1_P121_R31	2,28	2,28	2,21	2,22	2,33
20080328_L0_G1_P154_R126	3,34	3,54	3,27	2,57	3,37
20080525_L0_G1_P152_R126	4,46	5,02	4,84	3,77	4,75
20080712_L0_G1_P162_R102	4,06	4,54	4,42	3,12	4,42
20080713_L0_G1_P179_R118	3,81	4,20	3,81	3,55	3,86
20080716_L0_G1_P178_R105	4,45	4,99	4,99	3,07	4,89
20080719_L0_G1_P151_R127	4,82	6,11	6,05	6,79	6,19
20080719_L0_G1_P177_R130	3,30	3,63	3,34	3,31	3,43
20080805_L0_G1_P180_R103	4,09	4,49	4,70	2,99	4,57
20080811_L0_G1_P178_R119	3,39	3,53	3,16	3,01	3,30
20080814_L0_G1_P177_R124	3,63	3,92	3,61	3,52	3,70
20080826_L0_G1_P173_R103	3,37	3,77	3,56	3,00	3,64
20080829_L0_G1_P172_R100	3,01	3,25	3,15	2,70	3,17
20081212_L0_G1_P189_R112	3,56	3,87	3,48	3,39	3,63
20090103_L0_G1_P173_R103	3,31	3,39	3,42	3,06	3,31
20090201_L0_G1_P146_R137	3,33	3,58	3,51	3,61	3,50
20090202_L0_G1_P189_R112	3,25	3,40	3,28	3,24	3,31
Média	3,70	4,07	3,93	3,47	3,97
Média Total	3,83				

Tabela 9: Resultados de RC do JPEG-LS para o banco A.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	4,62	4,82	4,84	4,55	4,84
20080302_L0_G1_P154_R127	4,07	4,45	4,30	3,31	4,32

20080305_L0_G1_P153_R127	4,53	5,07	4,97	4,44	5,01
20080323_L0_G1_P121_R31	2,27	2,27	2,23	2,22	2,34
20080328_L0_G1_P154_R126	3,43	3,66	3,34	2,58	3,44
20080525_L0_G1_P152_R126	4,70	5,11	4,92	3,65	4,74
20080712_L0_G1_P162_R102	4,34	4,83	4,73	3,16	4,66
20080713_L0_G1_P179_R118	3,90	4,30	3,85	3,55	3,90
20080716_L0_G1_P178_R105	4,74	5,37	5,31	3,08	5,09
20080719_L0_G1_P151_R127	5,03	5,89	5,71	5,97	5,65
20080719_L0_G1_P177_R130	3,36	3,75	3,39	3,34	3,47
20080805_L0_G1_P180_R103	4,30	4,76	4,87	3,00	4,76
20080811_L0_G1_P178_R119	3,44	3,65	3,17	3,00	3,32
20080814_L0_G1_P177_R124	3,73	4,09	3,69	3,57	3,78
20080826_L0_G1_P173_R103	3,44	3,85	3,63	3,01	3,70
20080829_L0_G1_P172_R100	3,22	3,38	3,34	2,80	3,29
20081212_L0_G1_P189_R112	3,64	4,00	3,51	3,40	3,68
20090103_L0_G1_P173_R103	3,59	3,52	3,64	3,21	3,47
20090201_L0_G1_P146_R137	3,37	3,61	3,52	3,61	3,52
20090202_L0_G1_P189_R112	3,37	3,45	3,36	3,30	3,33
Média	3,86	4,19	4,02	3,44	4,01
Média Total	3,90				

Por fim, as razões de compressão para cada imagem do banco A quando da compressão em blocos, desde 2906x2906 até 5x5, do trabalho proposto são listadas da Tabela 10 à Tabela 19, mantendo o mesmo padrão de colorização.

Tabela 10: RCs obtidas da compressão em blocos de 2906x2906 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,89	4,12	4,02	3,70	4,18
20080302_L0_G1_P154_R127	3,63	3,91	3,79	2,93	3,78
20080305_L0_G1_P153_R127	3,98	4,33	4,33	3,84	4,29
20080323_L0_G1_P121_R31	1,95	1,96	1,89	1,89	1,97
20080328_L0_G1_P154_R126	3,11	3,28	2,99	2,22	3,12
20080525_L0_G1_P152_R126	4,12	4,40	4,23	3,09	4,15
20080712_L0_G1_P162_R102	3,84	4,21	4,09	2,70	4,08
20080713_L0_G1_P179_R118	3,50	3,73	3,35	3,04	3,52
20080716_L0_G1_P178_R105	4,18	4,65	4,49	2,64	4,36
20080719_L0_G1_P151_R127	4,34	5,02	4,94	5,24	4,94
20080719_L0_G1_P177_R130	3,04	3,25	2,94	2,87	3,15
20080805_L0_G1_P180_R103	3,80	4,11	4,22	2,68	4,15
20080811_L0_G1_P178_R119	3,15	3,24	2,84	2,64	3,06
20080814_L0_G1_P177_R124	3,39	3,61	3,30	3,17	3,47
20080826_L0_G1_P173_R103	3,03	3,36	3,15	2,63	3,27
20080829_L0_G1_P172_R100	2,71	2,85	2,77	2,31	2,81
20081212_L0_G1_P189_R112	3,29	3,56	3,19	3,04	3,36

20090103_L0_G1_P173_R103	2,92	2,95	2,98	2,61	2,97
20090201_L0_G1_P146_R137	2,88	3,14	3,05	3,13	3,14
20090202_L0_G1_P189_R112	2,87	2,99	2,86	2,78	2,94
Média	3,38	3,63	3,47	2,96	3,54
Média Total	3,40				

Tabela 11: RCs obtidas da compressão em blocos de 1453x1453 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,88	4,11	4,01	3,70	4,18
20080302_L0_G1_P154_R127	3,62	3,91	3,79	2,93	3,78
20080305_L0_G1_P153_R127	3,98	4,33	4,33	3,83	4,29
20080323_L0_G1_P121_R31	1,95	1,96	1,89	1,89	1,97
20080328_L0_G1_P154_R126	3,11	3,28	2,99	2,22	3,12
20080525_L0_G1_P152_R126	4,12	4,40	4,23	3,09	4,15
20080712_L0_G1_P162_R102	3,84	4,21	4,09	2,70	4,08
20080713_L0_G1_P179_R118	3,50	3,73	3,35	3,04	3,52
20080716_L0_G1_P178_R105	4,18	4,64	4,49	2,64	4,35
20080719_L0_G1_P151_R127	4,34	5,02	4,94	5,23	4,94
20080719_L0_G1_P177_R130	3,03	3,25	2,94	2,87	3,15
20080805_L0_G1_P180_R103	3,79	4,11	4,22	2,68	4,15
20080811_L0_G1_P178_R119	3,15	3,24	2,84	2,64	3,06
20080814_L0_G1_P177_R124	3,39	3,61	3,29	3,17	3,47
20080826_L0_G1_P173_R103	3,03	3,36	3,15	2,63	3,27
20080829_L0_G1_P172_R100	2,71	2,85	2,77	2,30	2,81
20081212_L0_G1_P189_R112	3,29	3,56	3,19	3,04	3,36
20090103_L0_G1_P173_R103	2,92	2,94	2,98	2,61	2,97
20090201_L0_G1_P146_R137	2,88	3,14	3,05	3,13	3,14
20090202_L0_G1_P189_R112	2,87	2,98	2,86	2,78	2,94
Média	3,38	3,63	3,47	2,96	3,54
Média Total	3,39				

Tabela 12: RCs obtidas da compressão em blocos de 726x726 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,88	4,11	4,01	3,69	4,17
20080302_L0_G1_P154_R127	3,62	3,90	3,78	2,92	3,78
20080305_L0_G1_P153_R127	3,97	4,33	4,33	3,83	4,29
20080323_L0_G1_P121_R31	1,95	1,95	1,89	1,89	1,97
20080328_L0_G1_P154_R126	3,11	3,28	2,98	2,21	3,12
20080525_L0_G1_P152_R126	4,11	4,39	4,23	3,09	4,14
20080712_L0_G1_P162_R102	3,83	4,20	4,08	2,70	4,08
20080713_L0_G1_P179_R118	3,50	3,72	3,34	3,03	3,52
20080716_L0_G1_P178_R105	4,18	4,64	4,48	2,63	4,35
20080719_L0_G1_P151_R127	4,33	5,01	4,93	5,23	4,94

20080719_L0_G1_P177_R130	3,03	3,24	2,94	2,86	3,14
20080805_L0_G1_P180_R103	3,79	4,11	4,22	2,68	4,15
20080811_L0_G1_P178_R119	3,15	3,23	2,84	2,64	3,06
20080814_L0_G1_P177_R124	3,38	3,61	3,29	3,17	3,46
20080826_L0_G1_P173_R103	3,03	3,36	3,14	2,62	3,27
20080829_L0_G1_P172_R100	2,71	2,85	2,77	2,30	2,81
20081212_L0_G1_P189_R112	3,29	3,56	3,19	3,04	3,36
20090103_L0_G1_P173_R103	2,92	2,94	2,98	2,61	2,97
20090201_L0_G1_P146_R137	2,88	3,14	3,05	3,13	3,14
20090202_L0_G1_P189_R112	2,87	2,98	2,86	2,78	2,94
Média	3,38	3,63	3,47	2,95	3,53
Média Total	3,39				

Tabela 13: RCs obtidas da compressão em blocos de 363x363 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,87	4,10	4,00	3,68	4,17
20080302_L0_G1_P154_R127	3,62	3,90	3,78	2,92	3,78
20080305_L0_G1_P153_R127	3,97	4,32	4,32	3,83	4,28
20080323_L0_G1_P121_R31	1,95	1,95	1,89	1,88	1,96
20080328_L0_G1_P154_R126	3,11	3,27	2,98	2,21	3,11
20080525_L0_G1_P152_R126	4,11	4,38	4,22	3,08	4,14
20080712_L0_G1_P162_R102	3,83	4,19	4,07	2,69	4,07
20080713_L0_G1_P179_R118	3,49	3,72	3,34	3,03	3,51
20080716_L0_G1_P178_R105	4,17	4,63	4,47	2,63	4,35
20080719_L0_G1_P151_R127	4,32	5,00	4,92	5,21	4,93
20080719_L0_G1_P177_R130	3,03	3,24	2,93	2,86	3,14
20080805_L0_G1_P180_R103	3,79	4,10	4,21	2,68	4,14
20080811_L0_G1_P178_R119	3,15	3,23	2,84	2,64	3,06
20080814_L0_G1_P177_R124	3,38	3,60	3,29	3,17	3,46
20080826_L0_G1_P173_R103	3,02	3,35	3,14	2,62	3,26
20080829_L0_G1_P172_R100	2,70	2,84	2,76	2,30	2,80
20081212_L0_G1_P189_R112	3,29	3,56	3,18	3,03	3,36
20090103_L0_G1_P173_R103	2,91	2,94	2,97	2,60	2,96
20090201_L0_G1_P146_R137	2,87	3,14	3,04	3,12	3,13
20090202_L0_G1_P189_R112	2,87	2,98	2,85	2,78	2,94
Média	3,37	3,62	3,46	2,95	3,53
Média Total	3,39				

Tabela 14: RCs obtidas da compressão em blocos de 181x181 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,86	4,08	3,98	3,67	4,15
20080302_L0_G1_P154_R127	3,61	3,89	3,77	2,91	3,77
20080305_L0_G1_P153_R127	3,96	4,31	4,31	3,81	4,27
20080323_L0_G1_P121_R31	1,94	1,95	1,88	1,88	1,96

20080328_L0_G1_P154_R126	3,10	3,26	2,97	2,20	3,10
20080525_L0_G1_P152_R126	4,10	4,37	4,20	3,07	4,13
20080712_L0_G1_P162_R102	3,82	4,18	4,06	2,69	4,06
20080713_L0_G1_P179_R118	3,48	3,70	3,33	3,02	3,50
20080716_L0_G1_P178_R105	4,16	4,61	4,46	2,62	4,34
20080719_L0_G1_P151_R127	4,31	4,98	4,90	5,19	4,91
20080719_L0_G1_P177_R130	3,02	3,23	2,92	2,85	3,13
20080805_L0_G1_P180_R103	3,78	4,09	4,20	2,67	4,14
20080811_L0_G1_P178_R119	3,14	3,22	2,83	2,63	3,05
20080814_L0_G1_P177_R124	3,37	3,59	3,28	3,16	3,45
20080826_L0_G1_P173_R103	3,02	3,34	3,13	2,61	3,25
20080829_L0_G1_P172_R100	2,69	2,83	2,75	2,29	2,79
20081212_L0_G1_P189_R112	3,28	3,55	3,18	3,03	3,35
20090103_L0_G1_P173_R103	2,90	2,92	2,96	2,59	2,95
20090201_L0_G1_P146_R137	2,87	3,13	3,03	3,12	3,12
20090202_L0_G1_P189_R112	2,86	2,97	2,84	2,77	2,93
Média	3,36	3,61	3,45	2,94	3,52
Média Total	3,38				

Tabela 15: RCs obtidas da compressão em blocos de 90x90 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,84	4,04	3,95	3,64	4,13
20080302_L0_G1_P154_R127	3,59	3,86	3,75	2,89	3,75
20080305_L0_G1_P153_R127	3,93	4,28	4,28	3,79	4,25
20080323_L0_G1_P121_R31	1,93	1,93	1,87	1,86	1,95
20080328_L0_G1_P154_R126	3,08	3,24	2,95	2,19	3,09
20080525_L0_G1_P152_R126	4,07	4,33	4,17	3,05	4,11
20080712_L0_G1_P162_R102	3,79	4,14	4,03	2,67	4,04
20080713_L0_G1_P179_R118	3,46	3,68	3,30	3,00	3,48
20080716_L0_G1_P178_R105	4,14	4,57	4,42	2,61	4,31
20080719_L0_G1_P151_R127	4,28	4,94	4,85	5,14	4,88
20080719_L0_G1_P177_R130	3,00	3,21	2,90	2,83	3,11
20080805_L0_G1_P180_R103	3,76	4,06	4,17	2,65	4,12
20080811_L0_G1_P178_R119	3,12	3,20	2,81	2,61	3,03
20080814_L0_G1_P177_R124	3,35	3,57	3,26	3,14	3,43
20080826_L0_G1_P173_R103	3,00	3,32	3,11	2,59	3,23
20080829_L0_G1_P172_R100	2,67	2,81	2,72	2,27	2,77
20081212_L0_G1_P189_R112	3,26	3,53	3,16	3,01	3,33
20090103_L0_G1_P173_R103	2,88	2,90	2,94	2,57	2,93
20090201_L0_G1_P146_R137	2,85	3,11	3,01	3,10	3,10
20090202_L0_G1_P189_R112	2,84	2,95	2,82	2,75	2,91
Média	3,34	3,58	3,42	2,92	3,50
Média Total	3,35				

Tabela 16: RCs obtidas da compressão em blocos de 45x45 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,79	3,96	3,87	3,57	4,07
20080302_L0_G1_P154_R127	3,55	3,81	3,70	2,85	3,71
20080305_L0_G1_P153_R127	3,89	4,23	4,23	3,74	4,20
20080323_L0_G1_P121_R31	1,91	1,91	1,84	1,84	1,92
20080328_L0_G1_P154_R126	3,05	3,21	2,92	2,16	3,06
20080525_L0_G1_P152_R126	4,02	4,27	4,11	3,01	4,06
20080712_L0_G1_P162_R102	3,75	4,08	3,96	2,63	3,99
20080713_L0_G1_P179_R118	3,42	3,62	3,25	2,96	3,44
20080716_L0_G1_P178_R105	4,09	4,49	4,35	2,58	4,27
20080719_L0_G1_P151_R127	4,23	4,84	4,76	5,04	4,81
20080719_L0_G1_P177_R130	2,97	3,16	2,86	2,79	3,08
20080805_L0_G1_P180_R103	3,72	4,01	4,12	2,62	4,08
20080811_L0_G1_P178_R119	3,09	3,17	2,78	2,58	3,00
20080814_L0_G1_P177_R124	3,32	3,53	3,22	3,10	3,40
20080826_L0_G1_P173_R103	2,96	3,28	3,06	2,56	3,20
20080829_L0_G1_P172_R100	2,62	2,77	2,68	2,24	2,73
20081212_L0_G1_P189_R112	3,23	3,49	3,13	2,98	3,30
20090103_L0_G1_P173_R103	2,83	2,86	2,90	2,54	2,88
20090201_L0_G1_P146_R137	2,82	3,07	2,98	3,06	3,07
20090202_L0_G1_P189_R112	2,80	2,91	2,79	2,72	2,87
Média	3,30	3,53	3,38	2,88	3,46
Média Total	3,31				

Tabela 17: RCs obtidas da compressão em blocos de 23x23 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,69	3,79	3,72	3,45	3,96
20080302_L0_G1_P154_R127	3,46	3,72	3,62	2,78	3,64
20080305_L0_G1_P153_R127	3,79	4,11	4,13	3,64	4,10
20080323_L0_G1_P121_R31	1,86	1,87	1,80	1,79	1,87
20080328_L0_G1_P154_R126	2,99	3,13	2,85	2,11	2,99
20080525_L0_G1_P152_R126	3,91	4,12	3,98	2,92	3,97
20080712_L0_G1_P162_R102	3,65	3,94	3,83	2,56	3,90
20080713_L0_G1_P179_R118	3,34	3,51	3,16	2,87	3,36
20080716_L0_G1_P178_R105	3,98	4,33	4,20	2,53	4,16
20080719_L0_G1_P151_R127	4,11	4,65	4,57	4,83	4,65
20080719_L0_G1_P177_R130	2,89	3,08	2,78	2,72	3,00
20080805_L0_G1_P180_R103	3,63	3,89	4,01	2,56	4,00
20080811_L0_G1_P178_R119	3,03	3,09	2,72	2,52	2,94
20080814_L0_G1_P177_R124	3,24	3,44	3,15	3,03	3,33
20080826_L0_G1_P173_R103	2,89	3,19	2,98	2,50	3,12
20080829_L0_G1_P172_R100	2,55	2,69	2,60	2,18	2,65
20081212_L0_G1_P189_R112	3,16	3,41	3,06	2,91	3,23

20090103_L0_G1_P173_R103	2,76	2,79	2,82	2,47	2,80
20090201_L0_G1_P146_R137	2,75	3,00	2,91	2,99	2,99
20090202_L0_G1_P189_R112	2,74	2,84	2,72	2,65	2,80
Média	3,22	3,43	3,28	2,80	3,37
Média Total	3,22				

Tabela 18: RCs obtidas da compressão em blocos de 11x11 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	3,45	3,40	3,36	3,14	3,69
20080302_L0_G1_P154_R127	3,25	3,47	3,40	2,61	3,43
20080305_L0_G1_P153_R127	3,54	3,81	3,85	3,38	3,83
20080323_L0_G1_P121_R31	1,77	1,77	1,70	1,70	1,77
20080328_L0_G1_P154_R126	2,82	2,95	2,69	1,99	2,84
20080525_L0_G1_P152_R126	3,64	3,77	3,65	2,72	3,72
20080712_L0_G1_P162_R102	3,40	3,59	3,50	2,40	3,65
20080713_L0_G1_P179_R118	3,13	3,24	2,93	2,68	3,17
20080716_L0_G1_P178_R105	3,70	3,92	3,82	2,40	3,89
20080719_L0_G1_P151_R127	3,79	4,17	4,11	4,31	4,27
20080719_L0_G1_P177_R130	2,73	2,87	2,60	2,54	2,83
20080805_L0_G1_P180_R103	3,40	3,60	3,73	2,43	3,77
20080811_L0_G1_P178_R119	2,86	2,91	2,57	2,38	2,79
20080814_L0_G1_P177_R124	3,06	3,21	2,96	2,85	3,15
20080826_L0_G1_P173_R103	2,72	2,98	2,80	2,35	2,94
20080829_L0_G1_P172_R100	2,40	2,52	2,44	2,06	2,50
20081212_L0_G1_P189_R112	2,99	3,21	2,89	2,75	3,06
20090103_L0_G1_P173_R103	2,61	2,63	2,66	2,34	2,64
20090201_L0_G1_P146_R137	2,60	2,82	2,75	2,81	2,82
20090202_L0_G1_P189_R112	2,59	2,68	2,58	2,50	2,64
Média	3,02	3,18	3,05	2,62	3,17
Média Total	3,01				

Tabela 19: RCs obtidas da compressão em blocos de 5x5 do trabalho proposto.

Imagem\Banda (RC)	B1	B2	B3	B4	B5
20071027_L0_G1_P153_R127	2,84	2,53	2,55	2,43	3,03
20080302_L0_G1_P154_R127	2,70	2,84	2,83	2,22	2,89
20080305_L0_G1_P153_R127	2,90	3,07	3,14	2,76	3,13
20080323_L0_G1_P121_R31	1,57	1,57	1,51	1,50	1,57
20080328_L0_G1_P154_R126	2,40	2,49	2,31	1,75	2,44
20080525_L0_G1_P152_R126	2,94	2,94	2,89	2,28	3,06
20080712_L0_G1_P162_R102	2,78	2,77	2,75	2,04	3,01
20080713_L0_G1_P179_R118	2,61	2,59	2,40	2,24	2,66
20080716_L0_G1_P178_R105	2,99	2,97	2,93	2,06	3,19
20080719_L0_G1_P151_R127	3,03	3,11	3,09	3,21	3,36
20080719_L0_G1_P177_R130	2,31	2,36	2,19	2,15	2,41

20080805_L0_G1_P180_R103	2,78	2,87	3,04	2,11	3,14
20080811_L0_G1_P178_R119	2,44	2,45	2,21	2,06	2,39
20080814_L0_G1_P177_R124	2,57	2,66	2,51	2,42	2,67
20080826_L0_G1_P173_R103	2,31	2,48	2,36	2,03	2,50
20080829_L0_G1_P172_R100	2,09	2,15	2,11	1,81	2,15
20081212_L0_G1_P189_R112	2,54	2,70	2,47	2,37	2,61
20090103_L0_G1_P173_R103	2,27	2,26	2,32	2,05	2,28
20090201_L0_G1_P146_R137	2,24	2,39	2,36	2,41	2,40
20090202_L0_G1_P189_R112	2,24	2,29	2,24	2,17	2,27
Média	2,53	2,57	2,51	2,20	2,66
Média Total	2,49				

Apêndice II: Resultado detalhado de compressão para o banco de imagens B

As razões de compressão obtidas para todas as imagens contidas no banco de imagens B em todos os métodos considerados, incluindo o trabalho proposto, são listadas na Tabela 20. Em azul são marcados os piores casos de cada caso e em vermelho os melhores.

Tabela 20: Quadro comparativo entre algoritmos de compressão sem perdas no banco de imagens B.

Resolução Radiométrica	Imagem	Razão de compressão						
		CCSDS/Rice	JPEG-LS	CCSDS	JPEG 2000	SPIHT	ICER	Trabalho Proposto
8 bits	coastal_b1	2,25	2,59	2,38	2,56	2,59	2,61	2,18
	coastal_b2	2,41	2,76	2,48	2,69	2,72	2,74	2,4
	coastal_b3	2,17	2,48	2,3	2,48	2,49	2,5	2,12
	coastal_b4	2,75	3,32	2,85	3,16	3,11	3,14	2,85
	coastal_b5	2,42	2,85	2,52	2,72	2,75	2,77	2,42
	coastal_b6h	2,91	3,2	2,65	3,08	2,95	3,15	2,94
	coastal_b6l	3,94	4,55	3,4	4,08	3,96	4,28	4,14
	coastal_b7	2,19	2,52	2,32	2,48	2,52	2,54	2,13
	coastal_b8	2,04	2,34	2,18	2,35	2,39	2,42	1,89
	spot-la_b3	1,54	1,77	1,67	1,71	1,7	1,75	1,55
	spot-la_panchr	1,64	2	1,88	1,94	1,95	1,99	1,71
	média para 8 bits	2,39	2,76	2,42	2,66	2,65	2,72	2,39
10 bits	ice_2kb1	1,84	2,11	2,09	2,11	2,17	2,16	1,93
	ice_2kb4	2,59	3,1	2,97	3,08	3,15	3,14	2,81
	india_2kb1	1,9	2,16	2,1	2,12	2,16	2,16	1,99
	india_2kb4	2,13	2,52	2,46	2,49	2,54	2,54	2,33

10 bits	landesV_G7_10b	1,59	2,26	1,98	2,19	2,05	2,26	1,93
	marseille_G6_10b	1,32	1,52	1,49	1,49	1,52	1,54	1,44
	ocean_2kb1	1,88	2,17	2,02	2,05	2,09	2,11	1,93
	ocean_2kb4	2,27	2,78	2,62	2,68	2,72	2,75	2,5
	média para 10 bits	1,94	2,33	2,22	2,28	2,3	2,33	2,11
12 bits	pleiades_portdebouc_b3	1,39	1,54	1,52	1,51	1,56	1,55	1,48
	pleiades_portdebouc_pan	1,52	1,69	1,67	1,65	*	1,71	1,61
	média para 12 bits	1,45	1,62	1,6	1,58	1,56	1,63	1,55
Resolução Radiométrica	Imagem	CCSDS/Rice	JPEG-LS	CCSDS	JPEG 2000	SPIHT	ICER	Trabalho Proposto
		Razão de compressão						
*dado indisponível em (THE CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS, 2007).								

Apêndice III:

A Low Complexity Lossless Data Compressor IP-Core for Satellite Images

Yuri Gonzaga Gonçalves da Costa, José Antonio Gomes de Lima and Guilherme Navarro
Informatics Center
Federal University of Paraíba
João Pessoa, Brazil
{yuriggc, jose}@di.ufpb.br, guilhermenavarro@gmail.com

Abstract—As the technology advances, space imaging systems use equipment of increasing resolutions. Hence, it is necessary to ensure that this great quantity of data arrives at its destination reliably. Among some variables involved, data compression plays an important role to accomplish this requirement. In this context, this paper proposes a digital hardware approach of a low complexity satellite image lossless compressor based on prediction and Golomb-Rice coding, which has achieved excellent results considering hardware and compression performance. In order to validate and analyze the compressor, a functional verification and FPGA prototyping methodology were followed. Given an image set from Brazilian's National Institute for Space Research (INPE, in Portuguese acronyms), its results on FPGA show that this compressor achieves compression ratio around 3.4, comparable value to related works in this area, and throughput of 28 MPixel/s (224 Mbit/s). Taking advantage of images nature, its compression can be parallelized through simultaneous multi-cores compressors. For example, using 5 cores, this work is able to compress those images in a rate of 142 MPixel/s (1.1 Gbit/s). All these features make it useful and effective as a current remote sensing imaging system.

Keywords—remote sensing; satellite; image; lossless data compression; FPGA; prediction; Golomb-Rice coding; IP-core

I. INTRODUCTION

Remote sensing makes use of satellites to expand human beings' sensory perception, both by capturing images, which provide a panoramic spatial view of the Earth, and by obtaining data from part of the electromagnetic spectrum invisible to the human eye. Such satellites have evolved to produce images of even higher spatial, radiometric and spectral resolutions [1]. Thus, space systems have to deal with a large amount of data, which can cause problems, especially in terms of storage and transmission [2]. The increasing of storage influences the satellite size, weight and power consumption, to name a few.

The availability of bandwidth to perform transmission between space systems and Earth is determined by combining some factors, such as antenna size, transmitter power and the number of allocated ground stations. Due to the increasing number of satellites in space, the bandwidth has become a scarce resource. The most straightforward solution to avoid the communication bottlenecks is data compression [3]. Then, onboard compression techniques have become crucial for space missions, mainly remote sensing ones due to the large amount of generated data. In addition, some missions require real time data transmission to Earth, demanding compressors to have very high throughput rates.

In this context, space mission requirements guide the choice among several data compression solutions, mainly those based on transform (for example, DCT and DWT), prediction and BTC (Block Truncation Coding) [2]. Each one holds its particular advantages and disadvantages.

This paper proposes a particular solution in hardware of lossless compression for satellite images suitable for this kind of application that was implemented on FPGA in order to validate and assess the technique but could be easily ported to ASIC technologies. The compression method presented uses a low complexity approach based on prediction and Golomb-Rice coding.

Section 2 introduces the theoretical background for this work. Based on a high-level software reference model, Section 3 analyzes the compression performance of proposed method. In order to accomplish the hardware implementation, Section 4 shows the compressor architecture, providing its details. In Section 5, simulation methodology is explained. Using the basis established by previous sections, an FPGA

prototyping scheme is detailed in Section 6. The achieved results and their comparisons to related works are exposed in Section 7. Finally, Section 8 presents conclusions and future works.

II. COMPRESSION MODEL AND CONCEPTS

Prediction-based techniques are well known for their effectiveness in achieving lossless data compression, which is a requirement for some space missions [2]. One way to explain this set of techniques comes from a specific generic compression model [4], as shown in Fig. 1, which was adopted in this work. Initially, the original image x_o is processed by a decorrelation stage, which objective is to reduce the inter-pixel correlation. The coding stage tries to code the decorrelated image x_d using a reduced number of bits. The result of the coding is the binary sequence x_c .

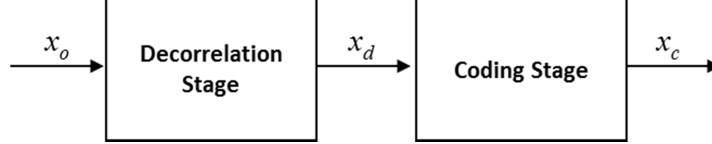


Figure 1. Generic Compression Model.

Entropy coders depend on a statistical model describing the sequence to be coded. When pixels are strongly correlated, high compression rates can be achieved if the coder is based on a contextual model that appropriately captures the correlations.

A. Prediction

The decorrelation stage used in the proposed compression is based on prediction because it is a common simple manner to guess a pixel level from a group of neighbor pixels. Prediction techniques define an estimate $\widetilde{x}_{i,j}$ for a given pixel $x_{i,j}$ of the image (on row i and column j) using past pixels, for instance $x_{i-1,j}$, $x_{i,j-1}$ and $x_{i-1,j-1}$. The pixel $x_{i,j}$ is substituted by the prediction error, $e_{i,j} = x_{i,j} - \widetilde{x}_{i,j}$. The bidimensional prediction adopted in this work uses an edge detection approach as proposed in the JPEG-LS standard [5] and is given by:

$$\widetilde{x}_{i,j} = \begin{cases} \min(x_{i,j-1}, x_{i-1,j}) & \text{if } x_{i-1,j-1} \geq \max(x_{i,j-1}, x_{i-1,j}), \\ \max(x_{i,j-1}, x_{i-1,j}) & \text{if } x_{i-1,j-1} \leq \min(x_{i,j-1}, x_{i-1,j}), \\ x_{i,j-1} + x_{i-1,j} - x_{i-1,j-1} & \text{otherwise.} \end{cases} \quad (1)$$

B. Golomb-Rice Coding

Golomb-Rice coding (also sometimes called only Rice coding) provides an easy and operational way to code any non-negative integer n [6]. Given a non-negative integer parameter k , the Golomb-Rice code for n is the unary code for n shifted k bits to the right, concatenated with the k less significant bits of n . For one-sided geometric distributions (OSGD), Golomb-Rice codes normally achieve nearly optimal compression bit rates [7].

It is generally accepted that prediction errors are well modeled by Laplacian distributions or, in the case of integer values, by two-sided geometric distributions (TSGD) [7]. A TSGD can be transformed in an approximation to an OSGD by the following mapping of values [6]:

$$M(v) = \begin{cases} 2v & \text{if } v \geq 0, \\ 2|v| - 1 & \text{if } v < 0. \end{cases} \quad (2)$$

C. Calculation of the Parameter k

Rice coding concept suggests that a variable and adaptive value of k gets better results than if it is applied in a static manner. Given a sequence whose elements follow a Laplacian distribution, the optimum Golomb-Rice parameter k to encode the sequence obtained after the mapping in (2) can be estimated as $k = \lceil \log_2 a' \rceil$, where a' is the average of the absolute values of the sequence before mapping [6]. In [4], parameter k is incrementally calculated as shown in (3) [6].

$$n \cdot 2^k \geq a, \quad (3)$$

where n is the number of errors seen so far, and a is the sum of the absolute values of these errors. Then, it avoids logarithmic calculation and division $a' = \frac{a}{n}$.

It is possible to avoid n and a overflow by changing it to the following equation. This one is a different approach using a'' instead of a' but provides similar results.

$$a'' = \frac{\sum_{i=1}^n 2^{i-1} |v_i|}{2^n} \quad (4)$$

In this case, a'' will be always in a reasonable fixed range.

III. COMPRESSION PERFORMANCE

A software reference model was developed (both compressor and decompressor) as a high-level step before the hardware implementation and to evaluate compression performance. It was selected a set of test images from CBERS-2B (China-Brazil Environment Resources Satellite) CCD (Charge Coupled Device) camera, as used in [1]. Some examples are shown in Fig. 2.

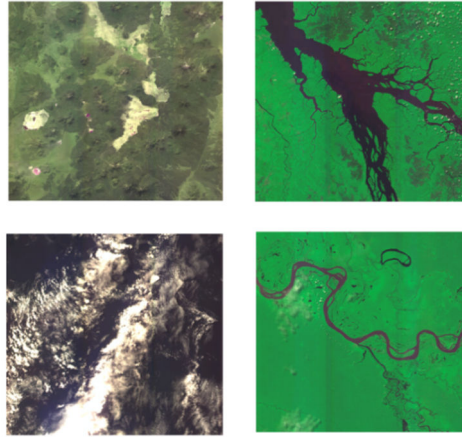


Figure 2. Some images from the test set.

Each of the five available bands (panchromatic, blue, green, red and near infrared) is compressed as a separate grayscale image, accessible as raw data, radiometric resolution of 8 bits and size of 5812 x 5812 pixels (~32MPixels). Twenty different images were chosen to evaluate the compression performance. Because each image has five different bands, it totalizes 100 testing units. The average, worst and best cases of compression ratio (CR) in each band are listed in Table 1. CR is defined as the ratio between the number of bits in the original image and the number of bits used to represent the compressed one.

TABLE I. COMPRESSION EVALUATION RESULTS.

Band	Average CR	Worst CR	Best CR
Panchromatic	3.54	1.97	4.94
Blue	3.38	1.96	4.34
Green	3.63	1.96	5.03
Red	3.47	1.89	4.94
Near Infrared	2.96	1.89	5.24

When uncompressed imaging data is transmitted over a telecommunications channel, the effect of a single bit error is constrained to the particular pixel in which it occurs. On the other hand, the effect of a single bit error on compressed data will naturally propagate over many pixels during the reconstruction process [8]. Since space communication can be very error-prone, this issue represents a serious problem about onboard data compression.

To overcome it, this work has used a straightforward solution similar to that applied in [1]: images are decomposed in several smaller frames. So, each frame is compressed separately as an independent image. Some frame sizes were tested to assess their influence in compression performance using the same image set. Fig. 3 shows selected average ratios obtained from this assessment.

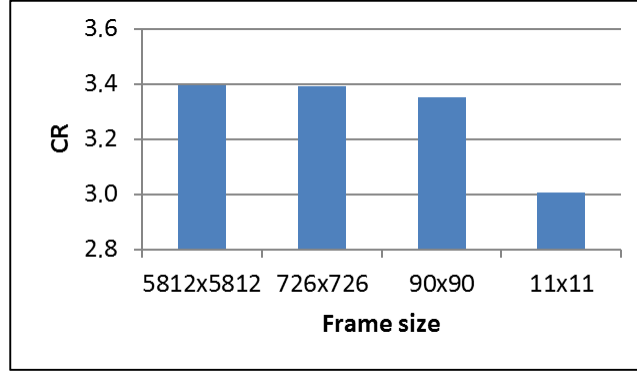


Figure 3. Average compression ratio by frame size comparison.

To illustrate the effectiveness of this technique, considering for example the case in which compression is done using frames of 90x90 (8100 pixels), the results show that average CR decreases around 0.04 while provides a strong resistance to error propagation since one communication error will lead to corruption of a small frame, which corresponds only to 0.02% of the original size.

IV. COMPRESSOR ARCHITECTURE AND IMPLEMENTATION

The compressor is composed of two main blocks: Predictor and Coder, as shown in Fig. 4a. An 8 bits pixel is read by the Predictor block, which computes the prediction error. It needs a memory to store the past 5813 pixels (considering the given images' width of 5812 pixels). The Coder block performs the Golomb-Rice coding and generates the codeword, bit by bit, on the output.

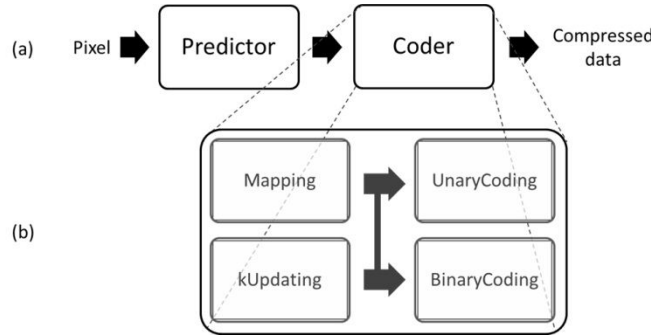


Figure 4. Compressor architecture.

As seen in Fig. 4b, the following blocks compose the Coder: Mapping, kUpdating, UnaryCoding and BinaryCoding. Some functional and implementations details are listed below.

- The Mapping block maps the prediction error according to (2). Positive error values are mapped into the set of even numbers, whereas negative ones are mapped into the set of odd numbers;
- The kUpdating block incrementally calculates the parameter k value according to (4), so it can assume values in the range between 0 and 8;
- The UnaryCoding block outputs serially the unary code;
- The BinaryCoding block outputs its bitstream once the UnaryCoding has finished its coding.

The compressor was described in RTL level using SystemVerilog and its hierarchical architecture was divided in blocks as shown in Fig. 4. This description was compiled and synthesized using Quartus II v.10.1 software from Altera.

This architecture can be employed in a multicore approach as a way to achieve better performance. It means a division of the image among a number of compressor cores, so each one is responsible to compress a different part of the image at the same time, which represents an increase proportional to the number of cores used.

Implementation depends on a signal protocol to establish data communication between blocks. It was used a handshake based on three main signals: ready, valid and last. 'Ready' is set by the receiver block to warn the sender that it is ready to receive data while 'valid' is set by the sender to mark the current data as valid to be read. 'Last' is used together to 'valid' by the sender to feature the current data as the last part in a certain transfer. Then, in a system clock's rising edge, if the pair 'ready' and 'valid' are high, it is the exact moment of data transfer.

Thus, in a more detailed view, Fig. 5 illustrates a graphical representation of compressor's internal blocks interaction and signalization.

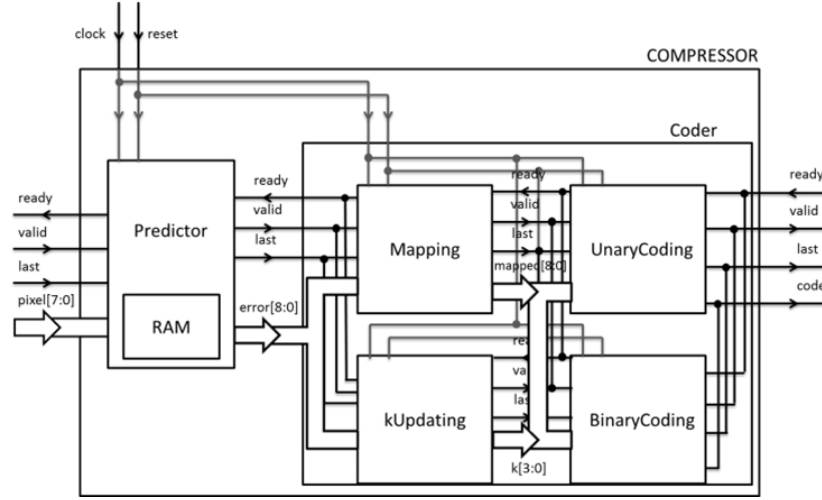


Figure 5. Compressor's block view.

V. SIMULATION METHODOLOGY

In order to validate the compression functionality, a set of simulations was performed using QuestaSim v.10.0 software from Mentor Graphics following BVM (Brazil-IP Verification Methodology) based on [9]. The simulations applied pixel stimulus into the compressor, at the same time to high level and RTL models (known as Reference Model and DUV, respectively), from which results were compared and monitored. This comparison helps in the process of finding functional errors, whereas the monitoring detects if certain expected conditions are reached.

In BVM, the testbenches are implemented in transaction level while the DUV (Design Under Verification) is implemented in signal level. A testbench is outlined in Fig. 6.

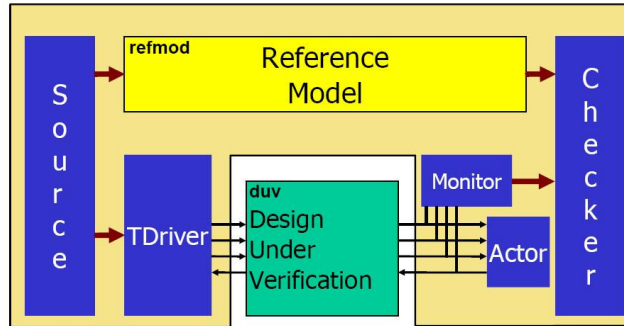


Figure 6. BVM's testbench overview.

Source entity is responsible for sending the same input transaction to the *TDriver* and *Reference Model*, which processes it and provides its corresponding output transaction. Meanwhile, *TDriver* translates the transaction into signals to lead the *DUV* to generate output signals by the interaction with *Actor*. These signals are observed by *Monitor* in order to produce new equivalent transactions, which go to the *Checker* together with *Reference Model*'s output transactions so they can be compared.

Transactions that flow from *Source* are generated according to a probability distribution. Both transactions coming from the *Source* as those that go to the *Checker* are used to decide the simulation's stopping time. This decision is based on coverage metrics, mainly functional coverage.

VI. FPGA PROTOTYPING

The Altera's Nios II Development Kit - Stratix Edition, was employed to set the FPGA prototyping scenario, enabling one more level of validation. It contains an EP1S10F780C6 device, featuring 10,570 logic elements and 920,448 RAM bits. Additionally, through this prototype, it was possible to measure the performance rate of the implemented compressor.

From a host computer using an Ethernet interface, it receives imaging data, which are compressed using the hardware compressor synthesized on FPGA considering its system clock of 50 MHz.

Compression result is sent back to the computer via the same communication channel to be decompressed by software. The reconstructed image has to be identical to the original one to validate the compressor. Nios II processor is applied to intermediate all this process and to measure compression time by making use of its timers. An overview of such prototyping environment is shown in Fig. 7.

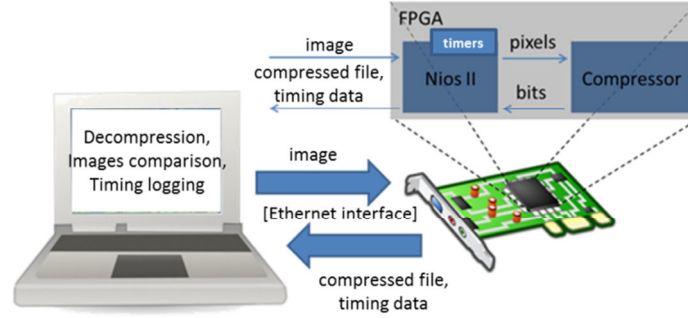


Figure 7. FPGA prototype scheme.

A client-server approach was implemented using pieces of software in the host and Nios II sides to establish a TCP connection. The compressor synthesized in the FPGA communicates to the Nios II processor through its internal bus to receive pixels and send back their compressed stream. Every time an image starts to be compressed, a hardwired clock cycles counter is triggered to increment until this compression process is over.

The sum of clock cycles for all these images ($\sim 3.3 \times 10^9$ pixels) is the measurement used to calculate and assess system's performance since the clock period/frequency is known, allowing the generalization of (5). Each value was sent to the host computer, where they were stored for further manipulation.

$$T = \frac{3.3 \cdot 10^9}{c \cdot p} \text{ or } T = 3.3 \cdot 10^9 \frac{F}{c} \quad (5)$$

T means throughput, C is clock cycles, p and F come from period and frequency.

Similar process was utilized to manage a multicore approach taking advantage of up to five simultaneous cores so it could measure its throughput rate in contrast to the first method.

VII. RESULTS

Altera's FPGA devices Stratix II EP2S60F484C3, Stratix EP1S10F780C6 and Cyclone III EP3C25E144C7 were employed in order to assess general performance. As seen before, implementation details show that the compressor requires 46,504 memory bits. Considering just one compressor core, Table 2 lists achieved synthesis data regarding area and memory compared to some related works.

TABLE II. HARDWARE RESULTS AND COMPARISON.

Method	Area	Memory (bits)	Technology
Proposed	588 (< 1%) logic elements	46,504 (2%)	Altera Stratix II EP2S60F484C3
	574 (5%) logic elements	46,504 (5%)	Altera Stratix EP1S10F780C6
	564 (2%) logic elements	46,504 (8%)	Altera Cyclone III EP3C25E144C7
[1]	2,375 (10%) logic elements	13 x 9k (20%)	Altera Cyclone III EP3C25
[10]	8,867 (86%) logic elements	18 x 9k (40%)	Altera Cyclone III EP3C10-6
	58,137 eq. gates	102,076	ASIC UMC 0.18 μ m
	51,506 eq. gates	102,076	ASIC TSMC 0.09 μ m

[11]	2,650 (24%) slices	25 x 18k (44%)	Xilinx Virtex II XC2V2000FG676
[12]	1,928 (13%) slices	63,078 (2%)	Xilinx Virtex-4 SX35

Since it is hard to establish a fair assessment between all these works, the comparison should be made carefully. Some variables, like algorithm, images employed in tests and technology chosen, can affect those results. However, it is clear that the current work uses a tiny amount of resources in an FPGA when compared to related works. For example, arithmetic coding used in [12], as a more complex algorithm, is likely to use larger and slower circuits. Besides, they can be compared with respect to compression rates in a trade-off between compression performance and hardware resources needed.

Proposed method's low resources feature supports the application of a multicore approach. Obviously, logic elements tend to grow proportionally to the number of cores placed in the three devices of synthesis, as illustrated in the graph of the Fig. 8.

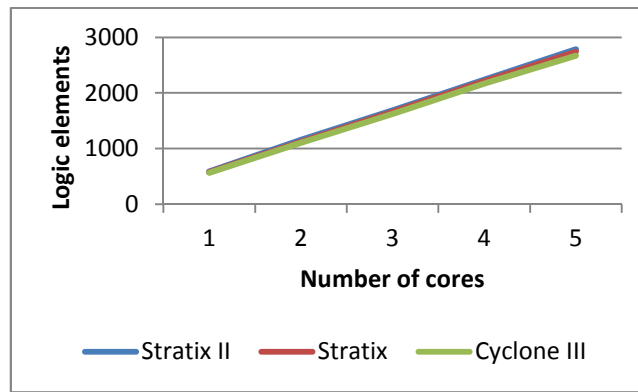


Figure 8. Variation of logic elements by increasing the number of compressor cores.

On the other hand, the amount of memory remains virtually unaffected because it is distributed among the cores once the image is shared by them, as well as the maximum frequency of operation, according to the synthesis reports, that is almost the same value in all the five cases.

The background defined by the proposed architecture, simulation methodology and prototyping environment allows this work to present its results of timing performance, which is an important step in the project of a space application compressor. First, it is presented the clock cycles counted by the prototype for the image set in the Fig. 9 applying up to five compressor cores in parallel.

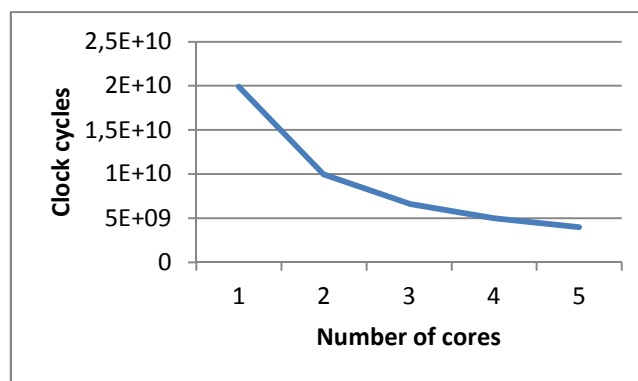


Figure 9. Clock cycles variation due to the application of multicore compression.

Timing measurements were concentrated in average values of throughput rate, calculated according to (5), and based on maximum frequencies achieved in the synthesis step for each specific device and number of cores. These values are shown in Fig. 10.

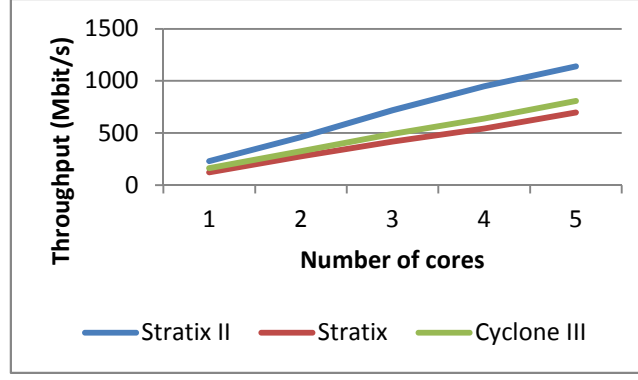


Figure 10. Throughput variation in function of number of cores.

Similarly, it is possible to list and compare these results to related works in the Table 3 following the same order of the Table 2.

TABLE III. FREQUENCY AND THROUGHPUT COMPARISON.

Method		Frequency (MHz)	Throughput (Mpixel/s)
Proposed (Stratix II)	1 core	172.65	28.62
	5 cores	171.38	142.21
Proposed (Stratix)	1 core	93.06	15.43
	5 cores	104.95	87.09
Proposed (Cyclone III)	1 core	122.41	20.29
	5 cores	121.52	100.84
[1]		50	12
[10]		100	100
		360	360
		700	700
[11]		75	75
[12]		123	15.3

As expected, core replication suggests improvements in throughput rates. Nevertheless, even only one core working in the maximum frequency of operation can reach a consistent performance. Again, increasing of cores means proportional rise of throughput as long as keeps area and memory features in an acceptable level, which points a way to meet performance requirements in the space application context.

VIII. CONCLUSION AND FUTURE WORKS

The FPGA design of a satellite image lossless compressor based on prediction and Golomb-Rice coding has been described. Its high level results indicate that the compression model performs appropriately reaching ratios compatible to other methods well known in the satellite imaging area, JPEG-LS for example. Furthermore, the compressor can handle the propagation error issue by dividing images in small frames without heavily affecting compression ratios.

As stated in Table 2, the synthesized circuit is definitely very simple. It reveals a real low complexity proposal, which could represent several benefits, including ease of implementation and optimization. Moreover, a lot of remaining room on FPGA foments future research to find possibilities of improvements

in compression, performance, area, memory, fault-tolerance, error propagation and so on, not to mention the multicore approach that was not totally explored in this work. Currently, it is not considering constraints like I/O, memory bandwidth limitations and cores' outputs synchronization.

Altogether, the discussed points confirm the main conclusion that its relative efficiency in terms of compression ratio, complexity and memory features is very suitable to hardware implementation, mainly considering satellite specific strong requirements. If this IP-Core solution is implemented in any ASIC technology, its performance will increase even more compared to results got on FPGA while its low complexity supports low area and low power design.

Consider, for example, the case of satellite CBERS-3 [13]. The current achieved performance is enough to work in real time with each MUXCAM band, for instance. It requires a performance of 68 Mbit/s per spectral band. Thus, national space missions could take advantage of an effective Brazilian data compression technology instead of importing expensive and proprietary foreign solutions.

ACKNOWLEDGMENT

Our thanks to *Agência Espacial Brasileira* (AEB, Brazil) - *Uniespaço* program - for supporting this work and *Instituto Nacional de Pesquisas Espaciais* (INPE, Brazil) for providing the image set and technical exchange largely used in this research.

REFERENCES

- [1] A. Lopes Filho and R. d'Amore, "A low complexity image compression solution for onboard space applications," in *Proc. 23rd Symp. on Integrated Circuits and System Design*, São Paulo, Brazil, 2010, pp. 174-179.
- [2] G. Yu., T. Vladimirova, and M. N. Sweeting, "Image compression systems on board satellites," *Acta Astronautica*, vol. 64, issues 9-10, pp. 988-1005, May-June, 2009.
- [3] European Space Agency. (2012, June 29). *Compression Stops Mission from Drowning in Data* [Online]. Available: http://www.esa.int/esaMI/Space_Engineering/SEM74PSRJHG_0.html
- [4] M. M. Meira, J. A. G. Lima and L. V. Batista, "An FPGA implementation of a lossless electrocardiogram compressor based on prediction and Golomb-Rice coding," *Proc. XI Workshop Iberchip*, Salvador, Brazil, 2005.
- [5] *Lossless and Near-Lossless Coding of Continuous Tone Still Images*, ISO-14495/ITU-T.87, 1999.
- [6] G. Seroussi and M. J. Weinberg, "On adaptive strategies for an extended family of Golomb-type codes," in *Proc. Data Compression Conf.*, Snowbird, UT, 1997, pp.131-140.
- [7] M. J. Weinberger, G. Seroussi and G. Sapiro, "LOCO-I: a low complexity, context-based, lossless image compression algorithm," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, 1996, p. 140-149.
- [8] R. F. Rice, "Channel coding and data compression system considerations for efficient, communication of planetary imaging data," Jet Propulsion Laboratory, Pasadena, CA, 33-695, June 1974.
- [9] L. M. L. Silva et al., "An MPEG-4 decoder design verification using functional coverage," in *Proc. Microelectronics Students Forum*, Rio de Janeiro, Brazil, 2007.
- [10] CAST, Inc. (2012, June 29). *JPEG-LS Compression Encoder Core* [Online]. Available: <http://www.cast-inc.com/ip-cores/images/jpegls-e/>
- [11] Z. Wang, T. Zhang; L. Yan and C. Gong , "A high performance fully pipelined architecture for lossless compression of satellite image," in *Proc. Int. Conf. on Multimedia Technology*, Ningbo, China, 2010, pp. 29-31.
- [12] X. Chen, N. Canagarajah, J. L. Nunez-Yanez and R. Vitulli, "Hardware architecture for lossless image compression based on context-based modeling and arithmetic coding," in *Proc. IEEE Int. SOC Conf.*, Newport Beach, CA, 2007, pp. 251-254.
- [13] National Institute for Space Research – Brazil (2012, June 29). China-Brazil Earth Resources Satellite: CBERS 3 and 4 Cameras [Online]. Available: http://www.cbers.inpe.br/ingles/satellites/cameras_cbers3_4.php

Glossário

ASIC: Circuito Integrado de Aplicação Específica (*Application Specific Integrated Circuit*).

Banda: Pequena faixa do espectro eletromagnético.

BVM: Metodologia de Verificação do Brazil-IP (*Brazil-IP Verification Methodology*).

CBERS: Satélite Sino-Brasileiro de Recursos Terrestres (*China-Brazil Earth Resources Satellite*).

CCD: Dispositivo de carga acoplada (*Charged-Coupled Device*) – sensor para captação de imagens.

CCSDS: Comitê Consultivo para Sistemas de Dados Espaciais (*Consultative Committee for Space Data Systems*).

Codificação de Golomb-Rice: Subconjunto dos códigos de Golomb que fixa um parâmetro como potência de dois. Por um lado, torna o esquema de codificação mais simples. Porém, ao mesmo tempo, pode não produzir os melhores resultados.

Codificação Unária: codificação que representa um número natural n por n '1' seguidos por um '0'.

Compressão de dados: Processo de codificação da informação utilizando menos bits que a representação original.

Deteção de bordas: ferramenta do processamento de imagens usada para identificar pontos onde o brilho da imagem possui descontinuidade.

Descorrelação: processo usado para reduzir a correlação/redundância em um sinal enquanto preserva outros aspectos do sinal em questão.

DUV: Projeto em verificação (*Design Under Verification*).

DWT: Transformada Wavelet Discreta (*Discrete Wavelet Transform*).

FPGA: Arranjo de portas programáveis em campo (*Field Programmable Gate Array*).

HD: Disco rígido (*Hard Disk*).

HDL: Linguagem de Descrição de Hardware (*Hardware Description Language*).

HRC: Câmera de alta resolução (*High Resolution Camera*).

ICER: formato de arquivo de compressão de imagem baseado em wavelet usado pelos veículos exploradores de Marte (*Mars Exploration Rovers*).

IEC: Comissão Internacional de Eletrotécnica (*International Electrotechnical Commission*).

Imageador: instrumento/equipamento de imageamento.

Imageamento: criação, modificação e processamento de imagens eletrônicas.

INPE: Instituto Nacional de Pesquisas Espaciais.

ISO: Organização Internacional de Padronização (*International Organization for Standardization*).

JPEG-2000: Padrão ISO/IEC 15444-1 para compressão de imagens a partir de um método baseado em wavelet.

JPEG-LS: Padrão ISO-14495-1/ITU-T.87 para compressão sem perdas e “quase” sem perdas de imagens.

Modelo de Referência: uma implementação correta do DUV em alto nível.

Multicore: múltiplos núcleos ou multinúcleo consiste em colocar dois ou mais núcleos de processamento (*cores*) no interior de um mesmo chip. Eles são responsáveis por dividir as tarefas entre si, permitindo que elas sejam executadas em paralelo, o que normalmente representa ganhos de desempenho.

Nios II: Processador embarcado de 32 bits cuja arquitetura foi especialmente desenvolvida para a família de FPGAs da Altera.

Observação da Terra: conjunto de técnicas envolvidas com o sensoriamento remoto.

Parâmetro k : parâmetro usado para ajustar a codificação de Golomb-Rice.

Pipeline: conjunto de elementos de processamento de dados conectados em série.

Pixel: menor ponto que forma uma imagem digital.

Predição: Forma comum de se reduzir a redundância na representação de um sinal qualquer. Cada amostra do sinal pode ser predito como outra amostra recente que já tenha sido processada (ou uma composição delas). Substitui-se a amostra codificada pelo erro da predição feita.

RAM: Memória de Acesso Aleatório (*Random Access Memory*).

RC: Razão de Compressão.

Resistência a erros: capacidade de um sistema em tolerar falhas que possam ocorrer com os dados que estão sendo manipulados.

Resolução radiométrica: determina quão precisamente um sistema pode representar ou distinguir diferenças de intensidade. Normalmente, é expresso por número de níveis ou de bits. Por exemplo, 8 bits ou 256 níveis, que é uma resolução radiométrica muito comum para imagens digitais.

Sensoriamento remoto: tecnologia de obtenção de imagens e dados da superfície terrestre através da captação e registro da energia refletida/emitida pela superfície sem que haja contato físico entre o sensor e a superfície estudada.

Símbolo: forma básica (tal qual uma letra, um pixel, etc.) de visualizar/organizar as informações para que elas possam ser representadas por dados.

SPIHT: método de compressão de imagens baseado em wavelet cujo nome é uma sigla em inglês para Particionamento de Conjuntos em Árvores Hierárquicas – *Set Partitioning in Hierarchical Trees*.

SystemVerilog: Linguagem combinada para descrição e verificação de hardware baseada em extensões da linguagem Verilog.

Tempo Real: refere-se a sistemas em que o tempo de execução de uma determinada tarefa é rígido. O sistema deve ser implementado visando à execução da tarefa no tempo correto.

Testbench: Ambiente virtual usado para verificar a corretude do DUV através da comparação com o modelo de referência.

Throughput: quantidade de dados processados em um determinado período de tempo.

Verificação Funcional: forma de verificação de hardware digital que estimula o DUV e o modelo de referência a fim de confrontar o funcionamento de ambos.

WFI: Câmera Imageadora de Amplo Campo de Visada (*Wide Field Imager*).