DESENVOLVIMENTO ARQUITETURAL PARA A COMPENSAÇÃO DE MOVIMENTO DO PADRÃO H.264/AVC

Fabiane Rediess[1], *José Luís Güntzel*[1,2], *Sergio Bampi*[2], *Luciano Agostini*[1,2]

[1]Grupo de Arquiteturas e Circuitos Integrados (GACI)
Universidade Federal de Pelotas (UFPel)
[2]Grupo de Microeletrônica (GME)
Universidade Federal do Rio Grande do Sul (UFRGS)

{frediess_ifm, guntzel, agostini}@ufpel.edu.br, {guntzel, bampi, agostini}@inf.ufrgs.br

RESUMO

Este artigo apresenta o desenvolvimento, simulação e prototipação de uma arquitetura para a compensação de movimento do padrão H.264/AVC de compressão de vídeo. Esta arquitetura foi descrita em VHDL e sintetizada para FPGAs das famílias Flex10K, Cyclone II, Stratix e Stratix II da Altera. A arquitetura desenvolvida foi prototipada em um dispositivo da família Cyclone II. O núcleo da arquitetura sintetizada para o dispositio Cyclone II é capaz de processar um novo bloco a cada ciclo de relógio, resultando em mais de 5.700 quadros SDTV por segundo. Considerando o atraso de memória, a arquitetura desenvolvida é capaz de processar entre 24 e 60 quadros SDTV (720x480 pixels) por segundo.

1. INTRODUÇÃO

O contínuo aumento no número de aplicações que manipulam vídeos digitais, como celulares, TV digital, etc., faz com que a compressão de vídeo seja uma técnica chave para permitir a disseminação destas aplicações.

Há vários anos a indústria vem propondo padrões para realizar a compressão de vídeo. Entre eles, o MPEG-2 [1] e, mais recentemente, o H.264/AVC [2][3].

Este artigo apresenta o desenvolvimento de uma arquitetura para realizar a compensação de movimento focando no perfil baseline do padrão H.264/AVC e em vídeos SDTV (resolução de 720x480 pixels).

2. O PADRÃO H.264/AVC

O H.264/AVC [2] [3] é o mais recente padrão de compressão de vídeo proposto pelo VCEG do ITU-T e pelo MPEG do ISO/IEC. Esse padrão atinge uma taxa de compressão duas vezes maior que os padrões anteriores [6]. Como era esperado, o aumento na eficiência e na flexibilidade da codificação acabou levando a um acréscimo na complexidade computacional em comparação com os padrões anteriores.

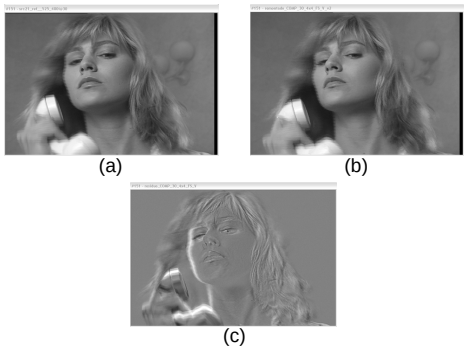

Fig. 1. Exemplo de MC (a) quadro de referência, (b) quadro remontado e (c) quadro residual.

Os principais módulos do codificador H.264/AVC são: transformadas diretas e inversas, quantização direta e inversa, filtro de deblocagem, codificador de entropia, predição intra quadro, estimação de movimento e compensação de movimento, que é o foco deste trabalho. O decodificador do H.264/AVC é composto por um sub-conjunto dos módulos do codificador, são eles: compensação de movimento, predição intra quadro, transformadas inversas, quantização inversa, filtro e decodificador de entropia.

3. COMPENSAÇÃO DE MOVIMENTO

O processo da compensação de movimento (MC) precisa estar adaptado aos parâmetros da estimação de movimento (ME). A ME encontra o melhor casamento para cada bloco do quadro atual entre os blocos do quadro de referência e, então, gera os vetores de movimento, indicando a posição deste bloco no quadro de referência.

Com a informação dos vetores de movimento, a MC encontra os blocos no quadro de referência e remonta o quadro atual usando apenas os blocos do quadro de referência. O quadro remontado é subtraído do quadro atual para gerar o quadro residual que será submetido aos próximos passos da codificação do vídeo (transformadas e quantização). A Fig. 1 apresenta um exemplo de

quadro de referência (a), de quadro remontado (b) e de quadro de resíduo (c).

O padrão H.264/AVC inseriu diversas novas ferramentas de codificação, para aumentar a eficiência deste processo. A maioria dessas novas ferramentas encontra-se no processo de predição inter quadro e, deste modo, a MC deve ser capaz de utilizar estas ferramentas. As principais inovações são o uso de blocos de tamanho variável (16x16, 16x8, 8x16, 8x8, 8x4, 4x8 ou 4x4), múltiplos quadros de referência, vetores de movimento apontado para fora dos limites do quadro, precisão de ¼ de pixel, etc. A arquitetura desenvolvida neste trabalho suporta todos os tamanhos de blocos permitidos pelo padrão. Outras ferramentas do MC não foram implementadas e serão desenvolvidas em trabalhos futuros.

4. ARQUITETURA DESENVOLVIDA

O principal objetivo deste trabalho é atingir uma solução em hardware com elevado *throughput* para suportar vídeos de alta resolução.

Na Fig. 2 é mostrado o diagrama de blocos da arquitetura desenvolvida, incluindo os detalhes de prototipação. Essa arquitetura foi prototipada em uma placa Altera DE2 [5], que tem um FPGA EP2C35F672C6 da família Cyclone II [6]. O núcleo da Compensação de Movimento foi mapeado para o FPGA disponível e esse núcleo usa duas memórias externas, uma SRAM e uma SDRAM. Para acessar a memória SDRAM, o núcleo do MC precisa de um controlador de SDRAM. A memória SDRAM é usada para armazenar o quadro de referência e o quadro remontado, enquanto que a memória SRAM é usada para armazenar as informações dos vetores de movimento.

A solução apresentada na Fig. 2 é a primeira solução para acessar os dados do quadro de referência e para armazenar os resultados do quadro remontado. A memória externa está sendo acessada diretamente pelo núcleo da compensação de movimento. Uma segunda alternativa, usando memória cache para reduzir o número de acessos a essas memórias, está sendo investigada. Essa memória cache será desenvolvida usando memória interna do FPGA.

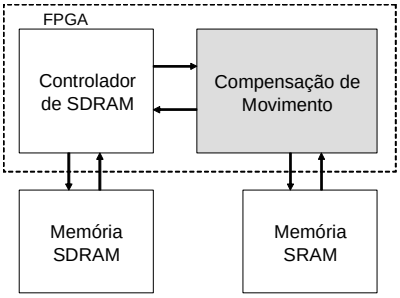

Fig. 2. Diagrama de blocos da arquitetura desenvolvida.

A SDRAM presente na placa DE2 está organizada em um milhão de palavras de 16 bits, que estão divididas em quatro bancos de memória. O acesso à SDRAM é feito pelo controlador de SDRAM, que introduz uma latência significante ao processo. Os comprimentos de acesso em rajada permitidos por este controlador são de 1, 2, 4 e 8 palavras. Neste trabalho, foi utilizado um acesso em rajada com 8 palavras. Após a latência do controlador, oito palavras de memória válidas são entregues na saída, uma a cada ciclo de relógio. Para processar completamente um macrobloco (16x16 pixels), a arquitetura desenvolvida precisa entre 48 e 192 leituras em rajada. Este valor depende dos valores dos vetores de movimento gerados pela estimação de movimento para este macrobloco. Cada requisição de leitura, após o controlador da SDRAM ativar o sinal de *acknowledge*, utiliza 5 ciclos de relógio para entregar o primeiro dado válido na entrada no núcleo do MC. Nenhuma outra operação pode ser solicitada para a memória até que o último dado apareça na sua saída. Então, para cada requisição de leitura de 8 palavras, são necessários 12 ciclos de relógio.

Esta memória SDRAM pode ser configurada com uma latência de 2 ciclos de relógio, se a freqüência de operação for de 100MHz, ou com uma latência de 3 ciclos de relógio, se a freqüência de operação for de 133MHz.

A organização do quadro da imagem na memória SDRAM começa com todas as informações de luminância (Y), então as informações de crominância são armazenadas, primeiramente a crominância azul (Cb) e, finalmente, a crominância vermelha (Cr). A relação das componentes de cor usadas neste trabalho é 4:2:0, que é definida no perfil *baseline* do padrão H.264/AVC[2].

A memória SRAM é organizada em 256 mil palavras de 16 bits. Esta memória não introduz latência adicional para ser acessada, pois o acesso pode ser realizado em paralelo com as outras operações. O acesso à SRAM é mais simples do que o acesso à SDRAM e este controle é realizado diretamente pelo controle do núcleo do MC.

A informação dos vetores de movimento é composta pela informação de tipo de bloco, ou seja, o tamanho de bloco selecionado pela estimação de movimento, e as componentes vertical e horizontal do vetor de movimento, que indicam a posição do melhor casamento encontrado para este bloco no quadro de referência. Esta informação também é gerada pela estimação de movimento.

O bloco operativo da compensação de movimento foi desenvolvido em *pipeline* e, após a latência inicial, um novo bloco pode ser compensado a cada ciclo de relógio. O *pipeline* é composto de 17 estágios, cada um destes estágios é responsável pelo cálculo de uma parte dos dados necessários à formação dos endereços de leitura e escrita, bem como aos deslocamentos ocasionados pelos vetores de movimento. No entanto, a arquitetura prototipada precisa aguardar pelas operações de leitura e escrita na memória SDRAM. Desta forma, o *pipeline* precisa ser parado para permitir o processamento correto das informações.

A Fig. 3 mostra o núcleo da compensação de movimento. O núcleo do MC é formado pelo seu bloco

operativo, seu bloco de controle, pelo bloco chamado próximo bloco e por um multiplexador que seleciona o endereço que será usado no acesso ao controlador da SDRAM.

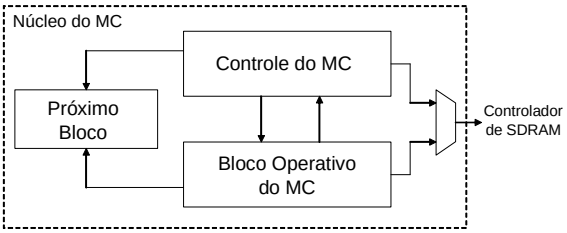

Fig. 3. Núcleo do Compensador de Movimento.

O multiplexador da Fig. 3 é necessário, pois, durante a fase de inicialização da memória, o endereço deve conter informações de configuração, as quais devem vir do controle do MC. Para as outras situações, o endereço precisa vir diretamente do bloco operativo do MC, porque neste momento, o endereço contém o endereço da leitura que deve ser requisitada.

O bloco operativo foi descrito em VHDL estrutural. Existem quatro estágios principais neste bloco operativo: cálculo do endereço inicial, cálculo do deslocamento, cálculo dos endereços secundários e reconstrução do bloco. O endereço inicial e o deslocamento são calculados em paralelo. Após este passo, existe um estágio que calcula os endereços secundários, que serão necessários para as operações do MC. Mas, dependendo dos vetores de movimento, pode não ser necessário passar por todos os estágios e apenas 4 estados podem ser suficientes.

O menor tamanho de bloco permitido pelo padrão H.264/AVC é o 4x4 [2], conseqüentemente, todos os outros blocos podem ser construídos a partir de combinações de blocos 4x4. Mas, neste caso, uma informação extra de controle será necessária para indicar qual bloco será o próximo bloco que deve ser processado. Esta informação é definida pelo módulo chamado próximo bloco, mostrado na Fig. 3. O controle do MC ativa um *flag* indicando que uma informação de próximo bloco será necessária. Este módulo foi desenvolvido como uma máquina de estados finitos (FSM) com 7 estados e pode produzir uma nova informação a cada ciclo de relógio.

O controle do MC foi também desenvolvido como uma máquina de estados finitos. Na Fig. 4 encontra-se uma FSM simplificada do controle do MC. Esta FSM usa apenas seis macro-estados. Cada macro-estado desta FSM simplificada é dividido em outros estados e a FSM completa usou 33 estados. O Estado0 na Fig. 4 indica os estados de inicialização da memória que são 9 no total. Primeiramente, é necessário aguardar por 200µs com um sinal de relógio estável ligado na memória SDRAM. Após este tempo, é necessário inicializar os registradores de configuração. O Estado1 na Fig. 4 é composto por 6 estados que controlam o bloco operativo para calcular o endereço inicial, o deslocamento causado pelos vetores de movimento e o primeiro endereço de leitura. O Estado2 na Fig. 4 é formado por 11 estados que calculam todos os outros endereços que podem ser solicitados pela arquitetura do MC, para acessos às informações de luminância e crominância. É importante notar que apenas no pior caso todos estes endereços serão necessários, no melhor caso apenas 4 destes estados serão suficientes. O Estado3, o Estado4 e o Estado5 são estados usados para requisições de leitura, manipulação dos dados lidos e solicitação de escrita para informações de luminância, crominância azul e crominância vermelha, respectivamente.

Quando o primeiro endereço de luminância é conhecido, os endereços de crominância podem ser definidos em função deste.

Cada bloco é remontado usando de uma até quatro leituras de memória. Existem 9 estados na FSM que são usados para cada requisição de leitura, desde o momento da requisição, até que o último dado apareça na saída da memória. Para operações de escrita, dois estados na FSM são usados, um para aguardar o momento exato quando a memória estará apta a receber o dado para escrita e o outro para entregar os demais dados da escrita em rajada na entrada da memória.

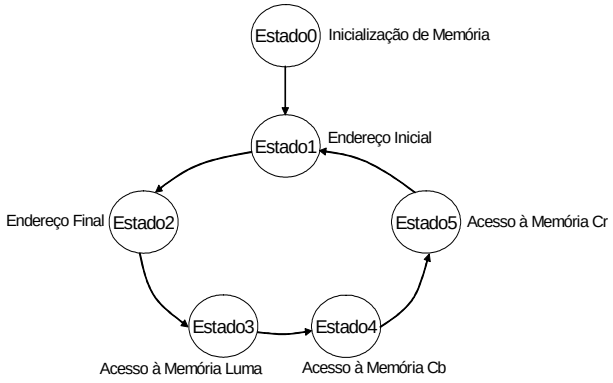

Fig. 4. Máquina de estados finitos simplificada para o controle da arquitetura.

5. RESULTADOS DE SÍNTESE

A Tab. 1 apresenta os resultados de síntese da arquitetura do MC, considerando diferentes famílias de FPGAs da Altera. Os FPGAs alvos de cada família foram: Flex10K EPF10K70RC240-2, Cyclone II EP2C35F672C6, Stratix EP1S10F484C5 e Stratix II EP2S15F484C3.

Tab. 1. Resultados de síntese para o núcleo do MC.

	Freqüência (MHz)	LEs/ALUTs	Registradores
Flex10K	17,7	3.203 (86%)	-
Cyclone II	124,7	2.146 (6%)	895
Stratix	123,1	2.148 (20%)	-
Stratix II	177,9	1.860 (15%)	900

A menor freqüência de operação foi encontrada no dispositivo Flex10K (17,7 MHz), enquanto que a maior freqüência de operação foi obtida com o dispositivo Stratix II (177,9MHz).

Considerando a arquitetura do núcleo do MC junto com a memória externa, quando o *pipeline* está cheio, esta solução pode remontar um novo bloco usando entre 96 (melhor caso) e 240 (pior caso) ciclos de relógio, considerando a latência de memória. Mas é importante considerar que, para operação a ser realizada na memória, o controlador de SDRAM precisa ativar o sinal de *acknowledge*. O número de ciclos até que o controlador ative este sinal é de aproximadamente 4 e este valor é incluído nos valores apresentados anteriormente.

As arquiteturas sintetizadas para as famílias Cyclone II, Stratix e Stratix II utilizaram 491 bits de memória e para Cyclone II foram usados também 3 multiplicadores internos ao FPGA. Estes dados não estão apresentados na Tab. 1.

A Tab. 1 não inclui os resultados de síntese para o controlador de SDRAM. Com a síntese direcionada para o dispositivo Cyclone II, o controlador de SDRAM usou 235 elementos lógicos, 215 registradores e 1 PLL.

Com os resultados apresentados na Tab. 1, é possível estimar o tempo usado para cada quadro e quantos quadros por segundo a arquitetura é capaz de processar. Estes resultados são apresentados na Tab. 2. A coluna "Prototipação" na Tab. 2 mostram o melhor e o pior caso, considerando as latências de memória externa (que são variáveis). A coluna "Núcleo do MC" mostra as taxas de processamento obtidas pelo núcleo da arquitetura. Uma simples comparação entre esses dados é suficiente para se perceber o tamanho do impacto que o uso de memória externa causa nesta arquitetura. O núcleo do MC é capaz de processar entre 819 e 8.236 quadros SDTV por segundo, enquanto que a arquitetura prototipada é capaz de processar entre 8 e 85 quadros por segundo, considerando o melhor caso. É importante notar que, mesmo com o impacto do uso de memória, a síntese para os dispositivos Cyclone II, Stratix e Stratix II são todos capazes de atingir processamento em tempo real para quadros SDTV, mesmo considerando o pior caso.

6. VALIDAÇÃO E PROTOTIPAÇÃO

O processo de validação começou pela implementação em software da compensação de movimento. O MC foi desenvolvido em linguagem C e, com este software, foi possível gerar um quadro remontado para comparar com os resultados gerados pela arquitetura. Uma arquitetura auxiliar foi também descrita em VHDL para realizar a comparação dos quadros remontados por software e por hardware. A comparação foi realizada com sucesso e a arquitetura foi considerada validada.

Após a validação por simulação da arquitetura desenvolvida, a arquitetura foi prototipada na placa DE2 da Altera e foi mapeada para o dispositivo EP2C35F672C6 da família Cyclone II. A resolução escolhida para o processo de prototipação foi a SDTV (720x480 pixels).

Tab. 2. Quadros SDTV remontados por segundo.

	Prototipação		Núcleo do MC
	Pior Caso	Melhor Caso	
Flex10K	3,14	8,54	819,44
Cyclone II	24,05	60,13	5.773,15
Stratix	23,74	59,37	5.699,07
Stratix II	34,32	85,79	8.236,11

Um *push-bottom* presente na placa de prototipação foi usado para iniciar o processo da compensação de movimento. Após isso, um LED é ligado, indicando que o processo foi finalizado. Então, um outro *push-bottom* é usado para iniciar o processo de comparação em hardware. Finalmente, se a comparação foi realizada com sucesso, um LED será ligado, caso contrário, outro LED será ligado, indiciando a ocorrência de erro na comparação.

7. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresentou o desenvolvimento de uma arquitetura para a compensação de movimento do padrão H.264/AVC e sua prototipação em FPGAs. A arquitetura foi descrita em VHDL e sintetizada para diversos FPGAs da Altera. A arquitetura também foi prototipada usando o dispositivo EP2C35F672C6 da família Cyclone II.

A arquitetura foi desenvolvida usando diretamente o acesso externo à memória e esta decisão gerou uma importante degradação na performance desta solução. Esta foi a primeira solução funcional que foi planejada. Uma segunda alternativa, usando memória cache, está sendo investigada e será implementada nos próximos trabalhos. Outro importante trabalho futuro é a investigação de soluções para incluir, na arquitetura proposta, outras ferramentas de codificação presentes no padrão H.264/AVC.

8. REFERÊNCIAS

[1] ITU-T e ISO/IEC JTC1, "Generic coding of moving pictures and associated audio information – Part 2: Video", ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2), 1994 November.

[2] JVT Editors (T. Wiegand, G. Sullivan, A. Luthra), Draft ITU-T Recommendation and final draft international standard of joint video specification (ITU-T Rec.H.264|ISO/IEC 14496-10 AVC), 2003.

[3] T. Wiegand; G. Sullivan; G. G. J. Bjntegaard; A. Luthra;."Overview of the H.264/AVC video coding standard". IEEE Trans. on Circuits and Systems for Video Tech.,, v. 13, Issue 7, 2003, pp: 560–576.

[4] Wiegand, T.; et all. "Rate-constrained coder control and comparison of video coding standards". IEEE Trans. on Circuits and Systems for Video Tech.,. v. 13, Issue 7, 2003, pp. 688-703.

[5] Altera Corporation, Altera DE2 Board. DE2 Development and Education Board – User Manual. [S.l.], 2006.

[6] Altera Corporation, Altera: The Programmable Solutions Company, <http://www.altera.com>, 2006.