A BASELINE H.264 VIDEO ENCODER HARDWARE DESIGN

by
AYDIN AYSU

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

Sabancı University
Spring 2010

A BASELINE H.264 VIDEO ENCODER HARDWARE DESIGN

APPROVED BY:

Assist. Prof. Dr. İlker Hamzaoğlu …………………………..

(Thesis Supervisor)

Assist. Prof. Dr. Ayhan Bozkurt…………………………..

Assoc. Prof. Dr. Erkay Savaş…………………………..

Assist. Prof. Dr. Hakan Erdoğan…………………………..

Dr. Mustafa Parlak …………………………..

DATE OF APPROVAL: …………………………..

A BASELINE H.264 VIDEO ENCODER HARDWARE DESIGN

Aydın Aysu

EE, Master Thesis, 2010

Thesis Supervisor: Assist. Prof. Dr. İlker Hamzaoğlu

ABSTRACT

The recently developed H.264 / MPEG-4 Part 10 video compression standard achieves better video compression efficiency than previous video compression standards at the expense of increased computational complexity and power consumption. Multiple reference frame (MRF) Motion Estimation (ME) is the most computationally intensive and power consuming part of H.264 video encoders. Therefore, in this thesis, we designed and implemented a reconfigurable baseline H.264 video encoder hardware for real-time portable applications in which the number of reference frames used for MRF ME can be configured based on the application requirements in order to trade-off video coding efficiency and power consumption. The proposed H.264 video encoder hardware is based on an existing low cost H.264 intra frame coder hardware and it includes new reconfigurable MRF ME, mode decision and motion compensation hardware.

We first proposed a low complexity H.264 MRF ME algorithm and a low energy adaptive hardware for its real-time implementation. The proposed MRF ME algorithm reduces the computational complexity of MRF ME by using a dynamically determined number of reference frames for each Macroblock and early termination. The proposed MRF ME hardware architecture is implemented in Verilog HDL and mapped to a Xilinx Spartan 6 FPGA. The FPGA implementation is verified with post place & route simulations. The proposed H.264 MRF ME hardware has 29-72% less energy consumption on this FPGA than an H.264 MRF ME hardware using 5 reference frames for all MBs with a negligible PSNR loss.

We then designed the H.264 video encoder hardware and implemented it in Verilog HDL. The proposed video encoder hardware is mapped to a Xilinx Virtex 6 FPGA and verified with post place & route simulations. The bitstream generated by the proposed video encoder hardware for an input frame is successfully decoded by H.264 Joint Model reference software decoder and the decoded frame is displayed using a YUV Player tool for visual verification. The FPGA implementation of the proposed H.264 video encoder hardware works at 135 MHz, it can code 55 CIF (352x288) frames per second, and its power consumption ranges between 115mW and 235mW depending on the number of reference frames used for MRF ME.

BİR H.264 VIDEO KODLAYICI DONANIM TASARIMI

Aydın Aysu

EE, Yüksek Lisans Tezi, 2010

Tez Danışmanı: Yard. Doç. Dr. İlker Hamzaoğlu

ÖZET

Yakın tarihte geliştirilen H.264 / MPEG4 Part 10 video sıkıştırma standardının sıkıştırma verimi daha önceki standartlara gore daha iyidir. Hareket Tahmini (HT) video sıkıştırma sistemlerinin en çok işlem yapılan ve en çok güç harcanan kısmıdır. H.264 standardında kullanılan çoklu referans çerçevesi (ÇRÇ) kullanan HT video kodlama kalitesini arttırmanın yanı sıra işlem karmaşıklığını ve güç kullanımını da arttırmaktadır. Bu nedenle, bu tez de, ÇRÇ HT'de kullandığı referans çerçeve sayısı ayarlanabilen bir H.264 video kodlayıcısı donanımı tasarlanmıştır. Bu sayede önerilen donanım video kodlama kalitesi ve harcadığı güç arasında ödünleşim yapabilmektedir. Önerilen H.264 video kodlayıcı donanımı daha önceden geliştirilen düşük maliyetli bir çerçeve içi kodlayıcı donanımı, ve yeni ÇRÇ kullanan HT, kip seçimi ve hareket dengelemesi donanımları içermektedir.

İlk olarak, işlem karmaşıklığı düşük ÇRÇ kullanan uyarlanır bir HT algoritması ve bu algoritmayı gerçekleyen düşük enerjili ÇRÇ kullanan uyarlanır bir HT donanımı tasarladık. Önerilen ÇRÇ kullanan HT algoritması, işlenen MB başına uyarlanır olarak belirlediği referans çerçeve sayısı ve erken sonlandırma tekniği ile işlem karmaşıklığını düşürmektedir. Önerilen H.264 ÇRÇ kullanan HT donanımı Verilog HDL ile düşük maliyetli Xilinx Spartan 6 FPGA'sı üzerinde gerçeklenmiştir. Önerilen H.264 ÇRÇ kullanan HT donanımı, Xilinx Spartan 6 FPGA'sı üzerinde ihmal edilebilir bir PSNR kaybı ile %29-72 enerji kazancı sağlamıştır.

Daha sonra, H.264 video kodlayıcı donanımı tasarlanmış ve Verilog HDL ile gerçeklenmiştir. Verilog RTL kodları Xilinx Virtex 6 FPGA'sına 135 MHz'de çalışacak şekilde yerleştirilmiştir. H.264 video kodlayıcı donanımı saniyede 55 CIF (352x288) çerçevesini kodlayabilmektedir ve güç kullanımı HT'de kullandığı referans çerçeve sayısına göre 115 mW ile 235 mW arasında değişmektedir. H.264 video kodlayıcı donanımı ile bir girdi çerçeve için oluşturulan bit dizgisi H.264 Joint Model kodçözücü ile başarılı bir biçimde

çözülmüş ve çözülen çerçeve “YUV Player” programı kullanılarak görsel olarak doğrulanmıştır.

To my mother, my father
and my brother Murat…

ACKNOWLEDGEMENTS

First of all, I would like to express my deep appreciation to my supervisor, Assist. Prof. Dr. Ilker Hamzaoglu, for his skills, enthusiasm, unconditional support, guidance and patience during the process of this thesis. I appreciate very much for his suggestions, detailed reviews, invaluable advices and life lessons. I feel myself privileged as his student.

I want to thank "System-on-a-Chip Lab" mates; Onur Can Ulusel, Abdulkadir Akın, Mert Çetin, Çağlar Kalaycıoğlu, Yusuf Adıbelli, Zafer Özcan and Murat Can Kıral for their great friendship and their collaboration during my MS study. I want to thank Gökhan Sayılar for his tolerance to me and significant contributions during my MS study.

My acknowledgements also go to Sabancı University and TÜBİTAK for supporting me with scholarships during my MS study.

I also would like to express my deepest gratitude to my family; my mother Ferda, my father Salim Öz and my brother Murat for their unlimited support, encouragement and trust made everything possible for me. It is very heartwarming to know that one has such family.

Last but not least; I want to thank Neşe for being there throughout the good, the bad and the ugly, it would be joyless, boring and a lonely journey without her.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ASIC	Application Specific Integrated Circuit
BM	Block Matching
BRAM	Block Random Access Memory
BRF	Best Reference Frame
CAVLC	Context Adaptive Variable Length Coding
CIF	Common Intermediate Format
DVD	Digital Versatile Disc
FD	Frame Decision
FPGA	Field Programmable Gate Array
FS	Full Search
FSM	Finite State Machine
HDL	Hardware Description Language
HDTV	High Definition Television
ISO	International Standards Organization
ITU-T	International Telecommunications Union, Telecommunications Standardization Sector
JVT	Joint Video Team
LUT	Look-up Table
MB	Macroblock
ME	Motion Estimation
MPEG	Motion Picture Experts Group
MRF	Multiple Reference Frame
MV	Motion Vector
NAL	Network Abstraction Layer

PE	Processing Element
PSNR	Peak Signal Noise Ratio
PVT	Process Voltage Temperature
QP	Quantization Parameter
R-D	Rate-Distortion
RTL	Register Transfer Level
SAD	Sum of Absolute Differences
SRF	Single Reference Frame
VBS	Variable Block Size
VCD	Value Change Dump
VGA	Video Graphics Array

CHAPTER I

INTRODUCTION

Video compression systems are used in many commercial products, from consumer electronic devices such as digital camcorders, cellular phones to video teleconferencing systems. These applications make the video compression hardware devices an inevitable part of many commercial products. To improve the performance of the existing applications and to enable the applicability of video compression to new real-time applications, recently, a new international standard for video compression is developed. This new standard, offering significantly better video compression efficiency than previous video compression standards, is developed with the collaboration of ITU and ISO standardization organizations. Hence it is called with two different names, H.264 and MPEG4 Part 10.

H.264 video coding standard has a much higher coding efficiency (capable of saving up to %50 bit rate at the same level of video quality) than the previous standards. Because of its high coding efficiency, flexibility and robustness to different communication environments, H.264 is expected to be widely used within a couple of years in many applications such as digital TV, DVD, video transmission and video conferencing over the internet and in wireless networks.

The video compression efficiency achieved in H.264 standard is not a result of any single feature but rather a combination of a number of encoding tools. The top-level block diagram of an H.264 Encoder is shown in Figure 1.1.

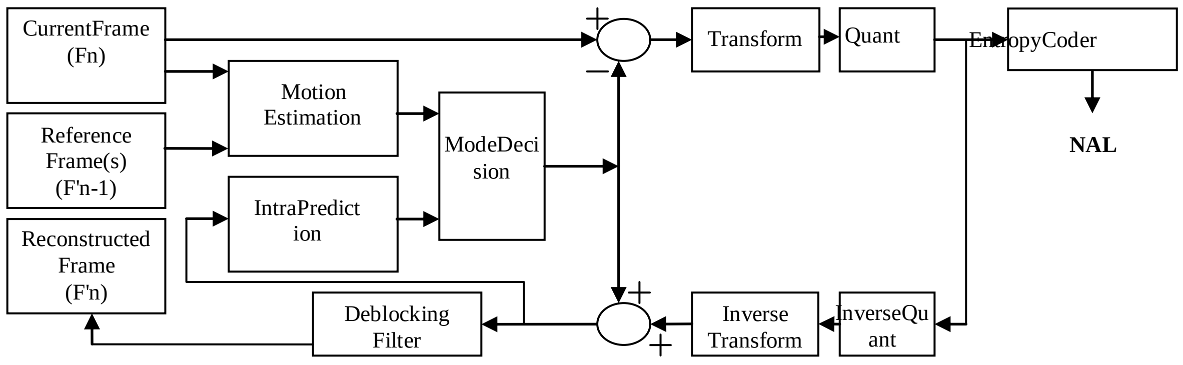

Figure 1.1 H.264 Encoder Block Diagram

As shown in Figure 1.1, an H.264 encoder has a forward path and a reconstruction path. The forward path is used to encode a video frame by using intra and inter predictions and to create the bit stream. The reconstruction path is used to decode the encoded frame and to reconstruct the decoded frame. Since a decoder never gets original images, but rather works on the decoded frames, reconstruction path in the encoder ensures that both encoder and decoder use identical reference frames for intra and inter prediction. This avoids possible encoder – decoder mismatches [1, 2, 3].

Forward path starts with partitioning the input frame into MBs. Each MB is encoded in intra or intermode depending on the mode decision. In both intra and inter modes, the current MB is predicted from the reconstructed frame. Intra mode generates the predicted MB based on spatial redundancy, whereas inter mode generates the predicted MB based on temporal redundancy. Mode decision compares the required amount of bits to encode a MB and the quality of the decoded MB for both of these modes and chooses the mode with better quality and bit-rate performance. In either case, intra or inter mode, the predicted MB is subtracted from the current MB to generate the residual MB. Residual MB is transformed using 4x4 and 2x2 integer transforms. Transformed residual data is quantized and quantized transform coefficients are re-ordered in a zig-zag scan order. The reordered quantized transform coefficients are entropy encoded. The entropy-encoded coefficients together with header information, such as MB prediction mode and quantization step size, form the compressed bit stream. The compressed bit stream is passed to network abstraction layer (NAL) for storage or transmission [1, 2, 3].

Reconstruction path begins with inverse quantization and inverse transform operations. The quantized transform coefficients are inverse quantized and inverse transformed to generate the reconstructed residual data. Since quantization is a lossy process, inverse quantized and inverse transformed coefficients are not identical to the original residual data. The reconstructed residual data are added to the predicted pixels in order to create the reconstructed frame. A deblocking filter is applied to reduce the effects of blocking artifacts in the reconstructed frame [1, 2, 3].

Motion estimation (ME) is the process of searching a search window in a reference frame to determine the best match for a block in a current frame based on a search criterion such as minimum Sum of Absolute Difference (SAD) [1]. ME is used to reduce the bit-rate in video compression systems by exploiting the temporal redundancy between successive frames and it is the most computationally intensive part of video compression systems. Block Matching (BM) is the most preferred method for ME. BM partitions current frame into non-overlapping NxN rectangular blocks and tries to find the block from the reference frame in a given search range that best matches the current block. SAD is the most preferred block matching criterion.

As shown in Figure 1.2, the location of a block in a frame is given using the (x,y) coordinates of top-left corner of the block. The search window in the reference frame is the [-p,p] size region around the location of the current block in the current frame. The SAD value for a current block in the current frame and a candidate block in the reference frame is calculated by accumulating the absolute differences of corresponding pixels in the two blocks as shown in the formula (1.1).

(1.1)

In formula (1.1), B_{mxn} is a block of size mxn, **d**=*(dx, dy)* is the motion vector, *c* and *r* are current and reference frames respectively. Since a motion vector expresses the relative motion of the current block in the reference frame, motion vectors are specified in relative coordinates. If the location of the best matching block in the reference frame is (x+u, y+v), then the motion vector is expressed as (u,v). Motion estimation is performed on the luminance (Y) component of a YUV image and the resulting motion vectors are also used for the chrominance (U and V) components.

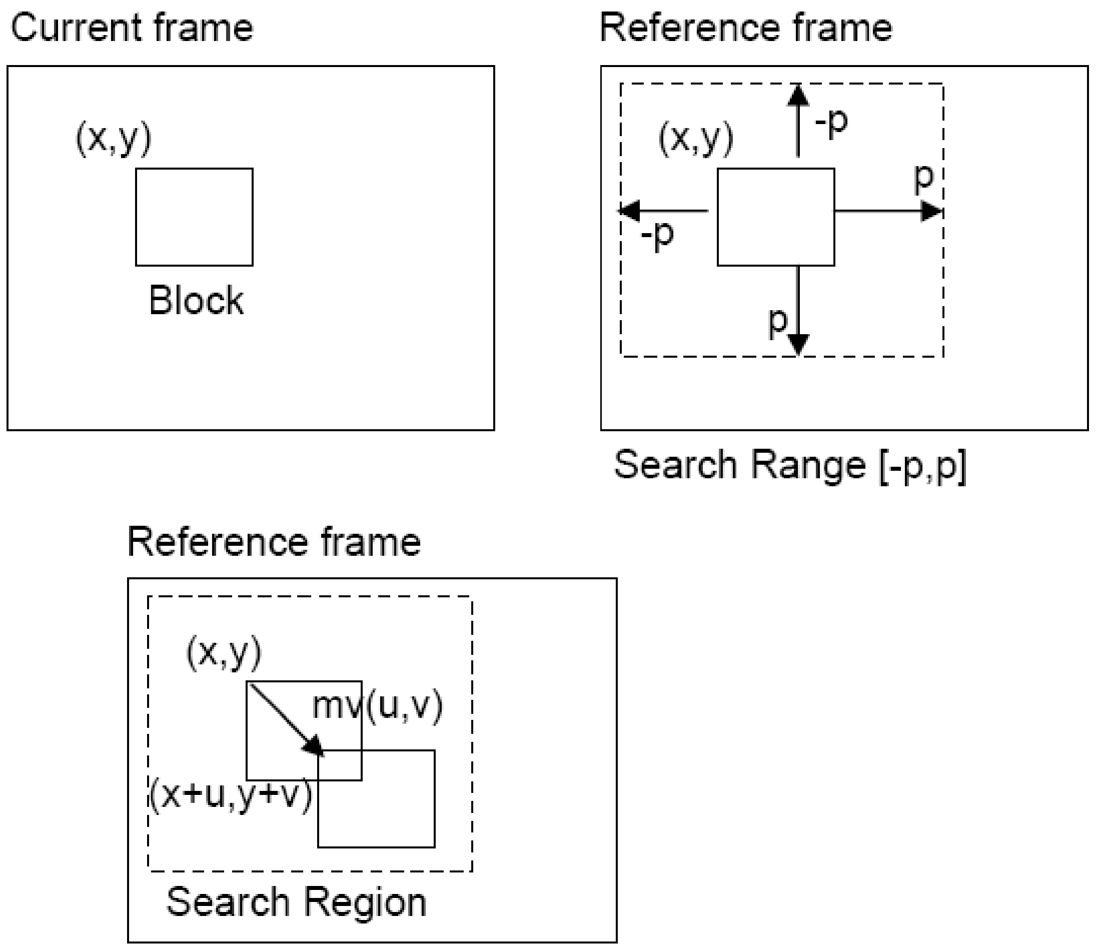

Figure 1.2 Motion Estimation [4]

Among the BM algorithms, Full Search (FS) algorithm achieves the best performance since it searches all search locations in a given search range. But the computational complexity of FS ME algorithm is high. In order to improve the ME performance, variable block size (VBS) and multiple reference frame (MRF) ME are used in H.264 standard. But the computational complexity of FS algorithm for VBS ME and MRF ME is even higher [5, 6, 7]. Although many fast search ME algorithms are developed, FS algorithm is a popular candidate for hardware implementation because of its regular dataflow and good compression performance [8, 9].

1.1 Thesis Contribution

In this thesis, we first propose a low complexity H.264 MRF ME algorithm and a low energyadaptive hardware for its real-time implementation. The proposed MRF ME algorithm reduces the computational complexity of MRF ME by using a dynamically determined number of reference frames for each Macroblock (MB) and early termination.

We designed a serial and a parallel H.264 MRF VBS ME hardware implementing the proposed low complexity MRF ME algorithm. The serial ME hardware has 256 processing elements (PEs) and it searches one reference frame at a time. The parallel ME hardware has 512 PEs and it searches two reference frames in parallel. We also designed an H.264 MRF VBS ME hardware which uses 5 reference frames for all MBs (Ref-5).This ME hardware has 256 PEs and it implements FS algorithm.

Allthree MRF ME hardware are implemented in Verilog HDL and mapped to a low cost Xilinx Spartan 6 FPGA using Synopsys Synplify synthesis tool and Xilinx ISE 11.4 place and route tool at 58 MHz under worst case PVT conditions.The Ref-5, serial and parallel MRF ME hardware implementations use 5k, 5.5k, 10.2k slices, and they can process9, between 11 and 45, between 22 and 45 VGA frames per second (fps), respectively on this FPGA.

The power consumptions of these MRF ME hardware implementations on this FPGA are estimated at 50MHz using Xilinx XPower 11.4 tool for a CIF size Foreman video frame. The results showed that the serial MRF ME hardware has 29-72% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss, and the parallel MRF ME hardware has 21-64% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss.

In this thesis, we then developed a reconfigurable H.264 video encoder hardware for real-time portable applications targeting level 2.0 of baseline profilein which the number of reference frames used for MRF ME can be configured based on the application requirements in order to trade-off video coding efficiency and power consumption.The proposed video encoder hardware is based on an existing low cost H.264 intra frame coder hardware [10, 11] and it includes new reconfigurable MRF ME, mode decision and motion compensation hardware.

The proposed H.264 video encoder hardware is implemented in Verilog HDL and mapped to Xilinx Virtex 6 FPGA. The FPGA implementation is verified with post place & route simulations. The bitstream generated by the proposed video encoder hardware for an input frame is successfully decoded by H.264 Joint Model reference software decoder and the decoded frame is displayed using a YUV Player tool for visual verification. The FPGA implementation works at 135 MHz, it can code 55 CIF (352x288) frames per second, and its power consumption ranges between 115mW and 235mW depending on the number of reference frames used for MRF ME.

The H.264 video encoder hardware presented in [12, 13] achieve higher performance than our hardware design at the expense of a much higher hardware cost and power consumption. The proposed H.264 video encoder hardware is a better solution for portable consumer electronics products.

The rest of the thesis is organized as follows;

Chapter II presents the proposed low complexity MRF ME algorithm and low energy hardware architectures for its implementation.

Chapter III presents the proposed baseline H.264 video encoder hardware.

Chapter IV presents the conclusions and the future work.

CHAPTER II

MULTIPLE REFERENCE FRAMEMOTION ESTIMATION ALGORITHM AND HARDWARE

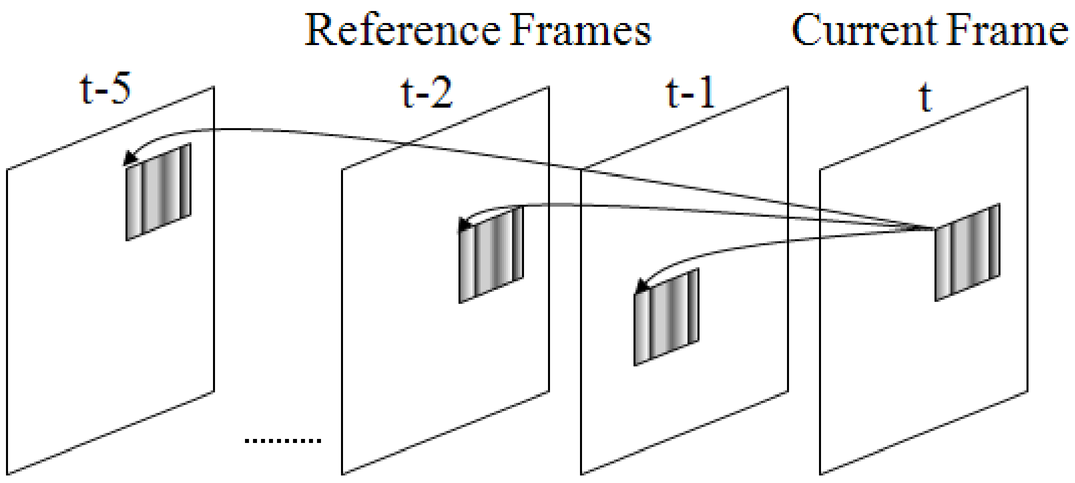

Figure 2.1 Multiple Reference Frame Motion Estimation

Multiple reference frame (MRF) ME is illustrated in Figure 2.1. MRF ME has better video coding efficiency than single reference frame (SRF) ME because of repetitive motions, uncovered backgrounds, alternating camera angles, camera shaking, sampling errors, shadow and lighting changes and noise in the source signal [14]. However, the amount of computation performed for ME increases with the number of reference frames used.

Therefore, this coding gain comes at the expense of increased computational complexity and energy consumption.

Several algorithms for reducing the computational complexity of MRF ME with small PSNR loss are proposed in the literature [14-21]. In [15-18], best reference frames (BRF) for the neighboring blocks are used to predict the BRF for the current block. BRF for a block is the reference frame that gives the best match to this block. In [15, 16], for some cases BRF is assumed to be the closest frame and temporal correlation is used. In [14], the spatial and temporal correlation of the motion vectors are used to compute motion vector trajectories and a short ranged search around the trajectory is done as the final refinement. In [15, 18], the spatial correlations of the motion vectors are also used as a criterion to determine the spatial correlation of the BRFs. Other techniques such as early termination, fast motion search and simplex minimization are proposed in [19], [20] and [21] respectively.

The MRF ME algorithms proposed in [14-21] are not implemented in hardware. MRF ME hardware implementations are proposed in [22, 23]. However, these MRF ME hardware use fixed number of reference frames for ME, and instead of computational complexity reduction, they try to reduce the required memory size and bandwidth by data reuse and frame-level scheduling.

In this thesis, we propose a low complexity H.264 MRF ME algorithm and a low energy adaptive hardware for its real-time implementation. The proposed MRF ME algorithm reduces the computational complexity of MRF ME by using a dynamically determined number of reference frames for each Macroblock (MB) and early termination.

We designed a serial and a parallel H.264 MRF variable block size (VBS) ME hardware implementing the proposed low complexity MRF ME algorithm. The serial ME hardware has 256 processing elements (PEs) and it searches one reference frame at a time. The parallel ME hardware has 512 PEs and it searches two reference frames in parallel. We also designed an H.264 MRF VBS ME hardware which uses 5 reference frames for all MBs (Ref-5). This ME hardware has 256 PEs and it implements FS algorithm.

All three MRF ME hardware are implemented in Verilog HDL and mapped to a low cost Xilinx Spartan 6 FPGA using Synopsys Synplify synthesis tool and Xilinx ISE 11.4

place and route tool at 58 MHz under worst case PVT conditions. The Ref-5, serial and parallel MRF ME hardware implementations use 5k, 5.5k, 10.2k slices, and they can process 9, between 11 and 45, between 22 and 45 VGA frames per second (fps), respectively on this FPGA.

The power consumptions of these MRF ME hardware implementations on this FPGA are estimated at 50 MHz using Xilinx XPower 11.4 tool for a CIF size Foreman video frame. The results showed that the serial MRF ME hardware has 29-72% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss, and the parallel MRF ME hardware has 21-64% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss.

2.1 Proposed Low Complexity Multiple Reference Frame Motion Estimation Algorithm

The proposed MRF ME algorithm uses spatial and temporal correlations for estimating the BRF for the current MB. The spatial and temporal correlations are shown in Figure 2.2. The median of the BRFs of the neighboring MBs A, B and C is taken as the spatially correlated BRF candidate for current MB E. The temporal correlation used in our algorithm is different from the temporal correlation used in [14-16]. The BRF of MB F is taken as the temporally correlated BRF candidate for the current MB E. The MB F is in the same location of the current MB E in the temporally closest frame to the current MB E.

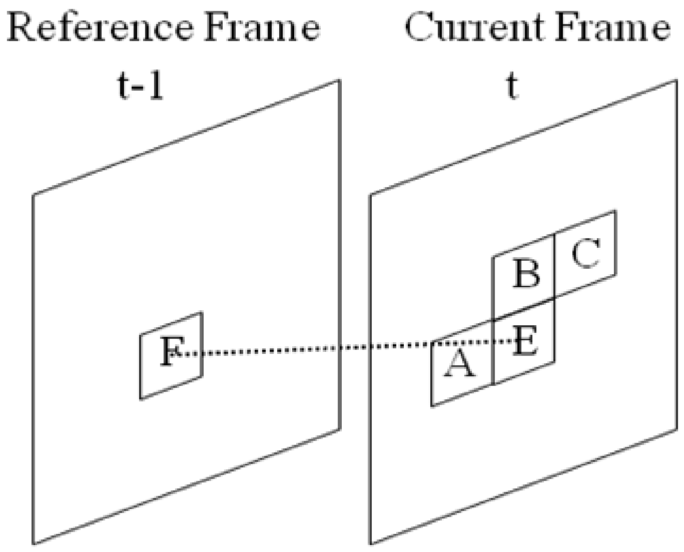

Figure 2.2 Spatial and Temporal Correlations

The proposed MRF ME algorithm is as follows:

Step 1 - Perform FS ME on the temporally correlated BRF candidate.

Step 2 - Perform FS ME on the spatially correlated BRF candidate.

Step 3 - Determine the BRF by performing mode decision between the reference frames of Step 1 and Step 2. If the cost of the BRF is below the threshold value, jump to Step 5.

Step 4 - Perform FS ME on two additional frames and determine the BRF by performing mode decision between these two additional frames and the ones form Step 3.

Step 5 - Finish ME for the current MB.

8x8, 8x16 and 16x8 partitions of a MB may be selected from different reference frames. However, only the BRFs for the 16x16 partitions are used to determine both the temporally and spatially correlated BRF candidates in order to reduce computational complexity. Since using only spatial and temporal correlation may cause the convergence of the BRFs of all MBs to the same frame, the proposed algorithm performs ME on two additional frames when the cost of the BRF obtained by spatial and temporal correlation is above a threshold value T_1.

The proposed algorithm also uses an early termination technique. While performing FS in Steps 1, 2 and 4 of the proposed MRF ME algorithm, if the SAD value of a search location is less than a threshold value T_2, the corresponding step for the current MB is early terminated and the algorithm jumps to Step 5.

The proposed MRF ME algorithm is implemented in H.264 encoder reference software (JM) version 14.2 [24]. Single reference frame ME (Ref-1), MRF ME using 5 reference frames for all MBs (Ref-5) and the proposed MRF ME algorithm are simulated using JM 14.2. The simulations are done for the threshold values $T_1 \in \{0, 500, 1000, 1500, 2000\}$, and the quantization parameters $QP \in \{20, 25, 30, 35, 40\}$. In the simulations T_2 is set to 500. The simulations are done for 100 frames of several CIF size (352x288) video sequences, and the luminance PSNR is used as the performance metric. The other simulation settings were as follows:

- RD optimization - Off
- Rate control - Off
- Fast ME - Off
- Search Range [-16,+16]
- P frames coding, only the first frame is I frame

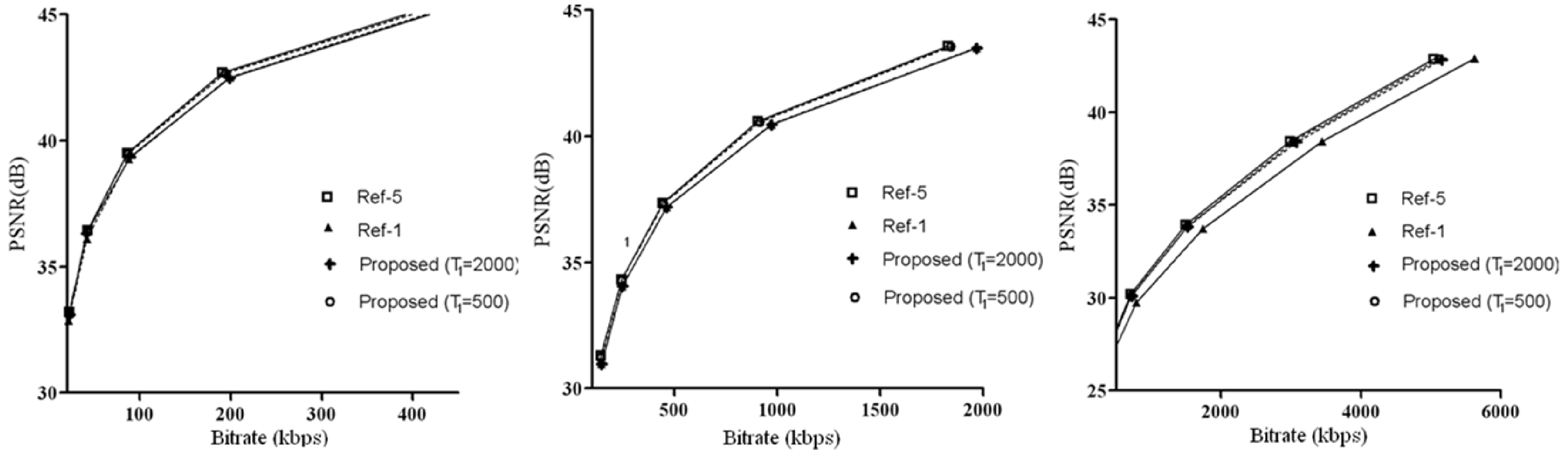

Figure 2.3 Rate Distortion Curves for a) Foreman, b) Akiyo, and c) Mobile

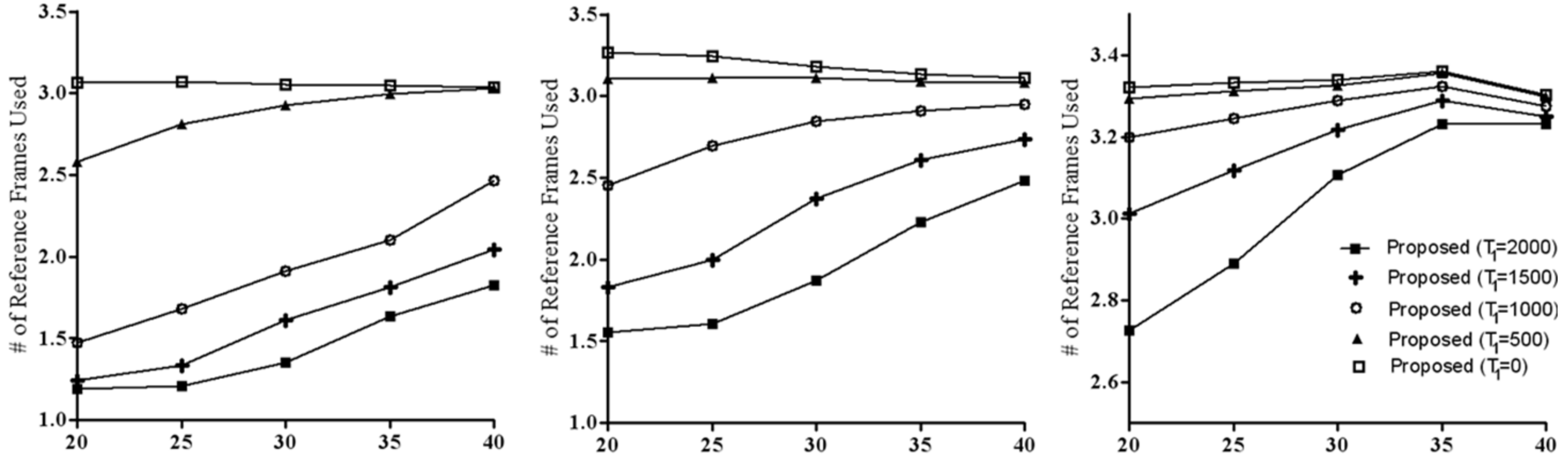

Figure 2.4 Number of Reference Frames Used vs. Quantization Parameter for a) Foreman, b) Akiyo, and c) Mobile

Table 2.1 Simulation Results of Proposed MRF ME Algorithm, Ref-1 and Ref-5

	QP	**Bitrate (kbps)**							**PSNR (dB)**						
		T_1=*0*	**T_1=*500***	**T_1=*1000***	**T_1=*1500***	**T_1=*2000***	***Ref-1***	***Ref-5***	**T_1=*0***	**T_1=*500***	**T_1=*1000***	**T_1=*1500***	**T_1=*2000***	***Ref-1***	***Ref-5***
Akiyo	20	415.69	415.91	422.14	427.24	429.15	426.41	411.24	45.192	45.187	45.143	45.125	45.124	45.136	45.218
	25	193.2	193.21	194.49	197.06	198.53	198.99	190.54	42.656	42.652	42.595	42.519	42.489	42.514	42.694
	30	86.7	86.78	87.01	87.7	88.7	87.43	86.31	39.465	39.467	39.45	39.394	39.343	39.263	39.506
	35	41.82	41.82	41.87	41.67	41.186	41.5	42.12	36.399	36.399	36.37	36.361	36.313	36.106	36.453
	40	22.32	22.32	22.32	22.14	22.2	21.33	21.95	33.127	33.127	33.125	33.105	33.092	32.866	33.197
Foreman	20	1841.9	1845.1	1868.7	1897.82	1921.8	1968.1	1828	43.551	43.547	43.516	43.486	43.474	43.495	43.566
	25	915.36	912.58	918.25	933.9	942.4	971.39	905.61	40.57	40.568	40.555	40.5	40.455	40.468	40.599
	30	446.57	445.86	447.74	450.6	453.84	462.73	443.12	37.32	37.323	37.321	37.294	37.246	37.186	37.35
	35	242.75	242.45	242.79	242.98	243.8	246.34	241.82	34.282	34.274	34.264	34.273	34.233	34.06	34.308
	40	142.73	143.05	140.99	141.32	141.9	143.38	140.58	31.23	31.231	31.262	31.235	31.247	30.973	31.29
Mobile	20	5113.3	5132.4	5121.7	5148.27	5159.8	5627.3	5041.1	42.863	42.856	42.851	42.843	42.843	42.912	42.856
	25	3033.2	3048.9	3043.3	3035.29	3073.8	3440.8	2989.7	38.408	38.413	38.398	38.389	38.383	38.42	38.437
	30	1519.5	1512.7	1513.1	1514.17	1518.5	1739.4	1488.8	33.858	33.858	33.851	33.847	33.834	33.713	33.941
	35	712.56	713.44	711.95	714.61	717.35	783.62	699.67	30.098	30.098	30.098	30.088	30.089	29.739	30.201
	40	331.5	331.5	330.47	331.38	331.15	353.56	328.12	26.702	26.702	26.702	26.677	26.704	26.241	26.775

Table 2.2 Average PSNR Results of Proposed MRF ME Algorithm, Ref-1 and Ref-5

	Average PSNR Differences from Ref-5 (dB)					
T_2=500	Proposed T_1=0	Proposed T_1=500	Proposed T_1=1000	Proposed T_1=1500	Proposed T_1=2000	Ref-1
Akiyo	0.024	0.018	0.022	0.120	0.130	0.280
Foreman	0.047	0.048	0.120	0.180	0.270	0.400
Mobile	0.150	0.181	0.170	0.180	0.250	1.070

The rate distortion (R-D) curves for Foreman, Akiyo, and Mobile video sequences are shown in Figure 2.3. Akiyo contains slow motion resulting in low bitrates and high PSNR values. Foreman contains moderate motion resulting in moderate bitrates and moderate PSNR values. Mobile contains fast motion over highly textured background resulting in high bitrates and low PSNR values.

The simulation results showed that using multiple reference frames increases the PSNR and decreases the bitrate. Since Akiyo contains slow motion, the R-D curves of Ref-1 and Ref-5 are close, whereas the difference between Ref-1 and Ref-5 is more visible for Mobile. Further reference frames are selected as the BRF for video sequences with fast motion.

The proposed MRF ME algorithm produces better PSNR results and uses less bitrate than Ref-1, whereas Ref-5 performs better than the proposed MRF ME algorithm. For low threshold values, the R-D curve of the proposed MRF ME algorithm is very close to the one for Ref-5. As the threshold value increases, the R-D curve gets closer to the one for Ref-1. The proposed MRF ME algorithm uses 1.2 to 3.5 reference frames depending on both the video sequence and the threshold value as shown in Figure 2.4. As the threshold value decreases, the number of reference frames used increases.

Simulation results of the proposed MRF ME algorithm, Ref-1 and Ref-5 are shown in Table 2.1. Average PSNR differences of Ref-1 and the proposed MRF ME algorithm for different threshold values from Ref-5 are shown in Table 2.2. If the early termination is not used (T_2=0), the average PSNR differences are increased by 0.01dB on the average. The threshold values T_1 and T_2 for the proposed MRF ME algorithm can be determined based on the quality and computational complexity requirements of the application.

2.2 Proposed Low Energy Multiple Reference Frame Motion Estimation Hardware

The block diagram of the H.264 SRF VBS ME hardware we proposed in [25] is shown in Figure 2.5. This ME hardware implements the FS algorithm in a search range of [-16, 15] pixels. It uses a 2-D systolic PE array with 256 PEs. Each circle in the figure represents a PE. All PEs are capable of shifting data down, up and left.

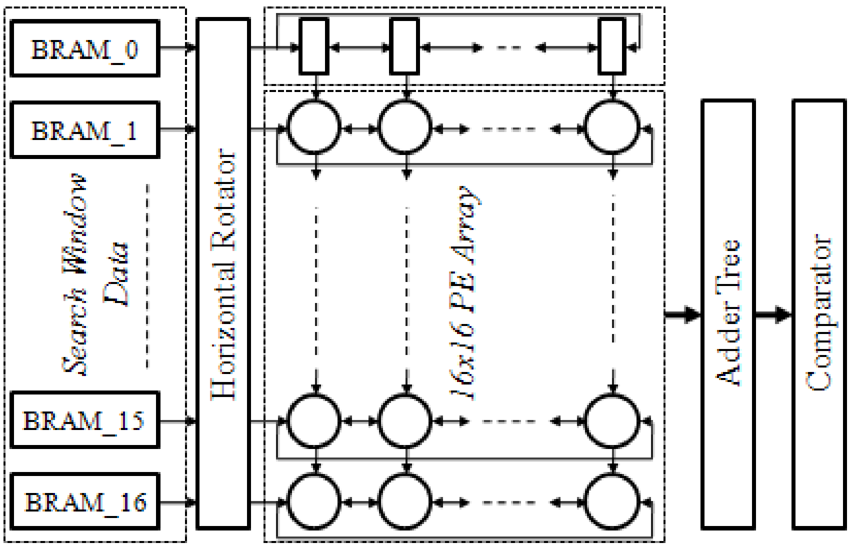

Figure 2.5 256 PE VBS ME Hardware Architecture

Each PE calculates the absolute difference between a pixel in the current MB and a pixel in the search window. The SAD of a search location is calculated by adding the absolute differences calculated by PEs using an adder tree. This ME hardware is highly pipelined and its latency is eight clock cycles; one cycle for synchronous read from BRAM, one cycle for horizontal shifting, one cycle for SAD computation in 2-D systolic PE array, two cycles for the adder tree generating 4x4 SADs and three cycles for the adder tree generating 41 MVs for 7 different block sizes.

Since this VBS ME hardware calculates the SAD value of one search location in each clock cycle, it can be easily early terminated after the SAD value of a search location is calculated by comparing the threshold value T_2 with this SAD value.

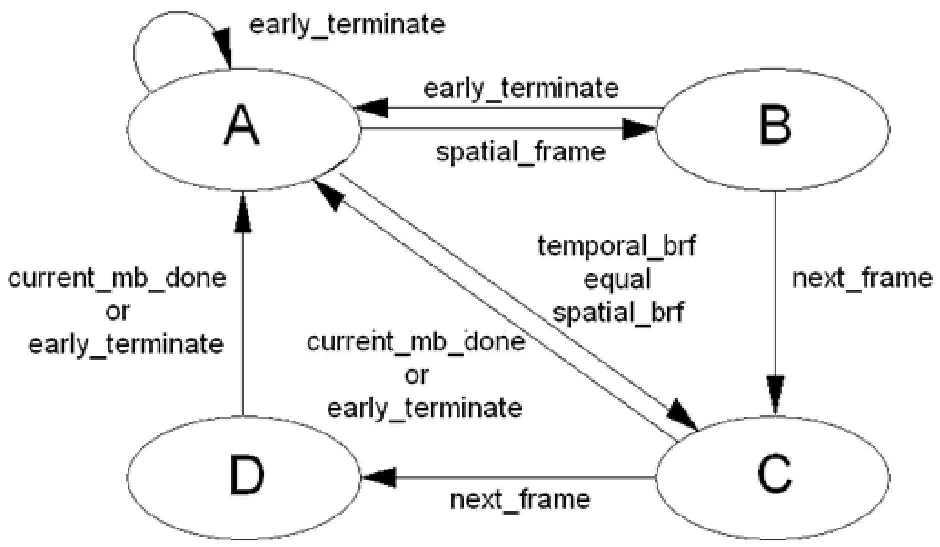

Figure 2.6 Top Level Control FSM of Serial Implementation

The top level control FSM of serial MRF VBS ME hardware is shown in Figure 2.6. This ME hardware implements the steps of the proposed MRF ME algorithm sequentially using 256 PEs. Steps 1, 2 and 3 are executed in states A, B, and C respectively. Step 4 is executed in states C and D. If the temporally and the spatially correlated BRF candidates are the same, state B is skipped. If there is an early termination in a state, FS for the current MB is stopped and the current reference frame is selected as the BRF. In this case, ME hardware jumps to state A and starts the search process for the next current MB.

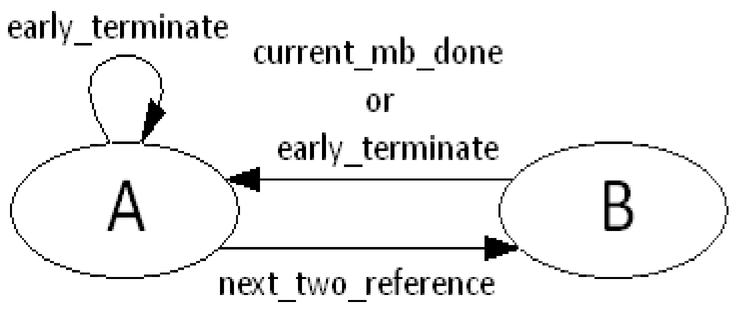

Figure 2.7 Top Level Control FSM of Parallel Implementation

The top level control FSM of parallel MRF VBS ME hardware is shown in Figure 2.7. In state A, steps 1 and 2 are executed in parallel. If the MRF ME hardware decides to search two additional frames, step 4 is executed in state B. In state B, these two additional frames are searched in parallel.

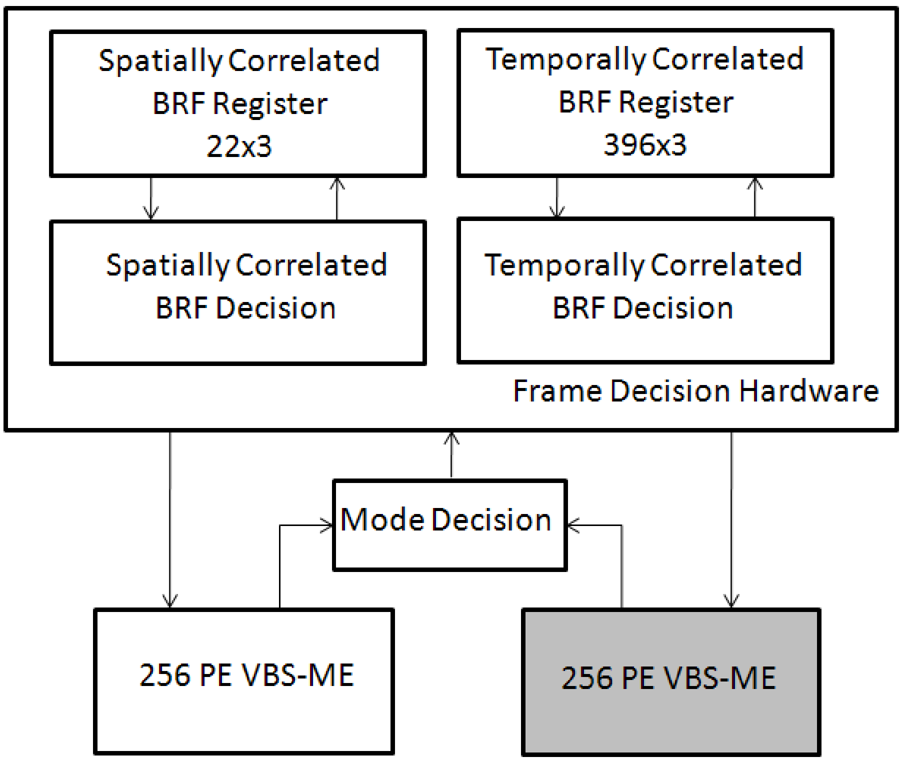
Spatially Correlated BRF Register 22x3
Temporally Correlated BRF Register 396x3
Spatially Correlated BRF Decision
Temporally Correlated BRF Decision
Frame Decision Hardware
Mode Decision
256 PE VBS-ME
256 PE VBS-ME

Figure 2.8 Top Level Block Diagram of Proposed MRF ME Hardware

The top-level block diagram of the proposed serial and parallel MRF ME hardware architectures is shown in Figure 2.8. 256 PE SRF VBS ME hardware we proposed in [25] is used in these ME hardware. The gray shaded 256 PE SRF VBS ME hardware and its interconnections do not exist in the serial MRF ME hardware. The parallel MRF ME hardware performs FS ME for the current MB in two reference frames in parallel by using two 256 PE SRF VBS ME hardware. However, if the temporally and the spatially correlated BRF candidates of the current MB are the same, only one 256 PE SRF VBS ME hardware is used and the other one is clock gated to reduce the dynamic power consumption.

Frame decision (FD) hardware determines the temporally and spatially correlated BRF candidates and sends them to SRF VBS ME hardware. Mode decision hardware determines the best partition size and the BRF based on the SAD values obtained by SRF VBS ME hardware for temporally and spatially correlated BRF candidates. If the cost of the BRF is above the threshold value T_1, FD hardware requests two additional frames. Mode decision hardware determines the best partition size and the BRF based on the SAD values obtained by SRF VBS ME hardware for the four reference frames searched.

Since the BRFs of left, top and top-right neighboring MBs are used for determining the spatially correlated BRF candidate for the current MB, as shown in Figure 2.9, BRFs of one row of MBs in the current frame are stored in the spatially correlated BRF register. Since the MRF ME hardware can use 4 of the 5 previous frames as reference frames, 3x22=66 bits on-chip memory is used for storing these BRFs for a CIF size frame. Spatially correlated BRF decision hardware, for example, determines the spatially correlated BRF candidate for the MB on the 4th column and 10th row by reading the BRFs of MBs 2, 3 and 4 from the spatially correlated BRF register and calculating their median.

Since the BRF of the MB in the same position in previous frame is used as the temporally correlated BRF candidate for the current MB, as shown in Figure 2.9, BRFs of all MBs in the previous frame are stored in the temporally correlated BRF register. Since the MRF ME hardware can use 4 of the 5 previous frames as reference frames, 3x396=1188 bits on-chip memory is used for storing these BRFs for a CIF size frame. Temporally correlated BRF decision hardware, for example, determines the temporally correlated BRF candidate for the MB on the 4th column and 10th row by reading the BRF of the MB in location 201.

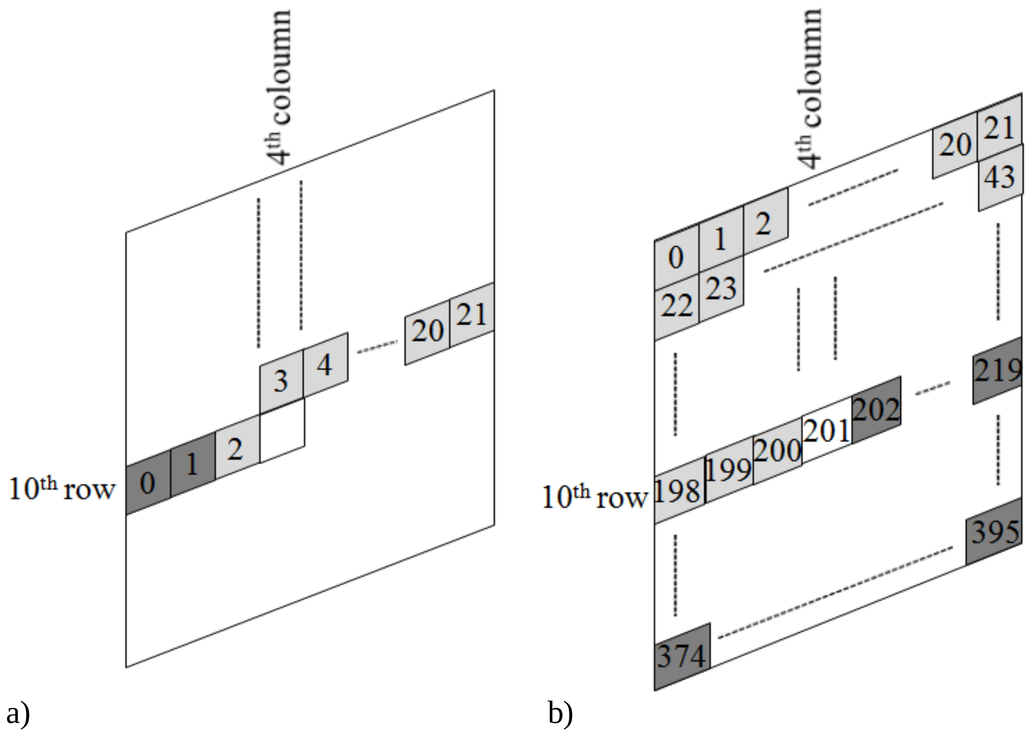

Figure 2.9 On-chip Memory for Storing BRFs of a) Spatially Correlated MBs b) Temporally Correlated MBs

The proposed MRF ME hardware determines the number of reference frames searched for each MB dynamically. For example, if the temporally and spatially correlated BRF candidates of the current MB are the same and the resulting cost after performing FS ME on that reference frame is below the threshold T_1, MRF ME hardware searches only one reference frame for that current MB. If the temporally and spatially correlated BRFs of the current MB are different and the resulting cost after performing FS ME on these two reference frames is above the threshold T_1, MRF ME hardware searches four reference frames for that current MB. Since the threshold values T_1 and T_2 are inputs to the MRF ME hardware, they can also be changed for each MB dynamically.

We also designed and implemented an H.264 MRF ME hardware which uses 5 reference frames for each MB (Ref-5). Ref-5 hardware has 256 PEs and it is also based on the SRF VBS ME hardware we proposed in [25]. The Control FSM of Ref-5 hardware is given in Figure 2.10. Each state represents performing ME for the current MB using one reference frame.

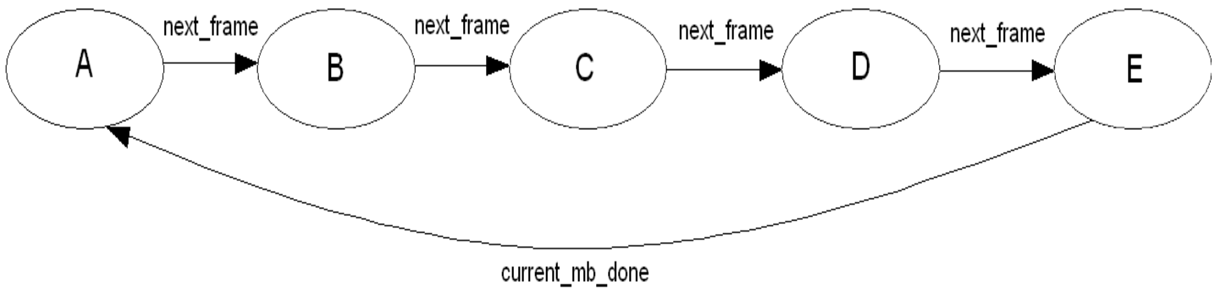

Figure 2.10 Top Level Control FSM of Ref-5 Hardware

2.3 Implementation Results

The serial, parallel and Ref-5 MRF ME hardware architectures are implemented in Verilog HDL and mapped to a low cost Xilinx Spartan 6 FPGA using Synopsys Synplify synthesis tool and Xilinx ISE 11.4 place and route tool at 58 MHz under worst case PVT conditions. The area and maximum operating frequency results are given in Table 2.3.

Table 2.3 Area and Performance Results of Proposed Serial, Parallel and Ref-5 MRF ME Hardware

	Proposed MRF ME Hardware (Serial)	Proposed MRF ME Hardware (Parallel)	Ref-5 Hardware
Function Generators	14248	28386	13692
CLB slices	9401	18518	9486
BRAMs	36	76	34
Frequency (MHz)	58	58	58

Ref-5 hardware can process 9 VGA fps. Serial hardware can process 11 VGA fps in the worst case and 45 VGA fps in the best case. Parallel hardware can process 22 VGA fps in the worst case and 45 VGA fps in the best case.

Table 2.4 Power and Energy Consumptions of Proposed Serial, Parallel and Ref-5 MRF ME Hardware

	Proposed MRF ME Hardware T_1=500 T_2=0		Proposed MRF ME Hardware T_1=500 T_2=500		Proposed MRF ME Hardware T_1=1000 T_2=0		Proposed MRF ME Hardware T_1=1000 T_2=500		Proposed MRF ME Hardware T_1=1500 T_2=0		Proposed MRF ME Hardware T_1=1500 T_2=500		Ref-5 MRF ME Hardware
	Serial	Parallel	Serial	Parallel	Serial	Parallel	Serial	Parallel	Serial	Parallel	Serial	Parallel	
Clock (mW)	80	137	80	137	80	137	80	137	80	137	80	137	76
Logic (mW)	47	109	41	98	40	84	37	76	35	74	34	69	49
Signal (mW)	76	178	73	176	66	135	65	134	61	121	60	119	78
BRAMs (mW)	15	33	14	30	14	30	13	29	13	29	12	28	16
Total Power (mW)	218	457	208	441	200	386	195	376	189	361	186	353	219
Total time (sec)	0.032	0.017	0.03	0.015	0.022	0.013	0.02	0.012	0.016	0.01	0.015	0.01	0.045
Energy (mJ)	6.97	7.77	6.24	6.62	4.4	5.02	3.9	4.51	3.02	3.61	2.79	3.51	9.86
Energy Reduction	29%	21%	37%	33%	55%	49%	60%	54%	69%	63%	72%	64%	-

The power consumptions of the Ref-5, serial and parallel MRF ME hardware implementations on the same FPGA are estimated at 50MHz for a CIF size Foreman video frame using Xilinx XPower 11.4 tool. The power and energy consumption results are shown in Table 2.4. The results showed that the serial MRF ME hardware has 29-72% less energy consumption on this FPGA than Ref-5 hardware, and the parallel MRF ME hardware has 21-64% less energy consumption on this FPGA than Ref-5 hardware. Therefore, serial or parallel MRF ME hardware can be used for an application based on its speed and power consumption requirements.

CHAPTER III

H.264 VIDEO ENCODER HARDWARE

3.1 Proposed H.264 Video Encoder Hardware

The top-level block diagram of the proposed H.264 video encoder hardware is shown in Figure 3.1. The proposed hardware has two pipeline stages. The first stage, search & mode decision hardware, includes intra prediction, motion estimation and mode decision blocks. The second stage, coder hardware, includes the rest of the blocks.

The white blocks exist in the H.264 Intra Frame Coder System presented in [10, 11]. The detailed descriptions of intra prediction, mode decision and entropy coding hardware are given in [10, 11], and the detailed descriptions of transform and quantization hardware are given in [27]. The gray shaded block, Deblocking Filter, is presented in [26]. In this thesis, the gray shaded blocks Motion Estimation, Mode Decision, Motion Vector Prediction, Motion Compensation and part of the Header Generation are designed, implemented and integrated to the H.264 Intra Frame Coder system together with the gray shaded block Deblocking Filter.

The current frame is divided into 16x16 MBs, and each MB is encoded in raster scan order. Both intra prediction and ME predicts the current MB. While intra prediction generates the predicted MB using spatial redundancies and the pixels in the current frame, ME generates the predicted MB using temporal redundancies and the pixels in previously reconstructed frame.

Mode decision is used to compare the MBs predicted by intra prediction and ME, and chose the predicted MB which produces better video quality and lower bit-rate for encoding the current MB. If the MB predicted by ME is selected, motion compensation is performed from the reference frame using the motion vectors to generate the predicted MB and motion vectors are transformed into difference vectors for coding by motion vector prediction.

Then, the difference between the selected predicted MB and the current MB is taken to find the residue MB. The residue MB is transformed by an Integer Transform and Hadamard Transform. The transformed coefficients are quantized by a uniform quantizer. The quantized transform coefficients are reordered using a zig-zag scan ordering, and the reordered coefficients are entropy coded.

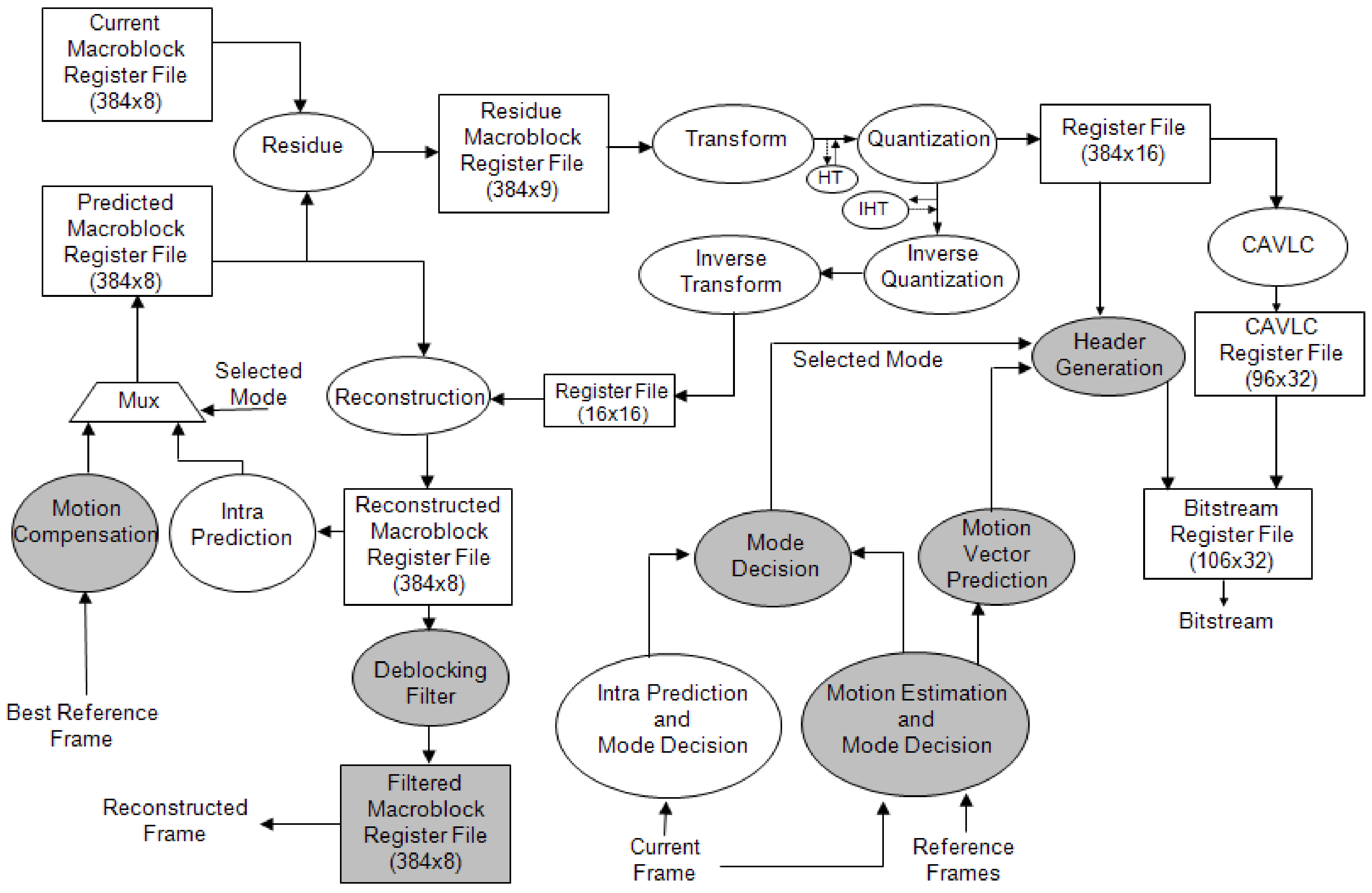

Figure 3.1 Top-Level Block Diagram of H.264 Video Encoder Hardware

The header including the prediction method (intra or inter) and modes, best reference frame, quantization parameter, motion vector differences and skipped MBs is coded with Exp-Golomb coding by the Header Generation hardware and combined with the entropy coded coefficients to generate the bitstream. The bitstream is sent to NAL for transmission or storage.

The reconstruction path begins with inverse quantization of the quantized coefficients. The inverse quantized coefficients are inverse transformed and added to the predicted MB to generate the reconstructed MB. Then, deblocking filter is applied to the reconstructed MB in order to reduce the blocking artifacts.

The reconfigurable MRF ME, Mode Decision, Motion Vector Prediction and Coder Hardware are shown in Figure 3.2. MRF ME hardware uses up to four reference frames depending on the configuration, and the configuration can be dynamically changed for each MB. Control Unit reads the search windows from the proper reference frames based on the configuration, and sends them to the Motion Estimation module together with the current MB. Mode Decision module gets the motion vectors and SAD values for all modes from the Motion Estimation module together with the number of reference frames used as inputs and determines the best reference frame and mode to encode the current MB.

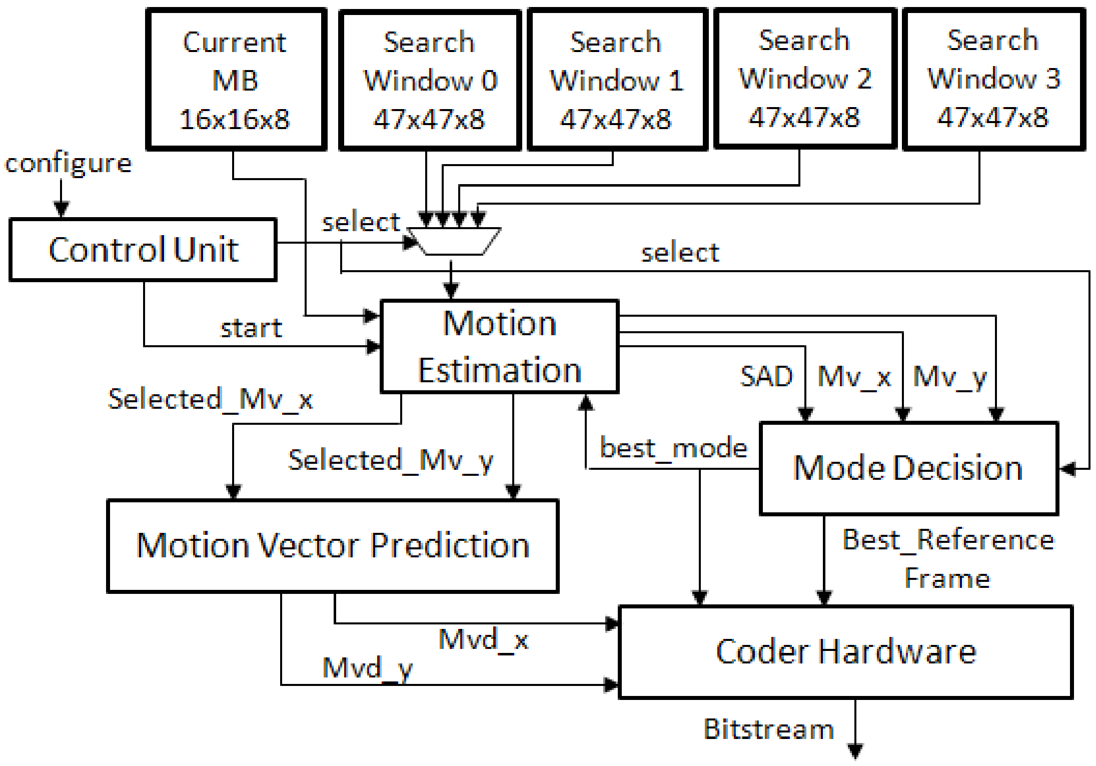

Figure 3.2 Top-Level Block Diagram of the Motion Estimation, Mode Decision, Motion Vector Prediction and Coder Hardware

3.1.1 Proposed Reconfigurable H.264 Motion Estimation Hardware

MRF ME increases the video coding efficiency at the expense of increased computational complexity and power consumption. The impact of MRF ME on PSNR (dB) obtained by using the H.264 reference software encoder [24] is shown in Chapter II.

In order to trade-off video coding efficiency and power consumption, the number of reference frames used for MRF ME in the proposed H.264 video encoder hardware can be dynamically configured for each MB based on the application requirements. The proposed reconfigurable MRF ME hardware uses up to four reference frames depending on the configuration. It is based on the serial MRF ME hardware implementation we proposed in Chapter II.

The top level FSM of the proposed reconfigurable H.264 MRF ME hardware is shown in Figure 3.3. In each state, ME is performed for the current MB in one of the search windows. The configure signal resets the FSM to state A based on the configuration, and the hardware starts the ME process for the next current MB.

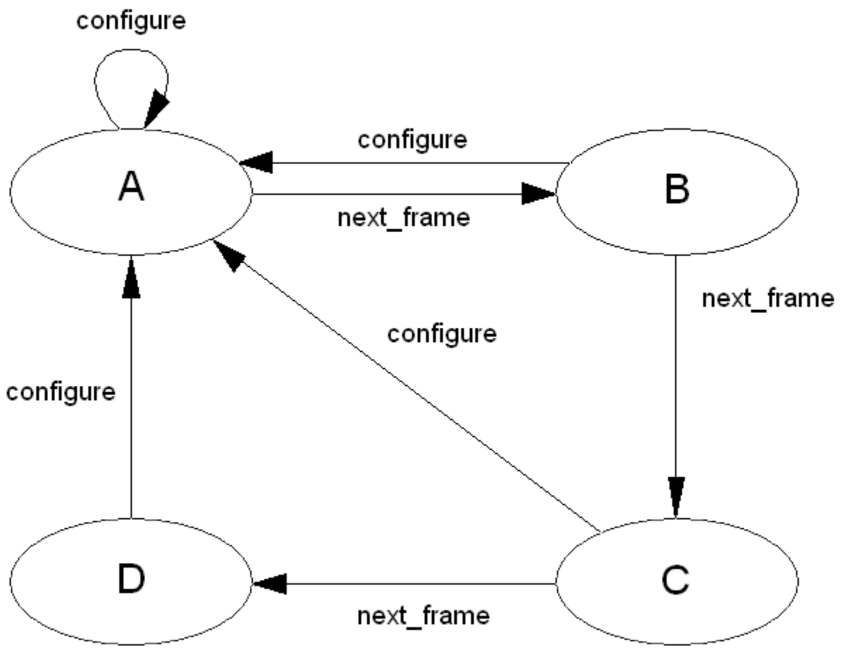

Figure 3.3 Top-Level FSM of Reconfigurable H.264 MRF Motion Estimation Hardware

3.1.2 Proposed Motion Compensation Hardware

Motion compensation hardware generates the predicted MB from the reference frames using the MV of the current MB. MRF VBS ME requires performing a tree structured block based motion compensation. Motion compensation hardware generates the read address, write address and write signals as the output. The read address is sent to search window memory of the best reference frame and the corresponding pixels are written into predicted MB register file.

Since the search range of the ME hardware is [-16, +15], the MVs are in the range [-16, +15]. However, in order to simplify the address generation in the motion compensation hardware, 16 is added to each MV before they are sent to the motion compensation hardware, making the input MVs in the range [0, +31]. In this way, the address generation for the predicted MB can be implemented using the formula (3.1).

$$Reference\_index = (48 \times Mv_y) + Mv_x + Offset \quad (3.1)$$

where Mv_x is the x component and Mv_y is the y component of the MV, *Offset* is the required offset value to find the address of the sub-block in the search window, and *Reference_index* is the address of the top-left corner pixel of the predicted MB in the search window.

An example for address generation is shown in Figure 3.4. The search window is a one dimensional 48 x 48 byte array since search range is [-16, +15]. If the prediction mode is 8x16 and both MVs are (0,0), the address of the top-left corner pixel of the upper 8x16 sub-block is 0. The address of the pixel right next to it is 1 and address of the pixel right below it is 48. The address of the top-left corner pixel of the lower 8x16 sub-block is 48 × 8 = 384 which is the *Offset* value for this sub-block. The offset values for different prediction modes are determined accordingly.

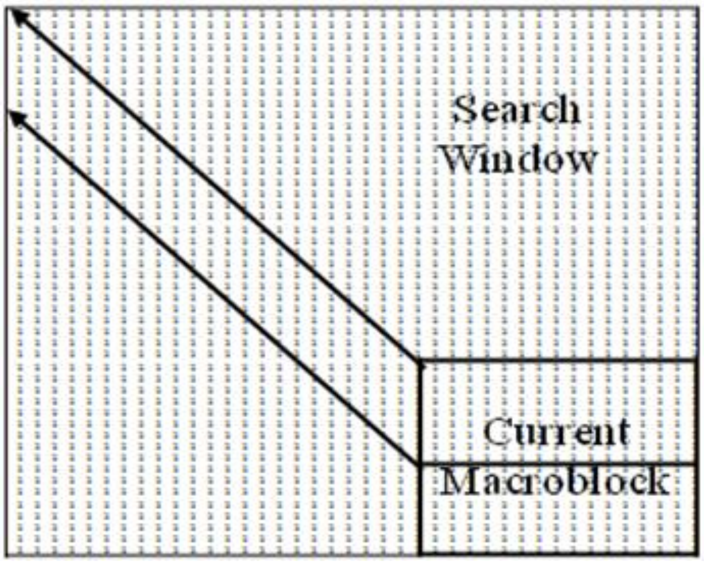

Figure 3.4 Motion Compensation of Current MB for 16x8 Mode

The proposed address generation hardware is shown in Figure 3.5. Its inputs are the mode information of the current MB and the x and y components of the MV. Its output is the address of the predicted block in the search window. The input MVs are processed sequentially and one output is generated in each clock cycle. The resulting hardware is, therefore, a low cost hardware consisting of 3 adders, 2 shifters and 1 counter. The offset value is determined by a lookup table which is controlled by the input mode and the position of the current sub-block inside the current MB. The counter counts up to generate the addresses of the consecutive pixels, jumps up to generate the address of the pixel in the next row or sets its value to the input position depending on the control signal.

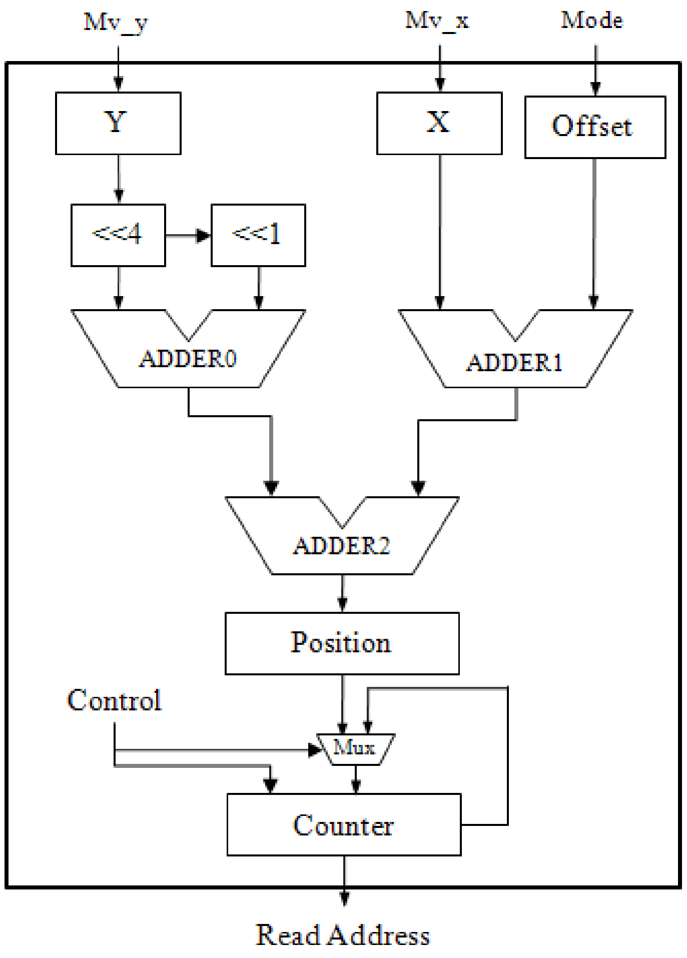

Figure 3.5 Motion Compensation Hardware

3.1.3 Proposed Mode Decision Hardware

Five different mode decisions are performed in H.264 video encoder hardware:

- Mode decision for 4x4 intra prediction modes
- Mode decision for 16x16 intra prediction modes
- Mode decision between 4x4 and 16x16 intra prediction modes
- Mode decision for MRF VBS ME modes
- Mode decision between intra and inter prediction

The first three mode decision hardware are presented in [10, 11]. The last two mode decision hardware are designed and implemented in this thesis. Since there are 7 different types of sub-blocks and 41 different sub-blocks, the best reference frame and sub-block partition for encoding a MB by MRF VBS ME should be selected. In H.264 reference software [24], the best sub-block partition minimizing the cost function (3.2) is selected.

$$J(m, \lambda_m) = SAD(s, c(m)) + \lambda_m \times R((m-p), ref) \qquad (3.2)$$

where J is the cost of a mode, $m = (mx, my)$ is the motion vector, $p = (px, py)$ is the prediction for motion vector, λ_m is the Lagrange multiplier, $R((m-p), ref)$ is the number of bits used to encode the motion vectors and best reference frame, s is the coded video signal, and c is the predicted video signal.

A low-cost hardware implementation of mode decision for MRF VBS ME is given in Figure3.6. The inputs of the mode decision hardware are the SAD values and x and y components of the MVs of all sub-blocks, and the reference frame number. Its outputs are the selected modes and the total cost. The inputs are processed sequentially. Therefore, the datapath has 4 adders, 1 shifter, 1 comparator and 3 LUTs. LUTs are used for calculating the R((m-p), ref) and shifter is used for multiplication with Lagrange multiplier.

The total cost is sent to the top level mode decision hardware which performs the mode decision between intra prediction and inter prediction, i.e. motion estimation. The top level mode decision hardware compares the costs of the intra prediction and inter prediction using a comparator, and selects the one with the smaller cost.

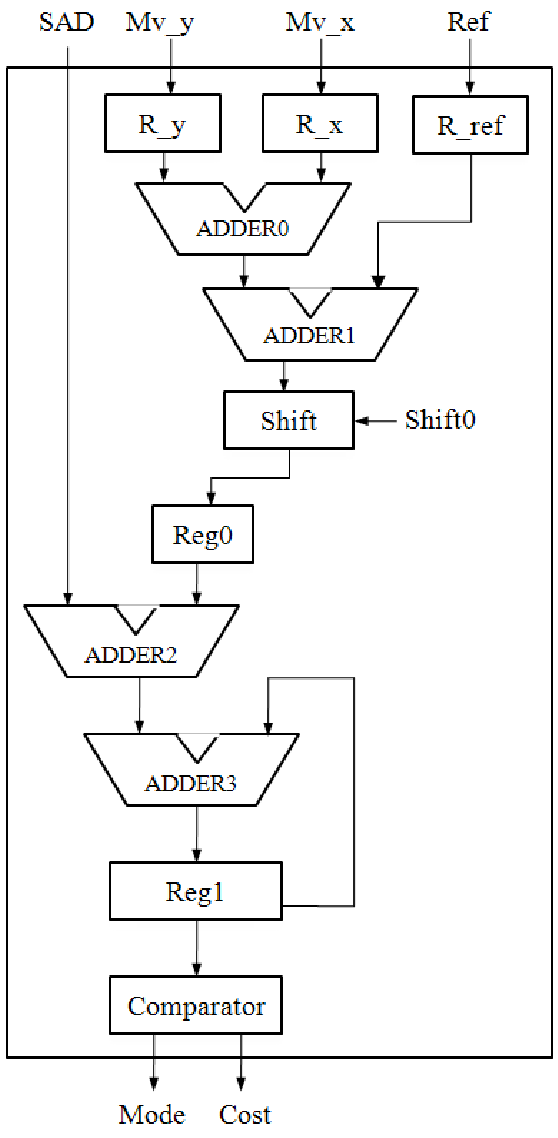

Figure 3.6 Mode Decision Hardware

3.1.4 Proposed Motion Vector Prediction Hardware

The MVs of the first MB in a frame are coded by entropy coder. However, in order to reduce the length of the resulting bitstream, for the other MBs the difference between the MV of a MB and the median of the MVs of its neighboring MBs are coded by entropy coder.

The neighboring MBs of a MB are shown in Figure3.7. MV prediction hardware calculates the median of the MVs of the MBs labeled A, B and C and subtracts it from the MV of the current MB. If the inter prediction modes of the neighboring MBs A, B, or C are different from 16x16, the median of the MVs of their corresponding sub-blocks are calculated. The medians of the x and y components of the MVs of the neighboring MBs are calculated separately. If MB A is not available, (0,0) MV is used instead. If both MBs B and C are not available, the MV of MB A is taken as the median MV. If MB C is not available, MV of MB D is used instead.

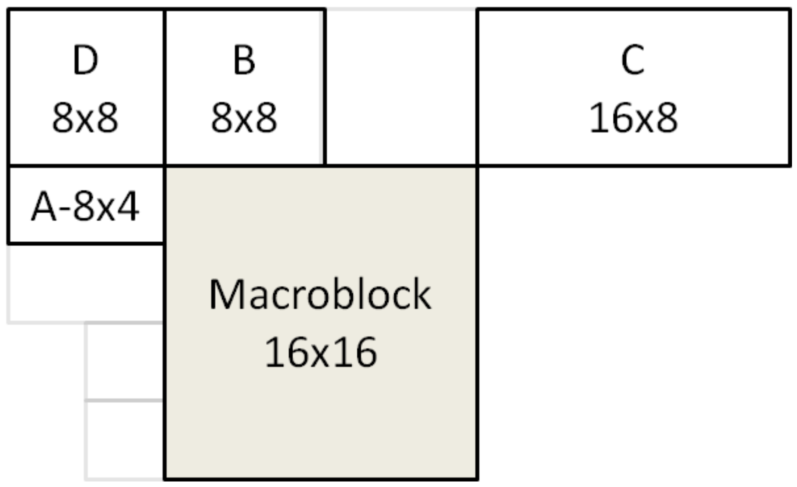

Figure 3.7 Neighboring Blocks used in Motion Vector Prediction

In the proposed MV prediction hardware, the MVs of all the MBs in a row of a video frame are stored in local memory. Although a MB can have up to 16 MVs, only MVs of the sub-blocks neighboring a below MB are stored. Depending on the inter prediction mode of a MB, up to 4 of its sub-blocks can be neighboring a below MB. However, the proposed hardware always stores 4 MVs, even if it means storing the MV of a larger sub-block multiple times. In this way, the proposed hardware can read the MVs of the neighboring MBs of the current MB without knowing the inter prediction modes of the neighboring MBs. Therefore, it avoids storing the mode information.

3.2 Implementation Results

The proposed H.264 video encoder hardware is implemented in Verilog HDL. The implementation is verified with RTL simulations. The Verilog RTL is then synthesized to Xilinx Virtex 6 FPGA. The resulting netlist is placed and routed to the same FPGA at 135 MHz under worst-case PVT conditions. The proposed H.264 video encoder hardware uses 14k slices in this FPGA. The FPGA slice usages of the modules in the H.264 video encoder hardware are shown in Figure 3.8. ME and Intra Prediction modules use most of the slices.

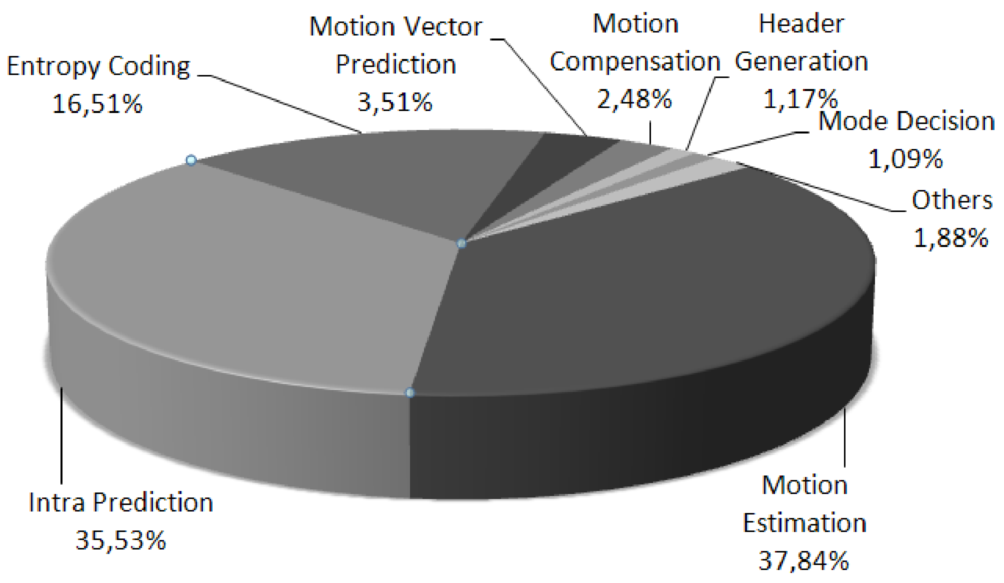

Figure 3.8 FPGA Slice Usages of H.264 Video Encoder Modules

The bitstreams generated by both the RTL and post place & route simulations of H.264 video encoder hardware for two Foreman video frames are successfully decoded by H.264 reference software decoder [24] and decoded frames are displayed using a YUV Player for visual verification.

The dynamic power consumption of the proposed H.264 video encoder hardware on this FPGA is estimated using Xilinx XPower tool at 50 MHz for CIF size Foreman video frames. It consumes 115mW, 153mW, 193mW, and 235mW when 1 reference frame, 2 reference frames, 3 reference frames and 4 reference frames are used respectively. The dynamic power consumptions of the modules in the H.264 video encoder hardware are shown in Figure 3.9. ME module consumes 66 - 89% of the total power depending on the number of reference frames used.

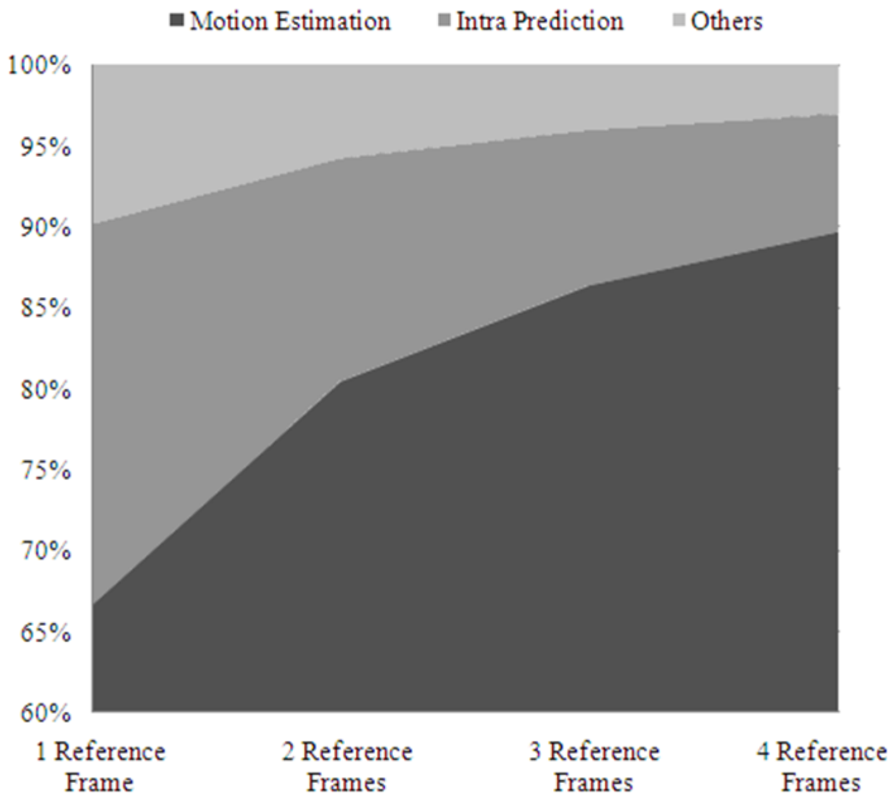

Figure 3.9 Power Consumptions of H.264 Video Encoder Modules

The schedule of the proposed H.264 video encoder hardware is shown in Figure 3.10. In the worst case, intra prediction search and mode decision takes 5151 clock cycles for a MB. ME and mode decision takes (1098+44) × *number of reference frames used for ME* clock cycles for a MB which is 4568 clock cycles in the worst case. Since ME works in parallel with intra prediction, the search & mode decision hardware takes 5151 clock cycles to process a MB.

The coder hardware in the H.264 intra frame coder takes 3680 clock cycles for a MB [11]. The coder hardware in the H.264 video encoder in addition performs motion compensation and deblocking filter. In the worst case, the deblocking filter hardware takes 5248 clock cycles to process a MB [26]. The coder hardware schedule for 4x4 intra prediction, 16x16 intra prediction and inter prediction are shown in Figures 3.11, 3.12, 3.13 respectively. The worst case occurs for the 16x16 intra prediction where the deblocking filter starts 880 clock cycles after the coder starts. Therefore, the coder hardware takes 6128 clock cycles for a MB.

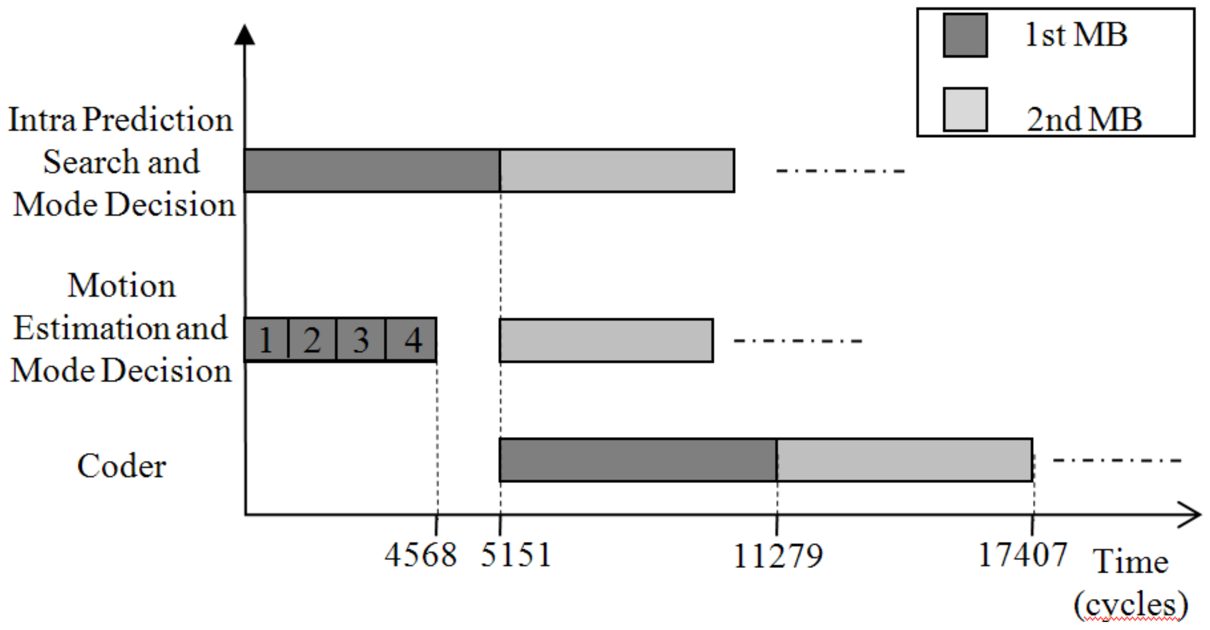

Figure 3.10 Scheduling of H.264 Video Encoder

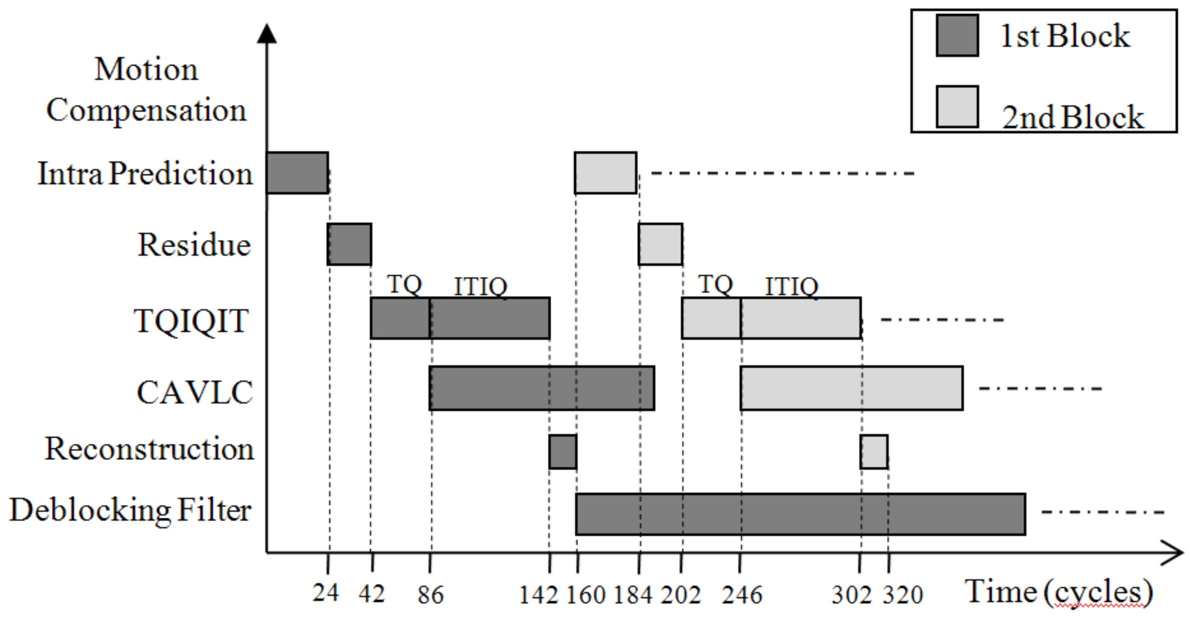

Figure 3.11 Scheduling of Coder - 4x4 Intra Prediction

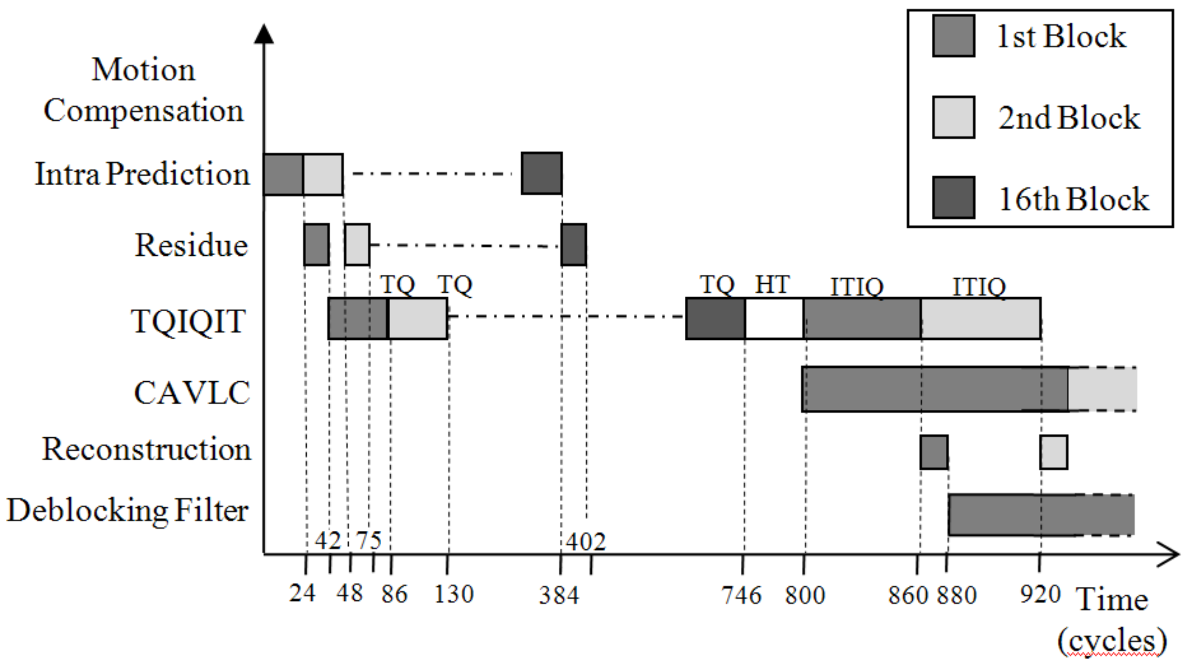

Figure 3.12 Scheduling of Coder - 16x16 Intra Prediction

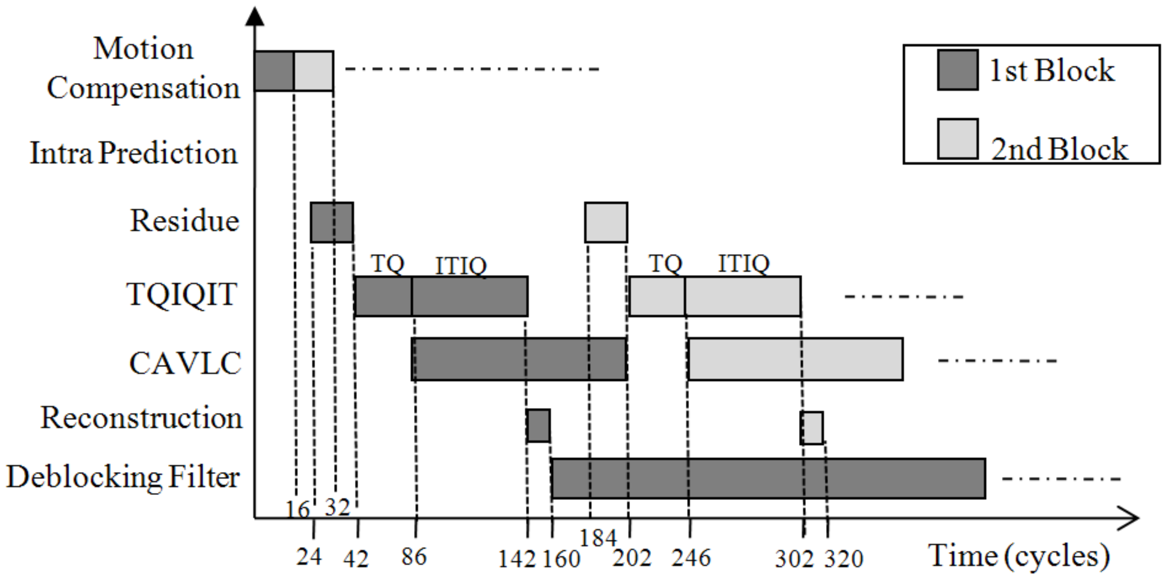

Figure 3.13 Scheduling of Coder - Inter Prediction

Since the execution of search & mode decision hardware and coder hardware are overlapped, the H.264 video encoder hardware takes 6128 clock cycles for a MB. Therefore, the FPGA implementation of the H.264 video encoder can process a CIF frame in 396 MB * 6128 clock cycles per MB * 7.4 ns clock cycle = 18 msec. Therefore, it can process 1000/18 = 55 CIF (352x288) frames per second.

As shown in Table 3.1, the H.264 video encoder hardware presented in [12, 13] achieve higher performance (720p HDTV and 1080p HDTV, respectively) than our hardware design at the expense of a much higher hardware cost and power consumption. The information for the entries shown as – in the table is not provided in the papers. Therefore, the proposed H.264 video encoder hardware is a better solution for portable consumer electronics products.

Table 3.1 Comparison with H.264 Video Encoders

	[12]	**[13]**	**Proposed**
Profile	Baseline	Baseline	Baseline
Frame Size	1280x720	1920x1080	352x288
Power Consumption (mW)	785	1409	115-235
Technology	ASIC	ASIC	FPGA
Area			
ME - Search Range	128x64	192x128	16x16
ME - PEs	128x8	64x32	256
Intra Prediction - Datapaths	4	4	3
Transform - Inv.Transform (Adders & Internal Reg. Files)	16 & 16	–	3 & 6
Residue - Subtractors	4	–	1

CHAPTER IV

CONCLUSIONS AND FUTURE WORK

In this thesis, we proposed a low complexity H.264 MRF ME algorithm and a low energyadaptive hardware for its real-time implementation. The proposed MRF ME algorithm reduces the computational complexity of MRF ME by using a dynamically determined number of reference frames for each MB and early termination.

We designed a serial and a parallel H.264 MRF VBS ME hardware implementing the proposed low complexity MRF ME algorithm. The serial ME hardware has 256 PEs and it searches one reference frame at a time. The parallel ME hardware has 512 PEs and it searches two reference frames in parallel. We also designed an H.264 MRF VBS ME hardware which uses 5 reference frames for all MBs (Ref-5). This ME hardware has 256 PEs and it implements FS algorithm.

All three MRF ME hardware are implemented in Verilog HDL and mapped to a low cost Xilinx Spartan 6 FPGA using Synopsys Synplify synthesis tool and Xilinx ISE 11.4 place and route tool at 58 MHz under worst case PVT conditions.The Ref-5, serial and parallel MRF ME hardware implementations use 5k, 5.5k, 10.2k slices, and they can process9, between 11 and 45, between 22 and 45 VGA fps, respectively on this FPGA.

The power consumptions of these MRF ME hardware implementations on this FPGA are estimated at 50MHz using Xilinx XPower 11.4 tool for a CIF size Foreman video frame. The results showed that the serial MRF ME hardware has 29-72% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss, and the parallel MRF ME hardware has 21-64% less energy consumption on this FPGA than Ref-5 hardware with a negligible PSNR loss.

We then proposed a reconfigurable baseline H.264 video encoder hardware in which the number of reference frames used for MRF ME can be configured based on the application requirements in order to trade-off video coding efficiency and power consumption. The proposed video encoder hardware is based on an existing low cost H.264 intra frame coder hardware [10, 11] and it includes new reconfigurable MRF ME, mode decision and motion compensation hardware.

The proposed H.264 video encoder hardware is implemented in Verilog HDL and mapped to Xilinx Virtex 6 FPGA. The FPGA implementation is verified with post place & route simulations. The bitstream generated by the proposed video encoder hardware for an input frame is successfully decoded by H.264 Joint Model reference software decoder and the decoded frame is displayed using a YUV Player tool for visual verification. The FPGA implementation works at 135 MHz, it can code 55 CIF (352x288) frames per second, and its power consumption ranges between 115mW and 235mW depending on the number of reference frames used for MRF ME.

The H.264 video encoder hardware presented in [12, 13] achieve higher performance (720p HDTV and 1080p HDTV, respectively) than our hardware design at the expense of a much higher hardware cost and power consumption. The proposed H.264 video encoder hardware is a better solution for portable consumer electronics products.

As future work, half-pixel and quarter-pixel accurate motion estimation can be added to the proposed H.264 video encoder hardware. Low complexity motion estimation algorithms such as diamond search and hexagon search can be used in the proposed H.264 video encoder hardware and their effect on the power consumption can be analyzed. The proposed H.264 video encoder hardware can be implemented as an ASIC and prototypes can be fabricated.

REFERENCES

[1] R. Schäfer, T. Wiegand and H. Schwarz, “The Emerging H.264/AVC Standard”, *EBU Technical Review*, January 2003.

[2] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra “Overview of the H.264/AVC Video Coding Standard”, *IEEE Trans. on Circuits and Systems for Video Technology vol.* 13, no. 7, pp. 560–576, July 2003.

[3] I. Richardson, H.264 and MPEG-4 Video Compression, Wiley, 2003.

[4] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, Kluwer Academic Publishers, 2nd Edition, 1997.

[5] C. Wei, H. Hui, T. Jiarong, and M. Hao, “A High-performance Reconfigurable VLSI Architecture for VBSME inH.264,” *IEEE Trans. on Consumer Electronics*, vol. 54, no. 3, pp. 1338-1345, Aug. 2008.

[6] T. Moorthy, and A. Ye, “A Scalable Computing and Memory Architecture for Variable Block Size Motion Estimation on Field-Programmable Gate Arrays,” *International Conference on Field Programmable Logic*, pp. 83-88, Sept. 2008.

[7] K. Lee, G. Jeon, and J. Jeong, “Fast Reference Frame Selection Algorithm for H.264/AVC,” *IEEE Trans. on Consumer Electronics*, vol. 55, pp. 773–779, May 2009.

[8] G. Stewart, D. Renshaw, and M. Riley, “A Novel Motion Estimation Power Reduction Technique,” *International Conference on Field Programmable Logic*, pp. 546–549, August 2007.

[9] S. Yalcin, H. F. Ates and I. Hamzaoglu, “A High Performance Hardware Architecture for an SAD Reuse based Hierarchical Motion Estimation Algorithm for H.264 Video Coding”, *International Conference on Field Programmable Logic*, August 2005.

[10] E. Sahin, and I. Hamzaoglu, “An Efficient Hardware Architecture for H.264 Intra Prediction Algorithm,” *DATE Conference,* April 2007.

[11] I. Hamzaoglu, O. Tasdizen, and E. Sahin, “An Efficient H.264 Intra Frame Coder System,” *IEEE Trans. on Consumer Electronics*, vol. 54, no. 4, November 2008.

[12] T. C. Chen, et.al., “Analysis and Architecture Design of an HDTV720p 30 Frames/s H.264/AVC Encoder,” *IEEE Trans. on Circuits and Systems for Video Technology,* vol. 16, no.4, pp. 507-522, April 2006.

[13] Z. Liu, et.al., “HDTV1080p H.264/AVC Encoder Chip Design and Performance Analysis,” *IEEE Journal of Solid-State Circuits,* vol. 44, no.2, pp. 594-608, February 2009.

[14] Y. Su and M.-T. Sun, “Fast multiple reference frame motion estimation for H.264/AVC,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 447–452, March 2006.

[15] Wu, P., Xiao, C.-B, An adaptive fast multiple reference frames selection algorithm for H.264/AVC, *IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 1017–1020, April 2008.

[16] L. Shen, Z. Liu, Z. Zhang, and G. Wang, “An Adaptive and Fast Multiframe Selection Algorithm for H.264 Video Coding,” *IEEE Signal Processing Letters,* vol. 14, pp. 836-839, Nov. 2007.

[17] H. Li, C. Hsu and M. Chen, “Fast Multiple Reference Frame Selection Method for Motion Estimation in JVT H.264”, *IEEE Asia-Pacific Conference on Circuits and Systems,* December 2004.

[18] SW Ho, SD Kim, MH Sunwoo, “Fast multiple reference frame selection methods for H. 264/AVC”, *IEEE Workshop on Signal Processing Systems,* 2008.

[19] Y. Shen and C. Huang, “Fast multi-frame motion estimation algorithm in H.264”, *ICSP 2004.*

[20] C.W. Ting, L. M. Po, and C. H. Cheung, “Center-biased frame selection algorithms for fast multi-frame motion estimation in h.264,” *IEEE International Conference on NeuralNetworks and Signal Processing*, pp. 1258–1261, December 2003.

[21] M E Al-Mualla, C N Canagarajah, D R Bull, “Simplex Minimization for Single- and Multi-Reference Motion Estimation*”, IEEE Trans. on Circuits and Systems for Video Technology,* Vol 11, No 12, pp 1209-1220, December 2001.

[22] T.-C. Chen, C.-Y. Tsai, Y.-W. Huang, and L.-G. Chen, “Single Reference Frame Multiple Current Macroblocks Scheme for Multiple Reference Frame Motion Estimation in H.264/AVC,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 17, no. 2, pp. 242-247, February 2007.

[23] K. Y. Min, J. W. Chong “A novel frame reordering scheme and a high speed VLSI architecture of multiple reference frame motion estimator for H.264/AVC”, *IEEE Trans. on Consumer Electronics*, pp. 2394-2400, November 2009.

[24] Joint Video Team (JVT) of ITU-T VCEG and ISO/IEC MPEG, *Joint Model (JM) Reference Software Version 14.2*, http://bs.hhi.de/suehring/.

[25] C. Kalaycioglu, O. Ulusel and I. Hamzaoglu, “Low Power Techniques for Motion Estimation Hardware”, *Int. Conference on FPL*, pp. 180–185, September 2009.

[26] M. Parlak, and I. Hamzaoglu, “Low Power H.264 Deblocking Filter Hardware Implementations,” *IEEE Trans. on Consumer Electronics*, vol. 54, no. 2, May 2008.

[27] O. Tasdizen, I. Hamzaoglu, “A High Performance And Low Cost Hardware Architecture for H.264 Transform And Quantization Algorithms", *13th European Signal Processing Conference*, September 2005.