H.264 Motion Estimation Engine v1.0

DS648 April 23, 2008 **Product Specification**

Introduction

The H.264 Motion Estimation Engine Version 1.0 is a fully functional netlist implemented on a Xilinx® FPGA. The Motion Estimation core accepts input parameters and macroblocks and generates output motion vectors and Sum of Absolute (SAD) values in accordance with the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG) as the product of a collective partnership effort known as the Joint Video Team (JVT). The collaborative effort is also known as H.264/AVC/MPEG4 Part 10.

Features

- H.264/MPEG-4 Part10 Baseline/Main/High Profiles at Level 4.2
- Compliant with International Standard ISO/IEC 14496-10:2005 (E) Rec. H.264 (E)
- 1080i@60 fields per second or 1080p@30 frames per second operation at 275 MHz
- 720p@30 frames per second operation at 130 MHz
- Integrated variable block pattern generation
- 8x4 blocks search for full pel locations
- 112 x 128 search range
- 120 candidate positions (10 predictors with 4 x 3 region around each predictor)
- External memory bandwidth 1.65 Gbps (720p@30) to 3.74 Gbps (1080i@60)
- Selectable reference frame at frame update through frame pointer parameter
- User input prediction vectors
- Optional output port for all SAD values and Motion Vectors

LogiCORE™ Facts Core Specifics	
Supported Device Families	Virtex®-5, Virtex-4, Spartan®-3A DSP
Resources Used	
Virtex-5 FPGA	2952 LUTs, 3040 FFs, 26 DSP48s, 15 block RAMs
Virtex-4 FPGA	2954 LUTs, 3252 FFs, 30 DSP48s, 19 block RAMs
Spartan-3A DSP FPGA	3297 LUTs, 3274 FFs, 30 DSP48s, 19 block RAMs
Provided with Core	
Documentation	Data Sheet, User Guide
Design File Formats	EDIF
Instantiation Template	VHDL Wrapper
Design Tool Requirements	
Synthesis	Synplicity Synplify_Pro 8.8.04
Xilinx Implementation Tools	Xilinx ISE™ 9.2.01i
Verification	VHDL Test Bench ModelSim® 6.1c SE, MicroSoft Studio v6.0, ActivePerl 5.8.3
Simulation	ModelSim 6.1c SE
Support	
Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing functionality, or support of product if implemented in devices not listed in the documentation, or if customized beyond that allowed in the product documentation, or if any changes are made in sections of the design marked DO NOT MODIFY.	

Applications

The H.264 Motion Estimation core can be utilized in H.264 standards encoding applications where hardware acceleration is needed to achieve real time operation. Typical video applications are video surveillance, video conferencing, and video broadcast. Other video encoding applications where 8x4 block searches are desired can also use this core.

Functional Overview

The H.264 Motion Estimation core computes the sum of absolute difference (SAD) for a set of 120 search locations within a 112 x 128 search window for 8x4 blocks. The search locations are determined by a set of 10 seeds that are provided by the user and the 4 x 3 region to the right and down from each seed. The core provides as output the 120 SAD calculated values and the motion vectors. In addition, the coded block pattern is computed for a macroblock and the best motion vector for each sub-block is provided.

The functional inputs and outputs to the H.264 Motion Estimation Core are:

- Inputs:
  - New Macroblock
  - Parameter for the New Macroblock
    - MB location (h, v)
    - MV predictors
  - Reference Frame pointer
  - Reference Frame
- Outputs
  - Output Parameters
    - Best MV
    - Motion Vectors
    - SAD values
    - VBS pattern

Motion estimation (ME) requires the associated portion of the reconstructed frame to be available before any processing can start. The Reference frame is stored in the external memory and is accessible through an external memory controller. One port is reserved for writing the reconstructed frame, and the user is responsible for loading it prior to starting the motion estimation process. The second port is reserved for the motion estimation reading the necessary data from the external memory. Figure 1 shows the H.264 Motion Estimation architecture.

The motion estimation is delivered as a netlist. The Motion Estimation core requires the user to provide a list of initial motion vectors where the searches will be performed. The Motion Estimation module is responsible for performing the searches around the provided location, as well as for the search area data management. The motion estimation algorithm uses the input information such us macroblock location for processing the current macroblock.

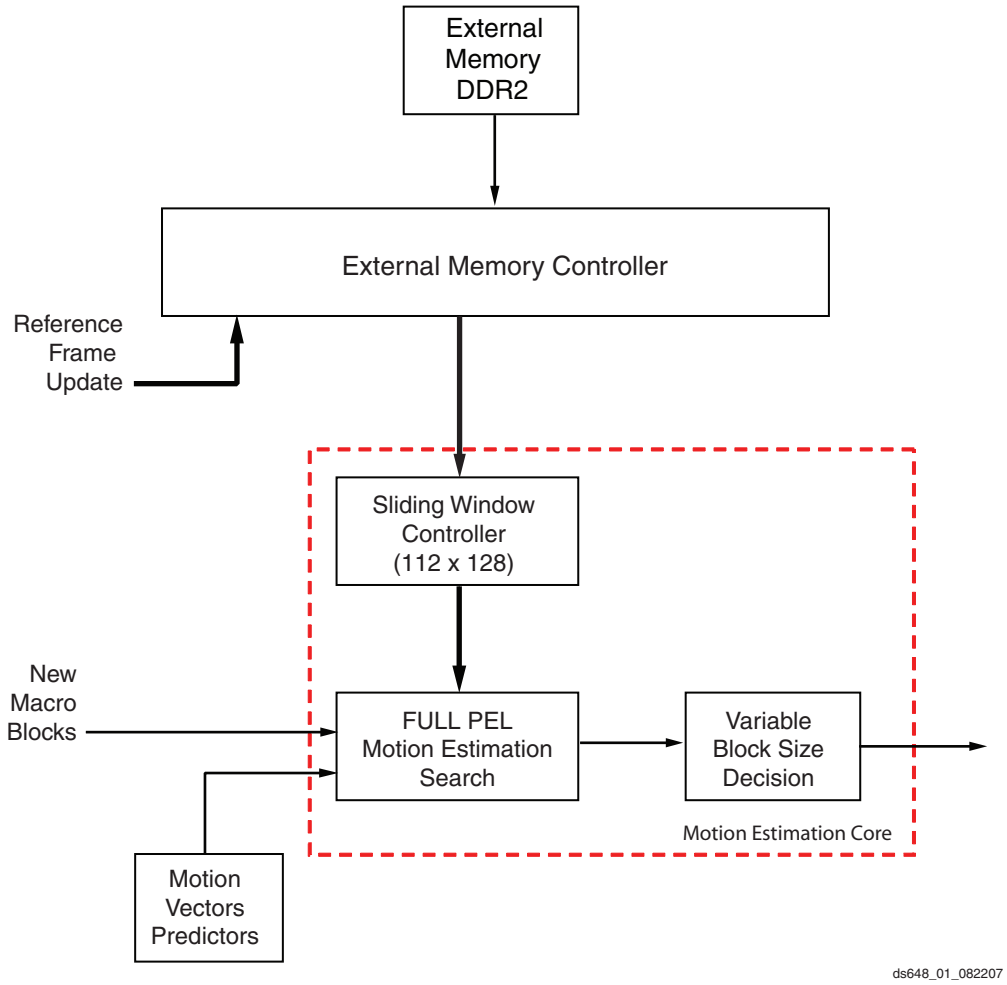

Figure 1: **H.264 Motion Estimation Architecture**

Performance

Target clock FMax and subsequent macroblock throughput are summarized in Table 1.

Table 1: **Performance Summary**

FPGA Family	Clock FMax	Macroblock Throughput (Macroblocks/s)	External Memory Bandwidth (Gbps)	Notes
Spartan-3A DSP speed grade -4	130 MHz	108,000	1.65	Supports up to 720P@30
Virtex-4 speed grade -12	225 MHz	200,000	3.1	Supports up to 720P@50
Virtex-5 speed grade -3	275 MHz	244,800	3.74	Supports up to 1080P@30, 1080i@60

System Overview

The H.264 Motion Estimation core is intended to be used in a video encoding setting where motion estimation is key to system performance. The motion estimation core requires connection to an external frame buffer to perform block matching and provide the SAD value to be used in rate distortion criteria for best mode selection between inter and intra modes in an H.264 encoder. The system diagram of Figure 2 gives a high-level view of how the core should be connected into the encoder system. The operation of the core proceeds with first the reference frame being written to external memory. Upon completion of writing the reference frame, the first macroblock can be written to the core. This triggers a frame update where frame parameters are read into the core for processing of the frame. The core processes macroblock-by-macroblock with a latency of six macroblocks.

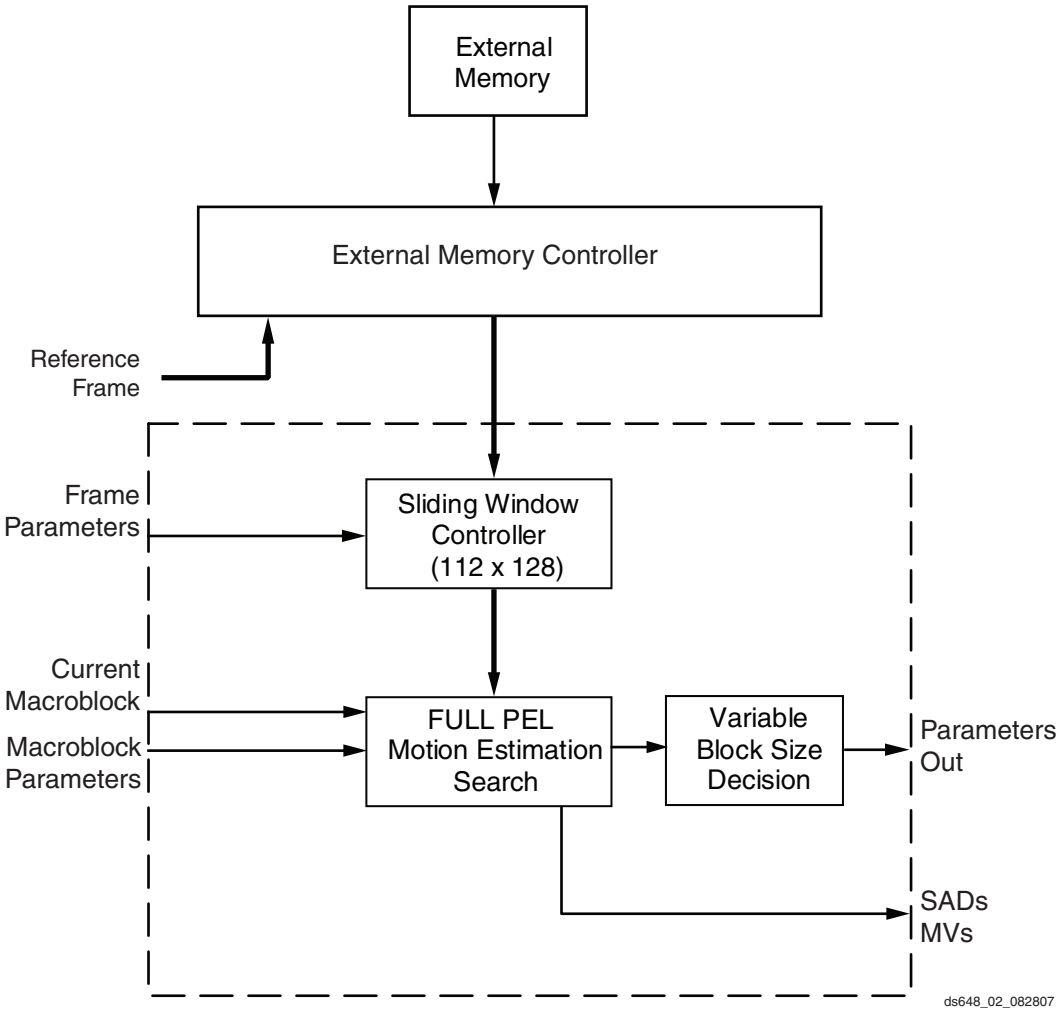

Figure 2: **Motion Estimation Functional Block Diagram**

H.264 Motion Estimation Interface

Figure 3 shows the I/O diagram. Table 2 lists the interface signals.

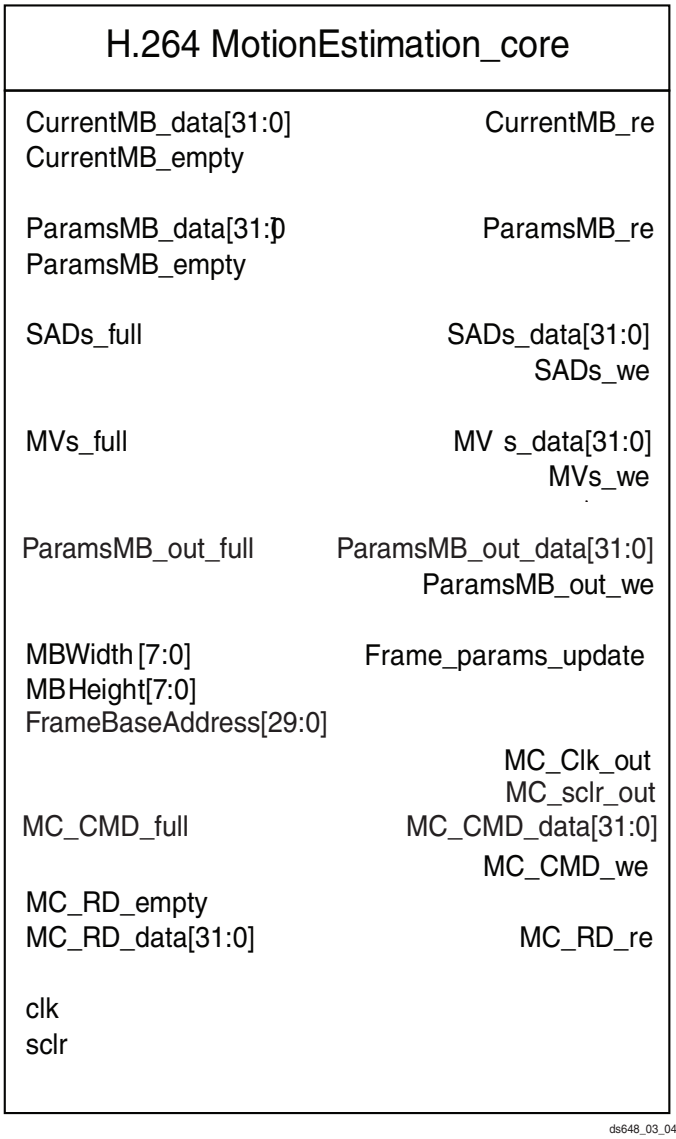

Figure 3: **H.264 Motion Estimation I/O Diagram**

Table 2: **H.264 Motion Estimation Interface Signals**

Name	Direction	Description
clk	Input	All systems and interface operations are synchronous to this clock.
sclr	Input	Active High clear signal that resets all internal states to a known state.
Inputs on Macroblock Basis		
CurrentMB_data[31:0]	Input	Macroblock input data; format described in Figure 4.
CurrentMB_empty	Input	Active High input to signal there is no more data to read for Current MB.
CurrentMB_re	Output	Read enable for reading Current Macroblock data.
ParamsMB_data	Input	Parameter input data; format described in Table 3.
ParamsMB_empty	Input	Active High input to signal there is no more data to read for Parameter MB.
ParamsMB_re	Output	Read enable for reading Parameter Macroblock data.
Outputs on Macroblock Basis		
SADs_data[31:0]	Output	SADs output data.
SADs_we	Output	Write enable for SADs data.
SADs_full	Input	Full flag for SADs external to core FIFO.
MVs_data[31:0]	Output	MVs output data.
MVs_we	Output	Write enable for MVs data.
MVs_full	Input	Full flag for MVs external to core FIFO.
ParamsMB_out_data	Output	Parameter output data; format described in Table 4.
ParamsMB_out_we	Output	Write enable for Parameter output data.
ParamsMB_out_full	Input	Full flag for Parameter output external to core FIFO.
Frame Based Signals		
MBWidth[7:0]	Input	Number of Macroblocks in the current frame width.
MBHeight[7:0]	Input	Number of Macroblocks in the current frame height.
FrameBaseAddress[29:0]	Input	Pointer to external frame.
Frame_params_update	Output	Signal for indicating the frame based parameters have been updated and pulled into the core.
External Memory Controller Signals		
MC_Clk_out	Output	Clock for external memory controller, synchronous with input core clock.
MC_sclr_out	Output	Synchronous High reset for the external memory controller.
MC_CMD_data[31:0]	Output	External memory controller command data bus.
MC_CMD_we	Output	Write enable for the external memory controller command.
MC_CMD_full	Input	Full signal for command bus.
MC_RD_data[31:0]	Input	External memory controller read data bus.
MC_RD_re	Output	Read enable for the external memory controller read bus.
MC_RD_empty	Input	Empty signal for external memory controller read bus.

Data Formats and Timing

Input and output formats for Macroblock based interfaces are FIFO handshaking with read enable and write enable for respective inputs and outputs. Data processing begins after the first Macroblock and associated parameters have been input into the core. The data format for the CurrentMB is shown in Figure 4.

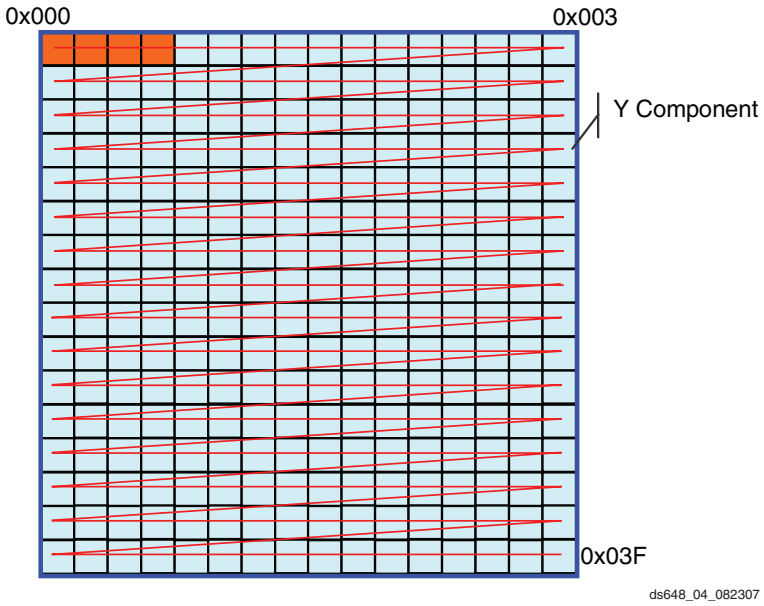

Figure 4: **Macroblock Linear Addressing for CurrentMB**

The data format for the ParamsMB input is shown in Table 3 with the 8x4 block ordering shown in Figure 5.

Table 3: **Parameter Input Data Ordering (PMV = Predicted Motion Vector)**

Location	ParamsMB_data[31:16]	ParamsMB_data[15:0]
0	MBy	MBx
1	Open	Open
2	PredMV_block0[0]y	PredMV_block0[0]x
3	PredMV_block0[1]y	PredMV_block0[1]x
4	PredMV_block0[2]y	PredMV_block0[2]x
5	PredMV_block0[3]y	PredMV_block0[3]x
6	PredMV_block0[4]y	PredMV_block0[4]x
7	PredMV_block0[5]y	PredMV_block0[5]x
8	PredMV_block0[6]y	PredMV_block0[6]x
9	PredMV_block0[7]y	PredMV_block0[7]x
10	PredMV_block0[8]y	PredMV_block0[8]x
11	PredMV_block0[9]y	PredMV_block0[9]x
12-21	PredMV_block1[0-9]y	PredMV_block1[0-9]x
22-31	PredMV_block2[0-9]y	PredMV_block2[0-9]x
32-41	PredMV_block3[0-9]y	PredMV_block3[0-9]x

Table 3: **Parameter Input Data Ordering (PMV = Predicted Motion Vector) *(Cont'd)***

Location	ParamsMB_data[31:16]	ParamsMB_data[15:0]
42-51	PredMV_block4[0-9]y	PredMV_block4[0-9]x
52-61	PredMV_block5[0-9]y	PredMV_block5[0-9]x
62-71	PredMV_block6[0-9]y	PredMV_block6[0-9]x
72-81	PredMV_block7[0-9]y	PredMV_block7[0-9]x

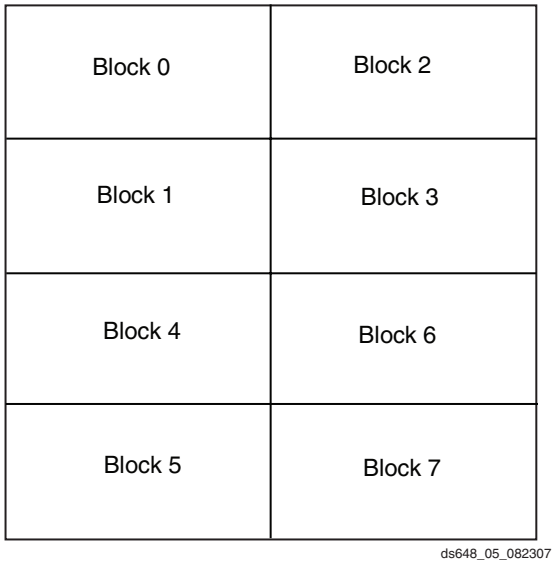

Figure 5: **Parameter Block Ordering**

The output data format for SADs_data output follow block ordering of Figure 5 with 120 SADs per 8x4 block. There are 8 blocks of 8x4, thus 960 SAD values per Macroblock. The MVs_data follows the same data structure with 960 motion vector pairs per Macroblock.

The ParamsMB_out_data is given in Table 4 with 4x4 block ordering described in Figure 6.

Table 4: **Parameter Output Data Ordering**

Location	ParamsMB_out_data[31:16]	ParamsMB_out_data[15:0]
0	MBy	MBx
1	Open	Open
2	PredMV_4x4block[0]y	PredMV_4x4block[0]x
3	PredMV_4x4block[1]y	PredMV_4x4block[1]x
4	PredMV_4x4block[2]y	PredMV_4x4block[2]x
5	PredMV_4x4block[3]y	PredMV_4x4block[3]x
6	PredMV_4x4block[4]y	PredMV_4x4block[4]x
7	PredMV_4x4block[5]y	PredMV_4x4block[5]x
8	PredMV_4x4block[6]y	PredMV_4x4block[6]x
9	PredMV_4x4block[7]y	PredMV_4x4block[7]x
10	PredMV_4x4block[8]y	PredMV_4x4block[8]x
11	PredMV_4x4block[9]y	PredMV_4x4block[9]x

Table 4: **Parameter Output Data Ordering *(Cont'd)***

Location	ParamsMB_out_data[31:16]	ParamsMB_out_data[15:0]
12	PredMV_4x4block[10]y	PredMV_4x4block[10]x
13	PredMV_4x4block[11]y	PredMV_4x4block[11]x
14	PredMV_4x4block[12]y	PredMV_4x4block[12]x
15	PredMV_4x4block[13]y	PredMV_4x4block[13]x
16	PredMV_4x4block[14]y	PredMV_4x4block[14]x
17	PredMV_4x4block[15]y	PredMV_4x4block[15]x
18-33	Best_MVs_4x4block[0-15]y	Best_MV_4x4block[0-15]x
34-49	Best_SAD_4x4block[0-15]	Best_SAD_4x4block[0-15]

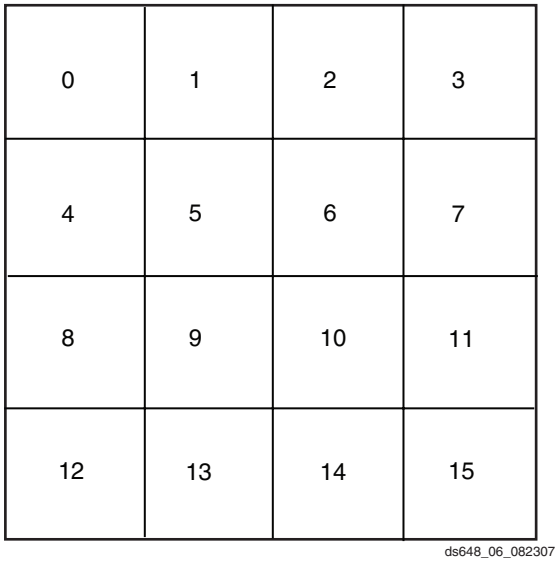

Figure 6: **4 x 4 Block Ordering**

The frame-based signals of MBWidth, MBHeight, and FrameBaseAddress should be valid when the first Macroblock is input to the core. The Frame_params_update signal indicates to the user that the values have been taken internally to the core. This signal is driven at the end of the first Macroblock being put into the core. During the operation of motion estimation for a given frame, the frame-based signals are ignored until the next frame time when the first macroblock of the new frame is put into the core. This allows for updating these three values anytime after the first macroblock is put into the core and the Frame_params_update signal is given until the first macroblock of the next frame is put into the core.

External Memory Controller signals are used to drive an external memory controller for reference frame use during the Motion Estimation operation. The command bus (CMD) is used to give 4-word commands for rectangular region access in the reconstructed frame. The reconstructed frame is stored in raster scan order with 4 pixels per word and luminance only. The access to a rectangular region is through 4 command words, C0 is X size, C1 is the address, C2 is Y size, C3 is the horizontal resolution of the frame store.

After a valid 4-word sequence is given as a command to the external memory controller, the MC_RD_empty signal indicates that there is data ready to be read into the Motion Estimation core and

the core uses the read enable signal MC_RD_re to move data into the core. This is a standard FIFO handshake.

See Figure 7 and Figure 8.

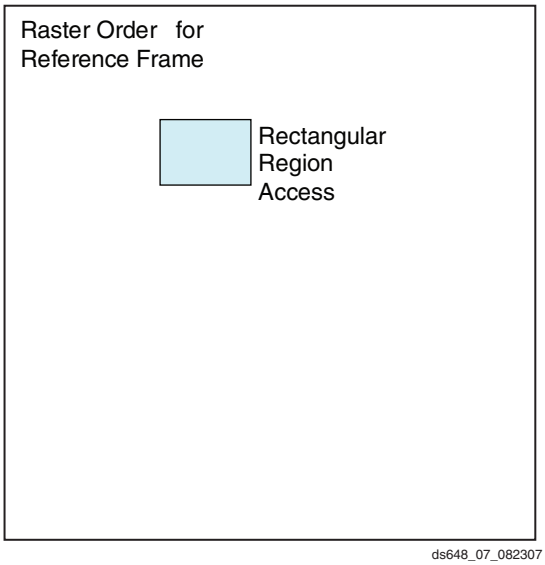

Figure 7: **Reference Frame and Region Access**

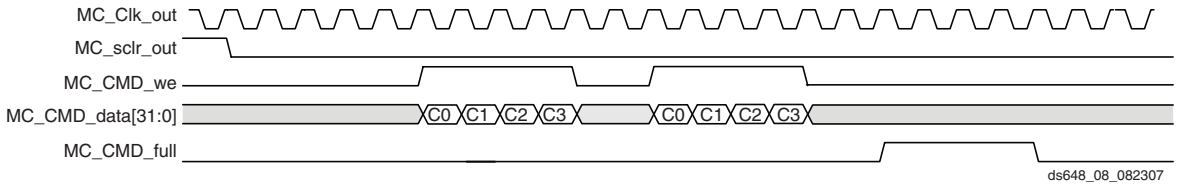

Figure 8: **External Memory Controller Command Timing Diagram**

Parameters

Macroblock-Based Parameters

Macroblock-based input parameters are the set of parameters that are sent to the H.264 Motion Estimation core through the ParamsMB_data FIFO interface. For the proper operation of the core, a new set of parameters must be sent with each macroblock. The parameters consist of the MBx and MBy Macroblock location of the CurrentMB and a set of 10 predicted motion vectors for each 8x4 block in the Macroblock. See Table 5.

Table 5: **Macroblock-Based Input Parameters**

Parameter	Description
MBx(15:0):	X location of current macroblock.
MBy(31:16):	Y location of current macroblock.
Open(31:0):	This location open.
PredMV_block0[15:0]x:	Predicted x motion vector for 8x4 block 0.
PredMV_block0[31:16]y:	Predicted y motion vector for 8x4 block 0.
PredMV_block1[15:0]x:	Predicted x motion vector for 8x4 block 1.
PredMV_block1[31:16]y:	Predicted y motion vector for 8x4 block 1.
PredMV_block2[15:0]x:	Predicted x motion vector for 8x4 block 2.
PredMV_block2[31:16]y:	Predicted y motion vector for 8x4 block 2.
PredMV_block3[15:0]x:	Predicted x motion vector for 8x4 block 3.
PredMV_block3[31:16]y:	Predicted y motion vector for 8x4 block 3.
PredMV_block4[15:0]x:	Predicted x motion vector for 8x4 block 4.
PredMV_block4[31:16]y:	Predicted y motion vector for 8x4 block 4.
PredMV_block5[15:0]x:	Predicted x motion vector for 8x4 block 5.
PredMV_block5[31:16]y:	Predicted y motion vector for 8x4 block 5.
PredMV_block6[15:0]x:	Predicted x motion vector for 8x4 block 6.
PredMV_block6[31:16]y:	Predicted y motion vector for 8x4 block 6.
PredMV_block7[15:0]x:	Predicted x motion vector for 8x4 block 7.
PredMV_block7[31:16]y:	Predicted y motion vector for 8x4 block 7.

Macroblock-based output parameters are the set of parameters that are provided from the H.264 Motion Estimation core through the ParamsMB_out_data FIFO interface. The Predicted motion vector, best motion vectors, and SADs are provided through 16 4x4 block values. See Table 6.

Table 6: **Macroblock-Based Output Parameters**

Parameter	Description
MBx(15:0):	X location of output macroblock.
MBy(31:16):	Y location of output macroblock.
Open(31:0):	This location open.
PredMV_4x4block[15:0]x: 16 locations	Predicted x motion vectors for 4x4 blocks.
PredMV_4x4block[31:16]y: 16 locations	Predicted y motion vectors for 8x4 blocks.
Best_MVs_4x4block[15:0]x: 16 locations	Best x motion vectors for 4x4 blocks.
Best_MVs_4x4block[31:16]y: 16 locations	Best y motion vectors for 4x4 blocks.
Best_SAD_4x4block[31:0]: 16 locations	Best SAD for 4x4 blocks.

Image-Based Parameters

Image-based parameters are signals provided to the core and synchronized with the Frame_params_update signal. See Table 7. Frame synchronization is achieved through controlling when the first Macroblock is put into the core.

Table 7: **Image-Based Parameters**

Parameter	Description
MBWidth[7:0]:	Number of macroblocks across one frame of current image.
MBHeight[7:0]:	Number of macroblocks from top to bottom of one frame of current image.
FrameBaseAddress[29:0]:	Physical address of the start of the frame buffer for the reference frame.
Frame_params_update:	Signal indicating the frame-based parameters have been taken inside the core.

The order of operations for core operation is:

1. Send the reference frame to external memory.
2. Set the values of MBWidth, MBHeight, and FrameBaseAddress.
3. Send the first input Macroblock parameters and first Macroblock. This results in Frame_params_update going active for one cycle and the parameters being moved into the Motion Estimation Core.
4. Send new Macroblocks in Macroblock raster order through the whole frame.
5. After the last Macroblock is sent to the core and the results are read at the output, send the next reference frame and proceed.

Core Module Functional Descriptions

The main computational engine of the Motion Estimation core is the Full Pel search engine. To keep the engine operating efficiently enough to make the macroblock throughput, both the external memory bandwidth and the input of macroblocks to the core must be sufficient. The Sliding Window controller is used to aid with the external memory bandwidth. There are two other blocks that work in conjunction with the Full Pel search engine. The variable block size design is a post processing step and the motion vector prediction is a pre-processing step to the search engine. See Figure 1.

Full Pel Search Engine

The full pel search engine is a block of 32 SAD engines that continuously compute for a given motion vector seed the 12 SADs in the 4x3 region to the right and down from the seed motion vector (predicted motion vector). See Figure 9.

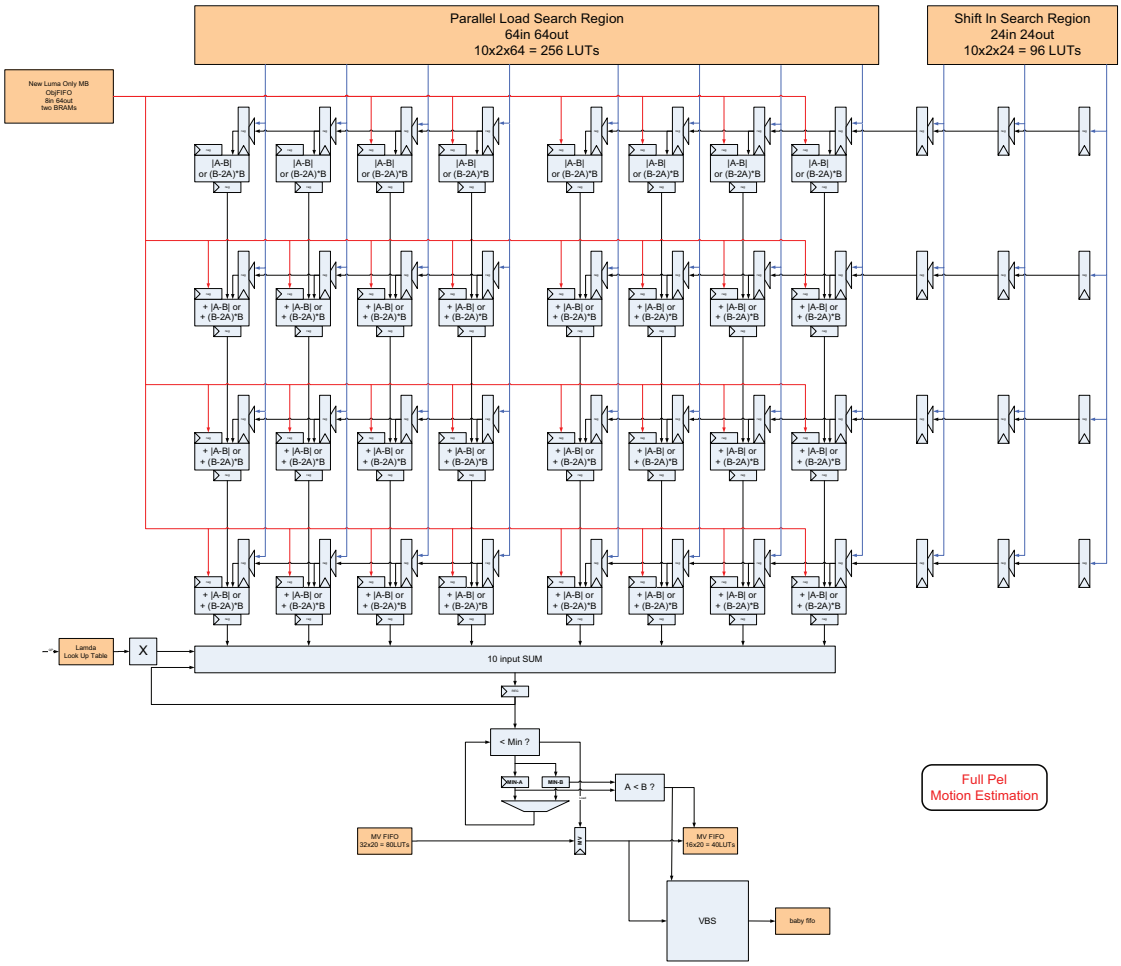

Figure 9: **H.264 Motion Estimation Full Pel Search Engine**

In the Motion Estimation search process, 10 predictors or seeds are given to the search engine and the search engine works in conjunction with the sliding window controller to cycle through these 10 seeds and provide 120 SADs and motion vectors for each 8x4 block searched. The operations are pipelined such that the computational engine is processing on every cycle. The resulting SADs and motion vectors are passed on to the variable block decision post processing to determine the coded block pattern for the Macroblock.

The Full Pel engine operates with a single reference frame although it is not a requirement that this reference frame be the previous frame.

Sliding Window Controller

The Motion Estimation sliding window controller reads from the external memory and organizes the data in a 2D circular buffer for local cache access. The sliding window controller has a parallel output to enable parallel load of the full pel search engine when doing a 4x3 SAD calculation in the area of a seed. See Figure 10 and Figure 11.

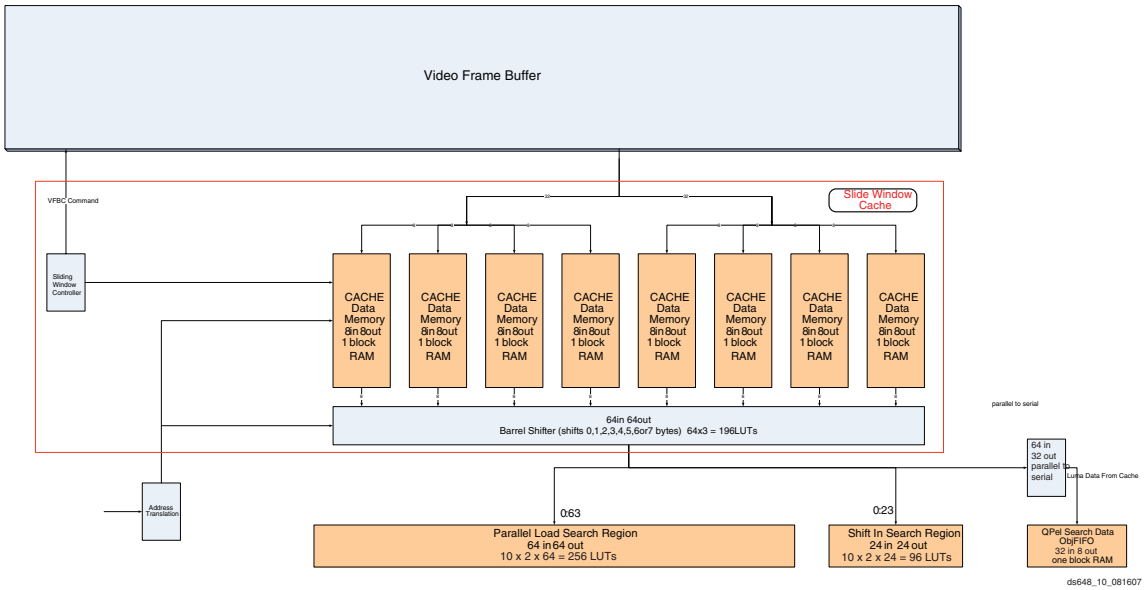

Figure 10: **H.264 Motion Estimation Sliding Window Controller**

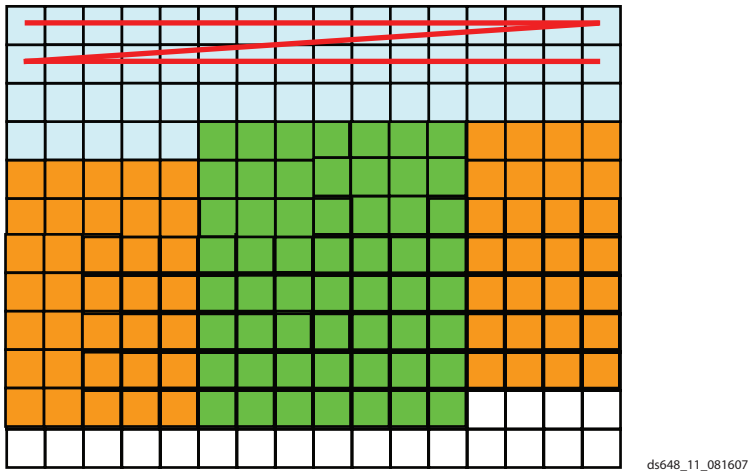

Figure 11: **H.264 Motion Estimation Sliding Window Concept**

Variable Block Decision

The function of the Variable Block Decision is to merge motion vectors from 8x4 blocks that are an exact match. The resulting merge follows the rules of allowable codec block patterns for the H.264 standard. For rate distortion decision with respect to inter or intra blocks should use appropriately summed 8x4 SAD values to determine the overall cost for each mode of operation. Figure 12 shows the decision tree for the variable block decision.

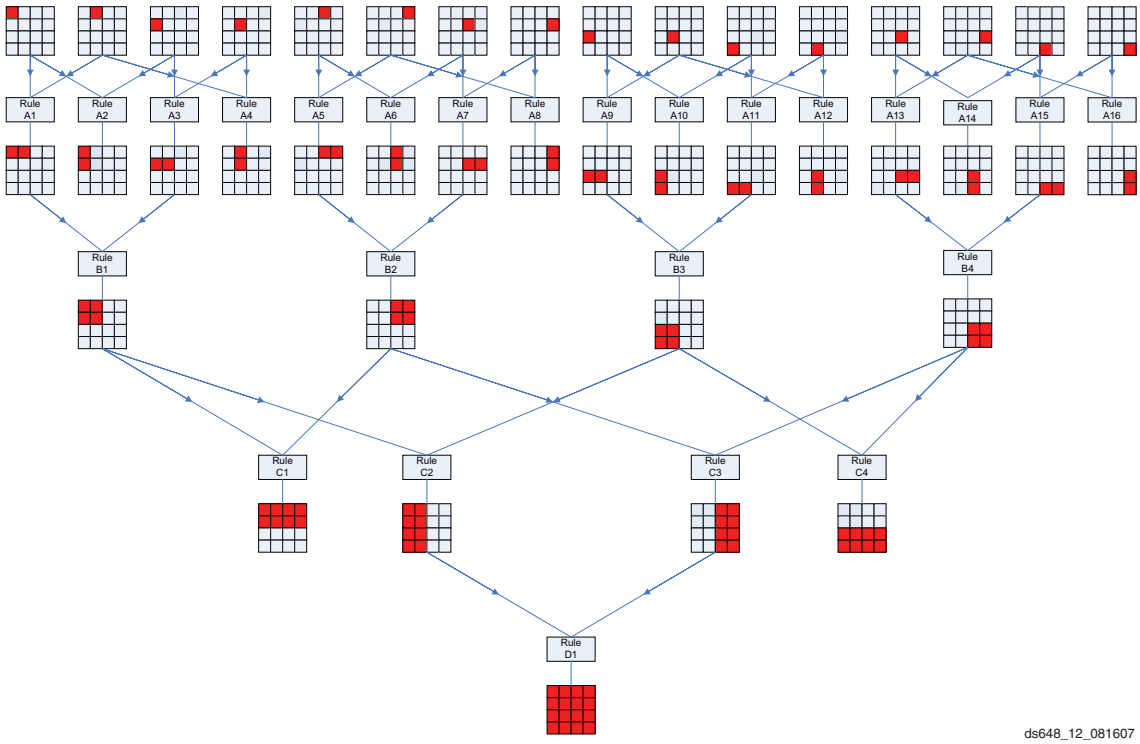

Figure 12: **H.264 Motion Estimation Variable Block Decision**

Unsupported Options

The main H.264 Motion Estimation features not supported by this core in version 1.0 are:

- Block search other than 8x4
- MBAFF mode (Macroblock Adaptive Field/Frame mode)
- More than 1 Reference Frame
- > 8 bit pixels
- 1080p@60 frames per second

Ordering Information

The H.264 Motion Estimation Engine core is provided in netlist format. The core is licensed under the terms of the Xilinx LogiCORE Site License Agreement, which conforms to the terms of the SignOnce IP Site License standard defined by the Common License Consortium. A free evaluation version of the core is available at the Xilinx IP Evaluation Lounge.

The core may be licensed through your local Xilinx sales representative. For part number and ordering information, refer to the H.264 Motion Estimation Engine product page.

References

1. ITU-T/ISO/IEC, *Advanced Video Coding For Generic Audiovisual Services*, H.264 03/2005.

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
10/24/07	1.0	Initial Xilinx release.
04/23/08	1.1	Minor edits to Table 1, Figure 3, and Table 2.

Notice of Disclaimer