Hardware Integration Guide

7280 Encoder

Version 1.4

Copyright Information

Hantro Products Oy has used its best efforts in preparing this document. Hantro Products Oy makes no warranty, representation, or guarantee of any kind, expressed or implied, with regards to this document. Hantro Products Oy does not assume any and all liability arising out of this document including without limitation consequential or incidental damages.

Hantro Products Oy reserves the right to make changes and improvements to any of the products described in this document without prior notice.

Hantro Products Oy
Kiviharjunlenkki 1
90220 Oulu
FINLAND
www.hantro.com

Glossary

4:2:0	YCbCr sampling format, where Cb and Cr components are sub-sampled by two both horizontally and vertically.
4:2:2	YCbCr sampling format, where Cb and Cr components are sub-sampled by two only horizontally.
720p	Progressive high-definition resolution of 1280 x 720 pixels
AC	Alternating Current
AHB	Advanced High Performance Bus, introduced in AMBA 2.0 specification.
AMBA	Advanced Microcontroller Bus Architecture, on-chip bus specification and an open standard
API	Application Programming Interface
APB	AMBA Peripheral Bus protocol specification
AXI	AMBA Advanced eXtensible Interface protocol specification
BIST	Built-In Self Test
CAVLC	Context Adaptive Variable Length Coding (video encoding algorithm)
CEN	Chip Enable, an input pin to an internal memory
CIF	Common Intermediate Format (352x288 pixels)
D1	Standard television data rate; 720x576 pixels with 25 pictures per second in a PAL system, or 720x480 pixels with 30 pictures per second in an NTSC system
DC	Zero frequency component
DFFRXL	D-type flip-flop example component
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
fps	Frames per second
I-VOP	See Intra picture
Inter picture	Coding of a picture using inter prediction, which is derived from samples of reference pictures other than the current encoded picture
Intra picture	Coding of a picture using intra prediction, which is derived from the samples of the current picture
H.263	A video coding standard developed by ITU-T
H.264	A video coding standard developed jointly by ITU-T and ISO/IEC
HW	Hardware
IP	Intellectual Property
JFIF	JPEG File Interchange Format
JPEG	An image coding standard developed by Joint Photographic Experts Group
kbps	Kilobits per second
Macroblock	A data unit of four 8x8 luminance pixel blocks, one 8x8 Cb and one 8x8 Cr block
MB	See macroblock
Mbps	Megabits per second
MPEG-4	A video coding standard developed by Motion Picture Experts Group
MV	Motion Vector
NONSEQ	Non-sequential memory transaction (the first transfer of a burst or a single transfer)
NAL	Network Abstraction Layer
NAND2XL	Two input port nand example component
NTSC	National Television System Committee, defines resolution 720x480
OCP	Open Core Protocol
P-VOP	See Inter picture

Planar	A YCbCr storage format, where all three components form a separate plane in memory
QCIF	Quarter CIF (176x144)
QVGA	Quarter Video Graphics Array (320x240)
QP	Quantisation Parameter, determines the coarseness of quantisation employed during encoding
RAM	Random Access Memory
RGB	Red-Green-Blue color space representation
RLC	Run-Length Coding
RTL	Register Transfer Level
SDRAM	Synchronous Dynamic Random Access Memory
Semi-planar	A YCbCr storage format, where the luminance samples form one plane in memory, and the pixel by pixel interleaved Cb and Cr samples form another
SEQ	Sequential memory transaction
SRAM	Static Random Access Memory
SXGA	Super eXtended Graphics Array (1280x1024)
Verilog	Verilog Hardware Description Language
VGA	Video Graphics Array (640x480)
VHDL	Very high-speed integrated circuit Hardware Description Language
VLC	Variable-Length Coding
WEN	Write Enable, an input pin to an internal memory
YCbCr	A color space representation, where color and intensity data are in separate components: Y contains the black and white image (luminance), Cb and Cr the color information (chrominance)

Version History

Document Version	Changes/Comments
1.0	Original version.
1.1	Synthesis results changed to TSMC's tcbn65lpwc library, Embedded memories updated after optimization, power simulation results added.
1.2	Encoding performance chapter updated after final optimization.
1.3	Performance figures in tables 3 – 11 updated after long latency time transfers optimization. Tables are valid for HW version hw7280_0_46 and later versions.
1.4	Fixed SAQ RAMs sizes to Table 16. Added minimum output bitrate to Table 5 and Table 6.

Table of Contents

COPYRIGHT INFORMATION **2**

GLOSSARY **3**

VERSION HISTORY **5**

TABLE OF CONTENTS **6**

1 INTRODUCTION **8**

2 FEATURES OF THE PRODUCT **9**

2.1 SUPPORTED STANDARDS AND TOOLS 9
2.2 ENCODING FEATURES 11
2.3 PRE-PROCESSING FEATURES 12
2.4 VIDEO STABILIZATION FEATURES 12
2.5 CONNECTIVITY FEATURES 13
2.6 PRODUCT CONFIGURABLE OPTIONS 14

3 SYSTEM REQUIREMENTS **15**

3.1 SILICON AREA REQUIREMENT 15
3.1.1 Embedded memory usage 16
3.2 HARDWARE POWER CONSUMPTION 20
3.2.1 Active state power consumption 20
3.2.2 Shutting down the encoder between pictures 21
3.2.3 Average power consumption 21
3.3 LOGIC CLOCK FREQUENCY REQUIREMENT 23
3.3.1 Performance in ideal environment 23
3.3.2 Minimum frequency in real systems 24
3.3.3 Performance charts, AHB bus scenario 24
3.4 EXTERNAL MEMORY REQUIREMENT 33
3.5 BUS USAGE 33
3.5.1 Encoder data flow 34
3.5.2 Bus requesting 35
3.5.3 Burst types and the amount of non-sequential accessing 35
3.5.4 Bus load 37

4 INTEGRATION OF THE PRODUCT **40**

4.1 7280 ENCODER RTL HIERARCHY 40
4.2 MASTER AND SLAVE INTERFACES 44
4.3 CONNECTING ENCODER TO AN AHB INTERFACE 44
4.3.1 Non-AHB interface signals 45
4.3.2 AHB master interface signals 46
4.3.3 AHB slave interface signals 48
4.4 CONNECTING ENCODER TO AN AXI INTERFACE 49
4.4.1 Non-AXI interface signals 49
4.4.2 AXI master interface signals 49
4.4.3 AXI slave interface signals 53
4.5 CONNECTING ENCODER TO AN APB SLAVE INTERFACE 57
4.6 CONNECTING ENCODER TO AN OCP INTERFACE 57
4.6.1 Non-OCP interface signal 58
4.6.2 OCP master interface signals 59
4.6.3 OCP slave interface signals 60

4.7 SELECTING A SIGNED MULTIPLIER COMPONENT 62
4.8 INTEGRATING THE EMBEDDED MEMORY MODELS 62
4.9 INTEGRATING THE CLOCK GATING COMPONENT 64
4.10 SYNTHESIS CONSIDERATIONS 64
4.10.1 Clocks and resets 65
4.10.2 Synthesis-time automatic clock gating 65
4.10.3 Critical path 65
4.10.4 Multi-cycle paths 65
4.10.5 False paths 65
4.10.6 Defensive timing strategy 66
4.10.7 Glue logic 66
4.10.8 Expected synthesis warnings 66
4.11 INTEGRATION TEST STRUCTURES 66
4.11.1 Interrupt test 66
4.11.2 Register test 66
4.11.3 Memory test 66
4.12 MEMORY-MAPPED REGISTERS 67
4.12.1 Core identification 69
4.12.2 Delivery identification 69
4.12.3 Encoder interrupt register 70
4.12.4 Device configuration register for system bus 71
4.12.5 Integration test register for encoder 72
4.13 HARDWARE VERIFICATION 72
4.13.1 RTL testing on customer site 73
4.13.2 BIST and scan test 74

REFERENCES 75

1 Introduction

This document describes the features, functionality and system requirements of the 7280 encoder product, and instructs in the hardware integration and synthesis process. It is assumed that the reader is familiar with the concepts introduced in the H.264, H.263, MPEG-4 and JPEG standards [1]-[4].

7280 encoder product is configurable. Please note that all features and formats presented in this document may not be included in the release you have received.

In chapter two the features of the product are described. Functionality and system requirements are presented in chapter three and the actual HW integration is discussed in chapter four. It may be useful to read this document along with the Software Integration Guide [5] to form a solid overall picture of the product.

`Courier new` font is used when describing code or parameters in the document. *Italic* expression is used in case of file names, paths or commands.

2 Features of the Product

2.1 Supported standards and tools

The 7280 multi-format encoder is compatible with OCP [6], AHB [7], AXI [8] and AMBA 3 APB [9] bus interfaces. The encoder supports 32-bit or 64-bit AHB, AXI or OCP master interface. The supported slave interfaces are 32-bit OCP, AHB, AXI or APB.

The supported standards, profiles and levels are presented in Table 1. The supported video tools are shown in Table 2, Table 3 and Table 4. The block diagram of the encoder is shown in Figure 1.

TABLE 1. SUPPORTED STANDARDS, PROFILES AND LEVELS

Standard	Encoder support
H.264 Profile and level	Baseline Profile, levels 1-3.2
MPEG-4 Visual profile and level	Simple Profile, levels 0-6 Main Profile, level 4[1)]
H.263 profile and level	Profile 0, levels 10-70. Image size up to 720x576, time code extensions not supported
JPEG profile and level	Baseline

[1)] Only Simple profile tools are supported.

TABLE 2. SUPPORTED H.264 TOOLS

Tool	Encoder support
Slices	I and P slices
Entropy encoding	CAVLC
Basic	• Constrained intra prediction • Maximum MV range +-16 pixels • MV accuracy ¼ pixels • All block sizes from 4x4 to 16x16 supported
Number of reference frames	1
Maximum number of slice groups	1

TABLE 3. SUPPORTED MPEG-4 VISUAL TOOLS

Visual tool	Encoder support
Basic	• I and P-VOPs • Maximum MV range +-16 pixels • MV accuracy ½ pixels • 1 or 4 MV/Macroblock • DC prediction
Error resilience	• Video packets (SW performs entropy encoding) • Data partitioning (SW performs entropy encoding) • Reversible VLC
Number of reference frames	1

Quantization	Method 2
Number of visual objects	1
Short Video Header	Yes

TABLE 4. SUPPORTED H.263 VISUAL TOOLS

Visual tool	Encoder support
Basic	• I and P-VOPs • Maximum MV range +-16 pixels • MV accuracy ½ pixels • 1 MV/Macroblock
Error resilience	• GOB
Number of reference frames	1

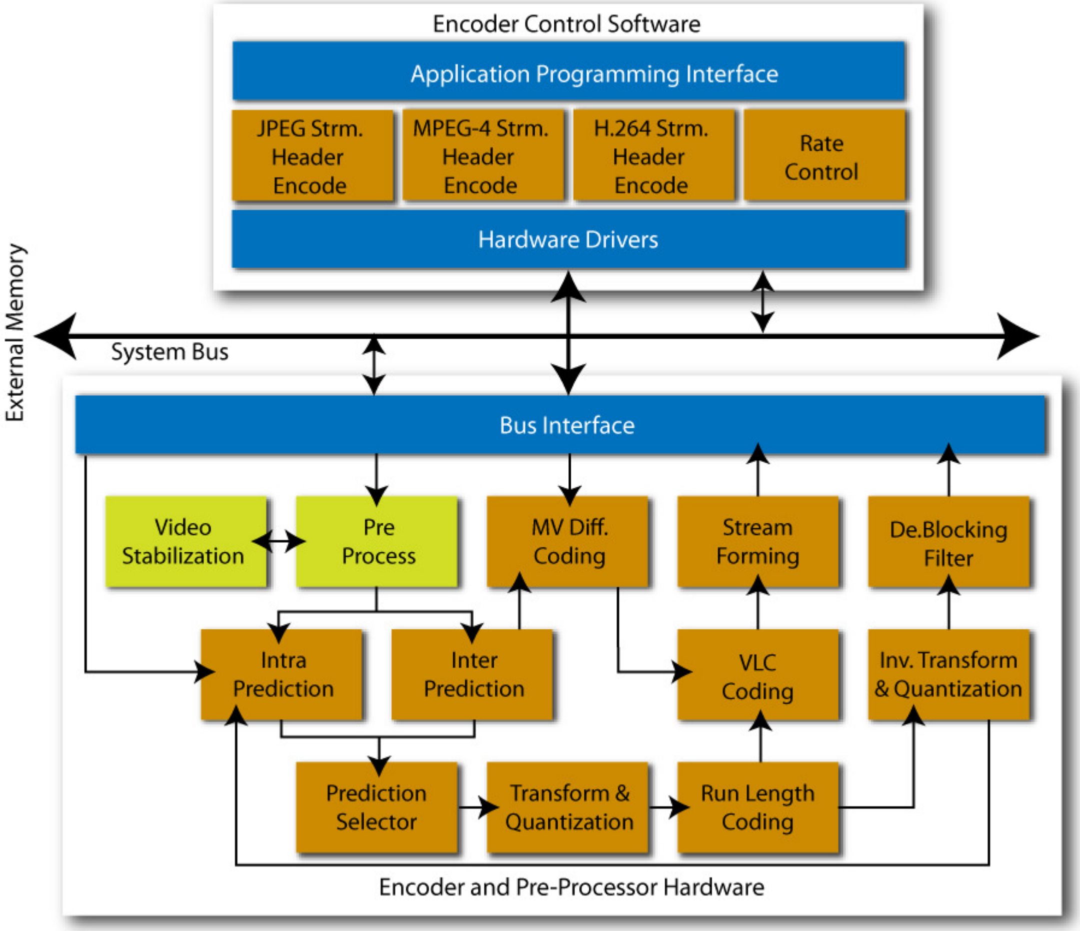

FIGURE 1. 7280 ENCODER FUNCTIONAL BLOCK DIAGRAM.

2.2 Encoding features

The features of the encoder for each supported standard are shown in Table 5 - Table 7.

TABLE 5. H.264 FEATURES

Feature	Encoder support
Input data format	YCbCr 4:2:0 planar or semi-planar YCbYCr and CbYCrY 4:2:2 Interleaved [1)]
Output data format	H.264 byte or NAL unit stream
Supported image size	• 96 x 96 to 1280 x 1024 (SXGA) • Step size 4 pixels
Maximum frame rate	25 fps at 720x576, or 30 fps at 720x480 or 30 fps at 1280x720 or 30 fps at 1280x1024[2)]
Output bit rate	Maximum 20 Mbps Minimum 10kbps

[1)] Internally encoder handles images only in 4:2:0 format

[2)] Actual maximum frame rate will depend on the logic clock frequency and the system bus performance. The given figure 30 fps at 1280x720 requires logic clock frequency 181 MHz. 30 fps at 1280x1024 requires logic clock frequency of 257 MHz.

TABLE 6. MPEG-4 / H.263 FEATURES

Feature	Encoder support
Input data format	YCbCr 4:2:0 planar or semi-planar YCbYCr and CbYCrY 4:2:2 Interleaved [1)]
Output data format	MPEG-4 / H.263 elementary video stream
Supported image size	• 96 x 96 to 1280 x 1024 (SXGA) • Step size 4 pixels
Maximum frame rate	25 fps at 720x576, or 30 fps at 720x480 30 fps at 1280x720 or 30 fps at 1280x1024[2)]
Maximum bit rate	Maximum 10 Mbps Minimum 10kbps

[1)] Internally encoder handles images only in 4:2:0 format

[2)] Actual maximum frame rate will depend on the logic clock frequency and the system bus performance. The given figure 30 fps at 1280x720 requires a logic clock frequency of 160 MHz. 30 fps at 1280x1024 requires logic clock frequency of 227 MHz.

TABLE 7. JPEG FEATURES

Feature	Encoder support
Input data format	YCbCr 4:2:0 planar or semi-planar YCbYCr and CbYCrY 4:2:2 Interleaved [1)]
Output data format	• JFIF file format 1.02 • Non-progressive JPEG
Supported image size	• 80x16 to 4672 x 3504 (16.4 million pixels) • Step size 4 pixels
Maximum data rate	Up to 90 million pixels per second [2)]
Thumbnail encoding	JPEG compressed thumbnails supported

[1)] Internally encoder handles images only in 4:2:0 format

[2] Actual maximum frame rate will depend on the logic clock frequency and JPEG compression rate.

2.3 Pre-processing features

Pre-processing is pipelined with the encoder and it can be used only with the 7280 encoder. Pre-processing features are presented in Table 8.

TABLE 8. PRE-PROCESSING FEATURES

Feature	Encoder support
Color space conversion	YCbYCr or CbYCrY 4:2:2 Interleaved or semi-planar 4:2:0 to YCbCr 4:2:0
Cropping	Video - from 4672 x 3504 to any supported encoding size
Rotation	90 or 270 degrees

2.4 Video stabilization features

Digital video stabilization detects and compensates undesired jitter effect on the video while the desired effects like panning are maintained. Stabilization operates with two input picture buffers simultaneously. Stabilization functionality requires at least 8 pixels larger input picture than the actual resolution which is wanted to be encoded. Figure 2 shows the relationship of the picture dimensions used by stabilization and demonstrates the effect of stabilizing a video frame. Frame 0 is the first frame and the stabilized picture is positioned in the middle of the camera picture. Frame 1 has been stabilized and the stabilized picture has moved 4 pixels left. The offsets around the stabilized picture (shown in dark) are cropped out when encoding the video thus creating a more stable video.

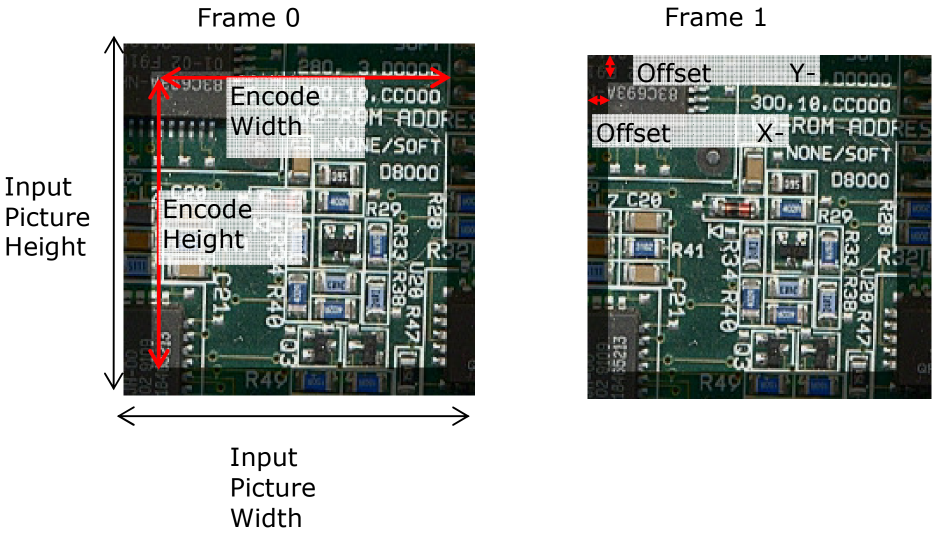

FIGURE 2. STABILIZATION PICTURE DIMENSION

Video stabilization can be used pipelined with 7280 video encoding or in standalone mode when 7280 video encoding is disabled. See more details about the usage of video stabilization in [11], [12], and [14]. Video stabilization features are explained in Table 9.

TABLE 9. VIDEO STABILIZATION FEATURES

Feature	Encoder support
Maximum stabilization displacement in pixels for two sequential input video pictures	+-16 pixels
Adaptive motion compensation filter	From 6 to 40 sequential video pictures noticed in unwanted and wanted movement separation
Offset around stabilized picture	• Minimum 8 pixels in standalone mode • Minimum 16 pixels when pipelined with video encoder • Recommended 64 pixels • Maximum not limited

2.5 Connectivity features

The encoder supports the connectivity features presented in Table 10. The usage of these features is described in more detail in paragraph 4.12. Note that the endian modes can be separately set for input and output data.

TABLE 10. CONNECTIVITY FEATURES

Feature	Encoder support
AHB precise burst / data discard [1)]	Yes
Restricting maximum issued AHB burst length	Yes, to 4, 8 or 16
Restricting maximum issued AXI burst length	Yes, to any value between 1-16
Restricting maximum issued OCP burst length	Yes, to any value between 1-31
Interrupt method	Polling or level based interrupting
32-bit little endian	Yes, byte order 3-2-1-0
32-bit big endian	Yes, byte order 0-1-2-3
64-bit little endian	Yes, byte order 7-6-5-4-3-2-1-0
64-bit big endian	Yes, byte order 0-1-2-3-4-5-6-7
Mixed 32-bit little endian in a 64-bit bus	Yes, byte order 3-2-1-0-7-6-5-4
Mixed 32-bit big endian in a 64-bit bus	Yes, byte order 4-5-6-7-0-1-2-3

[1)] When enabled, the bus interface will convert all INCR type read bursts into INCR4 and internally discard the extra data.

2.6 Product configurable options

Depending on the licensee needs and the agreement the 7280 encoder product can be scaled to support limited a feature set. This is called product configurability.

The 7280 product can be configured to leave out some of the encoding standards or video stabilization according to Table 11, change the maximum video resolution according to options in Table 12 or to select different bus protocols according to Table 13. The amount of supported video encoding standards affects the required silicon area. The maximum resolution affects the required embedded memory amount.

7280 Encoder product release can be any combination of configurable items.

TABLE 11. 7280 ENCODER PRODUCT DELIVERY CONFIGURATION

7280 Encoder Product Configurability Options
H.264 Video Encoder
MPEG-4/H.263 Video Encoder
JPEG Encoder
Video Stabilization

TABLE 12. 7280 ENCODER PRODUCT CONFIGURABLE VIDEO RESOLUTION

7280 Encoder Video Resolution Options	Maximum resolution
SXGA	1280x1024
720p	1280x720
D1	720x576
CIF	352x288

TABLE 13. 7280 ENCODER PRODUCT CONFIGURABLE SYSTEM BUS PROTOCOL

7280 System Bus Protocol Options	Interface Definition
OCP 2.0	32-bit or 64-bit master interface 32-bit slave interface
AHB 2.0	32-bit or 64-bit master interface 32-bit slave interface
AXI 1.0	32-bit or 64-bit master interface 32-bit slave interface
AMBA 3 APB 1.0	32-bit slave interface

3 System Requirements

This chapter describes the requirements the 7280 encoder has on the system it will be integrated to. In the following paragraphs the functionality of the product is described in terms of data flow and bus usage, and by showing typical performance plots in different bus and memory latency environments.

3.1 Silicon area requirement

The synthesis parameters and tools presented in Table 14 were used for obtaining the gate count estimate.

TABLE 14. LOGIC SYNTHESIS PARAMETERS AND TOOLS

Parameter / tool	Value
Target technology	TSMC 65nm Core Library, tcbn65lp, 1.2V
Library type	Worst case
Topographical synthesis	No
Clock frequency (HCLK)	300 MHz
Hardware description language	VHDL
Synthesis tool	Synopsys, Design Compiler, Version Y-2006.06-SP5 for linux – Jan 19,2007

The synthesis results using DC Ultra are shown in Table 15. In the gate count estimation, a TSMC 65 nm process NAND2XL component with area of 1,44 μm^2 is used as the reference gate. The gate count estimate includes only the logic, i.e. no test structures or wireload models. Embedded memories are not included in this estimate.

TABLE 15. 7280 HW ENCODER GATE COUNT ESTIMATES FOR FEATURE OPTIONS

Features	Component	Cell area [μm^2]	Gate count [gates]
All Formats with Video Stabilization	7280 Encoder logic	540055	375038
All Formats	7280 Encoder logic	522073	362550
MPEG-4/H.263 Video Encoder	7280 Encoder logic	367006	254865
H.264 Video Encoder	7280 Encoder logic	424970	295118
JPEG Encoder	7280 Encoder logic	193183	134156
MPEG-4/H.263 Video Encoder with Video Stabilization	7280 Encoder logic	383567	266366
H.264 Video Encoder with Video Stabilization	7280 Encoder logic	442850	307535
JPEG Encoder with Video Stabilization	7280 Encoder logic	210101	145903
MPEG-4/H.263 and H.264 Video Encoder with Video Stabilization	7280 Encoder logic	538704	374100
MPEG-4/H.263 and H.264	7280 Encoder logic	523810	363757

Video Encoder			
MPEG-4/H.263 Video and JPEG Encoder with Video Stabilization	7280 Encoder logic	430353	298856
MPEG-4/H.263 Video and JPEG Encoder	7280 Encoder logic	413859	287402
H.264 Video and JPEG Encoder with Video Stabilization	7280 Encoder logic	523319	363417
H.264 Video and JPEG Encoder	7280 Encoder logic	507134	352177

3.1.1 Embedded memory usage

Embedded memory amount in the encoder release package can vary depending on target resolution and selected features. All embedded memories in the product are single-port synchronous SRAM that can also be implemented as register files. Only the simulation models of the memories are included in the RTL delivery, so it is up to the customer do to the memory integration. Any memory vendor can be selected. Integration instructions for the memories are given in chapter 4.

The required embedded memory amount for H.264 and MPEG-4/H.263 configuration possibilities and supported maximum resolution are presented in Table 16.

TABLE 16. 7280 EMBEDDED MEMORIES FOR H.264 AND MPEG-4/H.263

SRAM instance name	Supported format	Supported resolution	Size	Embedded memory [bytes]
u_ram16x128_mqnb	H.264 MPEG-4/H.263	SXGA 720p SD CIF	16x128	256
u_ram572x64_sa	H.264 MPEG-4/H.263	SXGA 720p SD CIF	572x64	4576
u_ram240x64_np	H.264 MPEG-4/H.263	SXGA 720p SD CIF	240x64	1920
u_ram192x48_sax0	H.264 MPEG-4/H.263	SXGA 720p SD CIF	192x48	1152
u_ram192x48_sax1	H.264 MPEG-4/H.263	SXGA 720p SD CIF	192x48	1152
u_ram192x48_sax2	H.264 MPEG-4/H.263	SXGA 720p SD	192x48	1152

		CIF		
u_ram64x24_nbx	H.264 MPEG-4/H.263	SXGA 720p SD CIF	64x24	192
u_ram_dynamicx72_saq0h	H.264	SXGA 720p SD CIF	118x72	1062
	MPEG-4/H.263	SXGA 720p SD CIF	114x72	1026
u_ram_dynamicx72_saq1h	H.264	SXGA 720p SD CIF	118x72	1062
	MPEG-4/H.263	SXGA 720p SD CIF	114x72	1026
u_ram_dynamic x72_saq0l	H.264	SXGA 720p SD CIF	118x72	1062
	MPEG-4/H.263	SXGA 720p SD CIF	114x72	1026
u_ram_dynamic x72_saq1l	H.264	SXGA 720p SD CIF	118x72	1062
	MPEG-4/H.263	SXGA 720p SD CIF	114x72	1026
u_ram_dynamicx64_pr	H.264 MPEG-4/H.263	SXGA 720p SD CIF	32x64	256
u_ram_dynamicx88_tq	H.264	SXGA 720p SD CIF	68x88	748
	MPEG-4/H.263	SXGA 720p SD CIF	64x88	704
u_ram96x32_fb0	H.264 MPEG-4/H.263	SXGA 720p SD CIF	96x32	384

u_ram96x32_fb1	H.264 MPEG-4/H.263	SXGA 720p SD CIF	96x32	384
u_ram_dynamicx32_dfw0	H.264	SXGA 720p	1392x32	5568
		SD	832x32	3328
		CIF	464x32	1856
	MPEG-4/H.263	SXGA 720p SD CIF	96x32	384
u_ram_dynamicx32_dfw1	H.264	SXGA 720p	1392x32	5568
		SD	832x32	3328
		CIF	464x32	1856
	MPEG-4/H.263	SXGA 720p SD CIF	96x32	384
u_ram_dynamicx16_bsmv	H.264 MPEG-4/H.263	SXGA 720p	320x16	640
		SD	180x16	360
		CIF	88x16	176
u_ram_dynamicx32_ne	H.264	SXGA 720p	1048x32	4192
		SD	596x32	2384
		CIF	296x32	1184
	MPEG4	SXGA 720p	320x32	1280
		SD	180x32	720
		CIF	88x32	352
u_ram_dynamicx18_rlc	H.264	SXGA 720p	580x18	1305
		SD	510x18	1147,5
		CIF	464x18	1044
	MPEG-4/H.263	SXGA 720p SD CIF	384x18	864

Embedded memories required by JPEG only configuration (no H.264 or MPEG-4/H.263 support) are presented in Table 17.

Table 17. JPEG Encoder only Configuration embedded memories

SRAM instance name	Supported Format	Supported resolution	Size	Embedded memory [bytes]
u_ram_dynamicx64_pr	JPEG	16.4 Mpix	32x64	256
u_ram240x64_np	JPEG	16.4 Mpix	240x64	1920
u_ram_572x64_sa	JPEG	16.4 Mpix	572x64	4576
u_ram_dynamicx12_tq	JPEG	16.4 Mpix	64x88	704

u_ram_dynamicx18_rlc	JPEG	16.4 Mpix	384x18	864

Embedded memories required by video stabilization are presented in Table 18.

Table 18. 7280 Video stabilization embedded memories

SRAM instance name	Instance size	Embedded memory [bytes]
u_ram64x64_stbnb	64x64	512
u_ram102x64_stbnfnb	102x64	816
u_ram594x40_stb	594x40	2970

Table 19 summarizes embedded memory amounts of different configurations of 7280 product.

Table 19. 7280 configuration options and embedded memories

Encoder Configuration	Memory instance Amount	Supported Resolution	Embedded memory [kilo bytes]
H.264 MPEG-4/H.263 JPEG Video Stabilization	23	SXGA 720p	37.1
		SD	30.6
		CIF	26.2
H.264 MPEG-4/H.263 JPEG	20	SXGA 720p	32.9
		SD	26.4
		CIF	22.0
MPEG-4/H.263	20	SXGA 720p	19.3
		SD	18.5
		CIF	18.0
H.264	20	SXGA 720p	32.9
		SD	26.4
		CIF	22.0
JPEG	4	16.4 Mpix [1)]	8.1
MPEG-4/H.263 JPEG	20	SXGA 720p	19.3
		SD	18.5
		CIF	18.0
MPEG-4/H.263 JPEG Video Stabilization	23	SXGA 720p	23.5
		SD	22.7
		CIF	22.2
H.264 JPEG Video Stabilization	23	SXGA 720p	37.1
		SD	30.6
		CIF	26.2
H.264 JPEG	20	SXGA 720p	32.9
		SD	26.4
		CIF	22.0
MPEG-4/H.263 Video Stabilization	23	SXGA 720p	23.5

		SD	22.7
		CIF	22.2
H.264 Video Stabilization	23	SXGA 720p	37.1
		SD	30.6
		CIF	26.2
H.264 MPEG-4/H.263 Video Stabilization	23	SXGA 720p	37.1
		SD	30.6
		CIF	26.2

[1)] Only JPEG encoding supported

3.2 Hardware power consumption

This paragraph handles the power consumption of the 7280 encoder hardware. The presented power consumption figures apply for the full configuration of 7280 product with video stabilization, at SXGA resolution. The power consumption is measured using Synopsys Power Compiler tool and Verilog netlist simulation. The parameters for synthesis are shown in Table 20.

The estimates given in this document do not include the power consumption of the ARM processor, bus environment or external memory.

TABLE 20. SYNTHESIS AND SIMULATION PARAMETERS AND TOOLS FOR POWER MEASUREMENT

Parameter / tool	Value
Target library	TSMC 65nm Core Library, tcbn65lp, 1.2V [10]
Library type	Worst case
Topographical synthesis	Yes
Clock frequency	250 MHz
Hardware description language	Verilog
Synthesis tool	Synopsys, Design Compiler, Version Y-2006.06-SP5 for linux – Jan 19,2007
DC Ultra	Yes
Automatic clock gating	Yes
Measurement length in netlist simulation	30000 ns

Note: due to memory compiler licensing issues, the embedded memory power consumption is based TSMC 65 nm generic library. As the typical target library is low power, the memory power consumption figures may be somewhat too large.

3.2.1 Active state power consumption

The netlist simulation for power measurement is performed separately for each encoding format, with the dynamic module level clock gating first disabled and then enabled. When the module level clock gating is enabled, the clock signal is gated from blocks that are not required at that time. For example, when an H.264 video is being encoded, the clock can be disabled for MPEG-4 or JPEG and Video stabilization specifics sub modules.

The active state power consumption figures for logic and embedded memories are presented in Table 21. These figures show how much power the hardware consumes while encoding. The logic core figures contain the clock tree power consumption. The same test streams and picture numbers were used for measurement as in Table 25.

Typically the hardware is idling between pictures and does not consume much power during those periods. The following paragraphs Table 22 - Table 24 show the power consumption when the idling time is taken into notice.

TABLE 21. HW ACTIVE STATE POWER CONSUMPTION AT 250 MHZ

Encoding scheme	Power type	Power consumption [mW], dynamic clock gating disabled	Power consumption [mW], dynamic clock gating enabled
H.264	SRAM power	16.6	16.4
	Logic core power	41.4	35.0
	Total power	58.1	51.4
MPEG-4/H.263	SRAM power	12.4	12.2
	Logic core power	39.5	36.6
	Total power	51.9	48.8
JPEG	SRAM power	4.2	4.1
	Logic core power	32.0	23.1
	Total power	36.2	27.1
H.264 Video Stabilization	SRAM power	17.6	17.5
	Logic core power	40.4	34.5
	Total power	58.0	52.0
MPEG-4/H.263 Video Stabilization	SRAM power	15.7	13.9
	Logic core power	40.6	35.9
	Total power	56.3	49.7

3.2.2 Shutting down the encoder between pictures

The switching frequency of the encoder's nets is measured when the encoder is enabled. When running the encoder in a typical environment, it does not have to be activated all the time. This means that the power or the clock to the encoder can be turned off between pictures. The amount of idle time between encoding two consecutive pictures depends for example on the bus clock frequency, target frame rate and resolution, the type of stream to encode, and the SDRAM access time.

NOTE: In the following power estimates it is assumed that the clock is turned off between pictures. It is expected that the data bus width is 64 bits and the bus environment is ideal (zero wait states, zero latency). If the encoder is connected to a 32-bit bus, the given figures will increase roughly 5-10 % due to reduced processing speed and less idle time.

3.2.3 Average power consumption

The encoder hardware processes each format with a slightly different speed, as shown in Table 25.

These figures affect the active state versus passive state ratio of the hardware; the faster the encoding process is, the more time it can idle before SW will start the hardware for the next picture.

The average power dissipation for the supported encoding formats (for video formats video stabilization is enabled, for JPEG encoding disabled), when this activity ratio is taken into notice is given in Table 22, Table 23 and Table 24. In the tables the clock gating is enabled.

Note: Over 100% active state ratio implies that the used frequency is not adequate for encoding at that data rate. The power figures given at those ratios are not accurate.

TABLE 22. ENCODER HARDWARE POWER CONSUMPTION FOR H.264 AT 250 MHZ

Image size	Frame rate	Core activity [%]	Power dissipation [mW]
QCIF	15	0,99 %	0,51
QCIF	30	1,98 %	1,03
QVGA	15	3,00 %	1,56
QVGA	30	5,99 %	3,12
CIF	15	3,95 %	2,06
CIF	30	7,91 %	4,11
VGA	15	11,98 %	6,23
VGA	30	23,96 %	12,46
SD	15	13,48 %	7,01
SD	30	26,96 %	14,02
720p	15	35,94 %	18,69
720p	30	71,88 %	37,38
SXGA	15	51,12 %	26,58
SXGA	30	102,24 %	53,16

TABLE 23. ENCODER HARDWARE POWER CONSUMPTION FOR MPEG-4 AT 250 MHZ

Image size	Frame rate	Core activity [%]	Power dissipation [mW]
QCIF	15	0,85 %	0,42
QCIF	30	1,70 %	0,85
QVGA	15	2,58 %	1,28
QVGA	30	5,16 %	2,57
CIF	15	3,41 %	1,69
CIF	30	6,81 %	3,39
VGA	15	10,32 %	5,13
VGA	30	20,65 %	10,26
SD	15	11,62 %	5,77
SD	30	23,23 %	11,55
720p	15	30,97 %	15,39
720p	30	61,95 %	30,79
SXGA	15	44,05 %	21,89
SXGA	30	88,10 %	43,79

TABLE 24. ENCODER HARDWARE POWER CONSUMPTION FOR JPEG AT 250 MHZ

Image size	Frame rate	Core activity [%]	Power dissipation [mW]
VGA	1	0,33 %	0,09
1 Mpixel	1	1,07 %	0,29
2 Mpixel	1	2,14 %	0,58
3 Mpixel	1	3,21 %	0,87
5 Mpixel	1	5,34 %	1,45
8 Mpixel	1	8,55 %	2,32
16 Mpixel	1	17,10 %	4,63
32 Mpixel	1	34,20 %	9,27

3.3 Logic clock frequency requirement

The applicable logic clock frequency range is limited in the low end by the target data rate, and in the high end by the critical logic timing path. 7280 encoder has been synthesized successfully for 300 MHz clock frequency in TSMC 65nm Core Library (tcbn65lp) process with wireload models.

3.3.1 Performance in ideal environment

Ideal environment for the hardware is a system without bus or memory latencies or wait states, with the encoder HW constantly processing data without idling between pictures.

In Table 25 the ideal performance of encoding different formats is shown. The figures are given for 32 and 64 bit bus environments and they are based on RTL simulation.

In order to achieve the performance shown, hardware must have valid input data and be enabled all the time. However, some hardware processing time is lost between pictures as the software performs the writing of high level headers to the stream, but the effect of this is negligible.

Note: If polling is used as the interrupting mechanism instead of IRQ, more time may be lost. Software requires more time also when it is performing entropy encoding. See more details in the software integration guide [5].

The time lost when HW is idling can be compensated by increasing the logic clock frequency.

TABLE 25. PERFORMANCE FOR DIFFERENT ENCODING MODES IN IDEAL ENVIRONMENT

Operational mode	Performance with a 64-bit or 32 bit bus [used clock cycles per processed macroblock]
H.264 encoding	1664
MPEG-4 encoding	1434
JPEG 4:2:0 encoding	684

Using the measurements given in the table, the minimum frequencies for target resolution can be calculated.

As an example let's consider a case of H.264 encoding with CIF at 30 fps. The minimum logic clock frequency in a 64-bit bus environment is given by following equation:

(1664*396*30) / 1 000 000 = 19.8 MHz,

where 396 is the amount of macroblocks in one CIF picture.

3.3.2 Minimum frequency in real systems

Real systems typically have some latency to bus or SDRAM. Latency can be defined as the amount of clock cycles from “bus requested” to “bus granted” (AHB) or from “data requested” to “data received” (OCP, AXI). When latency and non-sequential memory access (the first access of an access burst) wait states are increased, the minimum required clock frequency starts to rise. It is assumed that the sequential memory accesses have always one clock cycle latency.

In the Table 26 transformation chart from clock cycles per macroblock to required clock frequency at different resolutions is shown.

Table 26. Encoding performance per macroblock vs. minimum frequency

Hardware encoding [clk / MB]	QCIF 30 fps [MHz]	QVGA 30 fps [MHz]	CIF 30 fps [MHz]	VGA 30 fps [MHz]	D1 30 fps [MHz]	720p 15 fps [MHz]	720p 30 fps [MHz]	SXGA 15 fps [MHz]	SXGA 30 fps [MHz]	5 Mpix 1 fps [MHz]	16 Mpix 1 fps [MHz]
1000	3,0	9,0	11,9	36,0	40,5	54,0	108,0	76,8	153,6	19,5	62,5
1500	4,5	13,5	17,8	54,0	60,8	81,0	162,0	115,2	230,4	29,3	93,8
2000	5,9	18,0	23,8	72,0	81,0	108,0	216,0	153,6	307,2	39,1	125,0
2500	7,4	22,5	29,7	90,0	101,3	135,0	270,0	192,0	384,0	48,8	156,3
3000	8,9	27,0	35,6	108,0	121,5	162,0	324,0	230,4	460,8	58,6	187,5
3500	10,4	31,5	41,6	126,0	141,8	189,0	378,0	268,8	537,6	68,4	218,8
4000	11,9	36,0	47,5	144,0	162,0	216,0	432,0	307,2	614,4	78,1	250,0
4500	13,4	40,5	53,5	162,0	182,3	243,0	486,0	345,6	691,2	87,9	281,3
5000	14,9	45,0	59,4	180,0	202,5	270,0	540,0	384,0	768,0	97,7	312,5

3.3.3 Performance charts, AHB bus scenario

The figures given in this paragraph are valid for 32-bit and 64-bit AHB environment, with a regular SDRAM controller model that has variable non-sequential waitstates, and a constant sequential waitstate of 1. The effects the bus latency and external memory wait states have on the encoding speed are shown in Figure 3 -Figure 11. The figures are based on RTL simulation.

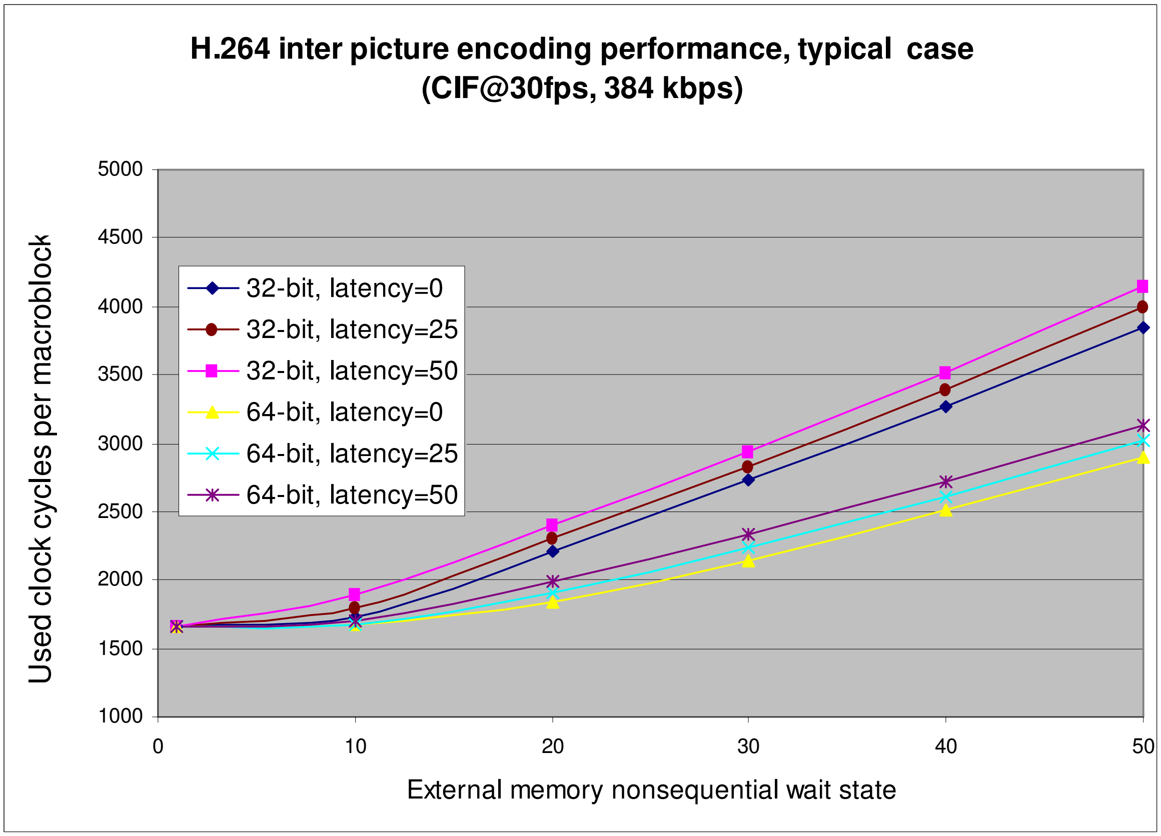

FIGURE 3. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON H.264 INTER FRAME ENCODING PERFORMANCE.

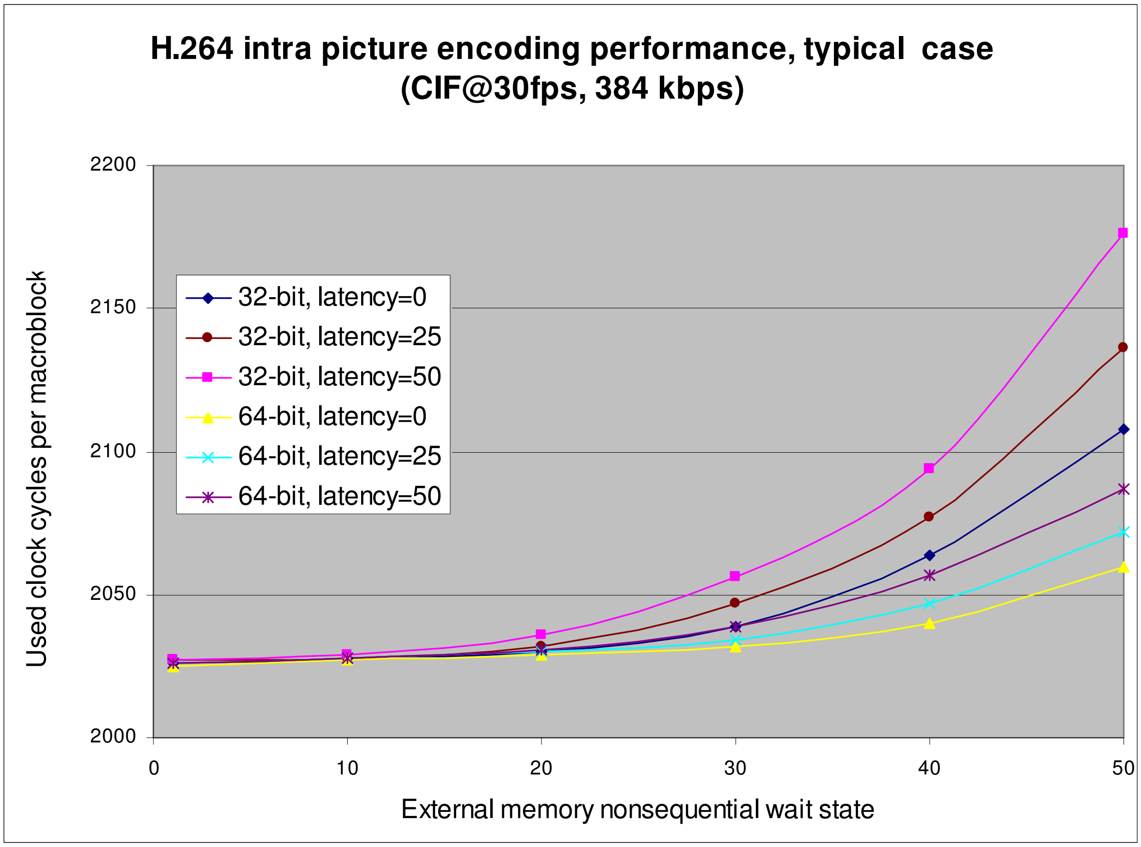

FIGURE 4. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON H.264 INTRA FRAME ENCODING PERFORMANCE.

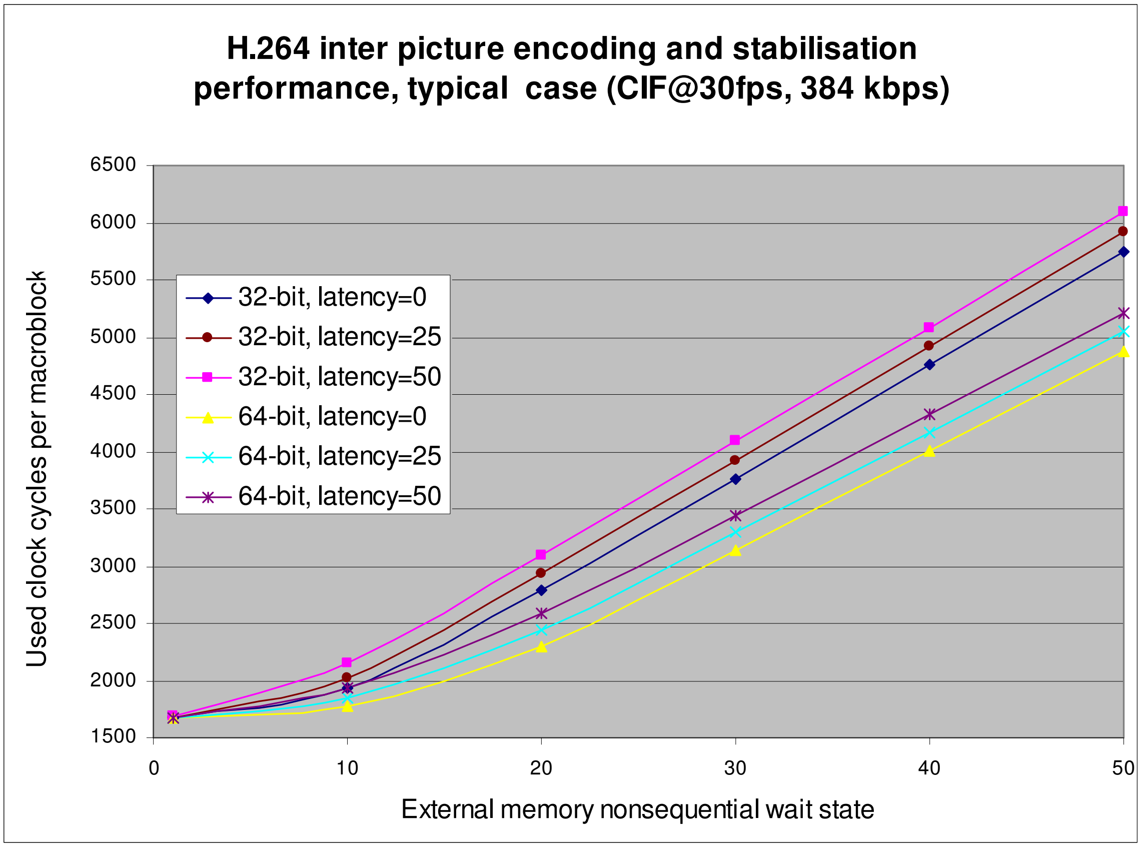

FIGURE 5.EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON H.264 INTER FRAME ENCODING AND VIDEO STABILIZATION PERFORMANCE.

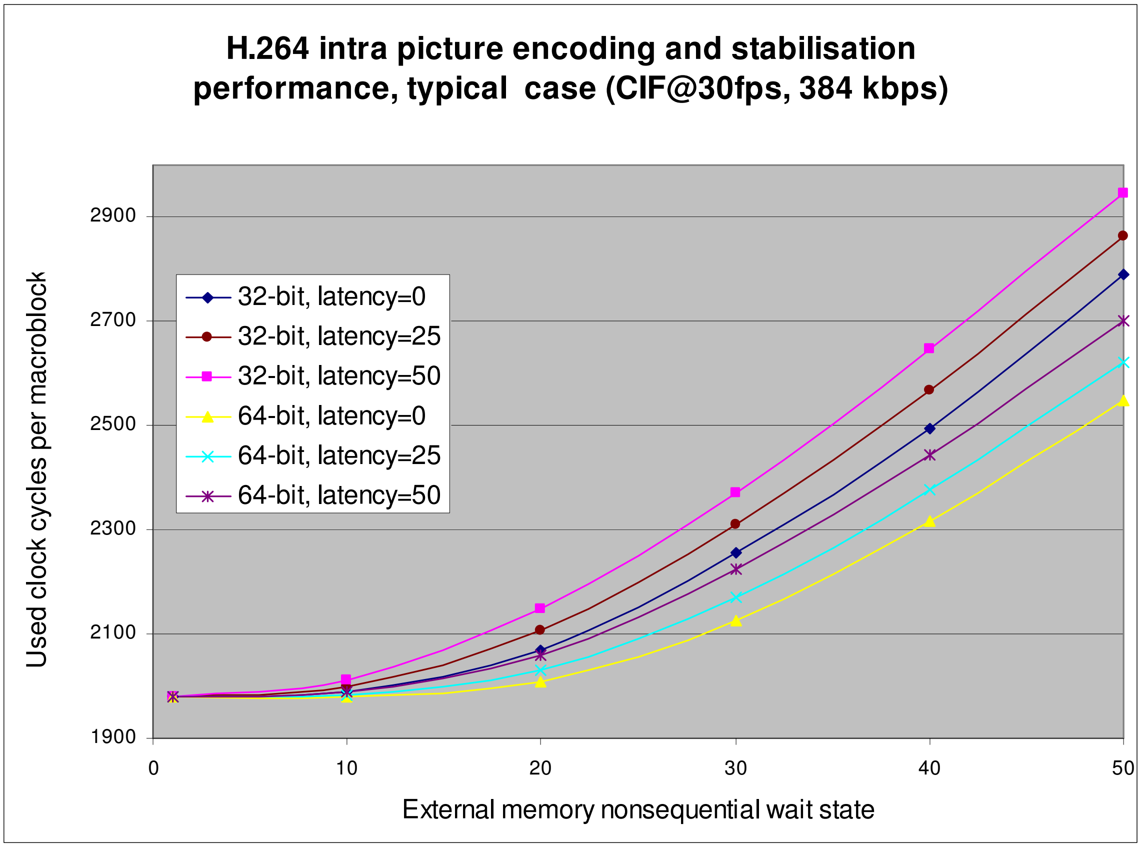

FIGURE 6.EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON H.264 INTRA FRAME ENCODING AND VIDEO STABILIZATION PERFORMANCE.

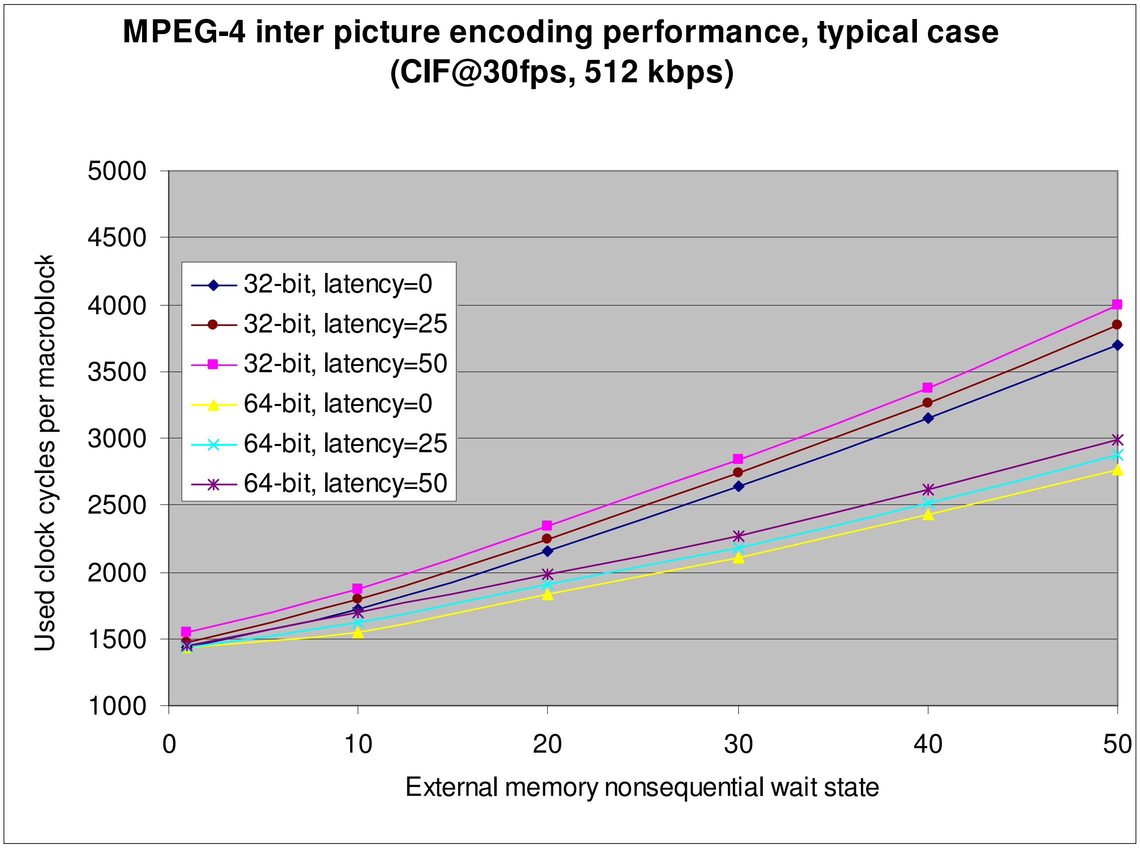

FIGURE 7. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON MPEG-4 ENCODING PERFORMANCE.

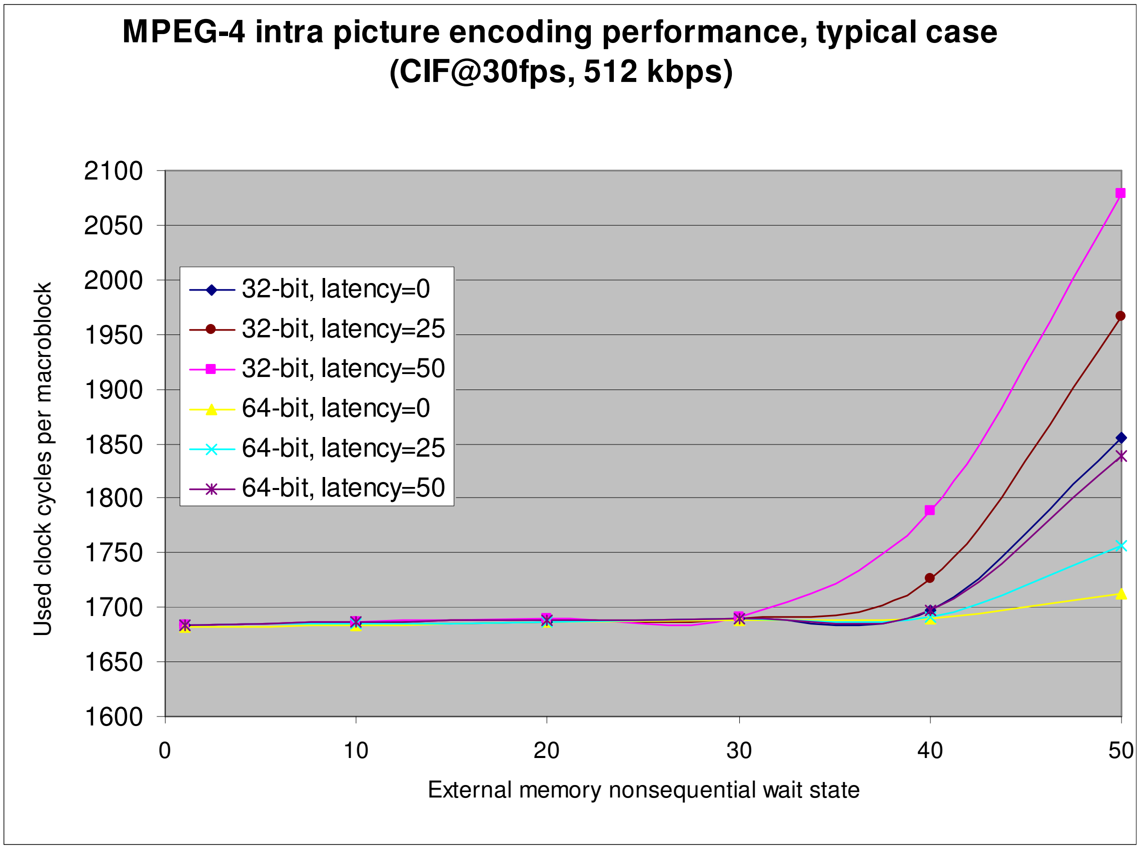

FIGURE 8. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON MPEG-4 INTRA FRAME ENCODING PERFORMANCE.

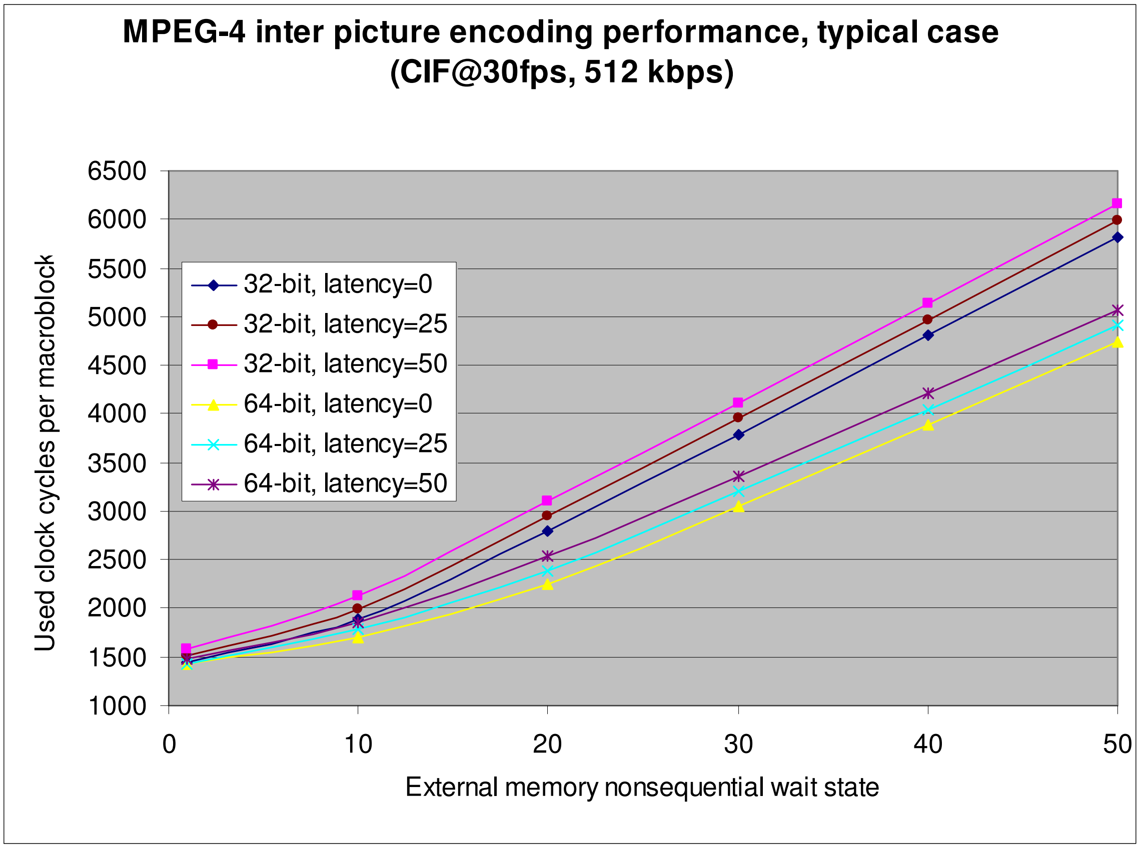

FIGURE 9. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON MPEG-4 ENCODING AND VIDEO STABILIZATION PERFORMANCE.

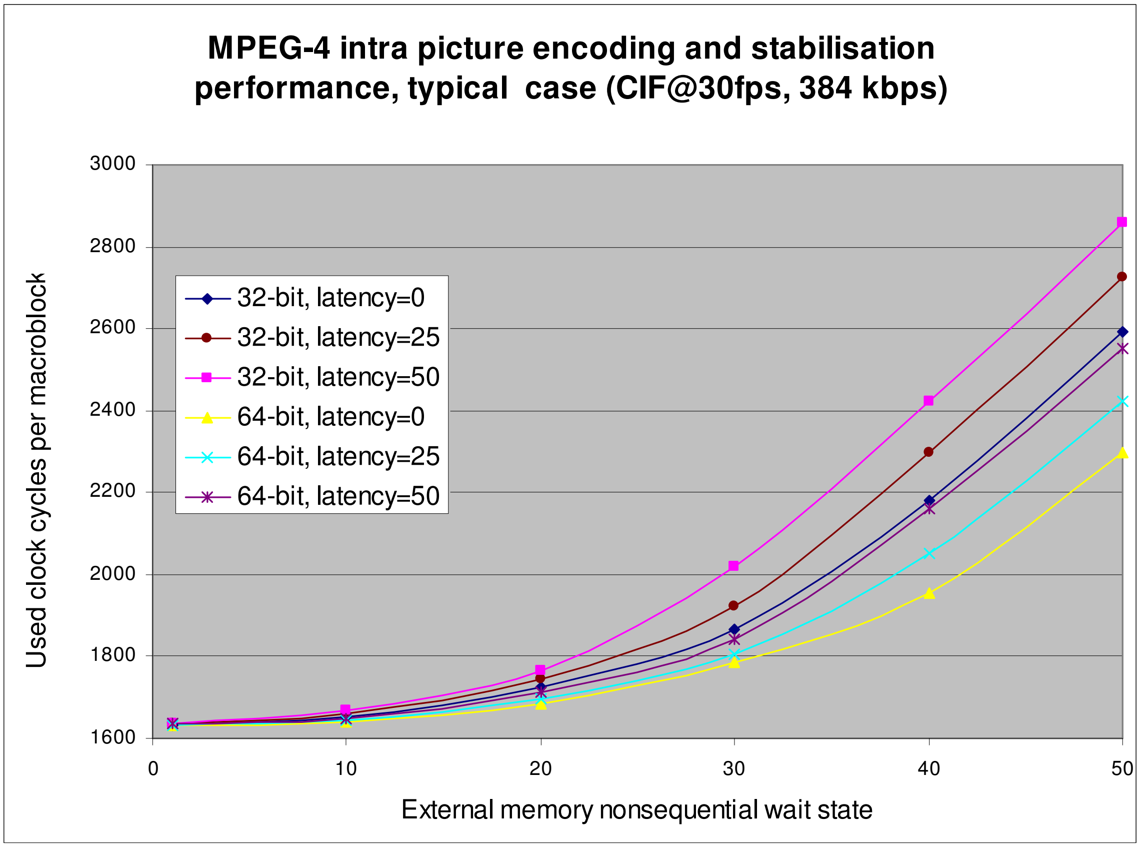

FIGURE 10. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON MPEG-4 INTRA FRAME ENCODING AND VIDEO STABILIZATION PERFORMANCE.

As shown in Figure 11, the low amount of data transferred during JPEG encoding makes it very resilient to latency.

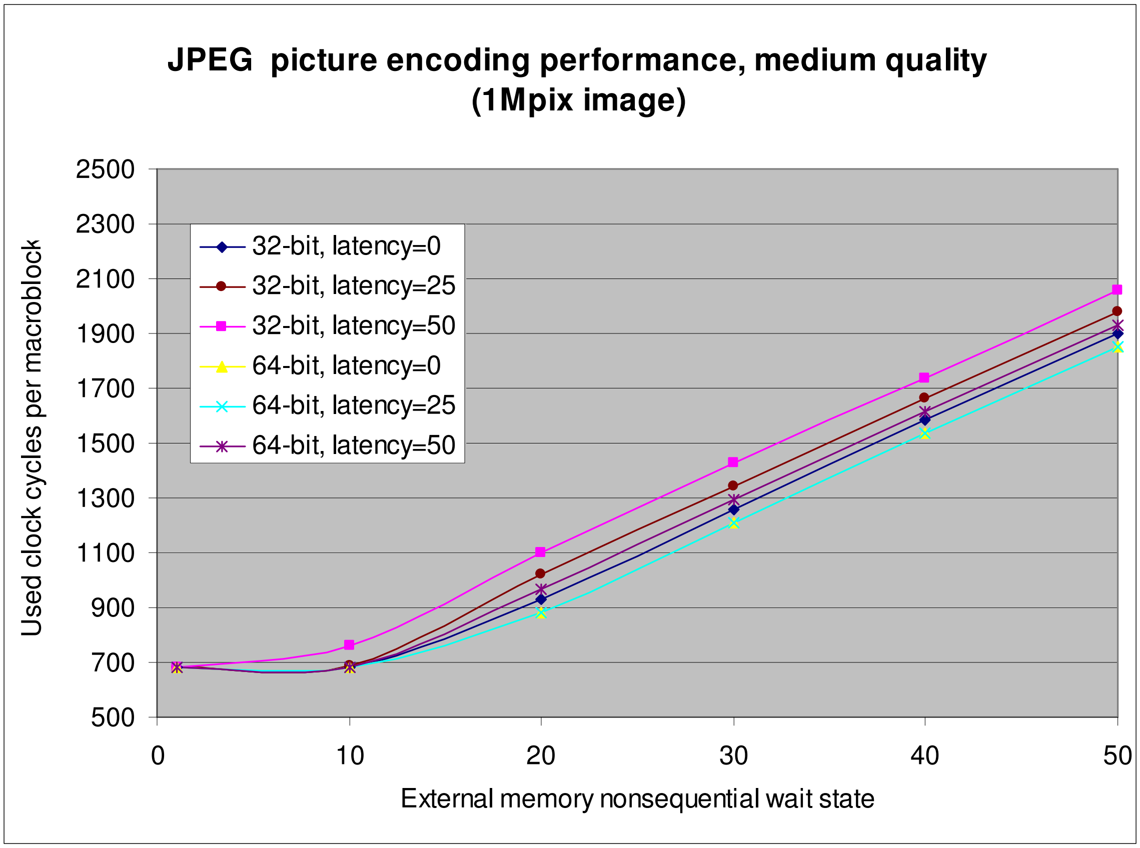

FIGURE 11. EFFECT OF THE LATENCY AND MEMORY WAIT STATES ON JPEG ENCODING PERFORMANCE.

3.4 External memory requirement

The external memory requirement of the product is formed of several components: The control software memory footprint, the stream buffer, the internal buffering of the encoder software, the SW/HW interface, the reference pictures and an input image.

Video stabilization requires that two input picture are buffered to external memory to enable general movement calculation between pictures.

Each item in memory space needed by hardware has to be linearly allocated to a contiguous memory block. Also, HW performance may increase if fast on-chip memory has been allocated for it. Especially the access time of the reference picture buffer is critical for the performance. Memory allocation issues are further handled in the Software Integration Guide [5].

3.5 Bus usage

In this paragraph the data flow, bus usage and bus load of H.264, MPEG-4 and JPEG encoding is presented. Please note that figures are presented only for the actual encoding process and video stabilization. When analyzing overall bus load, the effect of items listed below should be considered. These items are not taken into account in this document. More information can be found in SW integration guide. [5]

- Transferring the input image from the camera interface etc. to memory
- Internal software transfers
- Any other bus traffic e.g. audio encoding

3.5.1 Encoder data flow

Figure 12 illustrates the encoder data flow in H.264 encoding mode. Some of the transfers are omitted in MPEG-4 and JPEG encoding modes which are thus causing less bus load than H.264 encoding. Otherwise the encoder behaves in a similar way in each mode.

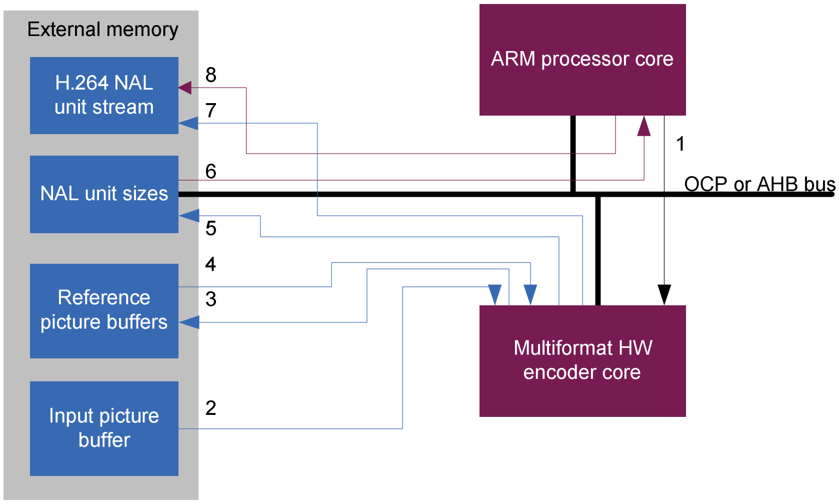

FIGURE 12. 7280 ENCODER AND EXTERNAL MEMORY DATA FLOW.

The numbers presented in figure 8 represent the following transactions:

1. Memory mapped register writes and reads
2. Input image read
3. Reference image write (not included in JPEG)
4. Reference image read (not included in JPEG)
5. NAL sizes write from HW
6. NAL sizes read to SW
7. Output byte or NAL unit stream write from HW
8. Output byte or NAL unit stream headers write from SW

NOTE 1: The encoder can be run in slice mode when encoding multimillion pixel JPEG images. In this case each slice behaves from the hardware's point of view as it was a separate picture. This approach keeps the required SW/HW interface memory in reasonable limits.

NOTE 2: If Video Stabilization is enabled two input picture buffers are needed. Reading is simultaneous with encoding read process. Stand alone Video Stabilization mode only two input picture buffers are required and 7280 encoder returns address offset for stabilized picture.

The encoder software starts encoding the first picture by initializing hardware and writing the stream headers. After HW has encoded the image, SW calculates new quantization values for HW, and initializes HW again.

3.5.2 Bus requesting

The encoder requests bus separately for each transaction, and some actions require several requests per macroblock. During one bus grant (or a memory accessing event), the encoder may make from one to 16 bursts to external memory.

3.5.3 Burst types and the amount of non-sequential accessing

A burst is defined as a combination of one non-sequential access and a variable number of sequential accesses that follow immediately. The length of a burst in a transfer may vary from 1 to 32, depending on the transaction. In VLC data write action the burst length varies also depending on the stream content.

The average amount of burst types in each encoding mode and video stabilization configuration is shown in Table 27 - Table 31. Maximum burst length is set to 16. The figures given in this paragraph are based on bus access monitoring during RTL simulation when running typical test cases. When compared typical case to high bit rate case, only difference is stream amount, which has only small effect to total busload.

NOTE: The encoder may issue bursts of any length between 1 and 31 in an OCP system. In AXI system burst length may vary between 1 and 16 . In AHB, the burst types are restricted to 1, 4, 8 and 16. In the following tables, the AHB burst types are used for clarity. The bus load figures given in this paragraph depict a typical case, and are based on the CIF resolution Knightshields video sequence and on a one megapixel JPEG reference picture.

TABLE 27. AVERAGE AMOUNT OF BURST TYPES PER MACROBLOCK IN H.264 ENCODING

Transfer type	32-bit bus	64-bit bus
Read SINGLE bursts	0.00	2.28
Write SINGLE bursts	0.00	0.00
Read INCR (undefined, < 4) bursts	10.53	7.84
Write INCR (undefined, < 4) bursts	0.00	8.44
Read INCR4 bursts	7.69	19.16
Write INCR4 bursts	8.00	0.00
Read INCR8 bursts	19.16	0.00
Write INCR8 bursts	0.00	0.36
Read INCR16 bursts	11.55	5.77
Write INCR16 bursts	4.02	1.72
Total amount of non-sequential transfers	60.96	45.60
Total amount of sequential transfers	425.51	197.13

TABLE 28. AVERAGE AMOUNT OF BURST TYPES PER MACROBLOCK IN H.264 ENCODING AND VIDEO STABILIZATION ENABLED

Transfer type	32-bit bus	64-bit bus
Read SINGLE bursts	0.00	2.86
Write SINGLE bursts	0.00	0.00
Read INCR (undefined, < 4) bursts	5.89	2.03
Write INCR (undefined, < 4) bursts	0.00	8.44
Read INCR4 bursts	1.80	37.28
Write INCR4 bursts	8.00	0.00
Read INCR8 bursts	37.28	0.68
Write INCR8 bursts	0.00	0.36
Read INCR16 bursts	12.24	5.77
Write INCR16 bursts	4.03	1.72
Total amount of non-sequential transfers	69.26	59.17
Total amount of sequential transfers	540.66	244.87

TABLE 29. AVERAGE AMOUNT OF BURST TYPES PER MACROBLOCK IN MPEG-4 ENCODING

Transfer type	32-bit bus	64-bit bus
Read SINGLE bursts	0.00	2.28
Write SINGLE bursts	0.00	0.00
Read INCR (undefined, < 4) bursts	10.77	8.14
Write INCR (undefined, < 4) bursts	0.00	8.00
Read INCR4 bursts	7.99	17.08
Write INCR4 bursts	8.00	0.00
Read INCR8 bursts	17.08	0.00
Write INCR8 bursts	0.00	0.01
Read INCR16 bursts	11.55	5.77
Write INCR16 bursts	4.01	2.00
Total amount of non-sequential transfers	59.42	43.30
Total amount of sequential transfers	411.90	191.90

TABLE 30. AVERAGE AMOUNT OF BURST TYPES PER MACROBLOCK IN MPEG-4 ENCODING AND VIDEO STABILIZATION ENABLED

Transfer type	32-bit bus	64-bit bus
Read SINGLE bursts	0.00	2.86
Write SINGLE bursts	0.00	0.00
Read INCR (undefined, < 4) bursts	5.76	1.89
Write INCR (undefined, < 4) bursts	0.00	8.00
Read INCR4 bursts	1.67	35.66
Write INCR4 bursts	8.00	0.00
Read INCR8 bursts	35.66	0.68
Write INCR8 bursts	0.00	0.00
Read INCR16 bursts	12.24	5.77
Write INCR16 bursts	4.00	2.00
Total amount of non-sequential transfers	67.35	56.90
Total amount of sequential transfers	528.42	240.08

TABLE 31. AVERAGE AMOUNT OF BURST TYPES PER MACROBLOCK IN JPEG ENCODING

Transfer type	32-bit bus	64-bit bus
Read SINGLE bursts	0.00	16.00
Write SINGLE bursts	0.00	0.00
Read INCR (undefined, < 4) bursts	16.00	16.00
Write INCR (undefined, < 4) bursts	0.00	0.00
Read INCR4 bursts	16.00	0.00
Write INCR4 bursts	0.00	0.00
Read INCR8 bursts	0.00	0.00
Write INCR8 bursts	0.00	0.25
Read INCR16 bursts	0.00	0.00
Write INCR16 bursts	0.25	0.00
Total amount of non-sequential transfers	32.25	32.25
Total amount of sequential transfers	67.88	17.81

3.5.4 Bus load

The memory access time in clock cycles for both accessing types is expressed as memory wait states. For example, the notation 15/1 stands for 15 clock cycle delay in non-sequential accessing and one cycle delay in sequential accessing. Bus load depends heavily on the memory wait states, as the encoder is reserving the bus while it is accessing memory. In the bus load estimates, Table 32 -Table 36, the wait state figures are the same for read access and write access. The bus load of hardware encoding is calculated using the total amount of non-sequential / sequential transfers presented in Table 27 –Table 31.

Note 1: OCP and AXI allow command pipelining and the issuing of bursts of any length which reduces the amount of bus load. The figures given in this paragraph are valid only for an AHB bus. OCP figures may be significantly lower.

Note 2: Data transfers of the encoder control software are not taken into account.

TABLE 32. BUS LOAD ESTIMATES FOR H.264 INTER PICTURE ENCODING

32-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	1.44	4.38	5.78	17.51	19.70	52.54	74.72
2/2	2.89	8.76	11.56	35.03	39.40	105.08	149.44
8/1	2.71	8.22	10.85	32.87	36.98	98.62	140.27
6/4	6.14	18.61	24.57	74.44	83.75	223.32	317.61
15/1	3.98	12.06	15.92	48.24	54.27	144.71	205.81
64-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	0.72	2.18	2.88	8.74	9.83	26.21	37.28
2/2	1.44	4.37	5.77	17.48	19.66	52.43	74.57
8/1	1.67	5.06	6.68	20.23	22.76	60.69	86.31
6/4	3.15	9.56	12.62	38.24	43.02	114.71	163.14

15/1	2.62	7.93	10.47	31.72	35.69	95.16	135.34

TABLE 33. BUS LOAD ESTIMATES FOR H.264 INTER PICTURE ENCODING AND VIDEO STABILIZATION ENABLED

32-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	1.81	5.49	7.25	21.96	24.70	65.87	93.68
2/2	3.62	10.98	14.49	43.91	49.40	131.74	187.37
8/1	3.25	9.85	13.01	39.41	44.34	118.23	168.15
6/4	7.66	23.20	30.63	92.82	104.42	278.45	396.01
15/1	4.69	14.22	18.77	56.86	63.97	170.59	242.62
64-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	0.90	2.74	3.61	10.95	12.31	32.84	46.70
2/2	1.81	5.47	7.22	21.89	24.63	65.67	93.40
8/1	2.13	6.46	8.53	25.86	29.09	77.57	110.32
6/4	3.96	12.01	15.85	48.04	54.05	144.13	204.98
15/1	3.36	10.19	13.45	40.77	45.86	122.30	173.94

TABLE 34. BUS LOAD ESTIMATES FOR MPEG-4 INTER PICTURE ENCODING

32-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	1.40	4.24	5.60	16.97	19.09	50.90	72.39
2/2	2.80	8.48	11.20	33.94	38.18	101.81	144.79
8/1	2.64	7.99	10.54	31.94	35.93	95.82	136.28
6/4	5.95	18.04	23.81	72.15	81.17	216.44	307.83
15/1	3.87	11.73	15.48	46.92	52.78	140.75	200.17
64-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	0.70	2.12	2.79	8.47	9.53	25.40	36.13
2/2	1.40	4.23	5.59	16.93	19.05	50.80	72.25
8/1	1.60	4.84	6.40	19.38	21.80	58.14	82.68
6/4	3.05	9.25	12.21	36.99	41.61	110.96	157.81
15/1	2.50	7.57	10.00	30.29	34.08	90.87	129.24

TABLE 35. BUS LOAD ESTIMATES FOR MPEG-4 INTER PICTURE ENCODING AND VIDEO STABILIZATION ENABLED

32-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	1.77	5.36	7.08	21.45	24.13	64.34	91.51
2/2	3.54	10.72	14.16	42.90	48.26	128.69	183.02
8/1	3.17	9.60	12.68	38.42	43.22	115.26	163.92
6/4	7.48	22.66	29.91	90.64	101.97	271.92	386.73
15/1	4.57	13.85	18.28	55.39	62.32	166.18	236.34

64-bit bus load [MHz]							
Wait states	QCIF 30 fps	QVGA 30 fps	CIF 30 fps	VGA 30 fps	SD 30 fps	720p 30 fps	SXGA 30 fps
1/1	0.88	2.67	3.53	10.69	12.03	32.07	45.62
2/2	1.76	5.35	7.06	21.38	24.06	64.15	91.23
8/1	2.06	6.26	8.26	25.03	28.16	75.09	106.80
6/4	3.87	11.72	15.46	46.86	52.72	140.59	199.94
15/1	3.25	9.84	12.99	39.37	44.29	118.11	167.97

TABLE 36. BUS LOAD ESTIMATES FOR JPEG ENCODING

Wait states	32-bit bus load [MHz]				
	1 Mpix	3 Mpix	5 Mpix	8 Mpix	16 Mpix
1/1	0.39	1.17	1.96	3.13	6.26
2/2	0.78	2.35	3.91	6.26	12.52
8/1	1.27	3.82	6.36	10.18	20.37
6/4	1.82	5.45	9.08	14.53	29.06
15/1	2.15	6.46	10.77	17.24	34.48
Wait states	64-bit bus load [MHz]				
1/1	0.20	0.59	0.98	1.56	3.13
2/2	0.39	1.17	1.96	3.13	6.26
8/1	1.08	3.23	5.39	8.62	17.24
6/4	1.03	3.10	5.17	8.27	16.55
15/1	1.96	5.88	9.80	15.67	31.35

4 Integration of the Product

This chapter guides through the actual hardware integration steps describing the RTL hierarchy and the bus and memory interfaces of the product. In addition this chapter covers the common synthesis related issues. Also the memory-mapped registers are described in this chapter.

Main steps of the hardware integration are:

- setting up the interface connections
- integrating the technology specific internal memory components
- integrating the technology specific clock gating component
- running the provided test cases for sanity checking

4.1 7280 Encoder RTL hierarchy

The RTL hierarchy of the product (version with AHB bus interface) is presented in Figure 13. Architecture names in parenthesis describe the type of file. STR stands for structure, i.e. the file contains only signal mappings, no logic. RTL codes contain the synthesizable logic.

The SRAM component is BEHAV type, meaning that it is a behavioral models only and do not synthesize. The *encoder_config_pkg* package file contains product configuration.

Note: In a Verilog RTL release there is no visible package file, as the configuration parameters are merged to the RTL at the time of VHDL to Verilog conversion.

```
ahb7280enc (str)
    + ahbwmife (rtl)
    + ahbwsife (rtl)
    + clkctrle (rtl)
        + clkgate (rtl)
    + hw7280enc (str)
        + bse (rtl)
        + busife (rtl)
            + busifeag (str)
                + busifeagaa (rtl)
                    + busifeagaaitest (rtl)
                    + hmulss (rtl)
                + busifeagctr (rtl)
                + busifeagrd (rtl)
                    + busifeagrdnbbuf (rtl)
                    + busifeagrdsach (rtl)
                + busifeagsa (rtl)
                + busifeagwd (rtl)
                + busifeagwide (rtl)
            + busifeswr (rtl)
                + encoder_config_pkg (PACKAGE)
        + dcprede (str)
            + dcprede_ctrl (rtl)
            + dcprede_dcdir (rtl)
            + dcprede_dcp (rtl)
            + dcprede_dcram_ctrl (rtl)
        + filtere (str)
            + filterecalc (rtl)
            + filterectrl (rtl)
            + filteretrans (rtl)
                + encoder_config_pkg (PACKAGE)
        + firme (str)
            + firmehintpfsm (rtl)
            + firmeinputbuf (rtl)
            + firmeintpfilter (rtl)
            + firmemainfsm (rtl)
            + firmeqpixfsm (rtl)
            + firmesaaddrgen (rtl)
            + firmesaqaddrgen (rtl)
            + firmesaqif (rtl)
            + firmevintp1fsm (rtl)
            + firmevintp2fsm (rtl)
        + ige (rtl)
        + ipe (str)
            + ipedir (str)
                + ipedircalc (rtl)
                + ipedirctrl (rtl)
                + ipedirpixels (rtl)
            + ipepred (str)
                + ipepredcalc (rtl)
                + ipepredctrl (rtl)
                + ipepredpixels (rtl)
        + mfsme (rtl)
        + mopee (str)
            + mopeefsm (rtl)
            + mopeesad1p (rtl)
        + mqpee (str)
            + mqpeectrl (rtl)
            + mqpeemad (rtl)
            + mqpeembtsel (rtl)
            + mqpeemux (rtl)
            + mqpeesad (str)
                + mqpeesadcalc (rtl)
                + mqpeesadsel (str)
                    + mqpeesad16x16 (rtl)
                    + mqpeesad16x8 (rtl)
                    + mqpeesad4x4 (rtl)
                    + mqpeesad4x8 (rtl)
                    + mqpeesad8x16 (rtl)
                    + mqpeesad8x4 (rtl)
                    + mqpeesad8x8 (rtl)
                    + mqpeesadmux (rtl)
        + mve (str)
            + mvead (rtl)
            + mvepme (rtl)
            + mvepmvd (rtl)
        + rce (rtl)
        + sce (str)
```

```
            + scectr (rtl)
            + scerlc (rtl)
        + stbe (str)
            + stbeinterp (rtl)
            + stbenbram (rtl)
            + stbenfnbram (rtl)
            + stbeonepixfilt (rtl)
            + stbestbram (rtl)
            + stbesumcalc (rtl)
        + tfe (str)
            + tfechbuf (rtl)
            + tfechfilt (rtl)
            + tfectrl (rtl)
            + tfeisum (rtl)
        + tqe (rtl)
            + regbank (rtl)
            + tqecalc (rtl)
                + hmulss (rtl)
            + tqectrl (rtl)
            + tqeiq (rtl)
                + hmulss (rtl)
            + tqeq (rtl)
                + hmulss (rtl)
            + encoder_config_pkg (PACKAGE)
        + vlce (str)
            + vlce_checkzero (rtl)
            + vlce_cntrl (rtl)
            + vlce_fsm_h264 (rtl)
                + encoder_config_pkg (PACKAGE)
            + vlce_mbin (rtl)
            + vlce_mpeg4_jpeg (str)
                + vlce_mpeg4_dctad (rtl)
                + vlce_mpeg4_jpeg_fsm (rtl)
                + vlce_mpeg4_jpeg_stream (rtl)
                + vlce_mpeg4_jpeg_tables (str)
                    + vlce_jpeg_tblvlcch (rtl)
                    + vlce_jpeg_tblvlclu (rtl)
                    + vlce_mpeg4_jpeg_tbls (rtl)
                    + vlce_mpeg4_tblmvdvlc (rtl)
                    + vlce_mpeg4_tblsmb (rtl)
                    + vlce_mpeg4_tblvlc (rtl)
            + vlce_obuff (rtl)
            + vlce_omem (rtl)
            + vlce_tables_h264 (rtl)
    + rse (rtl)
        + rseram (rtl)
            + vsram_dynamic (behav)
            + encoder_config_pkg (PACKAGE)
```

Figure 13. 7280 Encoder RTL hierarchy for AHB bus interface.

All RTL files can be found in *7280_encoder/hardware/code/vhdl* or *7280_encoder/hardware/code/verilog* folder.

Due to the configurable target bus environment, the product can be delivered with various top-level and bus interface wrapper codes.

If the product will be connected to an OCP bus, the delivered files are:

- Top-level *ocp7280enc*
- Wrapper for master interface *ocpwmife*
- Wrapper for slave interface *ocpwsife*

If the product will be connected to an AHB bus, the delivered files are:

- Top-level *ahb7280enc*
- Wrapper for master interface *ahbwmife*
- Wrapper for slave interface *ahbwsife*

If the product will be connected to an AXI bus, the delivered files are:

- Top-level *axi7280enc*
- Wrapper for master interface *axiwmife*
- Wrapper for slave interface *axiwsife*

If the product will be connected to an AXI master bus and an APB slave bus, the delivered files are:

- Top-level *axiapb7280enc*
- Wrapper for master interface *axiwmife*
- Wrapper for slave interface *apbwsife*

If the product will be connected to an AHB master bus and an APB slave bus, the delivered files are:

- Top-level *ahbapb7280enc*
- Wrapper for master interface *ahbwmife*
- Wrapper for slave interface *apbwsife*

If the product will be connected to an AXI master bus and an AHB slave bus, the delivered files are:

- Top-level *axiahb7280enc*
- Wrapper for master interface *axiwmife*
- Wrapper for slave interface *ahbwsife*

On the same hierarchy level with the bus wrappers and encoder core there is the wrapper for embedded memories, *rse* and the clock gating control block *clkctrle*.

The encoder core is partitioned into the following logical modules:

- Bus-independent encoder core top-level *hw7280enc*
- Video stabilization *stbe*
- RLC and scan encoder *sce*
- Integer transform and DCT encoder *tqe*
- Intra prediction module *ipe*
- DC prediction module *dcprede*

- Transform feeder *tfe*
- One pixel motion estimation *mopee*
- Index generation block *ige*
- Quarter pixel FIR block *firme*
- Quarter pixel motion estimation *mqpee*
- Bus wrapper interface *busife* performing data transferring between the encoder and the external memory
- In-loop deblocking filter *filtere*
- Differential motion vector predictor *mvde*
- Main control unit *mfsme*
- Boundary strength encoder *bse*
- Rate control *rce*
- Variable length coding block *vlce*

Most of the described blocks contain sub-blocks. On the lowest level of hierarchy there is a signed multiplier component *hmulss*. This has been separated as a component for ensuring minimal logic area. This issue is handled in chapter 0.

4.2 Master and slave interfaces

The encoder has two bus interfaces, master and slave.

The purpose of the master interface is to allow the encoder hardware an access to the external memory for reading e.g. the input image data and reference picture data, and for writing the output stream data.

The purpose of the encoder slave interface is to allow the CPU an access to the memory-mapped registers of the encoder for writing and reading the control and configuration data, enabling and resetting the encoder and reading the interrupt and status flags.

4.3 Connecting encoder to an AHB interface

The encoder can also be connected to a 32-bit or 64-bit single-layer or a multi-layer AHB bus. The master interface data bus width is selected using the *buswidth* interface signal. The slave interface data width is always 32 bits. The encoder hardware interface for AHB is described in Figure 14. The AMBA AHB signals and instructions about the master and slave interface integration to AHB bus are described in the AMBA specification [7].

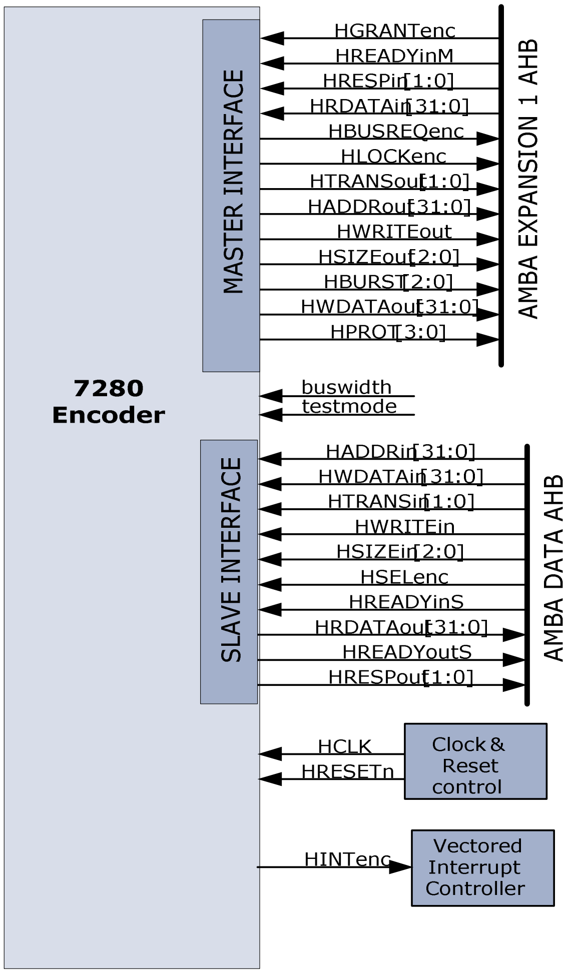

FIGURE 14. 7280 AHB INTERFACE.

4.3.1 Non-AHB interface signals

The encoder includes two input pins, *buswidth* and *testmode,* that are not related to the AHB specification.

Buswidth signal determines the data bus width. When the *buswidth* is connected to zero, the encoder assumes a 32-bit master interface data bus. When the *buswidth* is tied high, the AHB data bus has to be 64 bits. *Buswidth* does not affect the slave interface width; 32-bit interface is always assumed. It may be possible to remove the unnecessary data width logic inside the encoder by setting the *buswidth* to a constant value during synthesis.

Setting the *testmode* high will cause bypassing of the clock gating. This mode can be used to reach a better test coverage. In default operation mode the *testmode* must be tied to zero.

4.3.2 AHB master interface signals

In the following tables, the encoder master interface input and output signals are described. The data bus of the AHB master interface is either 32 or 64 bits.

TABLE 37. AHB MASTER INTERFACE INPUT SIGNALS

Input signals:	From:	Synchronous to clocks:	Function:
HCLK	Clock source		This clock times all bus transfers. All signal timings are related to the rising edge of *HCLK*.
HRESETn	Reset control	Asynchronous	The reset signal is active low and is used to reset the system and the bus. This is the only active low signal.
HGRANTenc	Expansion AHB (Arbiter)	HCLK	This signal indicates that the encoder is currently the highest priority master. Ownership of the address / control signals changes at the end of a transfer when *HREADY* is high, so a master gets access to the bus when both *HREADY* and *HGRANTenc* are high.
HREADYinM	Expansion AHB (Slave)	HCLK	When high, the *HREADY* signal indicates that a transfer has finished on the bus. This signal may be driven low to extend a transfer. *NOTE: Slaves on the bus require HREADY as both an input and an output signal.*
HRESPin[1:0]	Expansion AHB (Slave)	HCLK	The transfer response provides additional information on the status of a transfer. Four different responses are provided, OKAY, ERROR, RETRY and SPLIT.
HRDATAin0 [31:0]	Expansion AHB (Slave)	HCLK	The read data bus is used to transfer data from bus slaves to the bus master during read operations. Contains data bits [31:0].
HRDATAin1 [31:0]	Expansion AHB (Slave)	HCLK	The read data bus is used to transfer data from bus slaves to the bus master during read operations. Contains data bits [63:32]. *NOTE: If data bus width is set to 32 bits, this bus should be connected to 0*
buswidth	Vcc or ground	HCLK	When high, configures hardware to use data bus width of 64 bits. When low, data bus width is set to 32 bits. *NOTE: This input should be connected to constant 1 or 0*

TABLE 38. AHB MASTER INTERFACE OUTPUT SIGNALS

Output signals:	To:	Synchronous to clocks:	Function:
HBUSREQenc	Expansion AHB (Arbiter)	HCLK	A signal from bus master encoder to the bus arbiter, which indicates that the bus master requires the bus.
HLOCKenc	Expansion AHB (Arbiter)	HCLK	When high this signal indicates that the master requires locked access to the bus and no other master should be granted the bus until this signal is low. *NOTE: Fixed value '0' is used*
HTRANSout [1:0]	Expansion AHB (Arbiter)	HCLK	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL or IDLE.
HADDRout [31:0]	Expansion AHB (Encoder)	HCLK	The 32-bit system address bus.
HWRITEout	Expansion AHB (Slave)	HCLK	When high this signal indicates a write transfer and when low a read transfer.
HSIZEout[2:0]	Expansion AHB (Slave)	HCLK	Indicates the size of the transfer, which is typically byte (8-bit), half word (16-bit), word (32-bit) or double word (64-bit). *NOTE: Fixed value "010" is used for 32-bit bus, and "011" is used for 64-bit bus*
HBURST[2:0]	Expansion AHB (Arbiter, Slave)	HCLK	Indicates the burst mode of the transfer. Encoder supports the following values (wrapping bursts not supported): "000" = SINGLE "001" = INCR "011" = INCR4 "101" = INCR8 "111" = INCR16 *NOTE: Maximum burst length can be defined in the memory-mapped registers.*
HWDATAout0 [31:0]	Expansion AHB (Slave)	HCLK	The write data bus is used to transfer data from the master to the bus slaves during write operations. Contains data bits [31:0]
HWDATAout1 [31:0]	Expansion AHB (Slave)	HCLK	The write data bus is used to transfer data from the master to the bus slaves during write operations. Contains data bits [63:32]. *NOTE: this data bus is valid only if data bus width is set to 64 bits*
HPROT[3:0]	Expansion AHB (Slave)	HCLK	The protection control provides additional information about a bus access and is primarily intended to be used by any module that wishes to implement some level of protection. *NOTE: Fixed value "0001", no protections, is used*

4.3.3 AHB slave interface signals

In the following tables, the encoder slave interface input and output signals are described. The data bus of the AHB slave interface is 32 bits. The slave interface does not support burst accesses.

TABLE 39. AHB SLAVE INTERFACE INPUT SIGNALS

Input signals:	From:	Synchronous to clocks:	Function:
HCLK	Clock source		This clock times all bus transfers. All signal timings are related to the rising edge of the *HCLK*.
HRESETn	Reset control	Asynchronous	The bus-reset signal is active low and is used to reset the system and the bus. This is the only active low signal.
HSELenc	Address Decoder	HCLK	Each AHB slave has its own slave select signal and this signal indicates that the current transfer is intended for the selected slave. This signal is simply a combinatorial decode of the address bus.
HWDATA[31:0]	Master	HCLK	The write data bus is used to transfer data from the master to the bus slaves during write operations.
HADDR[31:0]	Master	HCLK	The 32-bit system address bus.
HTRANS[1:0]	Master	HCLK	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE or BUSY.
HWRITE	Master	HCLK	When high this signal indicates a write transfer and when low a read transfer.
HREADYinS	Master	HCLK	When high, the *HREADY* signal indicates that a transfer has finished on the bus.
HSIZE[2:0]	Master	HCLK	Indicates the size of the transfer, which is typically byte (8-bit), half word (16-bit), or word (32-bit). The protocol allows for larger transfer sizes up to a maximum of 1024 bits.

TABLE 40. AHB SLAVE INTERFACE OUTPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
HRDATAout [31:0]	Master	HCLK	The read data bus is used to transfer data from bus slaves to the bus master during read operation.
HREADYoutS	Master	HCLK	When high, the *HREADY* signal indicates that a transfer has finished on the bus. This signal may be driven lowe to extend a transfer. Slaves on the bus require *HREADY* as both an input and an output signal.
HRESPout[1:0]	Master	HCLK	The transfer response provides additional information on the status of a transfer. Four different responses are provided, OKAY, ERROR, RETRY and SPLIT. *NOTE: Fixed value OKAY is used*
HINTenc	Interrupt controller	HCLK	Hardware interrupt line. Active high.

4.4 Connecting encoder to an AXI interface

The encoder can also be connected to a 32-bit or 64-bit single-layer or a multi-layer AXI bus. The master interface data bus width is selected using the *buswidth* interface signal. The slave interface data width is always 32 bits. The AMBA AXI signals and instructions about the master and slave interface integration to AXI bus are described in the AMBA 3 specification [7].

4.4.1 Non-AXI interface signals

The encoder includes two input pins, *buswidth* and *testmode,* that are not related to the AXI specification.

Buswidth signal determines the data bus width. When the *buswidth* is connected to zero, the encoder assumes a 32-bit master interface data bus. When the *buswidth* is tied high, the AXI data bus has to be 64 bits. *Buswidth* does not affect the slave interface width; 32-bit interface is always assumed. It may be possible to remove the unnecessary data width logic inside the encoder by setting the *buswidth* to a constant value during synthesis.

Setting the *testmode* high will cause bypassing of the clock gating. This mode can be used to reach better test coverage. In default operation mode the *testmode* must be tied to zero.

4.4.2 AXI master interface signals

In the following tables, the encoder master interface input and output signals are described. The data bus of the AXI master interface is either 32 or 64 bits.

TABLE 41. AXI MASTER INTERFACE INPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
ACLK	Clock source		Global clock signal. All signals are sampled on the rising edge of the global clock.
ARESETn	Reset source		Asynchronous reset, active low.
AWREADY	Target slave	ACLK	Write address ready. Indicates when slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready
WREADY	Target slave	ACLK	Write ready. Indicates when slave is ready to accept write data: 1 = slave ready 0 = slave not ready
BID[7:0]	Target slave	ACLK	Response ID. The indication tag of the write response (not used, transactions are processed in the order they are issued).
BRESP[1:0]	Target slave	ACLK	Write response. “00” = OKAY, access has been successful “01” = reserved (EXOKAY, exclusive access not supported) “10” = SLVERROR, slave error “10” = DECERROR , decode error
BVALID	Target slave	ACLK	Write response valid. Write response is available. 1 = write response available 0 = write response not available
ARREADY	Target slave	ACLK	Read address ready. Indicates when slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready
RID[7:0]	Target slave	ACLK	Read ID tag. The ID tag of the read data group signals (not used, transactions are processed in the order they are issued).
RDATA[63:0]	Target slave	ACLK	Read data. *Note: if buswidth is set to zero, only bits [31:0] are used.*
RRESP[1:0]	Target slave	ACLK	Read response. “00” = OKAY, access has been successful “01” = reserved (EXOKAY, exclusive access not supported) “10” = SLVERROR, slave error “10” = DECERROR , decode error
RLAST	Target slave	ACLK	Read last. Indicates the last transfer in a read burst.
RVALID	Target	ACLK	Read valid. Indicates that the required

	slave		read data is available. 1 = read data available 0 = read data not available

TABLE 42. AXI MASTER INTERFACE OUTPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
AWIDout[7:0]	Target slave	ACLK	Write address ID. The indication tag for the write address group of signals. Fixed value used for every access. *Note: the ID value is configurable through the memory-mapped registers.*
AWADDRout[31:0]	Target slave	ACLK	Write address. The write address bus gives the address of the first transfer in a write burst transaction. The associated control signals are used to determine the address of the remaining transfers in the burst.
AWLENout [3:0]	Target slave	ACLK	Burst length. The burst length gives the exact number of transfers in a write burst.
AWSIZEout [2:0]	Target slave	ACLK	Burst size. The size of each transfer in a write burst. Supported values: "010" = 4 bytes in transfer "011" = 8 bytes in transfer
AWBURSTout [1:0]	Target slave	ACLK	Burst type. "00" = FIXED, fixed-address burst "01" = INCR, incrementing-address burst "10" = WRAP, incrementing-address burst that wraps to a lower address at the wrap boundary "11" = Reserved *Note: only INCR type is supported; value is fixed to "01".*
AWLOCKout [1:0]	Target slave	ACLK	Lock type. "00" = Normal access "01" = Exclusive access "10" = Locked access "11" = Reserved *Note: Exclusive or locked accesses are not supported; value is fixed to "00".*
AWCACHEout [3:0]	Target slave	ACLK	Cache type. *Note: Not supported; value is fixed to "0000".*
AWPROTout [2:0]	Target slave	ACLK	Protection type. *Note: Not supported; value is fixed to "0000".*
AWVALIDout	Target slave	ACLK	Write address valid. Indicates that write address and control information are available. 1 = address and control information

			available 0 = address and control information not available The address and control information remain stable until *AWREADY* goes high
WIDout [7:0]	Target slave	ACLK	Write ID tag. The indication tag for the write data transfer. Fixed value used for every access. *Note: the ID value is configurable through the memory-mapped registers.*
WDATAout [63:0]	Target slave	ACLK	Write data. *Note: if buswidth is set to zero, only bits [31:0] are used.*
WSTRBout [7:0]	Target slave	ACLK	Write strobes. Indicates which bytes to be written in to memory: “00001111” , 32 bit environment “11111111” , 64 bit environment
WLASTout	Target slave	ACLK	Write last. The last transfer in a write burst.
WVALIDout	Target slave	ACLK	Write valid. Indicates that valid write data and strobes are available. 1 = write data and strobes available 0 = write data and strobes not available
BREADYout	Target slave	ACLK	Response ready. Master can accept the response information. 1 = master ready 0 = master not ready
ARIDout [7:0]	Target slave	ACLK	Read address ID. The indication tag for the read address group of signals. Fixed value used for every access. *Note: the ID value is configurable through the memory-mapped registers.*
ARADDR[31:0]	Target slave	ACLK	Read address. The read address bus gives the address of the first transfer in a read burst transaction. The associated control signals are used to determine the address of the remaining transfers in the burst.
ARLENout [3:0]	Target slave	ACLK	Burst length. The burst length gives the exact number of transfers in a read burst.
ARSIZEout [2:0]	Target slave	ACLK	Burst size. The size of each transfers in a read burst. Supported values: “010” = 4 bytes in transfer “011” = 8 bytes in transfer
ARBURSTout [1:0]	Target slave	ACLK	Burst type. “00” = FIXED, fixed-address burst “01” = INCR, incrementing-address burst “10” = WRAP, incrementing-address burst that wraps to a lower address at the wrap boundary “11” Reserved Only INCR type supported, fixed to value “01”.
ARLOCKout [1:0]	Target slave	ACLK	Lock type. “00” = Normal access “01” = Exclusive access

			"10" = Locked access "11" = Reserved *Note: Exclusive or locked accesses are not supported; value is fixed to "00".*
ARCACHEout [3:0]	Target slave	ACLK	Cache type. *Note: Not supported; value is fixed to "0000".*
ARPROTout [2:0]	Target slave	ACLK	Protection type. *Note: Not supported; value is fixed to "0000".*
ARVALIDout	Target slave	ACLK	Read address valid. Indicates that write address and control information are available. 1 = address and control information available 0 = address and control information not available The address and control information remain stable until *ARREADY* goes high
RREADYout	Target slave	ACLK	Read ready. Master can accept the read data and response information. 1 = master ready 0 = master not ready

4.4.3 AXI slave interface signals

In the following tables, the encoder slave interface input and output signals are described. The data bus of the AXI slave interface is 32 bits. Burst lengths 1…16 are supported by the slave interface.

TABLE 43. AXI SLAVE INTERFACE INPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
ACLK	Clock source		Global clock signal. All signals are sampled on the rising edge of the global clock.
ARESETn	Reset source		Asynchronous reset, active low.
AWID[3:0]	Initiator master	ACLK	Write address ID. The indication tag for the write address group of signals. Not used, transactions are processed in the order they are issued.
AWADDR[31:0]	Initiator master	ACLK	Write address. The write address bus gives the address of the first transfer in a write burst transaction. The associated control signals are used to determine the address of the remaining transfers in the burst.
AWLEN[3:0]	Initiator master	ACLK	Burst length. The burst length gives the exact number of transfers in a write burst.
AWSIZE[2:0]	Initiator	ACLK	Burst size. The size of each transfer in a

	master		write burst. *Note: only value "010" = 4 bytes in transfers are supported. Slave wrapper gives SLVERR response (BRESPout) if other values used.*
AWBURST[1:0]	Initiator master	ACLK	Burst type. "00" = FIXED, fixed-address burst "01" = INCR, incrementing-address burst "10" = WRAP, incrementing-address burst that wraps to a lower address at the wrap boundary "11" = Reserved *Note: Only INCR type supported. Slave wrapper gives SLVERR response (BRESPout) if other values used.*
AWLOCK[1:0]	Initiator master	ACLK	Lock type. "00" = Normal access "01" = Exclusive access "10" = Locked access "11" = Reserved *Note: Exclusive or locked accesses not supported. Slave wrapper provides OKAY response (BRESP) to indicate the failure of an exclusive access.*
AWCACHE[3:0]	Initiator master	ACLK	Cache type. Not supported.
AWPROT[2:0]	Initiator master	ACLK	Protection type. Not supported.
AWVALID	Initiator master	ACLK	Write address valid. Indicates that write address and control information are available. 1 = address and control information available 0 = address and control information not available The address and control information remain stable until AWREADY goes high
WID[7:0]	Initiator master	ACLK	Write ID tag. The indication tag for the write data transfer. Not used, transactions are processed in the order they are issued.
WDATA[31:0]	Initiator master	ACLK	Write data.
WSTRB[3:0]	Initiator master	ACLK	Write strobes. Indicates which bytes to be written in registers.
WLAST	Initiator master	ACLK	Write last. The last transfer in a write burst.
WVALID	Initiator master	ACLK	Write valid. Indicates that valid write data and strobes are available. 1 = write data and strobes available 0 = write data and strobes not available
BREADY	Initiator master	ACLK	Response ready. Master can accept the response information. 1 = master ready 0 = master not ready

ARID[7:0]	Initiator master	ACLK	Read address ID. The indication tag for the read address group of signals. Not used, transactions are processed in the order they are issued.
ARADDR[31:0]	Initiator master	ACLK	Read address. The read address bus gives the address of the first transfer in a read burst transaction. The associated control signals are used to determine the address of the remaining transfers in the burst.
ARLEN[3:0]	Initiator master	ACLK	Burst length. The burst length gives the exact number of transfers in a read burst.
ARSIZE[2:0]	Initiator master	ACLK	Burst size. The size of each transfers in a read burst. Only value “010” = 4 bytes in transfer supported. Slave wrapper gives SLVERR response (BRESPout) if other values used.
ARBURST[1:0]	Initiator master	ACLK	Burst type. “00” = FIXED, fixed-address burst “01” = INCR, incrementing-address burst “10” = WRAP, incrementing-address burst that wraps to a lower address at the wrap boundary “11” Reserved *Note: only INCR type supported. Slave wrapper gives SLVERR response (BRESPout) if other values used.*
ARLOCK[1:0]	Initiator master	ACLK	Lock type. “00” = Normal access “01” = Exclusive access “10” = Locked access “11” = Reserved *Note: exclusive or locked accesses not supported. Slave wrapper provides OKAY response (BRESP) to indicate the failure of an exclusive access.*
ARCACHE[3:0]	Initiator master	ACLK	Cache type. Not supported.
ARPROT[2:0]	Initiator master	ACLK	Protection type. Not supported.
ARVALID	Initiator master	ACLK	Read address valid. Indicates that write address and control information are available. 1 = address and control information available 0 = address and control information not available The address and control information remain stable until ARREADY goes high
RREADY	Initiator master	ACLK	Read ready. Master can accept the read data and response information. 1 = master ready 0 = master not ready

TABLE 44. AXI SLAVE INTERFACE OUTPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
AWREADYout	Initiator master	ACLK	Write address ready. Indicates when slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready
WREADYout	Initiator master	ACLK	Write ready. Indicates when slave is ready to accept write data: 1 = slave ready 0 = slave not ready
BIDout [7:0]	Initiator master	ACLK	Response ID. The indication tag of the write response. Matches the AWID value.
BRESPout [1:0]	Initiator master	ACLK	Write response. "00" = OKAY, access has been successful "01" = reserved (EXOKAY, exclusive access not supported) "10" = SLVERROR, slave error "10" = DECERROR , decode error
BVALIDout	Initiator master	ACLK	Write response valid. Write response is available. 1 = write response available 0 = write response not available
ARREADYout	Initiator master	ACLK	Read address ready. Indicates when slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready
RIDout [7:0]	Initiator master	ACLK	Read ID tag. The ID tag of the read data group signals. Match to the ARID value.
RDATAout [31:0]	Initiator master	ACLK	Read data.
RRESPout [1:0]	Initiator master	ACLK	Read response. "00" = OKAY, access has been successful "01" = reserved (EXOKAY, exclusive access not supported) "10" = SLVERROR, slave error "10" = DECERROR , decode error
RLASTout	Initiator master	ACLK	Read last. Indicates the last transfer in a read burst.
RVALIDout	Initiator master	ACLK	Read valid. Indicates that the required read data is available. 1 = read data available 0 = read data not available
XINTenc	Interrupt controller	HCLK	Hardware interrupt line. Active high.

4.5 Connecting encoder to an APB slave interface

Optionally with the AHB or AXI master interface, encoder slave interface can be connected to an AMBA 3 APB interface. In the following tables the ABP slave interface input and output ports are listed.

TABLE 45. APB SLAVE INTERFACE INPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
PCLK	Clock source		Global clock signal. All signals are sampled on the rising edge of the global clock.
PRESETn	Reset source		Asynchronous reset, active low.
PADDR[31:0]	APB bridge	PCLK	The 32-bit system address bus.
PSEL	APB bridge	PCLK	The select signal indicates that this slave interface is selected and that data transfer is required.
PENABLE	APB bridge	PCLK	Enable signal indicates the second and subsequent cycles of an APB transfer.
PWRITE	APB bridge	PCLK	Direction. PWRITE indicates and APB write access when HIGH and an APB read access when LOW.
PWDATA[31:0]	APB bridge	PCLK	Write data.

TABLE 46. APB SLAVE INTERFACE OUTPUT SIGNALS

Input signals:	To:	Synchronous to clocks:	Function:
PREADY	Slave interface	ACLK	Ready signal. Encoder extends each read transfer for one clock cycle and each write transfer for three clock cycles with this signal.
PRDATA[31:0]	Initiator master	ACLK	Read data. Encoder drives this signal during read cycles when PREADY is HIGH.

4.6 Connecting encoder to an OCP interface

The encoder can be connected to an OCP compliant bus. Encoder hardware interface is described in Figure 15. The master interface data width is 64 bits, and the slave interface data width is 32 bits.

The OCP signals and instructions about the master and slave interface integration to OCP bus are described in the OCP specification [6].

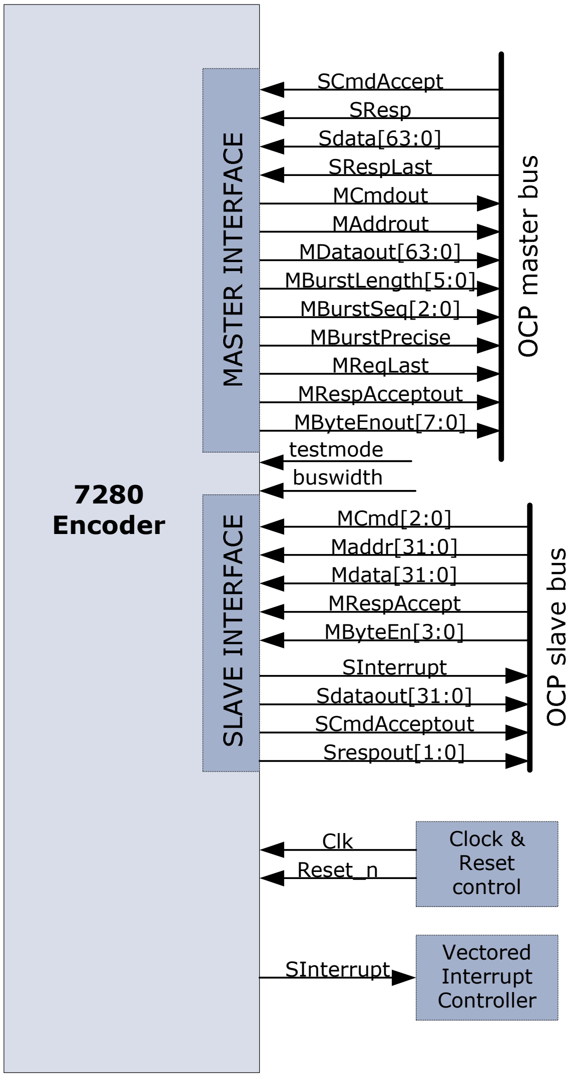

FIGURE 15. 7280 OCP INTERFACE.

4.6.1 Non-OCP interface signal

The encoder input pin *testmode* is not related to the OCP specification. Setting the *testmode* high will cause bypassing of the clock gating. This mode can be used to reach better test coverage. In default operation mode the *testmode* must be tied to zero.

Buswidth signal determines the data bus width. When the *buswidth* is connected to zero, the encoder assumes a 32-bit master interface data bus. When the *buswidth* is tied high, the OCP data bus has to be 64 bits. *Buswidth* does not affect the slave interface width; 32-bit interface is always assumed. It may be possible to remove the unnecessary data width logic inside the encoder by setting the *buswidth* to a constant value during synthesis.

4.6.2 OCP master interface signals

In the following tables, the encoder master interface input and output signals are described. The data bus of the OCP master interface is 64 bits.

TABLE 47. OCP MASTER INTERFACE INPUT SIGNALS

Input signals:	From:	Synchronous to clocks:	Function:
Clk	Clock source		This clock times all bus transfers. All signal timings are related to the rising edge of Clk.
Reset_n	Reset control	Asynchro-nous	Active low asynchronous reset.
SCmdAccept	Target slave	Clk	Target slave accepts transfer, active high.
SResp[1:0]	Target slave	Clk	Transfer response. "00" = No response "01" = Data valid "10" = Reserved "11" = Response error Encoder generates an interrupt in case of a response error and sets the bus error status flag.
SData[63:0]	Target slave	Clk	Read data from slave.
SRespLast	Target slave	Clk	Last response in burst, active high.

TABLE 48. OCP MASTER INTERFACE OUTPUT SIGNALS

Output signals:	To:	Synchronous to clocks:	Function:
MCmdout[2:0]	Target slave	Clk	Transfer command from master "000" = Idle "001" = Write "010" = Read "011" = ReadEx (not used) "100" = ReadLinked (not used) "101" = WriteNonPost (not used) "110" = WriteConditional (not used) "111" = Broadcast (not used)
MAddrout[31:0]	Target slave	Clk	Transfer address from master. 3 lowest bits are zeros.
MDataout[63:0]	Target slave	Clk	Write data from master.
MBurstLength [5:0]	Target slave	Clk	For precise bursts, the value indicates the total number of transfers in the burst, and is constant throughout the burst. For imprecise bursts, the value indicates the best guess of the number of transfers remaining (including the current request), and may change during

			the burst (with every request).
MBurstSeq[2:0]	Target slave	Clk	Burst address sequence "000" = INCR, incrementing "001" = DFLT1, custom, packed "010" = WRAP, wrapping "011" = DFLT2, custom, not packed "100" = XOR, exclusive "101" = STRM, streaming "110" = UNKN, unknown "111" = Reserved *Note: Fixed value "000" is used.*
MCmdout[2:0]	Target slave	Clk	Transfer command from master "000" = Idle "001" = Write "010" = Read "011" = ReadEx (not used) "100" = ReadLinked (not used) "101" = WriteNonPost (not used) "110" = WriteConditional (not used) "111" = Broadcast (not used)
MBurstPrecise	Target slave	Clk	Given burst length is precise, active high. *Note: Fixed value '1' is used.*
MReqLast	Target slave	Clk	Last request in burst, active high.
MRespAcceptout	Target slave	Clk	Master accepts the current response from the slave, active high. *Note: Fixed value '1' is used.*
MByteEnout[7:0]	Target slave	Clk	Byte enable. This field indicates which bytes in the OCP word are parts of the current transfer. There is one bit in *MByteEnout* for each byte of *MDataout*. *MByteEnout* is valid for both read and write type transfers. *Note: Fixed value "11111111" is used.*

4.6.3 OCP slave interface signals

In the following tables, the encoder slave interface input and output signals are described. The data bus of the OCP slave interface is 32 bits. The slave interface does not support burst accesses.

TABLE 49. OCP SLAVE INTERFACE INPUT SIGNALS

Input signals:	From:	Synchronous to clocks:	Function:
Clk	Clock source		This clock times all bus transfers. All signal timings are related to the rising edge of *Clk*.
Reset_n	Reset control	Asynchronous	Active low asynchronous reset.
MCmd[2:0]	Initiator master	Clk	Transfer command to slave interface *ocpwsifd*

			“000” = Idle “001” = Write “010” = Read “011” = ReadEx (not used) “100” = ReadLinked (not used) “101” = WriteNonPost (not used) “110” = WriteConditional (not used) “111” = Broadcast (not used)
MAddr[31:0]	Initiator master	Clk	Transfer address to slave interface. 3 lowest bits are zeros.
MData[63:0]	Initiator master	Clk	Write data to slave interface.
MRespAccept	Initiator master	Clk	Master accepts the current response from the slave, active high.
MByteEn[7:0]	Initiator master	Clk	Byte enable. This field indicates which bytes in the OCP word are parts of the current transfer. There is one bit in *MByteEn* for each byte of *MData*. *MByteEn* is valid for both read and write type transfers.

TABLE 50. OCP SLAVE INTERFACE OUTPUT SIGNALS

Output signals:	To:	Synchronous to clocks:	Function:
SCmdAcceptout	Initiator master	Clk	Slave accepts transfer, active high. *NOTE: Fixed value '1' is used.*
SRespout[1:0]	Initiator master	Clk	Transfer response from slave interface "00" = No response "01" = Data valid "10" = Reserved "11" = Response error
SDataout[63:0]	Initiator master	Clk	Read data from slave. Contains the data from the memory-mapped registers.
SInterrupt	Interrupt controller	Clk	Hardware interrupt. Active high.

4.7 Selecting a signed multiplier component

All signed multiplications performed by HW use a signed multiplier component *hmulss.* Two versions of this file are available in *7280_encoder/ hardware/verilog/*:

- *hmulss.v,* multiplier using *signed* datatype
- *hmulss_unsigned.v, unsigned* multiplier and sign handling with extra logic

If the product will be synthesised using VHDL or Verilog2001 language, the signed datatype is supported, and *hmulss* file should be synthesised.

However, using the original Verilog HDL for synthesis requires more attention, as it does not support the *signed* datatype. There are two options for integrating the signed multiplier in Verilog HDL:

- Integrator may manually map a third-party component into *hmulss* file thus getting the optimal signed multiplier component
- Integrator may synthesise the included *hmulss_unsigned.v* file instead of *hmulss.v*, which will not cause extra work or testing, but will result in somewhat larger silicon area and worse timing closure

4.8 Integrating the embedded memory models

RSERAM (rseram.v / rseram.vhd) is the wrapper for the internal memories of the encoder. *RSERAM* uses the memory models as components. The standard interface of each memory model is described in Table 51. Other interface signals (e.g. output enable) possibly appearing in a model used by customer should be connected to a fixed value.

Note: The memory simulation models use active low enable signals, but the encoder core uses active high CEN and WEN signals which are thus inverted in RSERAM. If the integrated memory has active high CEN and WEN, the inverters can be omitted from RSERAM code.

TABLE 51. INTERNAL MEMORY INTERFACE SIGNALS

Port Name	Direction	Function
CLK	Input	Clock signal
CEN	Input	Chip enable, active low
WEN	Input	Write enable, active low
A[addrW-1:0]	Input	Address
D[dataW-1:0]	Input	Data in
Q[dataW-1:0]	Output	Data out

The memories and their purpose are presented in Table 52. Hantro has no recommendation concerning the memory type selection or which memories should be generated as register arrays. However, it should be noticed that the memory output data access time is not a critical issue, since the data outputs are registered in *RSE* before they are taken to other modules.

TABLE 52. EMBEDDED MEMORIES OF THE 7280 ENCODER (ALL ENCODING MODES AND VIDEO STABILIZATION CONFGURATION)

SRAM instance name	Connected to block	Type
u_ram240x64_nb	BUSIFE,IGE,MQPEE, TFE	Single-port
u_ram64x64_stbnb	STBE	Single-port
u_ram594x40_stb	STBE	Single-port
u_ram102x64_stbnfnb	STBE	Single-port
u_ram572x64_sa	OCPIFE,IGE, FIRME	Single-port
u_ram64x24_nbx	IGE,MOPEE	Single-port
u_ram192x48_sax0	IGE,MOPEE	Single-port
u_ram192x48_sax1	IGE,MOPEE	Single-port
u_ram192x48_sax2	IGE,MOPEE	Single-port
u_ram_dynamicx72_saq0h	FIRME,MQPEE	Single-port
u_ram_dynamicx72_saq0l	FIRME,MQPEE	Single-port
u_ram_dynamicx72_saq1h	FIRME,MQPEE	Single-port
u_ram_dynamicx72_saq1l	FIRME,MQPEE	Single-port
u_ram16x128_mqnb	MQPEE	Single-port
u_ram_dynamicx64_pr	MQPEE,TFE,ITEST	Single-port
u_ramdynamicx16_bsmv	BSE,MVE	Single-port
u_ram_dynamicx32_ne	IPE / DCPREDE	Single-port
u_ram_dynamicx32_dfw0	FILTERE, BUSIFE	Single-port
u_ram_dynamicx32_dfw1	FILTERE, BUSIFE	Single-port
u_ram96x32_fb0	TFE,FILTERE	Single-port
u_ram96x32_fb1	TFE,FILTERE	Single-port
u_ram_dynamicx88_tq	TQE,TFE	Single-port
u_ram_dynamocx18_rlc	SCE,VLCE	Single-port

4.9 Integrating the clock gating component

The 7280 encoder contains a dynamic clock gating module *clkctrle*. The clock gating module can be enabled or disabled by software by writing the control bit to one or zero in a memory-mapped register. Clock gating is enabled by default from power-on. See 4.12.4 for details.

When enabled, the clock gate module investigates is the encoding mode (H.264, MPEG-4, H.263 or JPEG) and/or video stabilization active and decides internally for which parts of the hardware the clock signal is gated. Clock gating also turns off the clock from the whole encoder except the bus interface, if the encoder is not enabled.

The list of gated clocks is presented in Table 53.

Note: The clock gating module uses the clkgate component for gating each clock. This component is a simulation model only containing one OR and one AND gate, and must be replaced with a technology specific clock gating component.

TABLE 53. LIST OF GATED CLOCKS

Clock name	Purpose	Connected to
HCLK	AHB input clock signal name	
Clk	OCP input clock signal name	
ACLK	AXI input clock signal name	
clk_enc_always_on	The always enabled clock to the slave interface wrapper block and the memory-mapped registers. Created by passing the input clock through an always-on clock gating component to balance the delta delay.	WSIFE, MFSME
clk_enc_intra	If video encoders and JPEG encoder are disabled this clock is disabled.	BSE, TFE, TQE, FILTERE, SCE, RCE
clk_enc_inter	If inter type video is not encoded this clock is disabled.	IGE, MOPEE, FIRME, MQPEE, TFECHBUF, TFCHFILT, BUSIFEAGRDSACH, MVE
clk_enc_enc	If video encoders, JPEG encoder and video stabilization are disabled this clock is disabled.	WMIFE, BUSIFE
clk_enc_stab	If video stabilization is disabled this clock is disabled.	STBE
clk_enc_h264_intra	If H264 is not encoded this clock is disabled.	IPE
clk_enc_jpeg	If JPEG image nor inter type video is enabled this clock is disabled	RSE

4.10 Synthesis considerations

This paragraph handles the synthesis related issues, such as clocks and resets.

4.10.1 Clocks and resets

7280 encoder is a single clock domain design. The only sequential devices used are D-type flip-flops triggered on the rising edge of *HCLK (*AHB clock name). There are no latches in the design.

Note: the APB slave interface clock PCLK may have a lower frequency than the master interface clock. In this case, PCLK must be divided from the master interface clock. Asynchronous slave and master clocks are not supported.

*HRESETn (*AHB reset name) signal is used for resetting all flip-flops asynchronously.

Software can also reset the hardware synchronically by writing encoder enable bit to zero. This enable bit is located in the memory-mapped registers and it can be used for terminating or restarting the encoding at any time.

Only the control flip-flops of the encoder are reset with the enable signal. This prevents power consumption peaks.

4.10.2 Synthesis-time automatic clock gating

When a synthesis tool detects a load enable for a register, it can gate the clock signal of that register. Enabling the synthesis-time clock gating is recommended as it results in reduced gate count and power consumption.

The 7280 encoder has been implemented taking synthesis-time automatic clock gating into notice by adding load enable signals for all flip-flops where appropriate.

4.10.3 Critical path

The encoder has no distinct critical paths.

4.10.4 Multi-cycle paths

The encoder has no multi-cycle paths.

4.10.5 False paths

The input port *buswidth* used for selecting AHB master data bus width should be connected to constant high or low in synthesis and its input delay constraint set to zero. Timing paths starting from this port can be considered as false paths in synthesis and timing analysis.

The paths starting from *testmode* input port are also false paths.

More false paths are generated from the software accessible registers, as their values are never modified while the encoder is running. These paths start from the output ports of the *busifeswr* module labeled as *swreg[n],* where *n* is a running number from 1 to 36.

4.10.6 Defensive timing strategy

All output signals to the bus are registered. Also most output signals of each block in the design are registered to make block input delays predictable. Data outputs from internal RAM's are registered to remove dependency on the memory technology selected by customer.

4.10.7 Glue logic

There is no glue logic on the top-level of the design.

4.10.8 Expected synthesis warnings

Synthesis tools will generate some warnings that are expected. These harmless warnings are related to unreachable logic and register elements in the design, and are caused for instance by the unused bits in some of the memory-mapped registers and the configurable SRAM's address busses.

Note: if any format or feature is removed from the product via configuration, more unreachable logic warnings can be expected.

4.11 Integration test structures

The following test functions can be used in SW / HW integration.

- Interrupt test.
- Register cache coherency test.
- Input / Output memory cache coherency test.

4.11.1 Interrupt test

When interrupt test bit is set in integration test register then interrupt status bit is activated in interrupt register (swreg1). The activation of interrupt status bit causes an interrupt if usage of the interrupt line is enabled. The interrupt line is activated as long as interrupt test bit is in high state.

4.11.2 Register test

When register cache coherency test is enabled HW reads bits 31:28 from integration test register, increases value by one and write result back to the same bits. This function is performed once during register cache coherency test is enabled.

4.11.3 Memory test

When memory cache coherency test bit is set in integration test register then HW reads data from external memory to 7280 encoder and writes data back to external memory.

- Data is read from the base address which is described in swreg5 and it is written to base address which is described in swreg6.
- Data amount is described in integration test register bits 20:3. 2^18 are maximum amount of 64 bit data words which can be used in the test.

- The maximum burst and byte endian settings are described in encoder configuration register (swreg2).

4.12 Memory-mapped registers

Hardware parts of the 7280 encoder are controlled with registers which software has access to. A list of the control registers is presented in Table 54 for helping the product integration work. The content of the integration related registers is shown in Table 59.

TABLE 54. MEMORY-MAPPED REGISTERS

Off set Address + base	Function	Width	Read/ Write
0	Encoder ID register (swreg0)	32	R
4	Encoder interrupt register (swreg1)	32	R/W
8	Encoder device configuration register (swreg2)	32	R/W
C	Encoder integration test register (swreg3)	32	R/W
10	Encoder interrupt interval register (swreg4)	32	R/W
14	Encoder base address register for output data (swreg5)	32	R/W
18	Encoder base address register for output control (swreg6)	32	R/W
1C	Encoder reference picture base address register for luminance (swreg7)	32	R/W
20	Encoder reference picture base address register for chrominance (swreg8)	32	R/W
24	Encoder new reference picture base address register for luminance (swreg9)	32	R/W
28	Encoder new reference picture base address register for chrominance (swreg10)	32	R/W
2C	Encoder input picture base address register for luminance (swreg11)	32	R/W
30	Encoder input picture base address register for Cb (swreg12)	32	R/W
34	Encoder input picture base address register for Cr (swreg13)	32	R/W
38	Encoder common control register 1 (swreg14)	32	R/W
3C	Encoder common control register 2 (swreg15)	32	R/W
40	Encoder H.264 control register 1 (swreg16)	32	R/W
44	Encoder H.264 control register 2 (swreg17)	32	R/W
48	Encoder H.264 control register 3 (swreg18)	32	R/W
4C	Encoder MPEG-4 / H.263 control register (swreg19)	32	R/W
50	Encoder JPEG control register (swreg20)	32	R/W
54	Encoder favor control register (swreg21)	32	R/W
58	Encoder stream data buffer control register 1 (swreg22)	32	R/W
5C	Encoder stream data buffer control register 2 (swreg23)	32	R/W
60	Encoder data counter / buffer limit register (swreg24)	32	R/W
64	Encoder rate control parameter register (swreg25)	32	R

68	Read only register	32	R
6C	Encoder quantization parameter control register (swreg27)	32	R/W
70	Encoder rate control target count register for checkpoints 1,2 (swreg28)	32	R/W
74	Encoder rate control target count register for checkpoints 3,4 (swreg29)	32	R/W
78	Encoder rate control target count register for checkpoints 5,6 (swreg30)	32	R/W
7C	Encoder rate control target count register for checkpoints 7,8 (swreg31)	32	R/W
80	Encoder rate control target count register for checkpoints 9,10 (swreg32)	32	R/W
84	Encoder rate control count error table 1 register (swreg33)	32	R/W
88	Encoder rate control count error table 2 register (swreg34)	32	R/W
8C	Encoder rate control count error table 3 register (swreg35)	32	R/W
90	Encoder rate control delta QP table register (swreg36)	32	R/W
94	Encoder RLC sum register (swreg37)	32	R
98	Encoder MB counter register (swreg38)	32	R
9C	Encoder next input picture base address register for luminance (swreg39)	32	R/W
A0	Stabilization register 1 (swreg40)	32	R/W
A4	Stabilization register 2 (swreg41)	32	R
A8	Top-left matrix value around full-pixel minimum (swreg42)	32	R
AC	Top-middle matrix value around full-pixel minimum (swreg43)	32	R
B0	Top-right matrix value around full-pixel minimum (swreg44)	32	R
B4	Middle-left matrix value around full-pixel minimum (swreg45)	32	R
B8	Middle matrix value around full-pixel minimum (swreg46)	32	R
BC	Middle-right matrix value around full-pixel minimum (swreg47)	32	R
C0	Bottom-left matrix value around full-pixel minimum (swreg48)	32	R
C4	Bottom-middle matrix value around full-pixel minimum (swreg49)	32	R
C8	Bottom-right matrix value around full-pixel minimum (swreg50)	32	R
FC	Encoder config register(swreg63)	32	R
100	Encoder JPEG Q table write register (swreg64)	32	W
104	Encoder JPEG Q table write register (swreg65)	32	W
108	Encoder JPEG Q table write register (swreg66)	32	W
10C	Encoder JPEG Q table write register (swreg67)	32	W
110	Encoder JPEG Q table write register (swreg68)	32	W
114	Encoder JPEG Q table write register (swreg69)	32	W
118	Encoder JPEG Q table write register (swreg70)	32	W
11C	Encoder JPEG Q table write register (swreg71)	32	W

120	Encoder JPEG Q table write register (swreg72)	32	W
124	Encoder JPEG Q table write register (swreg73)	32	W
128	Encoder JPEG Q table write register (swreg74)	32	W
12C	Encoder JPEG Q table write register (swreg75)	32	W
130	Encoder JPEG Q table write register (swreg76)	32	W
134	Encoder JPEG Q table write register (swreg77)	32	W
138	Encoder JPEG Q table write register (swreg78)	32	W
13C	Encoder JPEG Q table write register (swreg79)	32	W
140	Encoder JPEG Q table write register (swreg80)	32	W
144	Encoder JPEG Q table write register (swreg81)	32	W
148	Encoder JPEG Q table write register (swreg82)	32	W
14C	Encoder JPEG Q table write register (swreg83)	32	W
150	Encoder JPEG Q table write register (swreg84)	32	W
154	Encoder JPEG Q table write register (swreg85)	32	W
158	Encoder JPEG Q table write register (swreg86)	32	W
15C	Encoder JPEG Q table write register (swreg87)	32	W
160	Encoder JPEG Q table write register (swreg88)	32	W
164	Encoder JPEG Q table write register (swreg89)	32	W
168	Encoder JPEG Q table write register (swreg90)	32	W
16C	Encoder JPEG Q table write register (swreg91)	32	W
170	Encoder JPEG Q table write register (swreg92)	32	W
174	Encoder JPEG Q table write register (swreg93)	32	W
178	Encoder JPEG Q table write register (swreg94)	32	W
17C	Encoder JPEG Q table write register (swreg95)	32	W

4.12.1 Core identification

The core can be precisely identified by reading the ID register value. The value consists of the product identification number, major and minor version numbers and the build number. The API function `H264EncGetBuild` returns the value of this register. See one of the encoder API manuals for more information [11]-[13].

TABLE 55. ID REGISTER (0X0)

Bits	Function	Default value	Width	Read/ Write
31:16	Product ID	0x7280	16	R
15:12	Major version number	0x1	4	R
11:4	Minor version number	0x00	8	R
3:0	Build version number	0x0	4	R

4.12.2 Delivery identification

7280 encoder product release configuration can be identified by reading the encoder config register (swreg63). The value informs supported encoding formats, video stabilization support status, supported system bus protocol and maximum supported encoding resolution by macroblocks.

TABLE 56. ENCODER CONFIG REGISTER (0XFC)

Bits	Function	Read/	Valid

		Write	state
31:28	Not used	R/W	0
27	Encoding format support, H.264 '0' = not supported '1' = supported	R	0,1
26	Encoding format support, MPEG-4/H.263 '0' = not supported '1' = supported	R	0,1
25	Encoding format support, JPEG '0' = not supported '1' = supported	R	0,1
24	Stabilization support '0' = not supported '1' = supported	R	0,1
23:20	Connected to standard bus: 0 = error 1 = AHB 2 = OCP 3 = AXI 4 = PCI	R	0…4
19:16	Synthesis language 0 = error 1 = vhdl 2 = verilog	R	0..2
15:12	Buswidth 0 = error 1 = 32 bit bus 2 = 64 bit bus 3 = 128 bit bus	R	0…4
11	Not used	R	0
10:0	Maximum video encoding resolution in macroblocks	R	0… 80

4.12.3 Encoder interrupt register

The encoder interrupt register contains critical information for integration such as the encoder enable bit and the interrupt flags. See the SW Integration Guide for information about selecting the interrupt method.

TABLE 57. ENCODER STATUS REGISTER (0X4)

Bits	Function	Read/ Write	Valid state
31:9	Not used	R	0
8	Interrupt status bit for interrupt interval reached. Active high. When this bit is set then encoder is processed the desired amount of MBs.	R/W	0,1
7	Interrupt status bit for memory cache coherency test. Active high. When this bit is set, hw has copied data from input buffer to output buffer.	R/W	0,1
6	Interrupt status bit for interrupt test. Active high. When interrupt test bit (bit0) in integration test register (swreg3) is set, status of this is activated.	R/W	0,1

5	Interrupt status bit for stream data buffer full. Active high. When this bit is high encoding is waiting until this bit is lowered.	R/W	0,1
4	Interrupt status bit for software reset. Active high. If software resets the hardware while encoding then this bit is written high.	R/W	0,1
3	Interrupt status bit for bus error. Active high.	R/W	0,1
2	Interrupt status bit for encoder frame ready. Active high. When this bit is high encoder has coded a picture.	R/W	0,1
1	Interrupt disable. Active high. When high, there are no interrupts (HINTenc) concerning encoder from HW. Polling must be used to see the interrupt.	R/W	0,1
0	Interrupt HINTenc. SW will reset this after interrupt is handled. HINTenc is not used if IRQ disable is high (swreg1 bit 1).	R/W	0,1

4.12.4 Device configuration register for system bus

In the device configuration register for example the SDRAM accessing related items of the hardware can be adjusted. Adjusting these values will affect the performance of the hardware.

In AHB environment maximum burst length can be set to 4, 8 or 16 or left undefined. In OCP environment burst length can anything less than 32. In AXI environment burst length can anything less than 16. Typically longer bursts give better performance as they reduce the amount of non-sequential transfer types, but they might increase the latency other bus masters are experiencing.

See the SW Integration Guide for information about changing the system configuration.

TABLE 58. ENCODER DEVICE CONFICURATION REGISTER (0X8)

Bits	Function	Read/ Write	Valid state
31:24	AXI write ID value. Used in all AXI write transfers.	R/W	0 …255
23:16	AXI read ID value. Used in all AXI read transfers.	R/W	0 …255
15:14	Not used	R/W	0
13:8	Maximum burst length for encoder bus transactions. Valid values for AHB are 4, 8 and 16. Other values will result in INCR type (undefined length) transfers. Valid values for OCP are 1-31, and 1-16 for AXI.	R/W	0 …63
7:6	Not used	R/W	0
5	AHB precise burst and data discard enable. When high, only the precise AHB bursts (SINGLE, INCR4, INCR8 and INCR16) are used in SDRAM read accesses. Extra data is discarded internally. When low, INCR bursts of undefined length 2 or 3 may be issued when necessary.		
4	Clock gating enable.	R/W	0,1

	0 = Clock gating is disabled. 1 = Clock gating is enabled.		
3	Swap of 32-bit words in 64-bit environment. For encoder output data. 0 = Swap disabled 1 = Swap enabled	R/W	0,1
2	Swap of 32-bit words in 64-bit environment. For encoder input picture. 0 = Swap disabled 1 = Swap enabled	R/W	0,1
1	Endian mode for encoder output data. 0 = Big endian 1 = Little endian	R/W	0,1
0	Endian mode for encoder input picture. 0 = Big endian 1 = Little endian	R/W	0,1

4.12.5 Integration test register for encoder

Integration test register controls integration test structures. With integration test structures, 7280 encoder master and slave interface can be tested without 7280 control software.

TABLE 59. ENCODER INTEGRATION TEST REGISTER (0XC)

Bits	Function	Read/ Write	Valid state
Bits	Function	Read/ Write	Valid state
31:28	Counter bits	R/W	0..15
27:21	Not used.	R/W	0
20:3	Data length for memory cache coherency test. Amount of 64 bit words.	R/W	0 … 262143
2	Input/Output memory cache coherency test enable. When this bit is set to one HW copies data from the location pointed by the base address of swreg5 to the location pointed by the base address of swreg6. Data length is specified in bits 20:3.	R/W	0,1
1	Register cache coherency test enable. When this bit is set to 1, HW reads bits 31:28 from register, increases value by one and writes result back to the same bits. Only once per enable HW self-reset this enable.	R/W	0,1
0	Interrupt test register, when this bit is set to 1, HW gives an interrupt.	R/W	0,1

4.13 Hardware verification

This paragraph deals with the hardware verification issues. RTL of the product has been extensively tested by both workstation simulations and on an ARM Versatile FPGA board, where both internally generated and conformance test streams, plus interoperability streams have been run.

4.13.1 RTL testing on customer site

If customer wants to run RTL simulation or perform sanity checking after hardware integration, Hantro can provide a VHDL or Verilog test bench and a set of test cases. See the 7280 Hardware Simulation Guide for test environment details and running the tests [15].

Instead of the releasing the internal test bench and test data for it, Hantro may also record test vectors to be used in another test environment. The vectors are recorded during RTL simulations and they describe the content of external memory before and after encoding, and the operations encoder software performs on the hardware memory-mapped registers. Typical format for recorded test data is the denali format as described in Table 60 and.Table 61.

TABLE 60. OPERATIONS ON THE MEMORY-MAPPED REGISTERS EXAMPLE IN DENALI FORMAT

```
####################################################
## SWREGISTER ACCESSES, Picture 0
####################################################
R 00000000/72800570
W 00000004/00000000
W 00000008/00001020
W 0000000c/00000000
W 00000010/00000000
W 00000014/00562200
W 00000018/00546000
W 0000001c/00384000
W 00000020/004b0000
W 00000024/001c2000
.
.
.
W 00000094/00000000
W 00000098/00000000
##
## Enable encoder
W 00000038/0058248f
##
## Poll encoder interrupt
P 00000004/00000005
##
## Interrupt noticed - check results
##
```

TABLE 61. EXTERNAL MEMORY CONTENT EXAMPLE IN DENALI FORMAT

```
#
# Encoder output stream data, Picture 0
#
00562200/800000000125b800
00562208/04fffffcf1400fbe
00562210/f778e31ccea81f5a
00562218/ab5b59d756b6b3ab
00562220/bdac7aaffe467fff
```

```
00562228/fc6d456efbf819df
00562230/241d27adaa1bc7a9
00562238/69ff83b683b6e38a
00562240/89c07a908fff1258
00562248/e7b1ea3a5a7ff6f0
00562250/41b6eac7a808f21f
.
.
.
00562920/db46197c4fef352a
00562928/0762f1acf0bd54f9
00562930/056797984a000000
#
# Encoder output stream data, Picture 1
#
00562200/0000000121e00023
00562208/d67ffffebfe2ae6f
00562210/37cc6efe63776b31
00562218/8ddd7108bef88370
00562220/44cdd9f11e23d7af
.
.
.
```

4.13.2 BIST and scan test

Hantro has no recommendation concerning BIST or scan testing. Customer may select the most convenient method.

References

[1] ISO/IEC 14496-10 / ITU-T Recommendation H.264 (2003). Advanced video coding for generic audiovisual services.

[2] ITU-T Recommendation H.263 (1998). Video Coding for Low Bit Rate Communication.

[3] ISO/IEC 14496-2 (2001). Information technology – Generic Coding for Low Bit Rate Communication.

[4] ITU-T Recommendation T.81 (1991). Information technology – Digital Compression and Coding of Continuous-tone Still Images – Requirements and Guidelines.

[5] Hantro Products (2008). 7280 Software Integration Guide.

[6] OCP International Partnership (2003). Open Core Protocol Specification Release 2.0.

[7] ARM (1999) AMBA 2 Specification Revision 2.0. URL: http://www.arm.com/products/solutions/AMBA_Spec.html

[8] ARM (2004) AMBA 3 AXI Protocol v1.0 Specification .URL: http://www.arm.com/products/solutions/AMBA3AXI.html

[9] ARM (2004) AMBA 3 APB Protocol v1.0 Specification. URL: http://www.arm.com/products/solutions/AMBA3AXI.html

[10] Taiwan Semiconductor Manufacturing Company (2005). TCBN65LP, TSMC 65nm Core library Databook, version 1.2, 2006

[11] Hantro Products (2008). 7280 H.264 API User Manual.

[12] Hantro Products (2008). 7280 MPEG-4 API User Manual.

[13] Hantro Products (2008). 7280 JPEG API User Manual.

[14] Hantro Products (2008). 7280 Video Stab API User Manual.

[15] Hantro Products (2008). 7280 HW Simulation Guide.

VISIBLY BETTER

Headquarters:
Hantro Products Oy
Kiviharjunlenkki 1, FI-90220 Oulu, Finland
Tel: +358 207 425100, Fax: +358 207 425299

Hantro Finland
Lars Sonckin kaari 14
FI-02600
Espoo
Finland
Tel: +358 207 425100
Fax: +358 207 425298

Hantro Germany
Ismaninger Str. 17-19
81675
Munich
Germany
Tel: +49 89 4130 0645
Fax: +49 89 4130 0663

Hantro USA
1762 Technology Drive
Suite 202. San Jose
CA 95110,
USA
Tel: +1 408 451 9170
Fax: +1 408 451 9672

Hantro Japan
Yurakucho Building 11F
1-10-1, Yurakucho
Chiyoda-ku, Tokyo
100-0006, Japan
Tel: +81 3 5219 3638
Fax: +81 3 5219 3639

Hantro Taiwan
6F, No. 331
Fu-Hsing N. Rd.
Taipei
Taiwan
Tel: +886 2 2717 2092
Fax: +886 2 2717 2097

Hantro Korea
#518 ho, Dongburoot, 16-2
Sunaedong, Bundanggu,
Seongnamsi, Kyunggido,
463-825 Korea
Tel: +82 31 718 2506
Fax: +82 31 718 2505

Email: sales@hantro.com
WWW.HANTRO.COM