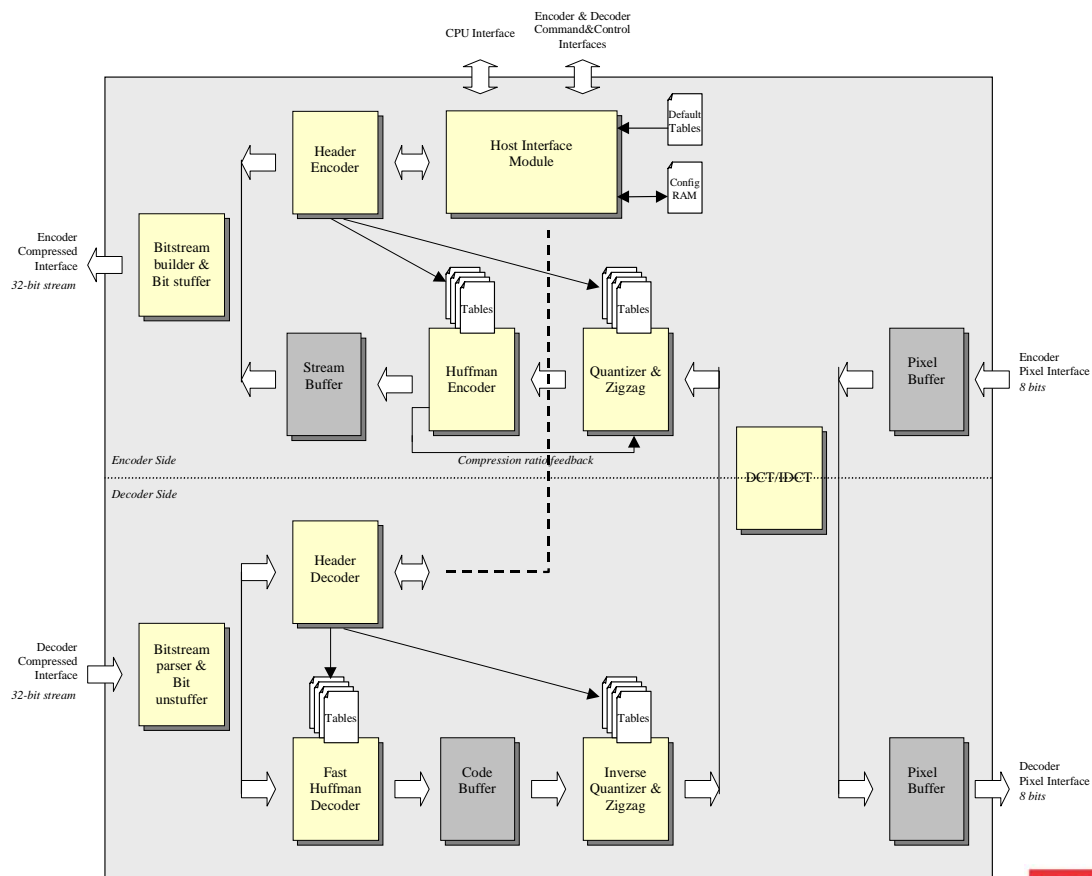


JPEG codec

BA119JPEGCOD Factsheet

Features

- Half duplex JPEG codec
- High-speed sustained single clock cycle per pixel component encoding and decoding
- Single clock cycle Huffman decoding and encoding
- 100% baseline ISO/IEC 10918-1 JPEG compliance for color images (single and multi-scan formats) extending to effective 255-scan support.
- Full header generation capability with default pre-programmed Huffman and quantization tables or user-definable custom tables in encoding mode.
- Full header parsing capability and automatic on-the-fly Huffman and quantization tables reprogramming from header data in decoding mode,
- One-pass encoding scheme with compression ratio regulation.
- Header error catching features
- Full baseline JPEG format and abbreviated format support, including restart markers and DNL.
- Simple FIFO interfaces for compressed data (32 bit) and pixel interface (8 bit)
- Simple CPU interface for codec control and programming.
- Easy-to-use status and control interface through 21 internal registers.
- Programmable external interrupts for event follow-up.
- For entropy tables (two DC, two AC), four quantization tables.
- Burst image-sequence encoding support for images with identical tables and color format.
- Burst image-sequence decoding support for images with identical tables.
- 8-bit/pixel components.
- 8x8 block format pixel interface with classical scan order (row by row from left to right).
- Fully synchronous hardware design.
- Fully stallable compressed data-interface, stallable pixel interface on a block-by-block basis.
- Throughputs ranging from sub CIF 25Hz to SDTV to HDTV1280x720 50Hz



Rue du Bosquet 7
B-1348 Louvain-La-Neuve

General description

The JPEG codec core is intended for high-speed encoding and decoding of gray-scale, color or multi-scan images according to the ISO/IEC 10918-1 baseline-coding standard. The codec supports all features of the baseline standard, including restart markers, DNL, user-definable comments and application markers. In decoding mode it performs stand-alone full header parsing. It is able to encode and decode abbreviated-format or full-format images, automatically extracting the quantization and entropy tables in decoding mode and automatically generating the fixed-length headers from its internal user-programmable header memory in encoding mode.

Its autonomous behaviour, its simple FIFO-like interfaces and its 100% synchronous structure allow to integrate it very easily in a complex system with few effort. This is reinforced by its stand-alone ability in decoding mode that allows to instantiate it in systems without CPU intervention.

Applications

The codec is suitable for applications involving baseline-DCT JPEG compression/decompression in a gray-level, colour or multi-scan environment. Applications requiring high pixel through-put can be addressed by the core. These include printers, scanners, digital cameras, medical imaging, archiving...

Technical description

The codec features a modular design and a highly optimized block architecture offering high performance encoding and decoding, and yielding sustained rate of one pixel component per clock cycle. Successive image decoding is also supported with the same global performance when processing image sequences with the same tables. Successive image encoding is also supported with the same global performance when processing image sequences with the same tables and color format.

The codec features an input pin allowing to configure it either in encoding either in decoding modes. Dynamic switching between both modes is allowed with the restriction that the codec has reached idle state before doing this.

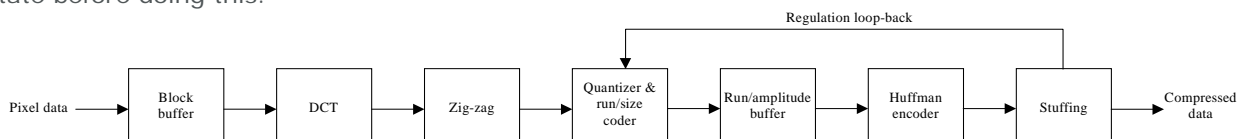


Figure 2: High-speed encoding modules

Image compression is achieved as represented in Figure 2 as soon as it is enabled by the internal sequencer once header building is over.

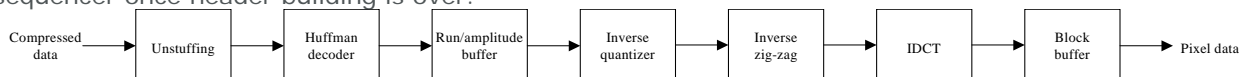


Figure 3: High-speed decoding modules

Image decompression is achieved as represented in Figure 3 as soon as it is enabled by the internal sequencer once header parsing is over.

For implementation effectiveness each module in Figures 2 and 3 is implemented with synchronization flip-flops at its inputs and outputs and communication through an effective Strobe/Acknowledge protocol. This allows for highly efficient automatic place and route flow on FPGA devices.

Encoding Mode

In encoding mode the core is fed with 8-bit pixel components under 8x8 block format with raster scan order and outputs 32-bit compressed data. The internal sequencer encodes the header data, programs the quantization and entropy tables and reports processing stage through the host-interface registers. Once the header is fully encoded the sequencer enables the high-speed encoding blocks, which execute the JPEG compression algorithm. Pixel components and Huffman-ready run-amplitude codes are processed at a rate of one per clock cycle. One pixel component is processed per clock cycle and this rate may be kept with minimal latency for the next image if it uses the same tables and color format as the first one.

Table programming is performed through the host interface (default pre-programmed tables are also available). The encoding is started either by activating the Go_enc pin either by writing 1 in the GO internal register. The encoder automatically stops encoding at the end of the current image when the LastPict_enc pin is activated or 1 is written in the LASTPICT internal register.

A single internal register allows the user to get status information about the encoding status (idle state, end of scan/image reached, end of header reached). Each status bit may be allowed to generate a maskable external interrupt. Moreover end-of-image and end-of-scan status bits are directly accessible as output pins (nEndI_enc, nEndS_enc).

Complementary internal registers accessible through the host interface allow to configure the encoder by defining the image size, the markers to be included into the header, the target compression ratio, the default tables to be used and the global weighting to be applied to the quantization coefficients. An internal address-mapped memory allows the CPU to prepare the header information, including unlimited-length comments by writing through the host interface.

The input interface accepts pixels in 8x8 block format. It features a block buffer that allows it to complete the acquisition of a current block of data even if the further pipeline is stalled. This allows to easily interconnect the JPEG encoder with block-burst devices that may not be stalled during the transmission of the current block. The input interface features pixel-level nWait_enc and block-level nStop_enc ready flags to stall the pixel source. nWait_enc and nStop_enc instruct the pixel source that the encoder is able to finish accepting the current block but will be full at this stage. nStop_enc can be used by pixel sources that cannot be interrupted during a pixel block transaction; nWait_enc can be used by pixel sources that are able to instantly stop sending new pixels.

The Discrete Cosine Transform (DCT) applies the first transformation on the input pixel data in order to output frequency coefficients. The pixel components are input in raster scan order into the DCT (line by line, from left to right).

The transformed coefficients are passed through the zigzag encoder which changes the coefficient order from raster scan order to zigzag order.

They are then transferred to the quantization block, which applies the quantization division according to the quantization tables and performs the run/amplitude encoding of the remaining non-zero coefficients, simplifying zero chains. The quantization block features a bit rate regulation mechanism allowing feedback from the accumulated amount of bits at the output of the encoder for the current image in order to be as close as possible to the targeted compression ratio.

The resultant run/amplitude pairs are stored in a buffer FIFO allowing to decouple with the variable-length processing stages.

The Huffman entropy compressor reads the FIFO and processes one run-size/amplitude code per system clock cycle. It is able to use the appropriate AC or DC table according to its own block-level synchronization without any time gap during switching.

Finally the entropy-encoded data are processed by a post-filter (stuffing) adding stuffing bytes and restart markers to the JPEG stream.

The compressed data interface features an easy FIFO-like protocol that can be stalled at any time by de-asserting the nJpegEn_enc input pin.

Decoding Mode

In decoding mode the core is fed with 32-bit compressed data and outputs 8-bit pixel components under 8x8 block format with raster scan order. The internal sequencer decodes the header data, updates the quantization and entropy tables and reports processing stage and header data through the host-interface registers. Once the header is fully decoded the sequencer enables the high-speed decoding blocks, which execute the JPEG decompression algorithm. Huffman-decoded run-amplitude codes and pixel components are processed at a rate of one per clock cycle. One pixel component is output per clock cycle and this rate may be kept with minimal latency for the next image if it uses the same tables as the first one.

The internal sequencer allows the JPEG core to decode images in stand-alone meaning that minimum user intervention is needed. No table programming is required through the host interface (the core extracts them from the header data). The decoding is started either by activating the nGo_dec pin either by writing 1 in the GO internal register. The decoder automatically stops decoding at the end of the current image when the nLastPict_dec pin is activated or 1 is written in the LASTPICT internal register.

Two internal registers allow the user to get status information about the decoding stage, either about the encountered markers in the header data (DQT, DHT, SOF, SOS) either about decoding status (idle state, end of image reached, header error detected, end of header reached). Each status bit may be allowed to generate a maskable external interrupt. Moreover end-of-image status bit is directly accessible as output pin (nJetEnd_dec).

Nine complementary internal registers accessible through the host interface report the crucial header information about image size, number of components, number of scans, sample precision and the sampling factors for the successive components encountered.

The Huffman decoder gets its data from an internal pre-filter (unstuffing) erasing stuffing bytes and restart markers. Restart markers are signaled to the Huffman decoder in order to reinitialize the predictive values for DC component decoding.

The Huffman entropy decompressor processes one run-size/amplitude code per system clock cycle. It is able to use the appropriate AC or DC table according to its own block-level synchronization without any time gap during switching. The decoder outputs one pair of run/amplitude code per cycle. These data are stored in a small internal FIFO in order to minimize the decoder stalling probability with images containing large amount of restart markers and stuffed bytes.

The coefficients are fetched from the FIFO by the inverse quantization block. This block expands the run/amplitude fields in order to generate zero chains. These data are dequantized by using the corresponding coefficients in the quantization table.

The dequantized coefficients are passed to the zigzag decoder which inverses the zigzag order and generates the coefficients in inverse raster scan order.

The Inverse Discrete Cosine Transform (IDCT) applies the last transformation on the block coefficients in order to output decompressed pixel data. This transform module is designed to comply to IEEE 1180-1990 standard that constrains the precision of this mathematical transform. The pixel components are output in raster scan order by the IDCT (line by line, from left to right) and they are stored in a 2x64-pixel output buffer that guarantees block burst capability. This allows the decoder to output a full 8x8-component block before being stopped either by an input FIFO underflow either by an external stall command at the pixel interface.

The output interface may be stopped when activating nStop_dec input pin. This instructs the decoder to finish to output the current block and then stop and wait for a nStop_dec deactivation. On the contrary the compressed data interface may be stalled at any time by de-asserting the nJpgEn_dec input pin.

Core Modifications

Core modifications can be performed on the CPU interface in order to adapt it to a particular bus protocol. However in most cases the simple synchronous behaviour of the interface will enable the customer to externally adapt it with few glue logic to his particular bus environment. If necessary core modifications can be applied by the IP provider in order to suit it to the customer bus protocol.

Implementation data

Device	Logic	# of Clk	Performance (MHz)	Needed Resource
Xilinx XC2V1500-6	7497 slices	1	82	36 RAMB16, 4 MULT
Xilinx XC2V1500-4	7497 slices	1	64	36 RAMB16, 4 MULT
Altera EP1S25F1020C7	12721 LE	1	57	49 M4K, 46 M512, 2 DSP MULT
Altera EP1S30F1020C5	12724 LE	1	83	45 M4K, 60 M512, 2 DSP MULT
Altera EP2A25F672C7	14239 LE	1	58	61 ESB
UMC 0.18 μ m	107 kgates	1	142	175 kbits

Pinout description

Name	I/O	Comments
Global		
nReset	I	Asynchronous reset
Clk	I	Clock
Mode	I	Encoding/decoding select
CPU Interface		
XDIn [7:0]	I	CPU data bus in input direction
XDOut [7:0]	O	CPU data bus in output direction
XDnOE	O	Data bus output enable
XA [10:0]	I	CPU address bus
WrRd	I	CPU transfer direction
nCs_enc	I	CPU chip select for encoder registers
nCs_dec	I	CPU chip select for decoder registers
nInt_enc	O	Encoder interrupt
nInt_dec	O	Decoder interrupt
ENCODING-MODE INTERFACES		

Pixel interface		
Pixel_enc[7:0]	I	Block-scan ordered pixels
nEOIout_enc	O	Last pixel of image flag
nEOSout_enc	O	Last pixel of scan flag
nEOBout_enc	O	Last pixel of block flag
nWait_enc	O	Pixel-level ready flag
nStop_enc	O	Block-level ready flag
nPixWen_enc	I	Pixel strobe (active low)
nLastPixOI_enc	I	Last pixel of image (for DNL support with single scan)
nLastPixOS_enc	I	Last pixel of image (for DNL support with multi-scan)
Command interface		
Go_enc	I	"Start encoding" command
LastPict_enc	I	Burst image mode enable/disable
LastScan_enc	I	Burst scan mode enable/disable
nEndI_enc	O	End of image encoding process
nEndS_enc	O	End of scan encoding process
Compressed data interface		
Code_enc [31:0]	O	Compressed data
nWrEn_enc	O	Synchronous FIFO write command (active low)
WrEn_enc	O	Synchronous FIFO write command (active high)
nFF_IR_enc	I	Synchronous FIFO full flag for compressed data
nWrEn_nOE_enc	O	nWrEn/WrEn output enable
nJpegEn_enc	I	Compressed data interface stalling command
DECODING-MODE INTERFACES		
Decoded pixel interface		
Comp_dec [7:0]	O	Component label
Pixel_dec[7:0]	O	Block-scan ordered decoded pixels
nDsync_dec	O	Start-of-block sync signal
nJetEoi_dec	O	End of image
nJetEos_dec	O	End of current scan
nStop_dec	I	Pixel stalling request
nPixWen_dec	O	Pixel strobe (active low)
PixWen_dec	O	Pixel strobe (active high)
Command interface		
nJetEnd_dec	O	End of decoding process
nGo_dec	I	"Start decoding" command
nLastPict_dec	I	End of image burst
Compressed data interface		
Code_dec [31:0]	I	Compressed data
nFFCREN_dec	O	Synchronous FIFO read command (active low) for compressed data
FFCREN_dec	O	Synchronous FIFO read command (active high) for compressed data
FFnCEF_dec	I	Synchronous FIFO empty flag for compressed data
FTMFIFO_dec	I	FIFO type: classical or fall-through-mode
nJpgEn_dec	I	Compressed data interface stalling command

Barco Silex overview

Barco Silex is a micro-electronic design house located in Belgium and France belonging to the Belgian Barco group.

Barco Silex offers a complete portfolio of high-end design services, from ASIC/FPGA design to advanced SoC/SoPC based system development, IP-core design and board design in the fields of:

- image processing
- communications
- consumer electronics
- industrial electronics.

Barco Silex IP products

Barco Silex design expertise is also made available through a wide portfolio of IP products, with a strong focus on high performance, standardized image processing and encryption functions.

All these IP cores have been designed and fully validated by Barco Silex and are hardware proven, which guarantees high IP quality as well as best support during your integration phase.

Deliverables include:

- RTL Code or netlist (depending on license type)
- Functional simulation testbench
- Synthesis script
- Full documentation

For some of them, we can also provide you with simulation models and a design kit.

These "off the shelf", high quality IP cores provide you with the fastest and most efficient way of integrating complex functionalities on FPGAs or ASICs, while meeting short time to market constraints.

More information

Order-reference: **BA119JPEGCOD**

For additional information and other IP products contact:

Barco – Silex

e-mail: barco-silex@barco.com

<http://www.barcodesignservices.com>

or the local Barco Silex design centers:

Belgium

Scientific Park
Rue du Bosquet 7
1348 Louvain-la Neuve
+32(0)10/45.49.04

France

ZI Peynier- Rousset
Route de Trets Imm CCE
13790 Peynier
+33(0)44/216.41.06