
JE150 and JE160 JPEG Encoder

9. February 2004

Product Specification 3.0



Penz VHDL
Frankenstraße 16
D-55299 Nackenheim
Germany

Phone: +49 (0)6135-950328
Fax: +49 (0)6135-950329
E-Mail: mail@penz-vhdl.de
URL: www.penz-vhdl.de

Core Facts	
Provided with core	
Documentation	Programmers Manual
Design File Formats	NGC netlist
Verification	Testbench, Testvectors
Constraints File	Not needed
Instantiation Template	VHDL
Reference design & application notes	Demo Application
Additional Items	
Simulator Tool Used	
ModelTech ModelSim 5.7c	
Support	
Penz VHDL	

Features

- Optimized for Xilinx FPGA
- Baseline Encoder
- Compliant with ISO/IEC 10918-1
- Motion-JPEG capability
- 12-bit/pixel input
- 8x8 block format pixel input
- Up to 16 components
- Up to 4 Quantization tables
- Up to 8 Huffman tables (four DC and four AC)
- Predefined luminance and chrominance tables
- Fully synchronous design
- Fully stall able design
- Simple CPU interface for table reprogramming
- Different clocks for encoder and CPU interface
- Single clock cycle per pixel encoding
- No pause cycles between blocks

Applications

The JE150 and JE160 are designed as coprocessor for a CPU to speed up the baseline encoding of a pixel stream to a JPEG encoded data stream. It is suitable where an image must be compressed in real-time, like in Frame grabbers, intelligent cameras, surveillance systems and scanners.

The JE150 is optimized for Virtex, Virtex-E, Spartan-II and Spartan-IIE devices.

The JE160 is optimized for Virtex-II, Virtex-IIP and Spartan-III devices.

Table 1: Core Implementation

Supported Family	Device Tested	Flip Flops	4 Input Luts	Block RAM	TBufs	Clock IOBs	IOBs	Performance (MHz)	Xilinx Tools
Spartan-IIE	XC2S600E-6	2789	5998	6	208	2	69	55	ISE 6.1
Virtex	XCV600-4	2789	5998	6	208	2	69	45	ISE 6.1
Virtex-II	XC2V250-4	2056	2634	3	-	2	69	100	ISE 6.1
Spartan-III	XC3S1000-4	2056	2634	3	-	2	69	90	ISE 6.1

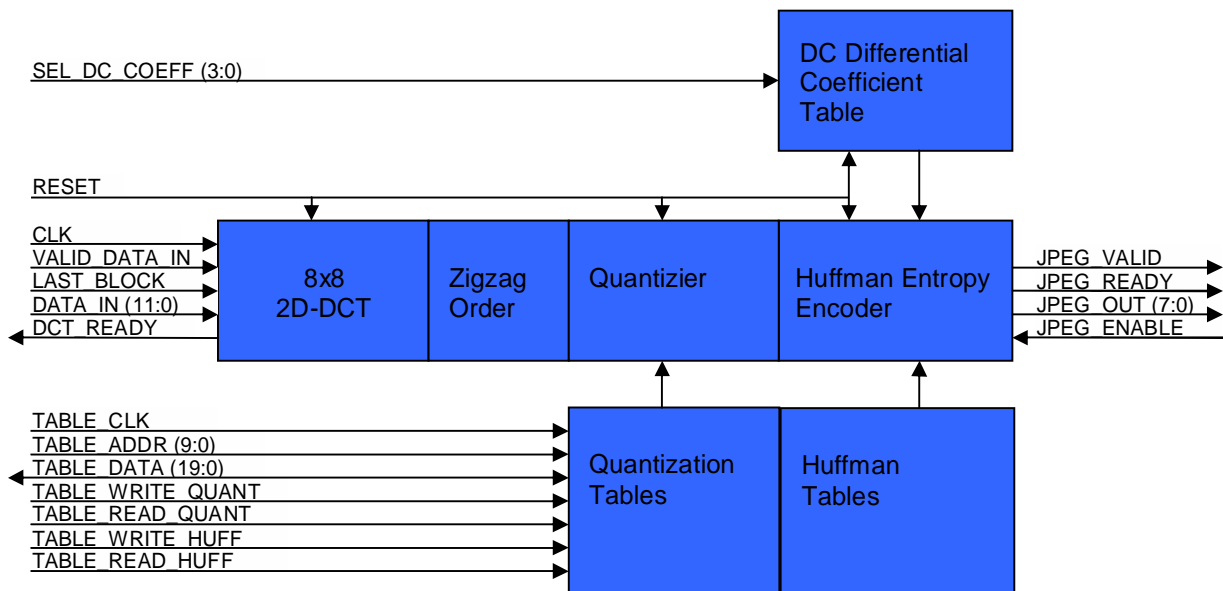
General Description

The JE150 and JE160 encode subsequent 8x8 pixel blocks into a JPEG standard conform bit stream. Baseline encoding with a pixel size of 8 bit is used. The bit stream appears on the 8 bit output register. The core is stall able by the handshake signals on the pixel input and the bit stream output interface. No Marker is generated, so the designer has the choice to generate the header with a CPU or use a fixed table. Sample header and a C++ class to generate the header for the most popular formats (monochrome, 4:2:2 color) are delivered with the core. Standard DC- and AC-tables for quantization and Huffman encoding are predefined, but can be changed through the programming interface, even if the encoding process is active.

Functional Description

Each incoming 8x8 pixel block goes through the two-dimensional discrete cosine transformation (2D-DCT) and will be transformed into 64 coefficients into the frequency domain. Now, the coefficients will be read out in a Zigzag order and quantized by dividing through the selected quantization table. The first (DC) quantized coefficient is differentially coded, using the most recently DC coefficient from the same component. Now all coefficients are run length coded to Run-Size symbols. Finally the symbols are Huffman coded, using the code from the selected Huffman table. The bit-stream of Huffman codes is divided into parts with 8 bit and stored in the output register.

Figure 1: Encoder Block Diagram



Core Modifications

Usually no core modifications are necessary, but when a special interface or more tables are needed, modifications can be applied. Please contact Penz-VHDL for more information.

Verification Methods

The encoder has been validated as functional and timing simulation with several test patterns under ModelSim XE 5.7c. A final verification has been done in a demo-application with some color and monochrome images.

Pinouts

Table 2: Core Signal Pinout

Signal Name	Direct.	Description
Pixel Input Interface		
CLK	In	Encoder clock
RESET	In	Reset encoder state machines
VALID_DATA_IN	In	Pixel is valid
LAST_BLOCK	In	Last block of field
DATA_IN (11 : 0)	In	Pixel input
DCT_READY	Out	Encoder is ready for new pixel
SEL_DC_COEFF (3:0)	In	Select a component
SEL_TABLE_QUANT (1:0)	In	Select a Quantization table for this component
SEL_TABLE_HUFF (1:0)	In	Select a Huffman table for this component
Compressed Data Output Interface		
JPEG_ENABLE	In	Data handshake
JPEG_VALID	Out	Output data is valid
JPEG_READY	Out	Last data for this field
JPEG_OUT (7:0)	Out	Output data
Tables Programming Interface		
TABLE_CLK	In	Clock for table reprogramming
TABLE_ADDR (9:0)	In	Table select and addressing
TABLE_DATA (19:0)	I/O	Table read- or write-data
TABLE_WRITE_QUANT	In	Write data into Quantization table
TABLE_READ_QUANT	In	Read data from Quantization table
TABLE_WRITE_HUFF	In	Write data into Huffman table
TABLE_READ_HUFF	In	Read data from Huffman table

Recommended Design Experience

Only basic VHDL knowledge is necessary, no manual placement is required. Advanced designers can reach a higher performance.

Ordering Information

The JPEG Encoder core is provided under license by Penz-VHDL. The VHDL source code is also available. Please contact Penz-VHDL for information about pricing and conditions of sale.