FIDCT Forward/Inverse Discrete Cosine Transform

CSELT S.p.A

Via G. Reiss Romoli, 274
I-10148 Torino, Italy
Phone: +39 011 228 5259
Fax: +39 011 228 5695
E-mail: viplibrary@cselt.it
URL: www.cselt.it

Features

- Forward, inverse, or multiplexed forward/inverse function
- Parametric Discrete Cosine Transform (DCT) for 8x8, 16x16 data
- Inverse DCT compliant with IEEE 1180-1990 standard
- Customizable RTL source code available
- Customized testbench for pre- and post-synthesis verification supplied with the module
- C environment supplied with the module to verify the adherence of the IDCT function to the IEEE 1180-1990 standard
- Core customization:
  - Parameterizable input/output word width (9-15) bits) and internal data path width (12-30 bits)
  - Parameterizable transpose memory word size (12-24 bits) and cosine coefficients width (10- 26 bits)
  - Parameterizable number of pipeline stages in multipliers
  - Variable I/O scanning format: column-by-column, row-by row, and zig-zag (MPEG styles)

Applications

Video coding systems based on the baseline JPEG, MPEG, and H.26x standards family.

AllianceCORE™ Facts

Core Specifics[1]	
Supported Family	Virtex
Device Tested	V200-6
CLBs[2]	1802
Clock IOBs	1
IOBs[3]	33
Performance (MHz)	78
Xilinx Tools	M3.1i
Special Features	1 BlockRAM
Provided with Core	
Document	User Manual
Design File Formats	EDIF netlist, XNF netlist,
Constraints Files	.ncf
Verification	VHDL testbench
Instantiation Templates	VHDL, Verilog
Reference Designs & Application Notes	None
Additional Items	C environment for IEEE 1180-1990 compliance verification
Simulation Tool Used	
Synopsys VSS	
Support	
Design and customization support provided by CSELT	

Notes:
1. Data refer to the following customization:
   - Inverse DCT functionality
   - Pixel Data Size =9, Coefficient Data Size =12
   - Cosine Coefficient Bit Width =15
   - Transpose Memory Word Size =14
   - Internal Datapath Size =20
   - One Pipeline Multiplier Stage
   - No MPEG2 Mismatch Control
   - No Input Data Re-Order
   - No Output Data Re-Order
   - Two Clock Cycle Memory Latency
2. Utilization numbers for Virtex are in CLB slices
3. Assuming all core I/Os are routed off-chip

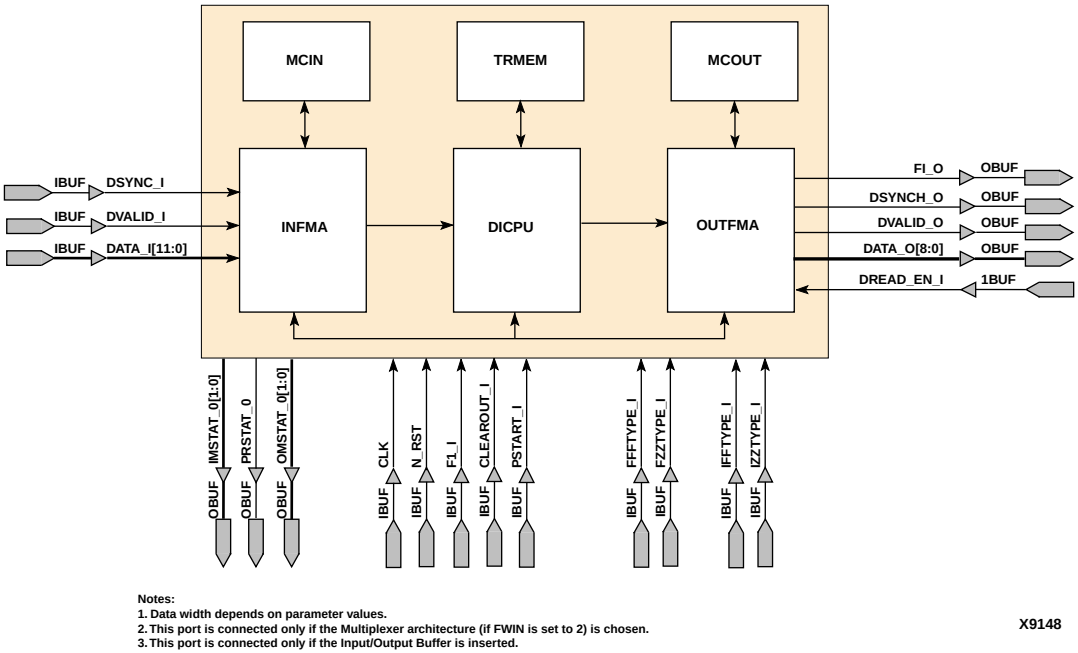

Figure 1: Arbiter Block Diagram

General Description

The FIDCT core performs 8x8 two-dimensional forward/inverse discrete cosine transform according to the following equations.

The forward DCT (FDCT) is defined as:

$$C(u,v) = \frac{2}{N} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} (x,y) \cos\frac{(2x+1)v\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

$$C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & uv = 0 \\ 1 & \text{otherwise} \end{cases}$$

The inverse DCT (IDCT) is defined as:

$$f(x,y) = \frac{2}{N} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} (x,y) \cos\frac{(2x+1)v\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

with u, v, x, y = 0, 1, 2, ... N-1

where x, y are spatial coordinates in the sample domain.

u,v are coordinates in the transform domain N = 8

The IDCT precision meets the constraints specified by the Joint CCITT/ISO committee which undergoes the IEEE Standard Specification for the Implementations of 8 by 8 Inverse Discrete Cosine Transform, Std. 1180-1990, December 6, 1990.

The macro programmability includes input and output memory buffers, input and output word length, internal bit width for temporary summations, cosine coefficients bit width, mismatch control block for the IDCT, MPEG1-2 compliance, field/frame reorder (MPEG2).

The FIDCT can be customized to perform either the Forward DCT (FDCT) or the Inverse DCT or the Forward/Inverse DCT functions.

Functional Description

The internal architecture of the FIDCT core is shown in Figure 1. A brief description of the operation of each module follows.

INFMA

This optional module re-orders input data (either pixels or DCT coefficients): these are first sent to the memory block MCIN, then read back and forwarded to DICPU block for processing. Different input data orders are accepted: 8x8 row scan, column scan, zig-zag scan. An additional 16x16 input row scan format, either field or frame, is allowed, which is useful for MPEG operation. Within INFMA, a fully parameterizable memory interface provides all the signals and protocols to interface to the most common types of memory blocks: synchronous/asynchronous protocol,

active edges of clock and control signals are some of the parameters the user can select.

MCIN, TRMEM, MCOUT

These Units represent the Input Buffer Memory, the Transpose Memory and the Output Buffer Memory, respectively. Their size can be controlled through the FIDCT parameter setting. MCIN and MCOUT, which are used when in/out data ordering is necessary, are required optionally according to the user parameter settings. However, TRMEM is required always to perform the DCT processing.

DICPU

This module performs the real DCT processing and includes a control and a processing pipeline. The control pipeline interacts with the FIDCT I/O signals by means of synchronization procedures and providing the correct sequence of activation to the functional units of the processing pipeline. The control logic provides automatic adjustment of the control signal timing, according to the parameter values concerning the number of multiplier pipeline stages as well as the number of latency access cycles to the memories.

The processing pipeline performs an I-D DCT operation (thus each 8x8 block passes through the pipeline twice), and is composed of 4 pipeline stages.

- a pre-processing unit containing register banks and add/subtract operators (according to the parameters used);
- a multipliers set to perform the cosine coefficients multiplication;
- a matrix accumulators bank to collect the matrix computation;
- a post-processing unit containing register banks and add/subtract operators.

A matrix transportation unit performs an intermediate 8x8 pixel row to column data reordering in TRMEM to provide data for the second processing step.

OUTFMA

This optional module re-orders the output data, either pixels or DCT coefficients. Data are received from the DICPU block; they are reordered using MCOUT, and subsequently output on the DATA_O bus. Different output data orders are generated: 8x8 row scan, column scan, zig-zag scan; an additional 16x16 output row scan format, either field or frame, is provided. A fully parameterizable memory interface provides all the signals and protocols to interface to the most common types of memory blocks.

Pinout

The pinout of this core has not been fixed to a specific FPGA I/O allowing flexibility with a user's application. Signal names are shown in the block diagram in Figure 1 and described in Table 1.

Table 1: Core Signal Pinout

Signal	Signal Direction	Description
CLK	Input	Master clock
N_RST	Input	Asynchronous reset
DATA_I[11:0][1]	Input	Data input
DSYNCH_1	Input	Data input
DVALID_I	Input	Data valid input
FI_I	Input	Forward or inverse transform selection input
PSTART	Output	Start operation
DATA_O[8:0][1]	Output	Data output
D_SYNCH_O	Output	Data synchronism output
D_VALID_O	Output	Data valid output
FI_O[2]	Output	Forward or inverse transform selection output
DREAD_EN_I	Output	Output data read enable
FFFTYPE_I[3]	Input	Input field/frame FDCT mode
FZZTYPE_I[3]	Output	Output zig-zag/scan IDCT mode
IFFTYPE_I[3]	Input	Output Field/Frame IDCT Mode
IZZTYPE_I[3]	Input	Input zig-zag/scan IDCT mode
CLEAR_OUT_I	Input	Clear output data request
IMSTAT_O[1:0]	Output	Input buffer memory status information
OMSTAT_O[1:0]	Output	Output buffer memory status information
PRSTAT_O	Output	Processing status(active/waiting for data)

Notes:
1. Port size is equal to PSIZE or CSIZE, where PSIZE and CSIZE are generic values.
2. This port is connected only if the multiplexed architecture (if FWIN is set to 2) is chosen.
3. This port is connected only if the Input/Output Buffer is inserted.

Core Modifications

The source code version of the FIDCT core is parametric. Parameters are implemented as a set of generics in the synthesizable VHDL source code of the core. Parameters allow the user to specify some architectural and functional features of the synthesized core netlist, so as to adapt it to a specific design or application.

Table 2: Core Parameters (VHDL Generics)

Parameter	Description
FWIN	Selects forward or inverse or muxed mode operation
PSIZE	Pixel data size
CZSIZE	Coefficient data size
CFSIZE	Cosine coefficient bit width
TWSIZE	Transpose memory word size
DPSIZE	Internal datapath size
NPIPE	Number of pipeline stages in multipliers
MISMCON	MPEG2 mismatch control for IDCT
FINORD	Forward DCT input order
FOUTORD	Forward DCT output order
FCHROMA	Forward DCT input chroma format
INORD	Inverse DCT input order
IOUTORD	Inverse DCT output order
ICHROMA	Inverse DCT input chroma order
I_LATENCY	INMC read latency
TR_LATENCY	TRMEM Read latency
O_LATENCY	OUTMC read latency

Verification Methods

Extensive functional and timing simulations have been performed for different values of the core parameters, using the Synopsys VSS simulator. Simulation scenarios (including data and command files) and parametric test bench used for design verification are provided with the core.

The test environment is split into two parts as shown in Figure 2: a VHDL test bench and a pair of executable programs written in C language used to test the standard compliance (IEEE 1180) of the macro.

The VHDL testbench environment is used to exercise all core configurations (Forward, Inverse, Multiplexed Forward/Inverse DCT)

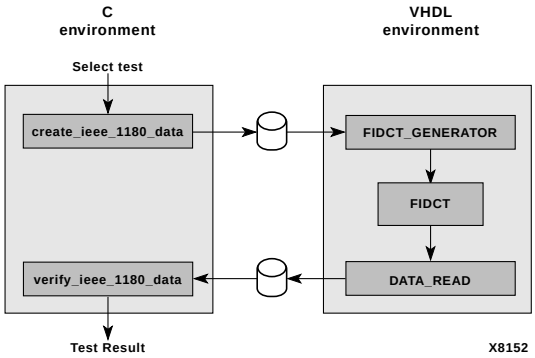

Figure 2: Test Environment for the FIDCT Core

The verification steps to check the IEEE 1180 standard compliance of the core are easily accomplished by automatic scripts, where the user simply fills in the desired parameter values to:

1. Generate the input data according to the parameters chosen
2. Run VHDL simulation;
3. Verify simulation results

Recommended Design Experience

Experience with the Xilinx design is recommended to the users of the netlist version of the core. For the source code version, users should also be familiar with the Synopsys FPGA synthesis tools (VHDL Compiler, FPGA Compiler) and simulator (VSS).

Strict C language expertise is not necessary as long as a C compiler is available; scripts compile automatically the C verification environment and run it.

Ordering Information

The FIDCT core is provided under license by CSELT S.p.A. for use in Xilinx programmable logic devices. Please contact CSELT S.p.A. for information about pricing, terms and conditions of sale.

CSELT S.p.A. reserves the right to change any specification detailed in this document at any time without notice, and assumes no responsibility for any error in this document.

Related Information

Xilinx Programmable Logic

For information on Xilinx programmable logic or development system software, contact your local Xilinx sales office, or:

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-559-7114
URL: www.xilinx.com

For general Xiilinx literature, contact:

Phone: +1 800-231-3386 (inside the US)
+1 408-879-5017 (outside the US)
E-mail: literature@xilinx.com

FIDCT Implementation Request Form

To: CSELT S.p.A.

FAX: +39 011 228 5695

E-mail: viplibrary@cselt.it

CSELT configures and ships Xilinx netlist versions of the Arbiter core customized to your specification. Please fill out and fax this form so that CSELT can respond with an appropriate quotation that includes performance and density metrics for the target Xilinx FPGA.

From: _____

Company:_____

Address:_____

City, State, Zip:_____

Country:_____

Phone:_____

FAX:_____

E-mail:_____

Implementation Issues

1. DCT function (forward, inverse, muxed)?_____

2. Internal bit width (normal, high)? _____

3. Transpose memory bit width (normal, high)? _____

4. Cosine multipliers bit width (normal, high)? _____

5. In/Out 8x8 Data Buffers (Yes, No)? _____

6. Pixel scanning (row, column)?_____

7. DCT coeffs scanning (row, col, zig-zag)? _____

8. Optimization choice (area, speed)? _____

Note: Items 6-7 are relevant only when item 5 is set to YES

Business Issues

1. Indicate time scales of requirement:
_____ date for decision
_____ date for placing order
_____ date of delivery

2. Indicate your area of responsibility:
_____ decision maker
_____ budget holder
_____ recommender

3. Has a budget been allocated for the purchase?
Yes _____ No _____

4. What volume do you expect to ship of the product that will use this core? _____

5. What major factors will influence your decision?
_____ cost
_____ customization
_____ testing
_____ implementation size

6. Are you considering any other solutions? _____