
Low Power High Throughput Configurable 2D-DCT/IDCT

Features

- IDCT IEEE1180-1990 compliant
- 0.25 clks/symbol throughput
- VGA 25fps 4:2:2 with 3.84MHz clk
- Very Low Power consumption
- Ultra low power when disabled
- High max clock speed
- Pipelined Design
- 28 clock cycle latency (19 initial)
- Flow control on input and output
- Output saturation logic included
- Up to 9bit 2's complement pixel data
- 12bit 2's complement DCT coeffs
- Fully synchronous design
- Asynchronous reset
- DCT or IDCT only variants
- ASIC or FPGA optimised variants
- Reduced area non-IEEE1180 variant
- "Drop-in" replacement for existing 2D-DCT/IDCT cores is possible
- RAM is not used, simplifying power-down, re-use, and reducing power consumption when function disabled

Applications

- Low Power or
- High Throughput applications
- JPEG, MPEG1/2/4, H261/3
- Digital Television
- Teleconferencing
- Medical Imaging
- Security Systems
- Battery operated products or
- Reduced chip package power rating
- Marketing advantage gained through low power or high throughput

Overview

The 2D-DCT/IDCT from **Jaskay Technology** has been designed as a low power alternative to 2D-DCT/IDCT functions that operate at 1 clk/symbol. Operation at 0.25clks/symbol enables the clock speed to be reduced to 1/4 of the frequency for the same data throughput. Advanced architecture and detailed design has kept logic area similar to 1 clk/symbol functions, enabling a power reduction of anywhere between 50% and 75%.

The logic activity in any 2D-DCT/IDCT implementation is typically high, close to 100%. Therefore, depending on the full product in which the 2D-DCT/IDCT is used as a sub-module, reducing power in this function could result in a very significant full product power reduction.

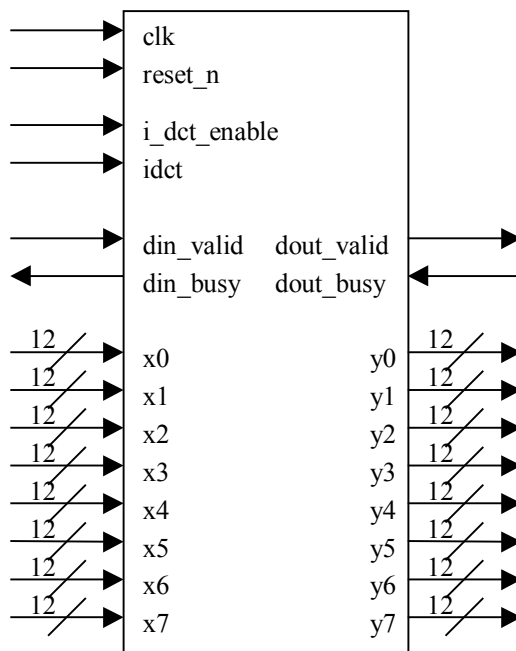
The 2D-DCT/IDCT from **Jaskay Technology** may therefore be used as a "drop-in" replacement, enabling power reduction in a product in one straightforward step. Note however that an extra clock may be required. Alternatively, it may form part of a new design, fully optimised for lower power operation.

The 2D-DCT/IDCT from **Jaskay Technology** may *also* be used in designs requiring high data throughput. The function is capable of up to four times the throughput of 1 clk/symbol cores. A Xilinx XC2V1000-5 implementation has achieved 75MHz operation with push-button synthesis using Leonardo Spectrum. Time spent with synthesis

Low Power High Throughput Configurable 2D-DCT/IDCT

could probably increase this clock speed. This particular implementation at 0.25 clks/symbol is therefore capable of real-time DCT or IDCT transform of SXGA (1280x1024) resolution at 100Hz with 4:2:2 component video, more than enough for many applications.

Symbol and Port Definitions



clk

The design is fully synchronous on the rising edge of clk.

reset_n

The reset signal is active low and asynchronous.

i_dct_enable

This signal enables the 2D-DCT/IDCT function when high. When low, the function enters a low power mode.

idct

This is the configuration signal to switch between the DCT and IDCT function. When high the IDCT function is selected, and the DCT function when low.

din_valid

This signal is active high and forms part of the flow control interface, indicating to the DCT or IDCT function when data on the x inputs is valid.

din_busy

This signal is active high and forms part of the flow control interface, indicating that the DCT or IDCT function is busy, and that current data must be held on the x inputs until this signal is low.

dout_valid

This signal is active high and forms part of the flow control interface, indicating that the DCT or IDCT function has valid data on the y outputs.

dout_busy

This signal is active high and forms part of the flow control interface, indicating that the recipient of data from the DCT or IDCT function is busy, and that current data, if valid, must be held on the y outputs until this signal is low.

x0-x7

These data inputs to the DCT or IDCT function are each 12bits wide. They accept 8bit or 9bit pixel data on the MSBs for the DCT function, and 12bit DCT coefficients for the IDCT function. Wrapper code is available to enable operation with one 12bit interface.

Low Power High Throughput Configurable 2D-DCT/IDCT

y0-y7

These data outputs from the DCT or IDCT function are each 12bits wide. They produce 8bit or 9bit pixel data on the MSBs for the IDCT function, or 12bit DCT coefficients for the DCT function. Wrapper code is available to enable operation with one 12bit interface.

width, 11bit or 12bit DCT coefficients).

4. Width of symbol interface (e.g. 8 symbols wide as shown in interface diagram, or 1 symbol wide, or other)
5. Target application (JPEG, etc)
6. Target implementation (SoC/ASIC or FPGA)

Verification

The DCT and IDCT functions have been verified with respect to behavioural models. The IDCT has been verified to be IEEE1180-1990 compliant following the methodology set out in the IEEE1180-1990 standard. The Flow Control Interface has been verified using a data source and data sink with random bursts of data and busy periods, over a large dataset, for both the DCT and IDCT functions.

If possible, please supply the following business information:

1. Time scales for requirement
 - a) Approximate decision date
 - b) Approximate order date
 - c) Approximate delivery date
2. Whether you are the decision maker for the purchase, the budget holder or are making a recommendation to the above people
3. What are the most important factors that will influence your decision?
 - a) Power
 - b) Cost
 - c) Implementation Area
 - d) Throughput
 - e) Interfacing factors
 - f) Customisation
 - g) Testing
4. Are you considering any other solutions?
5. How can **Jaskay Technology** make the product better suited to your needs?

Ordering Information

Please contact **Jaskay Technology** by email at enquiries@jaskay.biz for details about how to evaluate this product and for pricing information.

If possible, please supply the following implementation information:

1. Function required (DCT, IDCT or configurable DCT/IDCT)
2. IEEE1180-1990 compliance required for IDCT? Implementation area can be reduced to meet your accuracy requirement.
3. Width of data input and output required (e.g. 8bit or 9bit pixel

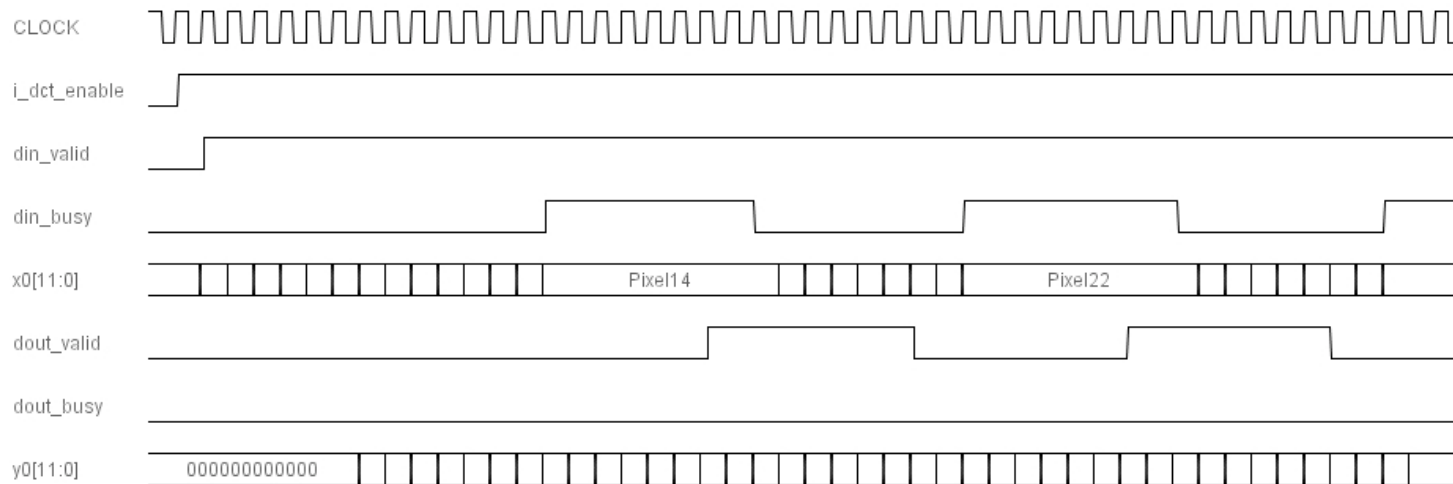
Low Power High Throughput Configurable 2D-DCT/IDCT

Flow Control Timing

1. *Maximum Throughput*

The following timing diagram shows the interface timing for maximum throughput, which assumes that data at the input is always available (`din_valid` always high) and that the data at the output may always be accepted (`dout_busy` always low). The initial latency is 19 clock cycles, followed by a regular 28 clock cycle latency. Throughput is 16 clocks per 8x8 block, or 0.25 clocks/symbol.

This timing diagram illustrates two important points about the flow control interface. Firstly, when `din_busy` goes high, data at the input must be held until `din_busy` goes low. Secondly, the data output will change every clock cycle, but should only be clocked when `dout_valid` is high.

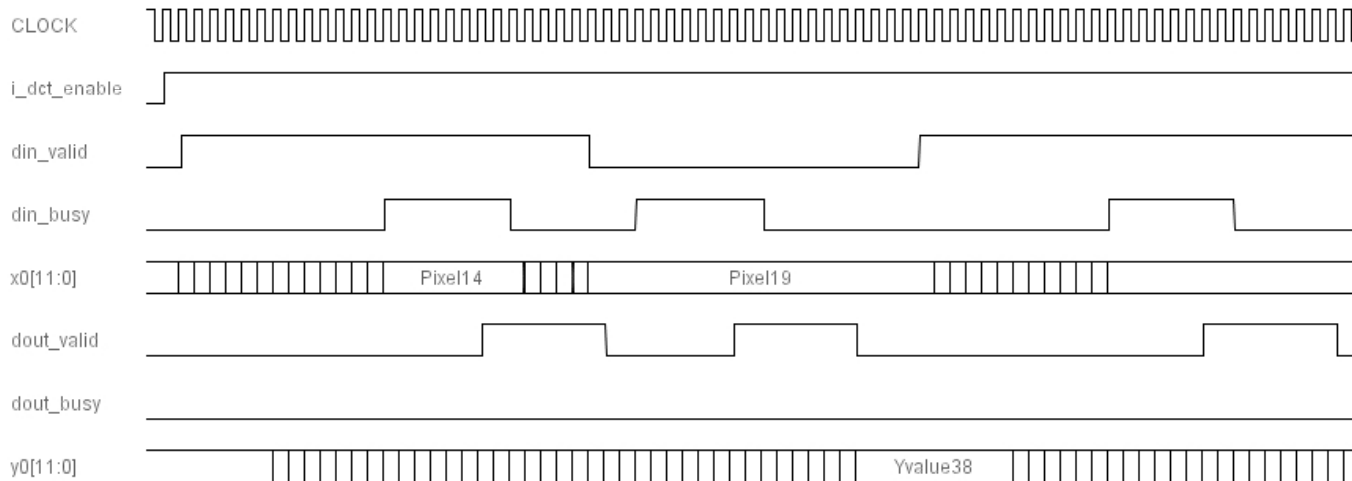


Low Power High Throughput Configurable 2D-DCT/IDCT

2. Timing For *din_valid* Signal

The timing diagram below shows timing behaviour for the *din_valid* signal. In particular, note that when data input is no longer valid, the internal pipeline continues to empty, see the *dout_valid* signal. The *din_busy* signal will also be asserted. Once *din_valid* is re-asserted, the internal pipeline begins to fill again. There is no minimum or maximum duration for the assertion or de-assertion of *din_valid*.

Note that in the timing diagram, the input data is shown held constant while *din_valid* is low. This need not be the case, the data input may take any value, and is ignored. New, valid data must be input when *din_valid* is re-asserted.



Low Power High Throughput Configurable 2D-DCT/IDCT

3. Timing for dout_busy Signal

The operation of the dout_busy signal may now be added to the Flow Control interface timing diagram. If the dout_busy signal is asserted while dout_valid is asserted, then the output data is held until the next clock edge after dout_busy is de-asserted. Dout_valid remains asserted.

Din_busy will also be asserted while dout_busy is asserted, if dout_valid is asserted. If Dout_valid is not asserted, then dout_busy is ignored.

