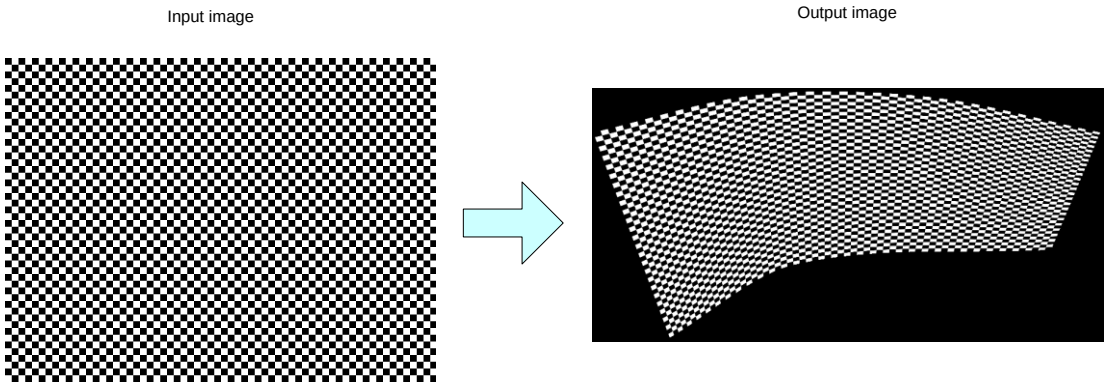
TES Warping Engine IP – Product Brief

Overview

TES Warping Engine is a specialized IP core for arbitrary high-performance re-mapping of bitmaps from memory to memory. Applications are for example pre-warping for projection on head-up displays or fisheye-correction of camera images.
The IP core adapts to different bus interfaces like AMBA APB and AHB/AXI as well as the Altera Avalon bus interface at different bus width (e.g. 32, 64, 128 bits).



Features

- Arbitrary mapping
  - Based on a output->input look-up table per pixel
  - Highly-efficient compression algorithm for the mapping LUT
- High-quality bilinear filtering
  - 16 x 16 subpixel positions
- High performance
  - Sophisticated caching mechanism proven in TES rendering engines
  - Allows real-time warping of videos or rendering output
- Flexible Color Formats
  - Input and output formats selectable at run-time
  - Available formats fully configurable at synthesis time, e.g.:
    - ARGB8888
    - ARGB4444
    - ARGB1555
    - RGB565
    - RGBA8888
    - RGBA4444
    - RGBA5551
    - 8 bit grayscale
    - etc.
- Definition of outside color
  - For seamless blending with possibly transparent background
- Easy configuration via registers (less than 20 functional registers)

Application Example: HUD warping

Projecting a rectangular image onto a curved surface yields a curved image. For a rectangular image to appear rectangular on a curved surface, it needs to be re-mapped first using the 'inverse curvature'. The projected image is then mapped into a non-rectangular region of the output image with 'transparent' (i.e. normally black) pixels all around:

Input image

Output image

When the output image is projected onto the windshield, it will appear rectangular from the driver position point of view.

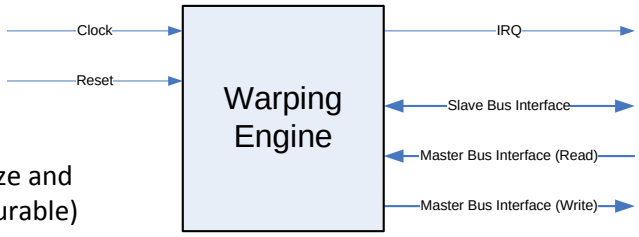
Application Example: Fisheye correction

Here the input image is distorted, while the output image is rectangular. Also the output image contains only pixels which exist in the input image (no 'outside' pixels like in the above HUD example). Still, the re-mapping is possible using an arbitrary look-up table per output pixel, which references the respective coordinates from the input image.

TES Warping Engine IP – Product Brief

December 3rd 2015

Easy Integration



- Low resource consumption
- Simple interface
- Single clock domain architecture
- High bus latency capable (cache/FIFO size and number of outstanding transfers configurable)
- Single bus master port (internal arbitration)
- Adaptors for common bus protocols
    - ARM AMBA: APB for register access, AHB or AXI for memory bus master access
    - Altera Avalon as bus adaptors for both register and bus master access
    - Other bus protocols can be easily adapted

Resource Usage

The actual resource usage of warping engine depends on the configuration (bus interface, run-time flexibility of image formats etc.). The following numbers give an indication of the resource usage for a standard configuration using a 32 bit bus interface.

FPGA Resource Usage

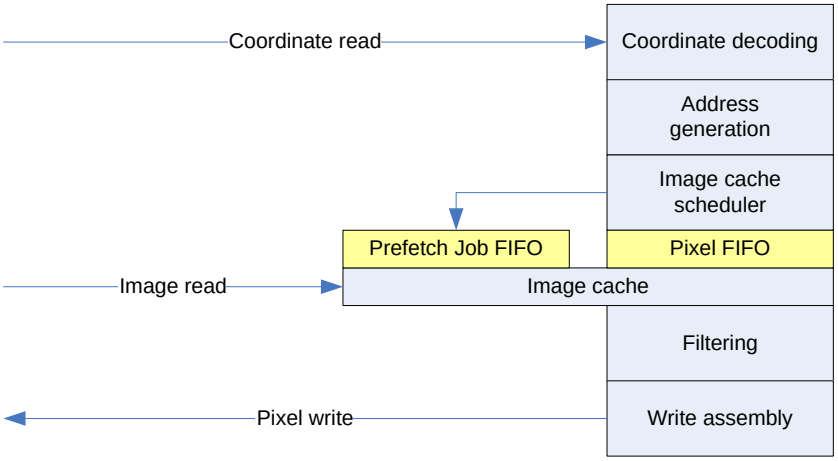
Logic cells	~5000 logic elements
DSP blocks	12 multipliers (8x8)
RAM blocks	7 M9K blocks

All numbers given for Altera Cyclone III device family

Power Efficient Design

- Memory blocks controlled by chip select port
- Prepared for efficient automatic clock gating

Block Diagram



Software Tools

TES provides free-of-charge tools:

- GUI tool for generation of example mappings using splines to model the mapping
- Command line tool to convert a plain LUT to the compressed format

Software Driver

Simple to use driver:

- Only about 20 different API functions
- Separate hardware-abstraction layer allowing easy porting to other platforms (e.g. different OS, different bus interface to warping engine etc.).
- Provided as C source code

FPGA Evaluation Kit

TES offers a free-of-charge evaluation kit for Altera FPGAs with the following content:

- Altera Quartus II / QSys projects for multiple Altera/Terasic standard evaluation boards (e.g. VEEK-MT, VEEK-MT-C5SoC)
- Warping Engine as Qsys component
  - Encrypted VHDL
  - OpenCore Plus hardware evaluation (timeout after 1 hour)
  - Nios II + Altera SoC Yocto Linux drivers as source
- Software tools to generate the compressed LUT format (see above)
- Software API documentation (HTML)
- Example software projects
- Tutorials/Howtos

Verification Details

A 100% algorithmic equivalent C model is used as reference for the verification of the RTL code.

The following test metrics are used:

- **Functional Coverage**
- **Code Coverage using Cadence IUS**
- **Static Code Checking using Cadence HAL**
- **Timing Checks**
- **FPGA prototyping**

Sales & Marketing Contact

graphics@tes-dst.com