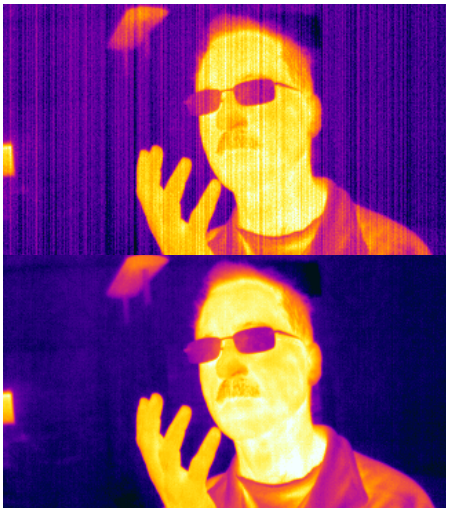
FLIR Accelerates Development of Thermal Imaging FPGA



Raw image (top) and image after applying filter developed with HDL Coder (bottom).

The Challenge

Accelerate the implementation of advanced thermal imaging filters and algorithms on FPGA hardware

The Solution

Use MATLAB to develop, simulate, and evaluate algorithms, and use HDL Coder to implement the best algorithms on FPGAs

The Results

- Time from concept to field-testable prototype reduced by 60%
- Enhancements completed in hours, not weeks
- Code reuse increased from zero to 30%

Thermal imaging infrared cameras are widely used in commercial applications, including security, firefighting, gas leak detection, and test and measurement. FPGAs within the cameras filter and process signals generated by sensors and detectors. Often, turning a new signal processing concept into an algorithm that runs in real time on a production camera is a lengthy process, because hardware engineers must translate algorithms developed by algorithm engineers into HDL, without intimate knowledge of how the algorithms work.

At FLIR Systems, engineers develop and simulate advanced algorithms in MATLAB® and then rapidly implement them on FPGAs with HDL Coder™. "In the past, we would rarely show simulations to our customers because it could take a long time for our ideas to make it into a product," says Nicholas Hogasten, manager of image processing technology at FLIR. "Recently, we showed a key customer some simulations of a new thermal imaging filter, the most complex filter we had ever developed. Our customer was ecstatic when, a few months later, we showed them the first working camera with this new filter, generated using HDL Coder, and the camera performed exactly like the MATLAB simulations."

The Challenge

Difficulties in FLIR's earlier development process stemmed from a disconnect between the algorithm engineers who developed new ideas and algorithms and the hardware engineers who implemented the algorithms on FPGAs. Algorithm engineers would evaluate new techniques for noise reduction or dynamic range compression and then hand off written specifications to hardware engineers, who may not have full knowledge of the algorithms.

"Often, the FPGA implementation would not perform like our simulations, and we never knew if there was a problem with the implementation or with the algorithm," says Hogasten. "Also, because the hardware engineers did not have a deep understanding of the algorithm, they did not know what assumptions were safe to make in optimizing it. If we later made a small enhancement to the algorithm, most of the HDL potentially had to be rewritten."

The Solution

FLIR established a new workflow for developing FPGA-based thermal imaging algorithms using MATLAB and HDL Coder.

Algorithm engineers use MATLAB and Image Processing Toolbox™ to explore new algorithms based on morphological operations and multidimensional image filtering.

These engineers select the algorithms for implementation and identify the algorithmic components that map to the target FPGA hardware. During this partitioning, the team replaces high-level functions from Image Processing Toolbox with MATLAB code that supports code generation. The Image Processing Toolbox algorithms provide a golden reference, easing verification of FLIR's custom MATLAB code.

To enable bit-true simulation and analysis, the engineers use integrated floating-point-to-fixed-point workflow in HDL Coder to

"With MATLAB and HDL Coder we are much more responsive to marketplace needs. We now embrace change, because we can take a new idea to a real-time-capable hardware prototype in just a few weeks. There is more joy in engineering, so we've increased job satisfaction as well as customer satisfaction."

—Nicholas Hogasten, FLIR Systems

automatically convert the floating-point MATLAB algorithms to fixed-point MATLAB code incorporating arithmetic and data types using Fixed-Point Designer™.

To support other test environments in use at FLIR, the team uses MATLAB Coder™ to generate C code and MEX-files from the generated fixed-point MATLAB code.

Next, the team generates synthesizable HDL code from the MATLAB algorithms using HDL Coder. The HDL code is then implemented and tested on the FPGA, and the results are verified against results from the fixed-point MATLAB algorithm.

In a related project, the engineers used MATLAB Compiler™ and Image Acquisition Toolbox™ to build an application that acquires images from cameras and frame grabbers, processes them using a variety of algorithms, and displays the results. This application enables other FLIR engineers to assess algorithms for a range of inputs, even if they do not have MATLAB installed.

The Results

Time from concept to field-testable prototype reduced by 60%. "With MATLAB and HDL Coder, we've eliminated the step of translating the initial algorithm to HDL by hand," says Hogasten. "Now algorithm developers can produce an FPGA prototype on their own, which has cut prototyping time by up to 60%."

Enhancements completed in hours, not weeks. "Recently, I asked one of our engineers to make a significant algorithmic change to a core filter," Hogasten reports. "Three hours later, he had made the change in MATLAB and redeployed the algorithm to the FPGA using HDL Coder. Previously, that type of change would have required six weeks."

Code reuse increased from zero to 30%. "We now have a common repository of algorithms, simple components, and MATLAB code that has been verified for HDL code generation," says Hogasten. "Previously, we had practically no code reuse, but today we reuse 30% of our MATLAB code to generate HDL for other projects."

Industry

- Electronics and semiconductors

Application Areas

- Image and video processing
- FPGA design

Capabilities

- Data acquisition
- Algorithm development
- HDL code generation and verification

Products Used

- MATLAB
- Fixed-Point Designer
- HDL Coder
- Image Acquisition Toolbox
- Image Processing Toolbox
- MATLAB Coder
- MATLAB Compiler

Learn More About FLIR

www.flir.com

MathWorks®
Accelerating the pace of engineering and science

mathworks.com

91997v01 06/13