

Reduce FPGA Design Time with PICO Express FPGA

Synfora's PICO Express accelerates algorithm-to-design completion on an FPGA.

by Fernando Martinez
Field Application Engineer
Synfora
fernando@synfora.com

The Xilinx® Spartan™ and Virtex™ product families offer designers a wide range of choices for embedded design. This array of products allows designers to create custom accelerators for embedded applications as well as entire systems on a single FPGA. The most vexing problem facing today's FPGA designers is shifting away from part capacity and toward design turn-around time.

Depending on the complexity of the target application, going from a written specification or reference C model to a working FPGA design can take anywhere from weeks to months. This increases the pressure on designers and makes the goal of first to market harder to reach.

Advanced Algorithmic Synthesis

The only way to accelerate design time on an FPGA is to automate the creation of the RTL. Although several attempts have been made to go from C to RTL, these solutions have fallen short of customer expectations because of language transformations, application complexity, and size limitations.

These concerns, which have limited the widespread adoption of high-level synthesis in the FPGA market, are addressed by Synfora's PICO platform, a complete development environment capable of taking sequential algorithmic ANSI-C and creating high-quality parallel hardware on par with RTL created by hand.

PICO is also fully integrated with FPGA back-end flows from both Xilinx and Synplicity. You can either use your own synthesis scripts or scripts provided by PICO to generate an FPGA programming bitstream.

PICO Express FPGA, which is a key component in the PICO platform, provides a complete development environment capable of taking sequential algorithmic ANSI-C and creating high-quality parallel hardware that is on par with RTL created by hand.

The design methodology in PICO Express FPGAs allows design teams to move to a higher level of abstraction without sacrificing design performance. Another aspect of the PICO design flow is the ability to retarget RTL generation across Xilinx product families in terms of part, speed grade, package, and clock frequency without having to modify the C code.

PICO is also fully integrated with FPGA back-end flows from both Xilinx and Synplicity. You can either use your own synthesis scripts or scripts provided by PICO to generate an FPGA programming bitstream.

Besides integration into standard FPGA back-end flows and the ability to target Spartan and Virtex parts, PICO gives you the ability to carry out what-if analysis. Under design space exploration, PICO allows side-by-side comparisons of multiple designs targeted at different throughputs, clock frequencies, part families, package options, and speed grades.

In this article, I'll show how PICO Express can quickly take an ANSI-C functional model and create a fully functional FPGA design. The application chosen for this demonstration combines real-time color space conversion from RGB to YUV and a Sobel edge detector. Although the C model for this application is part independent, Synpora has implemented a working system on the Spartan-3E video display kit.

Application Development on PICO Express

Application development on PICO Express is divided into three stages. The first two stages are the driver and synthesizable C model creation. As the example will show,

the synthesizable C model is completely sequential without any extra need for a user declaration of parallelism. The advanced compiler built into PICO automatically analyzes user applications to extract data dependencies and derive parallelism from sequential code.

The driver code comprises a test bench used to verify the correct operation of the function targeted for hardware synthesis. In the case of this example, the driver code reads input image files, passes the data to the hardware procedure, and gathers the results. The results of the hardware function are compared against the expected output created during test vector selection.

During the several steps needed to go from C to RTL, PICO simulates each transformation to ensure correct hardware by design. The simulations during hardware design are also useful in estimating design performance and resource utilization before RTL simulation and back-end place and route.

The second stage is the creation of the synthesizable C model. The synthesizable C model is the software procedure that will be implemented as a hardware block. This procedure code can encompass functionality as simple as a single accelerator block to as complex as a complete H.264 encoder/decoder. You only need to let PICO know the name of the top-level procedure in the synthesizable model. PICO Express automatically transforms all sub-procedures into hardware.

Color Conversion and Edge Detection

The color conversion and Sobel edge detection application is divided into two main components. Color conversion takes the RGB from the video source and

creates an on-the-fly YUV representation of this video. The conversion to YUV has the benefit of acting as a filtering operation, which reduces noise in the original RGB video stream.

Figure 1 shows a summarized pseudo code for the color converter; note that the type of C code accepted by PICO Express is no different from the C code accepted by any ANSI-C compatible software compiler. This allows for code development using open-source tools such as gcc.

```

unsigned char rgb_to_yuv(unsigned int rgb) {
...
result = ((66* Rf)>> 8) + ((129* Gf)>>8) + ((25* Bf)>>8) + 4096;
result = (result + 128) >> 8;
if(result >255) result =255;
...
}
    
```

Figure 1 – RGB to YUV converter

After color conversion, the new video stream is passed through a Sobel edge detector. Sobel edge detection is a classic filter used in image processing to reduce the information in both still images and video to a series of edges. The fundamentals of Sobel edge detection are based on:

$$N(x, y) = \sum_{k=-1}^1 \sum_{j=-1}^1 K(j, k) p(x-j, y-k)$$

This equation shows the use of a 3 x 3 window to calculate the strength of an edge at the center of the window. Depending on the chosen threshold, pixels will either have a value of 1 or 0 after the windowing function. Figure 2 shows the code used for the edge detector.

Like the code in Figure 1, the code in Figure 2 is completely compliant with ANSI-C and can be compiled on any C compiler. Figure 2 shows the use of arrays for storing both horizontal and vertical windows.

```
unsigned char sobel(  
    unsigned int window[3][3])  
{  
    ...  
    y00 = rgb_to_y(window[0][0]);  
    y01 = rgb_to_y(window[0][1]);  
    y02 = rgb_to_y(window[0][2]);  
    y10 = rgb_to_y(window[1][0]);  
    y12 = rgb_to_y(window[1][2]);  
    y20 = rgb_to_y(window[2][0]);  
    y21 = rgb_to_y(window[2][1]);  
    y22 = rgb_to_y(window[2][2]);  
  
    weight_horizontal = weight_calc(  
        y00, y01, y02, y20, y21, y22);  
  
    weight_vertical = weight_calc(  
        y00, y10, y20, y02, y12, y22);  
  
    total_weight = weight_horizontal + weight_vertical;  
    return total_weight >= THRESHOLD;  
}
```

Figure 2 – Sobel edge detector

PICO will analyze arrays in your code to determine the optimal mapping to minimize resource utilization and maximize performance. Depending on the size of an array and how it is used in the design application, PICO will determine the optimal storage medium for the array as LUTs, block RAMs, or external memories. It is the advanced memory analysis capabilities of PICO that allow for a simple coding model for memories. Arrays are a common storage structure with an easily understood usage model in the C domain. Therefore, designers can specify both simple and complex memory architectures with ease.

Figure 3 summarizes the data and con-

trol flow in this example. For the board implementation, the bitstream is mapped onto a Spartan-3E device. The base C code used to create the hardware is independent of the source video resolution. Like in most video processing systems, the PICO-designed hardware will use the first image frame to calculate the image resolution from the video source.

In this case, the video is of HD quality at 720p resolution. On the FPGA, the system runs at 133 MHz, which is sufficient to comply with the DVI standard for 720p HD video. In terms of synthesis time, this application takes less than five minutes in PICO to generate RTL and less than 10 minutes to generate a bitstream from Xilinx Synthesis Technology (XST).

Conclusion

PICO Express for FPGA provides the necessary tools and methodology for you to quickly go from algorithmic code to fully functional designs on an FPGA. The ANSI-C based input accepted by PICO presents you with a familiar coding language and helps reduce the learning curve associated with high-level synthesis tools. Also, PICO provides the capability to quickly compare price/performance trade-offs that allow you to choose the right part without having to modify your application code.

For additional information on how PICO Express can accelerate the development of FPGA-based embedded systems, visit www.synfora.com.

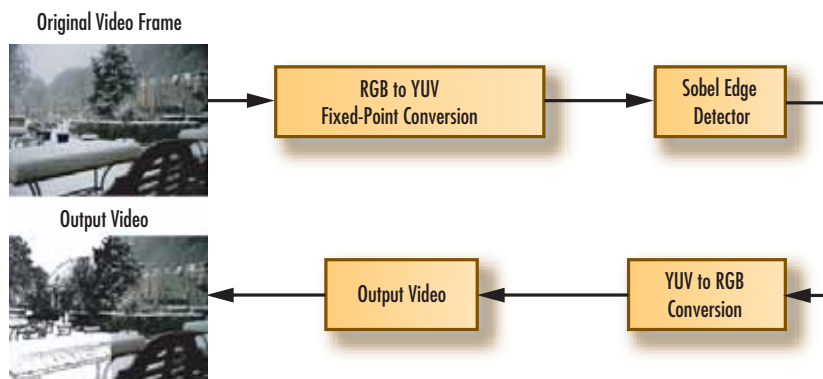


Figure 3 – Color conversion and Sobel edge detector

Topics Relevant to You Delivered in One Monthly Newsletter!

Thanks for your feedback! We have now created a new Xilinx newsletter that will provide you all of the information you need in one location. This newsletter will replace the multiple topic newsletters you have currently been receiving.

It's simple! Choose only the topics you want to receive and your customized newsletter will be delivered to your inbox every month.

<http://newsletter.xilinx.com>



XILINX®