Enhancing eFPGA Functionality with Speedcore Custom Blocks

achronix SEMICONDUCTOR CORPORATION

December 22, 2017 **WP009**

Abstract

Achronix Speedcore™ eFPGA IP can be integrated in an SoC for high-performance, compute-intensive and real-time processing applications such as AI, automotive sensor fusion, network acceleration and wireless 5G. Speedcore eFPGA IP is a game-changer for SoC developers, allowing them to add flexibility to their products by including FPGA technology in their ASICs. For SoC development, companies specify the quantity and mix of look-up-table (LUT) logic, embedded memory blocks, and DSP blocks that best meets their needs. Along with these functions, Achronix now offers the ability for companies to define custom block functions, optimized for their application, that can also be included in the eFPGA fabric. Speedcore custom blocks increase die area efficiency, increase performance and lower power.

Introduction

With an eFPGA, a customer defines their resource requirements for logic (look-up-tables), embedded memory blocks and DSP blocks as well as other parameters such as aspect ratio, number of clocks and configuration bus size. Once defined, Achronix delivers the configured eFPGA IP to the customer for integration into their SoC. In addition to the IP, Achronix also delivers a customized version of the Achronix ACE design tools that allows the end user of the SoC to place, route, and configure the eFPGA.

Along with the standard logic, embedded memory and DSP blocks, customers can also define their own custom block functions to be included in the Speedcore eFPGA. These customized blocks are integrated into the logic fabric alongside the traditional building blocks of LUTs, RAMs, and DSPs, greatly increasing the capability of the eFPGA by adding functions optimized to decrease area and/or increase performance of targeted applications. The Speedcore architecture is fully permutable, meaning Speedcore custom blocks (along with the standard Speedcore blocks) can be assembled in flexible columns to create the optimal programmable fabric for any given application.

The customer, with assistance from Achronix, defines the functionality of the custom block(s). As these custom blocks are distributed throughout the eFPGA fabric, optimal candidates are functions that are repeated many times in the typical applications for the SoC. Achronix delivers a Speedcore instance with the new custom block (s) integrated in the fabric (see Designing a Custom Block (see page 5) later).

With Speedcore custom blocks, the end user design flow for the eFPGA is the same as for a traditional FPGA. Customers instantiate the custom blocks in their RTL and use ACE tools to place and round the logic design. ACE offers full support for Verilog, SystemVerilog, and VHDL design flows.

> **Note**
>
> For a discussion of the teams, rolls, and responsibilities in a Speedcore engagement, see the blog post: Who's Who in the Zoo.

Custom Blocks Examples

Candidates for custom blocks can be any function that consumes a lot of programmable resources and is highly repeated in an the end application. Good candidates are functions that typically cause performance bottlenecks and/or are highly resource inefficient when implemented in an FPGA fabric. These functions can range from wide MUXes, custom math blocks, custom memory configurations to more specialized functions such as TCAMs. Barrel shifters and bit manipulation structures can also be fully implemented in Speedcore custom blocks, allowing for larger, sophisticated applications to be built in less area and operate at higher frequencies.

Speedcore custom blocks can be defined collaboratively with Achronix through a detailed architecture analysis of the customer's target application. What follows are some more complex examples that demonstrate the power of Speedcore custom blocks.

YOLO Object Recognition Algorithm

You only look once (YOLO) is a state-of-the-art, real-time object detection algorithm using neural networks that offers greatly increased performance over earlier methods. This algorithm relies on a large number of matrix multipliers. When implemented in an FPGA, these matrix multipliers are built using DSP and RAM blocks. The problem arises in the mismatch between the optimal configuration of the DSP and RAM blocks needed by YOLO versus what is found in a typical FPGA fabric. For example, an FPGA fabric may offer DSP blocks with 18 × 27 multiplication/accumulation and 32 × 128 RAMs, where the optimal solution would be a fabric with 16 × 8 DSP blocks with 48 × 1024 RAMs. Achronix built an example instance with DSP and RAM custom blocks for the YOLO algorithm, reducing the die area by 40% compared to the same function implemented using standard DSP and RAM blocks.

Large String Search Functions

String search is a highly recursive function used to match a search operand within a data stream, requiring a large number of parallel comparator arrays. Building a parallel search structure to operate on a 64-bit bus consumes more than 1,000 LUTs in a standard FPGA fabric. The entire structure can be implemented as a single custom block in Speedcore, which reduces the die area by over 90%.

400 Gbps Packet Processing Application

Central to a packet processing application is packet segment extraction/insertion. Trying to implement this type of functionality in a standard FPGA is not possible because of the f_{MAX} and large data bus size requirements. An example configuration would be a 512-bit bus running at 800 MHz, which far exceeds the performance capability of standard FPGAs. However, converting the packet segment extraction/insertion functionality to a Speedcore custom block results in a compact logic fabric capable of handing the performance requirements of the 400 Gbps packet processing data-path. All of the control functionality that operates at a lower bandwidth can then easily be implemented in the eFPGA fabric.

The Case for Custom Blocks

Die Area Reduction

In a traditional FPGAs, roughly 30%–50% of the die area is consumed by its complex I/O ring, consisting of I/O buffers, SerDes, PHYs, protocol controllers, PLLs, and programmable clocking routing. For any given FPGA design, some portion of the core programmable fabric is consumed with functionality that is dedicated for the FPGA to interface to other devices on the PCB. This wrapper logic is needed to handle functions such as chip-to-chip communication (supporting logic for the MAC/PHY, packed handling logic, protocol engine, etc.) and off-chip memory (interface logic, DMA, etc.). This wrapper logic can consume a significant percentage of the fabric resources in an FPGA. As an example, Microsoft found that for their Configurable Cloud project (a new cloud-scale, FPGA-based acceleration architecture) that roughly 40% of the FPGA's core resources were consumed by common wrapper logic needed for communication between network accelerators (see *A Cloud-Scale Acceleration Architecture* for details). This figure is probably not atypical of any large system partitioned across multiple FPGAs.

Assuming that 40% of the FPGA's die area is consumed by the FPGA's I/O ring, and then another 40% of the remaining core is consumed by wrapper logic, then only 36% of the die area of the FPGA die area implements the actual application logic.

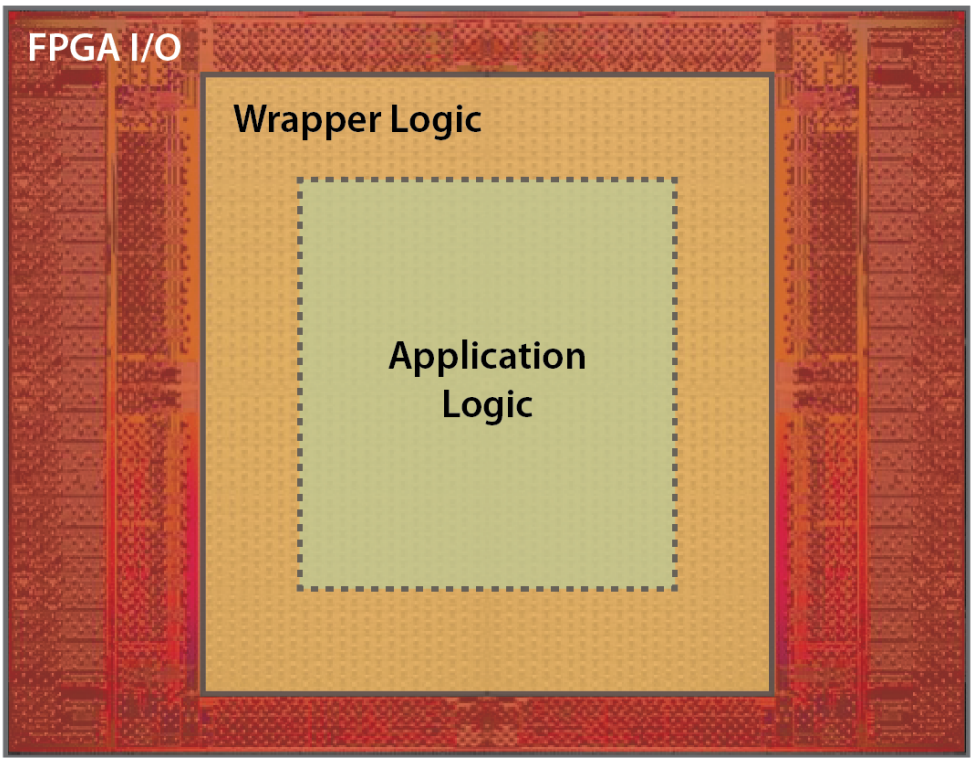

Figure 1: ***Two Thirds of a the Die Area of a Standalone FPGA is Consumed by Overhead***

With a Speedcore eFPGA-based solution, the I/O ring and wrapper logic functions originally found in a standalone FPGA-based solution can be implemented in the host SoC where needed. Therefore, A Speedcore instance must only be large enough to implement the actual application logic. As a result, the die size of a Speedcore instance is substantially smaller than that of a standalone FPGA implementing the same functionality.

By converting functions that consume a lot of programmable resources and are highly repeated in an application to custom blocks, customers can further reduce the die area of Speedcore instance. Assuming a 50% die size reduction when using Speedcore custom blocks, the end result is that the die size of the Speedcore instance is roughly one sixth the size of that of a standalone FPGA implementing the same application. Custom blocks greatly improve the die area efficiency of Speedcore eFPGAs, creating an programmable solution that is a fraction of the die size of a standalone FPGA.

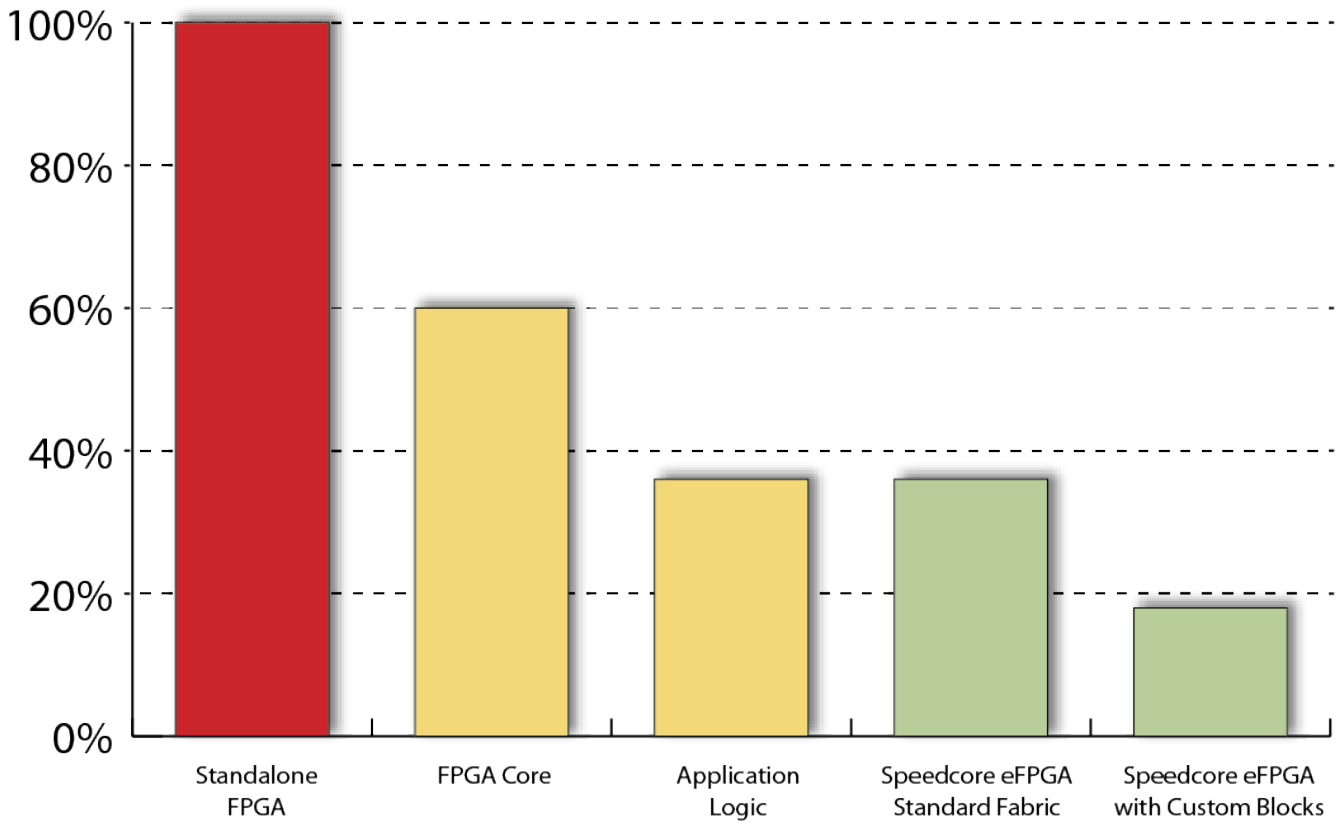

Figure 2: ***Speedcore eFPGA with Custom Blocks Results in Greatly Reduced Die Area***

Performance Enhancement

The f_{MAX} performance of FPGA fabrics is determined by routing and resource block delays for worst-case signal paths.Typical signal paths pass through multiple LUTs plus the routing delays between the LUTs. Improved placement of LUTs reduces the routing delays, but has no impact on LUT and block function delays. With custom blocks, functions that are typically used for a given application can be converted to a hard function in the eFPGA. These custom blocks can easily reduce complex logic with many levels of logic to a single logic level (see 400 Gbps Packet Processing Application (see page 2) for an example).

Speedcore look-up-tables (LUTs), RAM blocks, DSP64 blocks and custom blocks can then be assembled in flexible columns to create the optimal programmable fabric for the target application. Along with the performance increase, the custom blocks reduce the resource usage and improves place-and-route results because they eliminate the routing that would have been required to build the function with fabric resources.

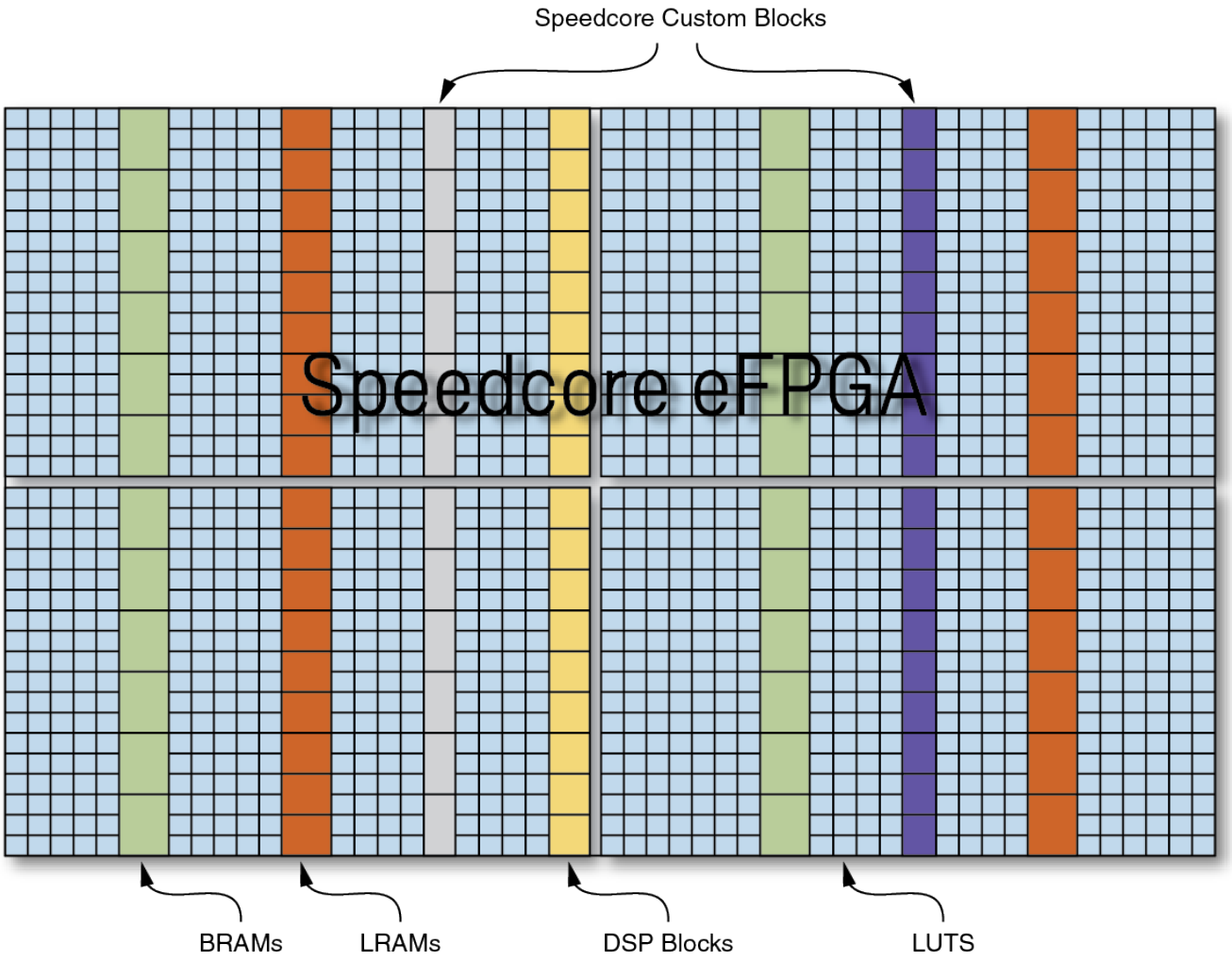

Figure 3: ***Example eFPGA Instance with Custom Blocks***

Designing a Custom Block

The design process for a custom block begins with a collaborative definition process between Achronix and the Speedcore customer. Achronix uses internal tools to profile a customer's typical designs to identify functions that are optimal candidates for conversion to Speedcore custom blocks. Good candidates are regularly repeated functions that when converted to a custom block result in significant area savings, performance gains and/or power reduction.

Achronix accepts multiple inputs for customers to define their custom blocks including a logic specification, RTL or they can keep the functionality completely confidential from Achronix and deliver a GDS file describing the block. Once the custom block design is completed, Achronix creates the new Speedcore instance in the ACE design tools for the customer to perform their benchmark analysis.

Once the custom block is defined, Achronix adds the new custom block to the Speedcore Builder tool to allow the customer to obtain die area and power details based on the number of instances that they want in their eFPGA. The new custom blocks are fully supported in Achronix's ACE design tools, including recognizing inputs, outputs, registers and timing information for the custom blocks. The customer instantiates the custom block function in their design and the ACE design tools will place and route the design accordingly.

Design Tool Support

Achronix ACE design tools fully support Speedcore custom blocks from design capture to bitstream generation and system debug in the same way as it does for memories and DSP blocks. Achronix creates a unique GUI for each Speedcore custom block that manages all configuration and validation rules. This GUI automatically creates a component for use in designs by the end user.

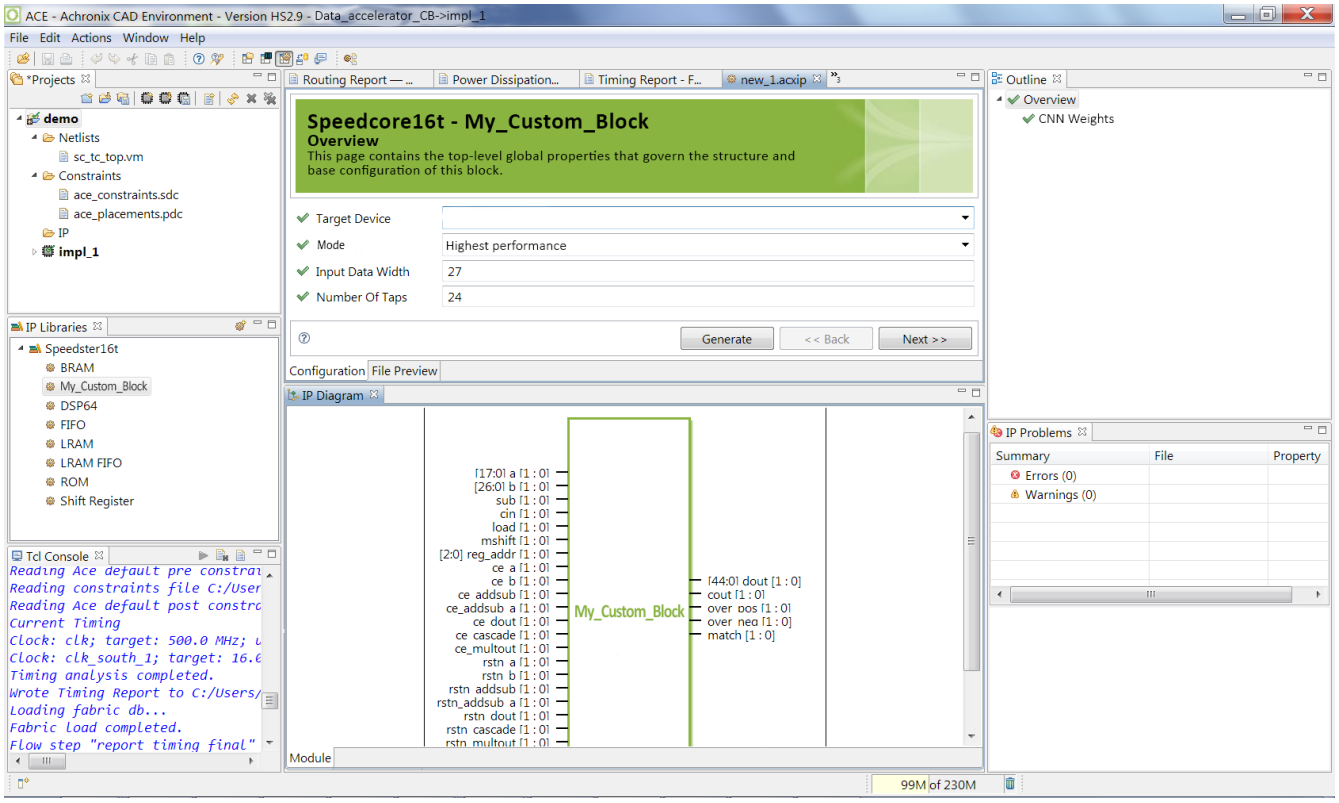

Figure 4: *ACE GUI for a Custom Block*

Customers can use the powerful ACE floorplanner tool for design optimization and to make regional or site assignments for all block instances. The floorplanner fully supports the custom block with all the same feature as standard Speedcore blocks, including the display of inputs and outputs plus detailed timing information. ACE also includes a critical path analysis tool that allows customers to analyze timing.

After floorplanning, ACE contains the full timing details for all configurations of the Speedcore custom block, which allows ACE to complete timing-based place-and-route for designs. Timing information for the custom block is also included in the standard timing report issued by ACE.

Customers can use ACE's powerful Snapshot embedded logic analyzer to create complex triggers and show run-time signals within a Speedcore instance.

Conclusion

Speedcore custom blocks massively improve performance, power, and area, enabling functionality that has never before been possible in standalone FPGAs. With Speedcore custom blocks, customers gain ASIC efficiency while retaining FPGA flexibility, resulting in a highly efficient implementation that minimizes power and area while maximizing data throughput with ASIC-level performance.

Achronix Semiconductor Corporation

2953 Bunker Hill Lane, Suite 101
Santa Clara, CA 95054
USA

Phone : 855.GHZ.FPGA (855.449.3742)
Fax : 408.286.3645
E-mail : info@achronix.com