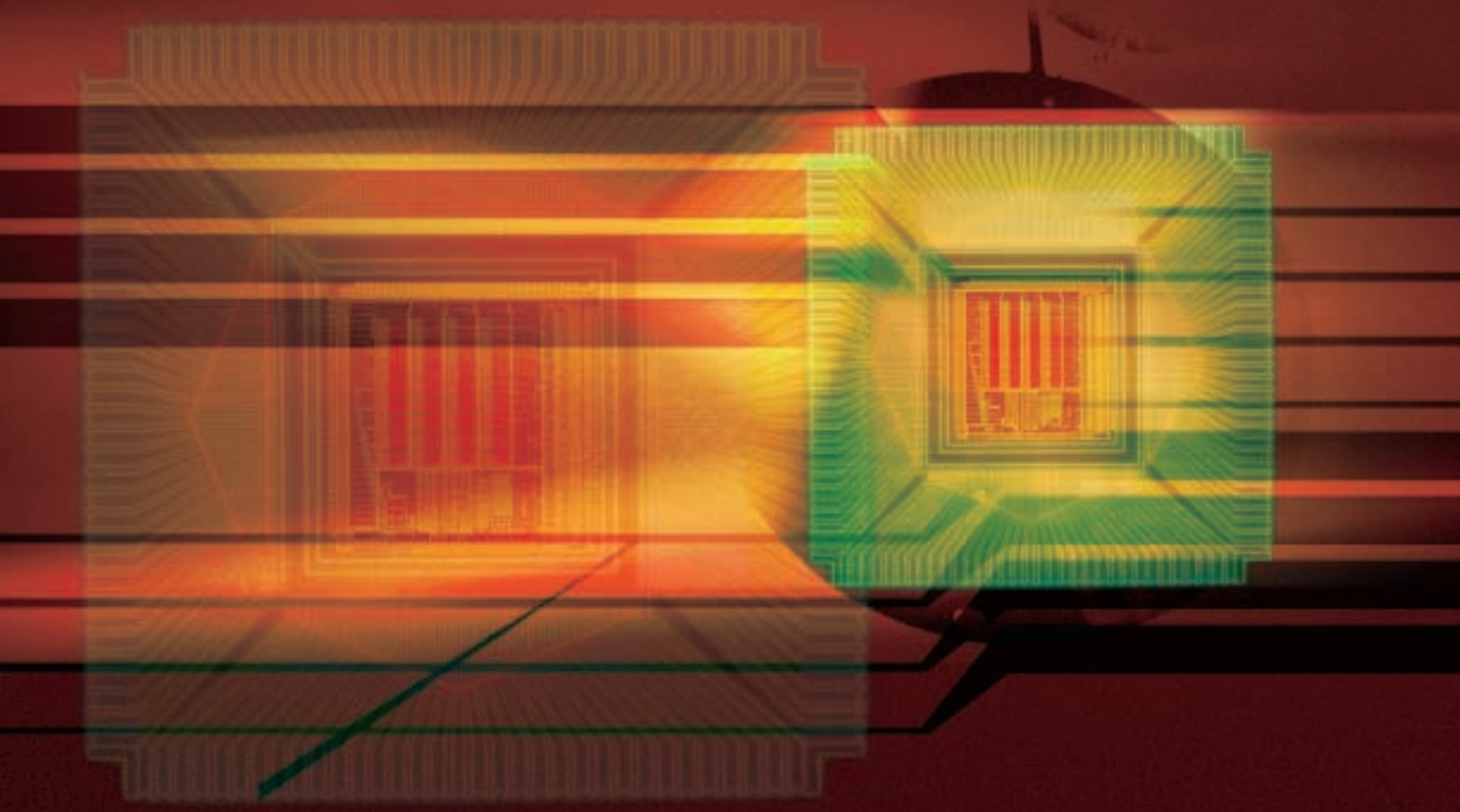
Co-Processor Acceleration and Design Reuse: Extending the Market for Platform FPGAs with DK1 Design Suite

Celoxica's DK1 design suite accelerates co-processing between a CPU and a Platform FPGA.



by Dennis Nye
Senior Vice President, Worldwide Sales and Marketing
Celoxica Limited
dennis.nye@celoxica.com

Few embedded applications will ever be fully created in programmable logic. Sequential algorithms are frequently best performed by software implementations using conventional CPU architectures. Optimal hardware acceleration comes most often from functions performed in parallel, normally in the physical form of a hardware co-processor closely coupled with the CPU.

In combination, the Xilinx Platform FPGA and the Celoxica™ DK1 system-level design suite represent a significant new approach to creating embedded solutions for a broad range of leading-edge applications, such as base stations, network compression, encryption, network traffic control, digital television, and more.

To exploit the multimillion-gate potential of advanced Platform FPGA architectures, companies need complementary methods that accelerate front-end logic design. C-based languages excel at implementing the algorithmic functions needed for system-level designs.

The DK1 design suite is a bridge across which Platform FPGAs can move into new markets. By addressing the needs of software engineers who want to benefit from the advantages of hardware acceleration, the DK1 suite speaks their language – Handel-C, a syntax based on standard ANSI-C. Thus, it is relatively simple for software engineers to learn and begin the process of converting C algorithms into Handel-C code to accelerate compute-intensive functions in parallel hardware.

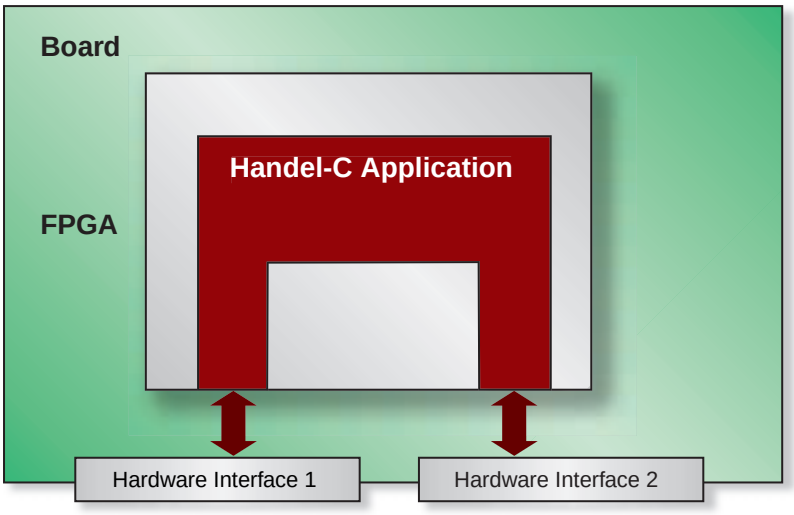


Figure 1 - Migrating software into an FPGA is just the first step.

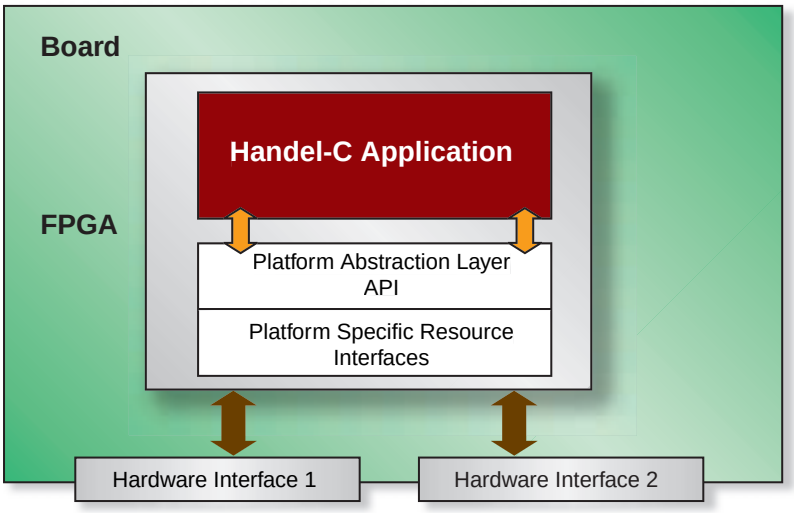


Figure 2 - Platform Specific Resource Interfaces reduce porting times.

PAL for FPGAs

Facilitating the migration of software into an FPGA is a useful first step (Figure 1), but engineers developing acceleration solutions are looking for further support. They expect the hardware independence offered by modern operating systems. Introducing this hardware independence into an FPGA environment requires the implementation of a layered and consistent approach to hardware interfacing – not dissimilar to the concept of device drivers within an operating system. Celoxica's recently announced Platform Abstraction Layer (PAL) strategy for FPGA implementations provides exactly this type of hardware independence.

PAL shields development engineers from low-level hardware interfaces in order to ease the integration of FPGAs with physical resources (Figure 2). It does this by providing a library of low-level interfaces to specific platform resources, such as I/O or memory. This library, called the Platform Resource Support Package (PRSP), is then accessed from the applications on the FPGA using a simple and consistent Application Programming Interface – the PAL API.

PRSPs are written once for each embedded system. Applications using the API to access platform resources are then portable between systems utilizing the PRSP concept. This approach can reduce porting times significantly. Development engineers can rapidly port designs from their prototype platform to their final system, or perform design re-use from one gener-

ation of a product to the next. So, at the same time as accelerating the development cycle for products utilizing FPGAs, the PAL approach allows companies to make more effective use of their design resources.

MPA Integrates CPU and FPGA

While PAL simplifies the implementation of solutions requiring access to the physical resources of the embedded solution, co-processing solutions will require sophisticated integration between those parts of the application running on the microprocessor and the accelerating routines provided on the FPGA. Developing this integration on a case-by-case basis would be unwieldy and complex (Figure 3).

Celoxica's Multi-Port Adaptor (MPA) technology provides the missing integration between the microprocessor and FPGA elements of a solution. By itself, MPA – using the PAL API – is also independent of the specific hardware solution.

The MPA provides a simple mechanism for function calls to be made from software within the microprocessor to functions in the FPGA or vice-versa (Figure 4). The developer may safely make multiple calls to the same or multiple functions as the MPA architecture ensures the management of the sequence of calls, delivering and returning data in the correct sequence, and to the correct function.

Conclusion

The benefits of a simple, layered approach for co-processor acceleration are:

- **Reduced solution complexity**
- **Reduced time to market**
- **Reduced cost**
- **Improved design re-use**

To learn more about Celoxica DK1 design suite, go to ***www.celoxica.com/products/design_suite/.***

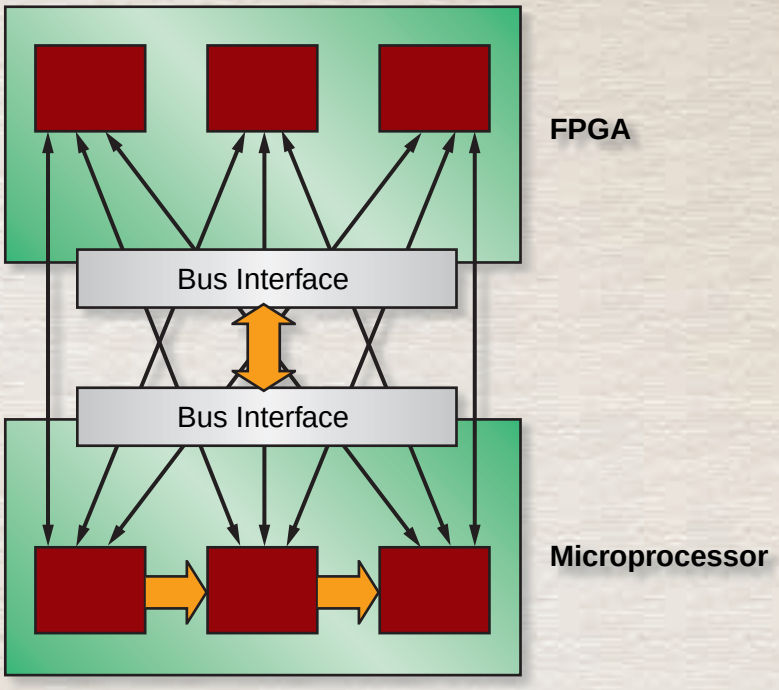


Figure 3 - FPGA-to-microprocessor without MPA support is too complicated.

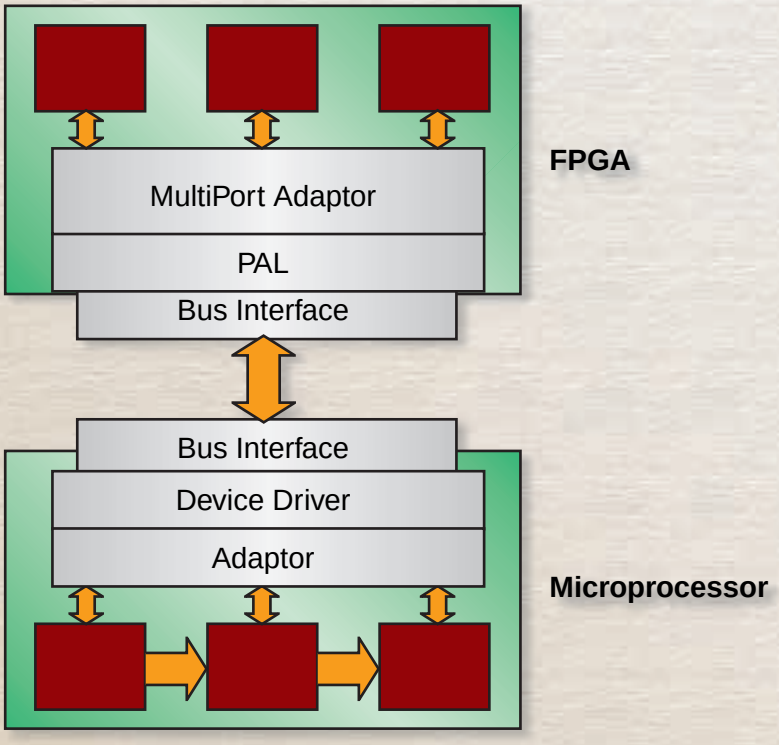


Figure 4 - DK1 design suite enhances and accelerates FPGA co-processing with a microprocessor.