StellarIP

To speed the development of better VHDL or FPGA designs, it is necessary to increase the level of automation available in design tools. StellarIP addresses this need to simplify FPGA firmware design by relying on proven IP cores that can be reused as building blocks for new designs. The foundation of StellarIP is an extensive library of firmware that is combined with an IP-centric development environment designed to increase productivity and minimize inefficiencies. The StellarIP library of working reference designs reduces the amount of time designers need to spend creating new projects which allows them to begin using their hardware immediately upon delivery. The tool enables FPGA firmware designers to quickly and easily incorporate and connect application-specific code to an existing design.

Creating a firmware design often involves grouping existing IP blocks together with new blocks. An IP block typically needs to receive configuration data and must have a means to return status information. In addition, one or more fast data input and output paths are required. StellarIP defines a standard command interface structure that is easy to use and supported all the way from the host software layer. The data paths between IP blocks are also standardized. A top-level wrapper design file is always required and a project must be created for FPGA tools such as Xilinx ISE or Vivado. StellarIP automates these steps. After IP blocks are connected using the graphical interface, StellarIP creates a top-level design file, synthesis constraint file (ISE/XDC), and a compatible project file for VIVADO and/or ISE.

"The foundation of StellarIP is an extensive library of firmware that is combined with an IP-centric development environment designed to increase productivity and minimize inefficiencies."

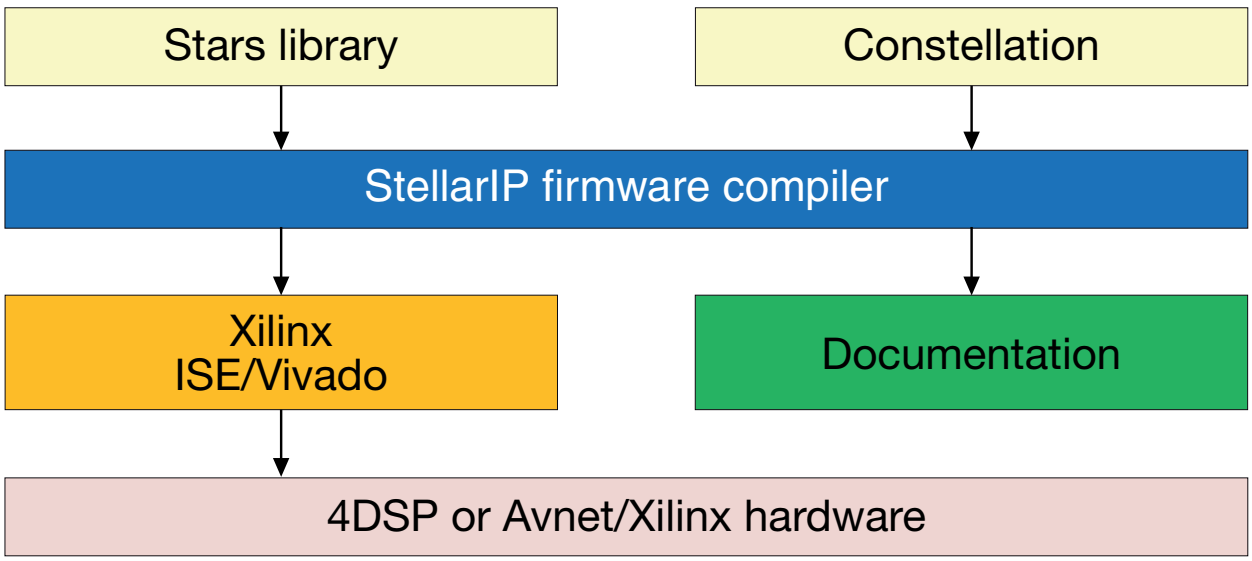

Figure 1: StellarIP design flow

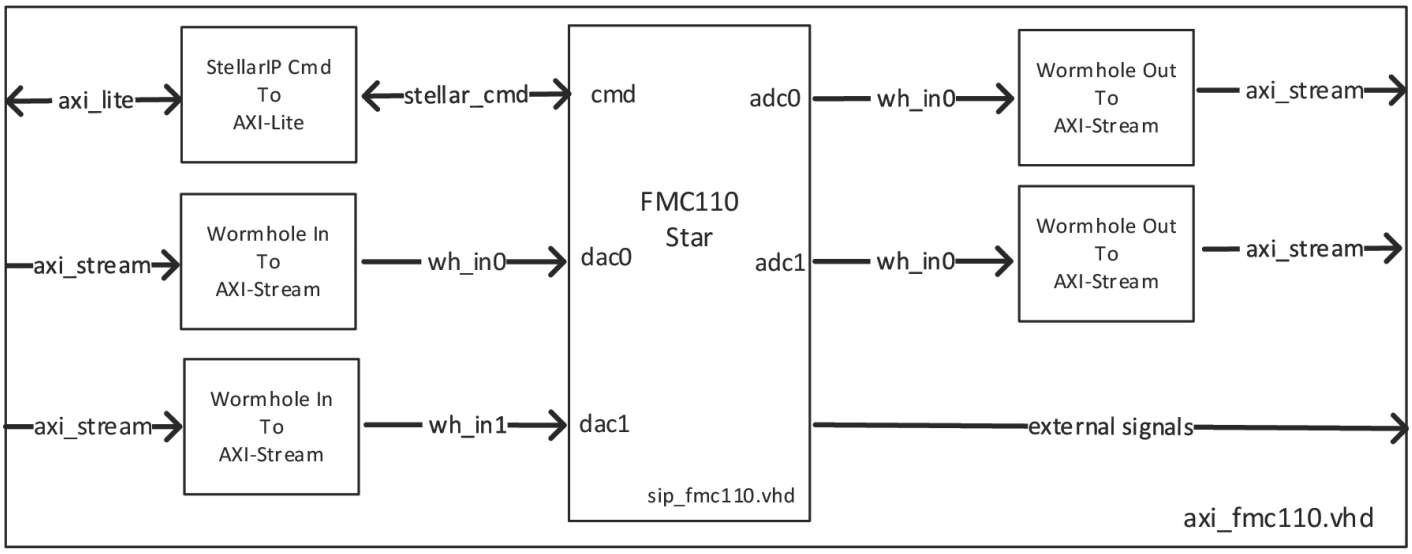

Figure 2: Block Diagram of a StellarIP Block with AXI Interfaces

The concept of StellarIP designates each core as a star (IP block). The stars communicate with one another through wormholes (channels), and a collection of stars forms a constellation (top-level entities or designs). Because StellarIP is based on proven stars that can be reliably and quickly reused, repetitive programming tasks are eliminated to minimize the introduction of errors into firmware. Xilinx AXI-compliant cores can be easily wrapped into StellarIP stars, making them ready to reuse across different designs. On the host software side, users can access the address space for any of the cores present in the design to more easily communicate with the hardware.

Without StellarIP, it is necessary to create UCF, XDC, and RTL top-level files. This can be problematic when engineers lacking the necessary expertise must learn how to write these files for their hardware. By providing a library of files and an interactive, object-oriented development environment designed to automatically connect essential IP interfaces and simplify schematic entry, StellarIP gives engineers without extensive firmware experience the ability to produce new and effective designs by selecting and interconnecting the available stars. Supporting a new carrier card in a star is as simple as creating the ball binding for it, in other words, by creating a UCF or XDC fragment. In this way, StellarIP accelerates design integration and enables users to rapidly leverage their designs as reusable IP.

In order to enable maximum performance of FPGA-based hardware, designs created with StellarIP feature ultra-fast PCIe DMA engines, advanced memory controllers, and flexible A/D and D/A interfaces. Another key benefit of StellarIP is a schematics entry tool that further simplifies the design process. Users can add their own features including DSP cores, interfaces, logic, and local memory to the reference designs included in 4DSP's Board Support Package (BSP) by connecting blocks in StellarIP's graphical workflow. This expedites the creation of complex designs for programmable devices.

The development roadmap for StellarIP features planned improvements for data routing and addressing functions. In upcoming versions of the tool, the data path will be automatically configured by the API and independent from the carrier type. The API will be able to identify the target FMC (FPGA mezzanine card) in the system and provide the appropriate support code. The reference design solution will also feature a simplified interface so users can focus on the source code of the target FMC without the need to configure firmware elements such as data routers and firmware buffering for the carrier card. Users will therefore not need to spend time learning a complete framework in order to use their FMC. This feature will further increase StellarIP's level of design workflow automation to help reduce product development times and make it easier to meet project schedule milestones with available resources.