Silaero

HYBRID-FPGA

ARCHITECTURE — DEVELOPMENT AND OPTIMIZATION

WHAT IS A HYBRID-FPGA?

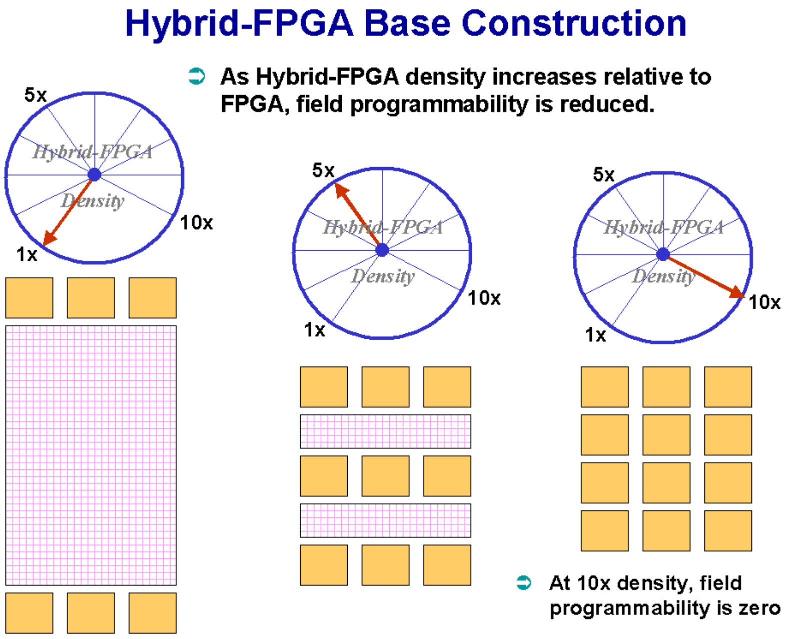
Well actually, it's an FPGA with a Hybrid Interconnect Structure. In other words, a Hybrid FPGA fabric consists of a combination of logic modules and a unique combination of interconnect. The logic modules may have either fixed function or field-programmable function, while the interconnection between the modules consists of a combination of fixed wiring and field-programmable wiring. As in conventional FPGAs this field programmable wiring consists of a combination of metal wiring tracks and field programmable switches, the most common implementation for such field-programmable switches being a transistor switch controlled by a RAM cell.

In general, a Hybrid-FPGA can be thought of as an application specific FPGA where there is some degree of field programmability and alterability, but only within the bounds of a specific customer application or industry application segment. There are different scenarios for using a hybrid FPGA:

1) Implementing a variety of programming patterns which are used for a specific FPGA in a specific application, but performing this implementation using a device having a much smaller die size.
2) Creating an application specific FPGA fabric that can perform a variety of functionalities which may be chosen field programmably, again with a smaller die size than what would result using conventional FPGA.

TARGETING A HYBRID-FPGA ARCHITECTURE

The issued patent incorporates Structured ASIC techniques for the final mask customization that targets the architecture at a particular application or industry segment. To clarify the point, before the final masks are determined for the Hybrid-FPGA, it is general purpose and can be used in any application. Once the final masks are determined, that particular device has now been targeted



Hybrid-FPGA Base Construction

- ➲ As Hybrid-FPGA density increases relative to FPGA, field programmability is reduced.
- ➲ At 10x density, field programmability is zero

at a particular application segment, and also retains a (lesser) degree of field programmability, the remaining programmability being focused within that segment to allow for changes.

The base wafer can be constructed with different degrees of field programmability as shown. Note that although the maximum density for Hybrid FPGA is approximately 10x that of conventional FPGA, the Hybrid has no field programmability at that level and is essentially equivalent to Structured ASIC. Our 5x density estimate may be conservative, but we don't want to discount the utility of having a reasonable degree of field programmability remaining.

At 10x, the Hybrid is essentially a Structured ASIC. Having invented the Structured ASIC at Lightspeed, we certainly feel it has significant value, especially given what mask costs have done in recent years. However, DSP designs usually interface with some form of real world phenomenon, and the result is often a need to change something during the integration and debug process.

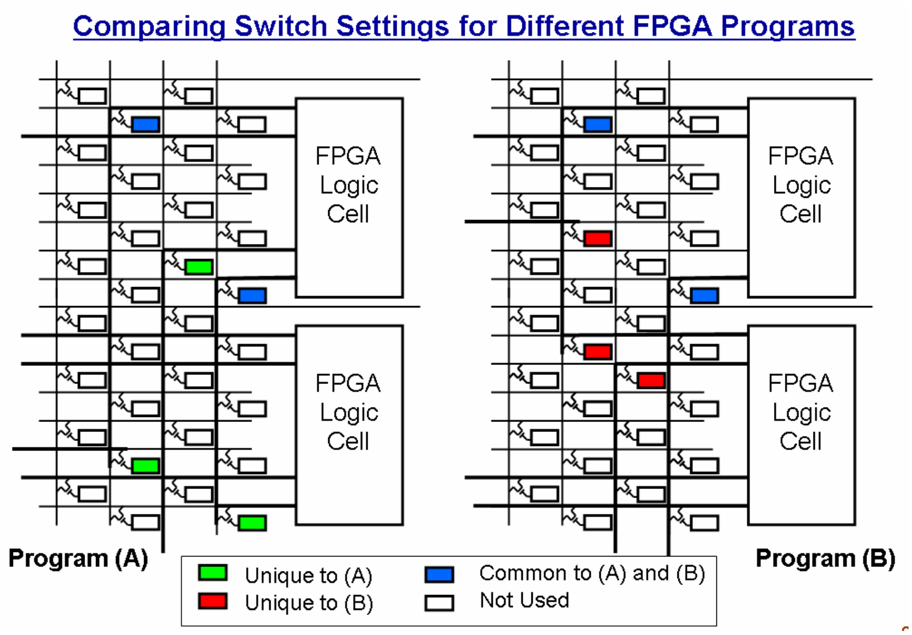
A specific Hybrid-ASIC base-wafer may be used in different known applications and also in emerging segments that have yet to be characterized. Although initially developing one base wafer focused on some initial target application, a Licensee may develop at least one additional base wafer with a different ratio of logic module area to routing switch area, as well as another with a different total amount of Hybrid-FPGA fabric, in order to support a variety of industry application segments.

If the initial base-wafer is targeted at video processing applications, for instance, it may turn out that it also functions quite well in broadband applications. The first base wafer configuration will have a specific degree of field programmability. This degree of programmability may be sufficient for, and even optimum for broadband applications. The realities of these issues will be determined during the architecture development process (see below).

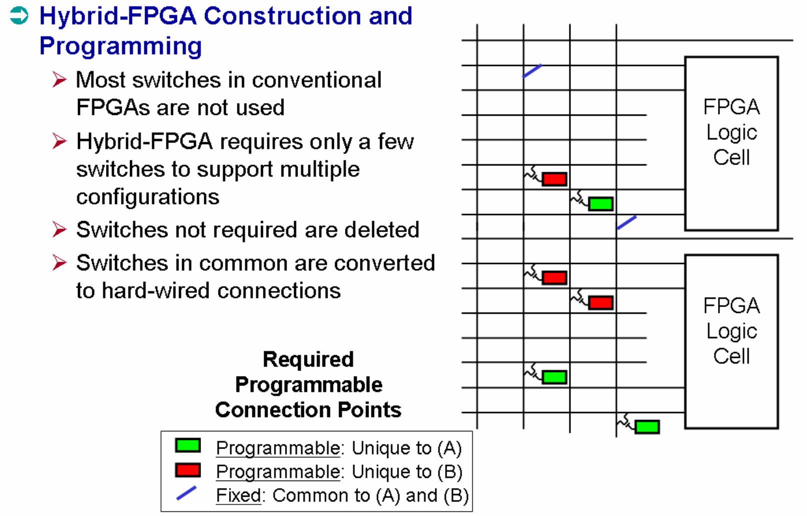
CREATING A HYBRID-FPGA

To explain the architecture and operation of a Hybrid-FPGA and how one would go about creating such a structure, it may be easiest to start with the concept of a conventional FPGA which must be re-programmed from time to time. Here, there are a number of programming patterns which must be implemented in the device. If such a device is also being used in a high-volume application, the die size for the conventional FPGA becomes a barrier to the viability of the implementation due to its high cost in volume. The Hybrid-FPGA can then be seen as a way to implement the identical functionality, including all of the known programming patterns and some unknown programming patterns, while providing a much smaller die size and therefore production cost.

So, let's start with these multiple programming patterns that the conventional FPGA must implement and examine them to look for commonalities and differences. First of all, it is common knowledge that the vast majority of interconnect switches in conventional FPGAs are never used. They are there in case they are needed for any program functionality, but since most of the possible wiring patterns are never required, the switches that enable these routing paths to be used are never turned on. If a switch is never turned on for any of the multiple programming patterns required, and also not required for a limited set of possible changes, then that switch is not necessary in the final implementation and can be deleted. If we compare the programming patterns for two possible implementations of the conventional FPGA as shown in this figure, we notice that some switches are only turned on for program A (green), some only turned on for program B (red), and a few turned on for both programs A and B (blue).


Comparing Switch Settings for Different FPGA Programs

Now, we can look at what programmable switches would be required, and how they would be programmed if a single device implementation was to be capable of implementing either program A or program B. In the next figure, switches that are not required for any programming pattern have been eliminated. Connections that were common to all programming patterns have been replaced with fixed metal wiring. Connections unique to programs A and B respectively are supported with specific field programmable switches (green and red). The result is a device architecture with an enormously smaller number of field programmable switches and RAM cells which control them, and thus a much smaller die size.

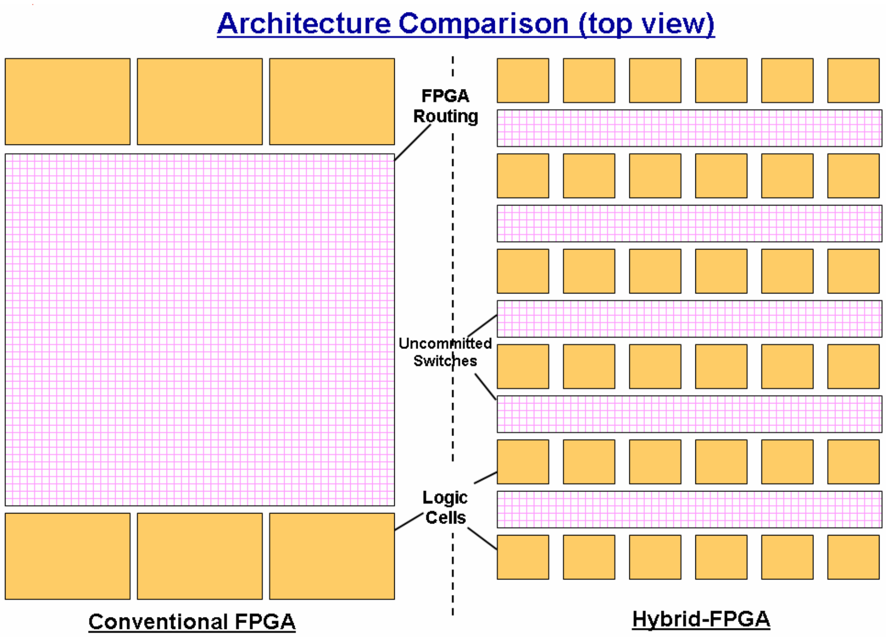

➲ Hybrid-FPGA Construction and Programming

- ➢ Most switches in conventional FPGAs are not used
- ➢ Hybrid-FPGA requires only a few switches to support multiple configurations
- ➢ Switches not required are deleted
- ➢ Switches in common are converted to hard-wired connections

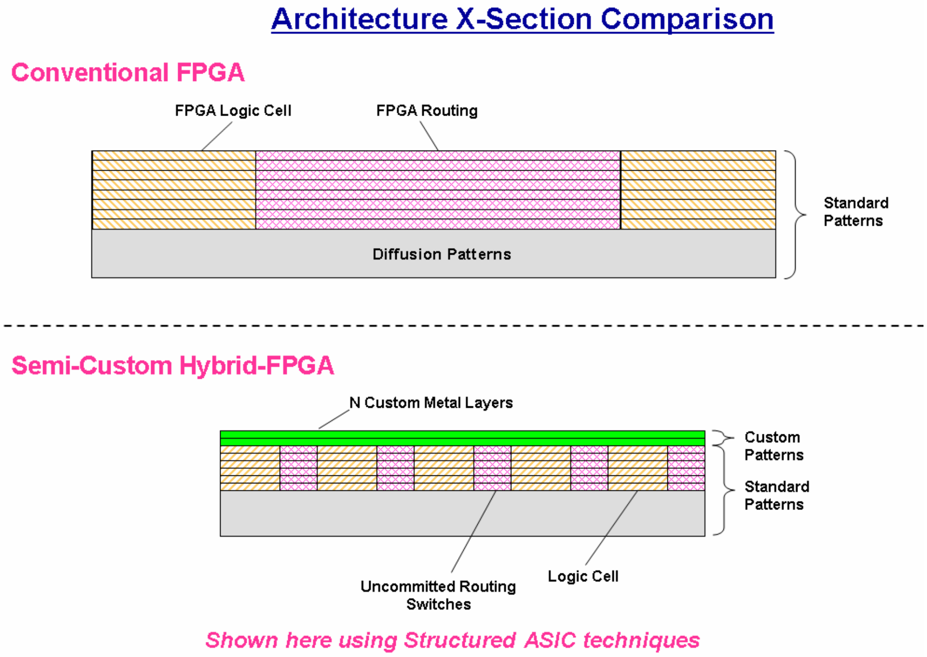
Required Programmable Connection Points
FPGA Logic Cell
FPGA Logic Cell
🟩 Programmable: Unique to (A)
🟥 Programmable: Unique to (B)
/ Fixed: Common to (A) and (B)

The physical implementation for this hybrid architecture can be constructed in more than one manner. First of all, custom silicon can be constructed using standard cell or full custom techniques, and for some applications this may be the best choice. In most applications, however, it may be more prudent to implement the hybrid FPGA architecture using Structured ASIC techniques. Structured ASIC's are configured with a relatively small number of custom metal masks, thus allowing changes to the base Hybrid-FPGA configuration to be made at relatively low cost when necessary.

Regardless of the chosen implementation, a top view comparing a Hybrid-FPGA fabric with conventional FPGA shows the main difference is the amount of switching fabric required relative to the number of logic modules. The logic modules used in the Hybrid-FPGA will often be identical to those used in a conventional FPGA, however they may also be reduced in functionality in accordance with the application segment to which the fabric is targeted.



The Structured ASIC implementation for the hybrid FPGA fabric enables a base wafer to be constructed which can be used in a variety of customer applications or even a variety of industry application segments, the fabric being tuned with only a few custom masks when necessary. In a structured ASIC implementation, the diffusion for logic modules and programmable switches is established as the die layout is planned, and it is how these resources are connected wherein the mask-based configurability lies.



Metal connections that are required for all possible designs, including metal structures required to construct the logic module and the transistor switch/RAM cell connections, are all implemented in the lower metal layers as are most of the power distribution connections.

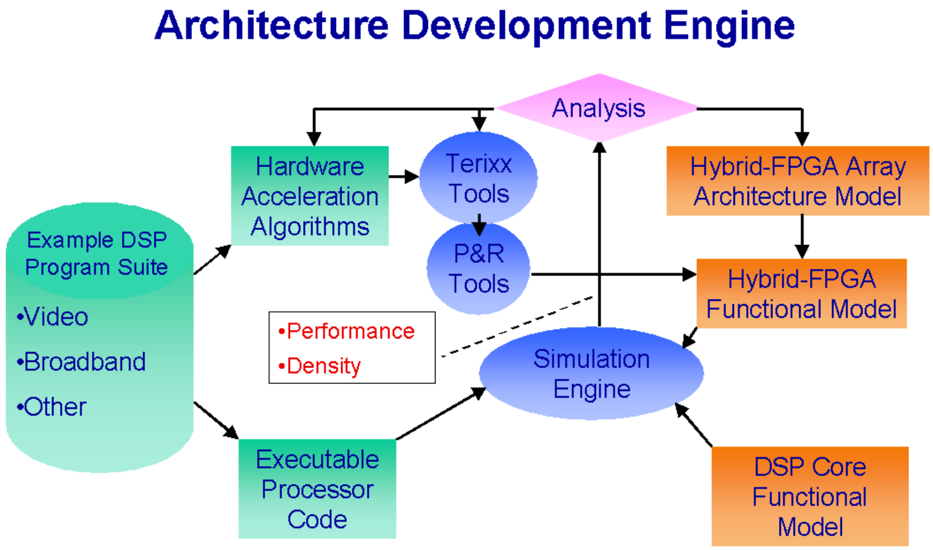
Connections which determine how programmable switches are routed, and what functional connections are implemented with hardwired metal, are left to the very top-most layers (or whichever layers are reserved in the metal stack for implementing customized functionality).

Thus, if a given implementation of the Hybrid-FPGA is not able to perform some new functionality that is desired, it is most probable that new wafers can be manufactured with changes to only a few metal masks, greatly reducing the cost as well as the turnaround time.

ARCHITECTURE DEVELOPMENT

One can think of the Architecture development process an engine where application information, architectural information and the tool flow may all be varied, with statistical information related to density and performance being extracted from the simulation results. The proper way to develop a new programmable architecture is well known, and variations on the theme shown here have been used at a number of well known FPGA and Structured ASIC companies.

In the case of the Hybrid-FPGA fabric for an rDSP device or any embedded (ASIC) application, the simulation should include a composite of both hardware simulation for functions within the Hybrid-FPGA as well as simulation of the software running on the processor including the interaction of the two.


Architecture Development Engine
Analysis
Hardware Acceleration Algorithms
Terixx Tools
P&R Tools
Hybrid-FPGA Array Architecture Model
Example DSP Program Suite
•Video
•Broadband
•Other
Hybrid-FPGA Functional Model
•Performance
•Density
Simulation Engine
Executable Processor Code
DSP Core Functional Model

OTHER HYBRID-FPGA APPLICATIONS

Hybrid-FPGA technology can be used by itself in a generic ASIC device with some reprogrammability included, or embedded in an SOC with or without various processors, including DSPs or RISC processors such as those from ARM. Hybrid-FPGAs could be used for FPGA replacement (with logic density approaching 10x FPGA) and allow some field programmability for minor changes. The Binning patent application currently pending (A/N 10/704,850) would be uniquely valuable for FPGA replacement, since conventional FPGAs are typically offered in a variety of speed grades. Initially, the Hybrid-FPGA may be most effective in vertical applications (such as video processing) where it will not require any mask customization for specific customers once the customization masks are tuned for video processing.

Numerous attempts have been made to embed FPGA in SoC ASICs, only to be deemed a failure for a variety of reasons. The primary failing has been the incompatibility of cost structures. SoC technology is typically utilized only in very high-volume cost sensitive applications while conventional FPGA technology is extremely silicon real estate consumptive. A significant reduction in real estate afforded by Hybrid-FPGA may make it the effective embedded FPGA technology of the future.