Maximizing Design Success with AMD Vivado Design Suite

WP559 (v1.1.1) February 12, 2025 **White Paper**

Abstract

The AMD Vivado™ Design Suite is a single, streamlined development tool for AMD FPGAs. A completely integrated tool suite from a single vendor provides benefits such as a shortened learning curve and unified channel for design support.

The Vivado Design Suite is a mature and robust solution, implementing both small and large designs with high efficiency. In this test, we see near 100% out-of-the-box timing closure across a suite of OpenCore designs. This saves engineering resources and shortens time to market with fewer design iterations.

The Vivado Design Suite is more than a place-and-route tool. With a robust IP catalog of more than 500 functions and IP for a wide range of applications, it is an end-to-end FPGA design suite built for developers and how they work.

Introduction

The AMD Vivado™ Design Suite is an Electronic Design Automation (EDA) tool built around three pillars that pave the way to a successful design outcome:

Get It Right the First Time: The Vivado Design Suite enables designers to achieve timing closure with out-of-the-box settings nearly 100% of the time. Additionally, the designers can reach their F_{MAX} targets using 40–60% fewer resources. This translates to fewer build iterations, minimizing designer fatigue.

Save Valuable Time and Reduce Risk: By offering a fully integrated environment, the Vivado Design Suite supports a comprehensive design flow from simulation to debug, eliminating the need for third-party tools. This seamless integration helps to reduce overall costs, save time, and minimize the risks associated with third-party tools and support.

Developer-Focused Tools for Better Results: With an extensive IP catalog encompassing over 500 functions and IP targeting a wide variety of applications, Vivado tools enable rapid innovation and accelerate time to market.

This white paper will review each of these three pillars in detail starting first with the AMD unified design flow.

AMD Unified Design Flow

The Vivado Design Suite is a streamlined development tool for our FPGAs and adaptive SoCs. It is fully integrated across the design flow, including all capabilities needed to go from RTL design to implementation and debug. The AMD Vitis Unified Software Platform, sits on top of the Vivado Design Suite, enabling processor development and deployment. A completely integrated tool suite from a single EDA vendor provides many advantages, including a shortened learning curve, and provides a unified channel for design support. Figure 1 illustrates the unified design flow.

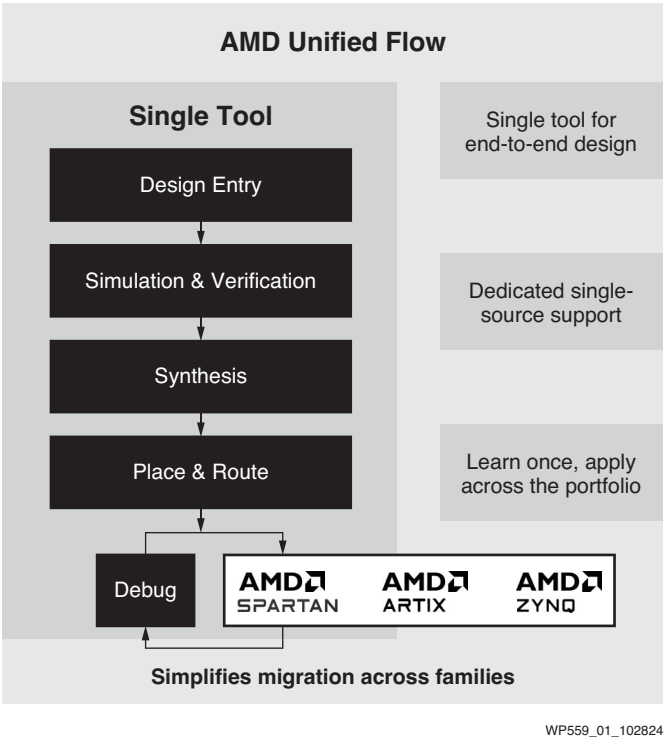

Figure 1: **AMD Vivado Unified Design Flow**

The Vivado Design Suite is fully integrated across the development flow, from design to implementation and debug:

- For design entry, you can import your RTL files or use our block-based, GUI-enabled IP integrator tool to visually connect your design.
- The Vivado synthesis tool and optimized place-and-route tools take advantage of our silicon architecture and deliver world class Quality-of-Results (QoR).
- The Vivado Design Suite has a built-in, integrated simulator that enables RTL, post synthesis, and post place and route simulation, as well as hardware co-simulation.
- Debugging the design in hardware is made possible with the ChipScope integrated logic analyzer (ILA) in the Vivado Design Suite. It can insert debug IP to validate your design and identify issues that can only be uncovered in hardware. It is effectively like having a modern logic analyzer sitting on your bench.

Competitors' Fragmented Design Flow

Smaller FPGA vendors, such as Lattice, require the use of a fragmented design flow that relies on various third-party tools for synthesis and simulation. When you are using tools from different vendors, typically 'some' effort is required to integrate these tools. Moveover, this fragmentation poses limitations to design analysis, discontinuities in technical support, and disparities in developer training. This also limits the extent to which the tools can be optimized and regression tested for silicon. See Figure 2.

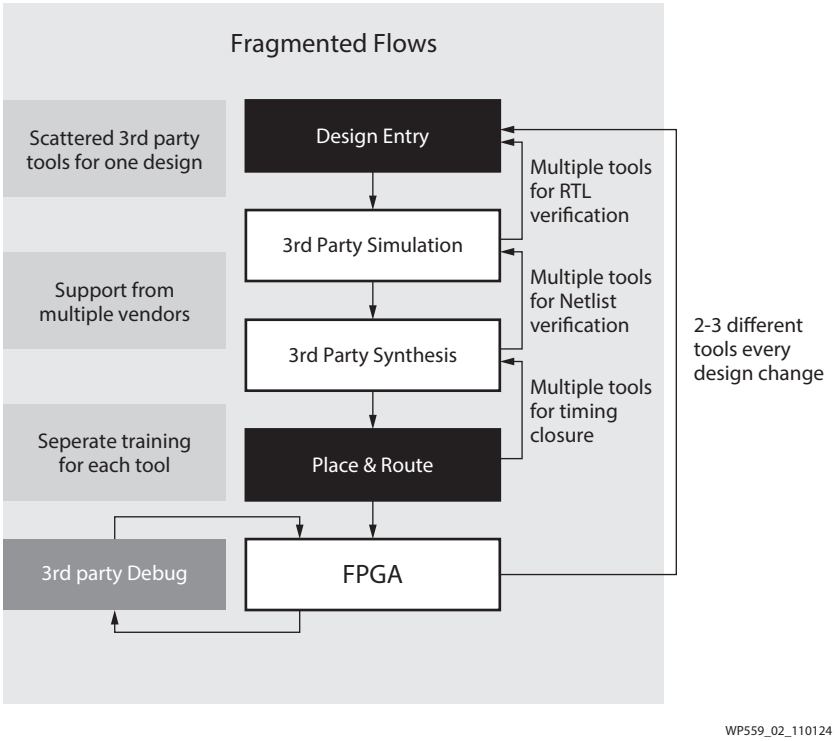

Figure 2: **Competitors' Fragmented Design Flows**

In contrast, with AMD Vivado Design Suite integration, you have an end-to-end design flow, a single point of support and training, and easy migration between different device families.

Additionally, Lattice only introduced devices above 100K logic cell densities in 2022 for their mid-range Avant FPGA platform. Effectively mastering place and route for larger density devices takes years, leaving Lattice much room for error and improvement. AMD, however, has decades of experience building synthesis and place-and-route tools for mid-range and high-density FPGAs.

Stepping beyond the inefficiencies of a fragmented tool flow, a less mature tool can also result in poorer Quality of Results (QoR). For example, the place-and-route engine developed by Lattice for the Radiant tools is relatively new and untested—especially when it comes to meeting performance requirements with optimized resource utilization. To measure QoR for these tools, AMD ran benchmarks using simple OpenCore designs, which are freely available at OpenCores.org.

Background of the Experiment

As a first step, we used 26 designs from OpenCores.org, ranging in size from a few hundred LUTs to a few thousand LUTs. These designs were compiled using AMD Vivado 2024.1 and Lattice Radiant 2024.1 with default tool settings. All benchmarks were conducted with the lowest speed grade devices (-1LV) from AMD, and the fastest speed grade from Lattice (-9 for Nexus and -3 for Avant). The rationale behind this match up is described in detail in WP558, *AMD FPGA Advantages over Competing Legacy LUT4 Architectures*. See Table 1 below for designs used in the experiment.

Table 1: **OpenCore Designs Used in Experiment**

OpenCore Designs		
oc_reed_solomon_decoder	oc_verilog_cordic_core	oc_qfp32
oc_md5_pipelined	oc_jt51	oc_product_code_iterative_decoder
oc_avr_hp	oc_usbhostslave	oc_potato_processor
oc_fpu100	oc_vgafb	oc_video_dithering
oc_wrap_tmu	oc_bluetooth	oc_graphicsaccelerator
oc_cordic	oc_trigonometric_functions_in_double_fpu	oc_wb_size_bridge
oc_bcd_adder	oc_rc4-prbs	oc_bilinear_demosaic
oc_cf_cordic	oc_cavlc	oc_can
oc_usbf	oc_pid_controller	

The same workstation was used for design compilation supporting both the AMD Vivado and the Lattice Radiant toolchains.

We first compared the place-and-route effectiveness when targeting two Lattice devices: the 28 nm Mach-XO5 family and then the 16 nm Avant-E family.

Comparison of Achieved Timing Closure for All 26 Designs

The first benchmark used an AMD Artix™ UltraScale+™ AU10P FPGA and a Lattice MachXO5-NX LFMXO5-100T device. The devices have similar logic cell counts (~96,000 logic cells), which is significantly more capacity than is required for these small designs. It is noteworthy, and important in the context of this white paper, that logic cells are not the same as LUTs. LUTs are the true representation of the logic capacity in an FPGA. Refer to WP558, *AMD FPGA Advantages over Competing Legacy LUT4 Architectures* for a comparison of LUT6 to LUT4 architectures.

Our test results show that even at a moderate target F_{MAX}, the Lattice tools had a problem closing the timing on more than half of the designs. See Figure 3.

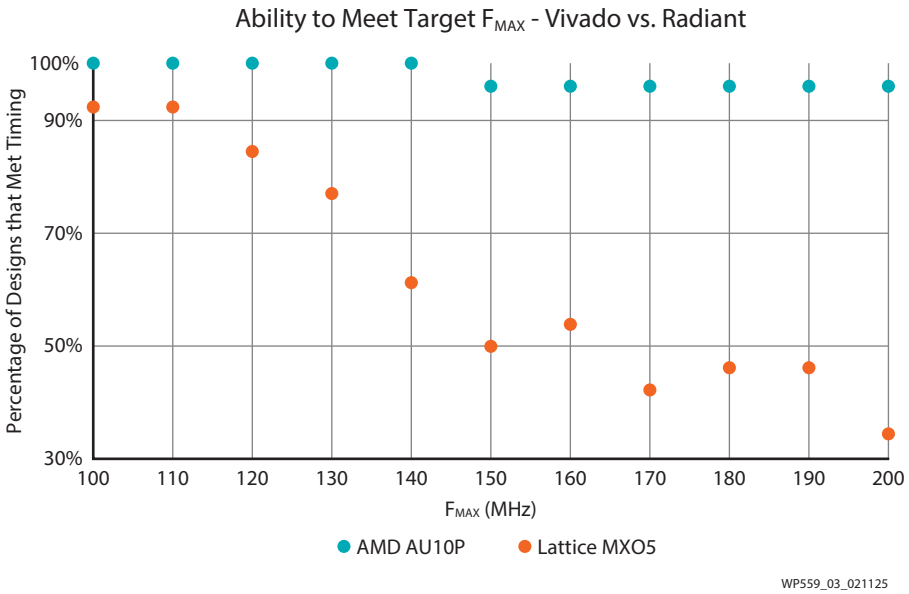

Figure 3: **Benchmark Results: AMD Artix UltraScale+ vs. Lattice MachXO5-NX Devices**

At a target performance of 200 MHz, only about one-third of designs met timing requirements when using Lattice Radiant, while 97% succeeded with the AMD Vivado tools. If your design fails to meet these requirements with Lattice, you will need to invest additional time for iterating and manually tuning the design and tool settings to close timing.

The fact that this suite of small designs, ranging from a few hundred to a few thousand LUTs, failed to meet timing in a resource-rich 96k logic cell device suggests that the root of the problem might be the quality of the place-and-route algorithms.

To understand if the Lattice place-and-route algorithms have been enhanced for newer architectures, the same design suite was benchmarked targeting Lattice's Avant-E device. This family is fabricated on a 16 nm process node, and the device targeted was the E70 with over 600k logic cells (just shy of 400k LUT4s). Note that at the time of publication of this white paper, only Avant E was available for testing in the tools. For Vivado tools, we targeted the AMD 16 nm Kintex™ UltraScale+™ FPGA family, specifically the XCKU5P with 217K LUT6s. The results are illustrated in Figure 4.

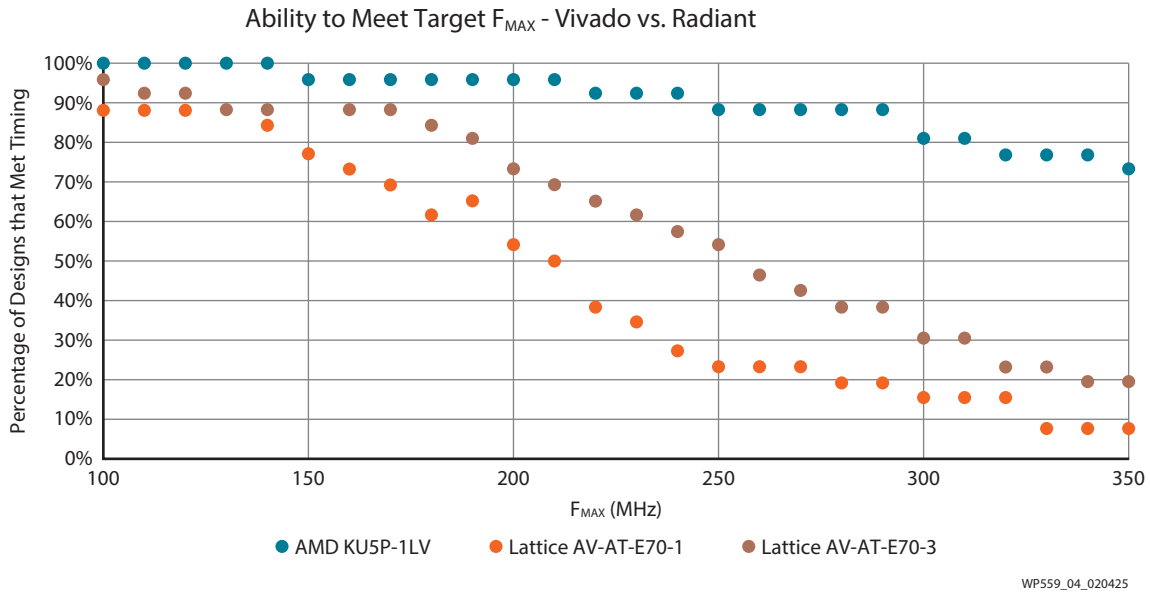

Figure 4: **Benchmark Results: 16 nm Devices**

While the place and route algorithms targeting the Avant-E family performed a little better at a 150 MHz target frequency, at 200 MHz target F_{MAX}, these tools still struggled to achieve timing closure. Lattice Avant does have an improvement in timing closure over Nexus, as expected. This is Lattice's first attempt at mid-range, introducing a new level of complexity for place and route.

Figure 4 shows the rates of successful designs across F_{MAX} targets from 100 MHz to 350 MHz for both the -1 and -3 speed grades available for Avant devices. Note that as described in WP558, *AMD FPGA Advantages over Competing Legacy LUT4 Architectures*, the slowest speed grade (-1LV) in the AMD UltraScale+ device matches up to the fastest device from Lattice (-3).

Lattice's high performance starts off well, but drops off at around 170 MHz, and continues to consistently (linearly) keep dropping, coming down to 30% success rate at 300 MHz. At the same frequency, AMD holds strong at 90%. Given that these are mid-range devices, the performance expectation is higher than that of low-end devices. AMD continues to offer consistent push-button timing across densities.

The low-performance speed grade from Lattice starts off poorer, and behaves considerably poorer throughout—peaking at just short of 90% success rate, and dropping off at a mere 130 MHz. For a mid-range application design, this translates to considerably more iterations and frustration to achieve the same result.

The overall conclusion of this performance study is that Lattice tools in conjunction with their devices are only relevant for designs with a target performance of 150 MHz and below. Anything higher than that will require a designer to manually tweak the designs to achieve timing closure. Also remember these are small designs—push-button timing closure will likely be more problematic for larger designs.

Comparison of Device Utilization (LUTs) When Closing Timing at 100 MHz

With a large percentage of OpenCore designs failing at 150 MHz and beyond with the Lattice Radiant tools, no attempts were made to analyze tool efficiency in terms of device utilization at higher frequencies. Thus, this analysis was done with the designs that met timing at 100 MHz with the Lattice tools.

To analyze resource utilization, we targeted the Lattice MachXO5-NX devices (LFMXO5-100T) and compared the results to the Artix UltraScale+ AU10P device. The results, shown below in Figure 5, illustrate that for the MachXO5 family of devices, Lattice synthesis and mapping algorithms are very inefficient in terms of device resources used to implement even simple, small designs.

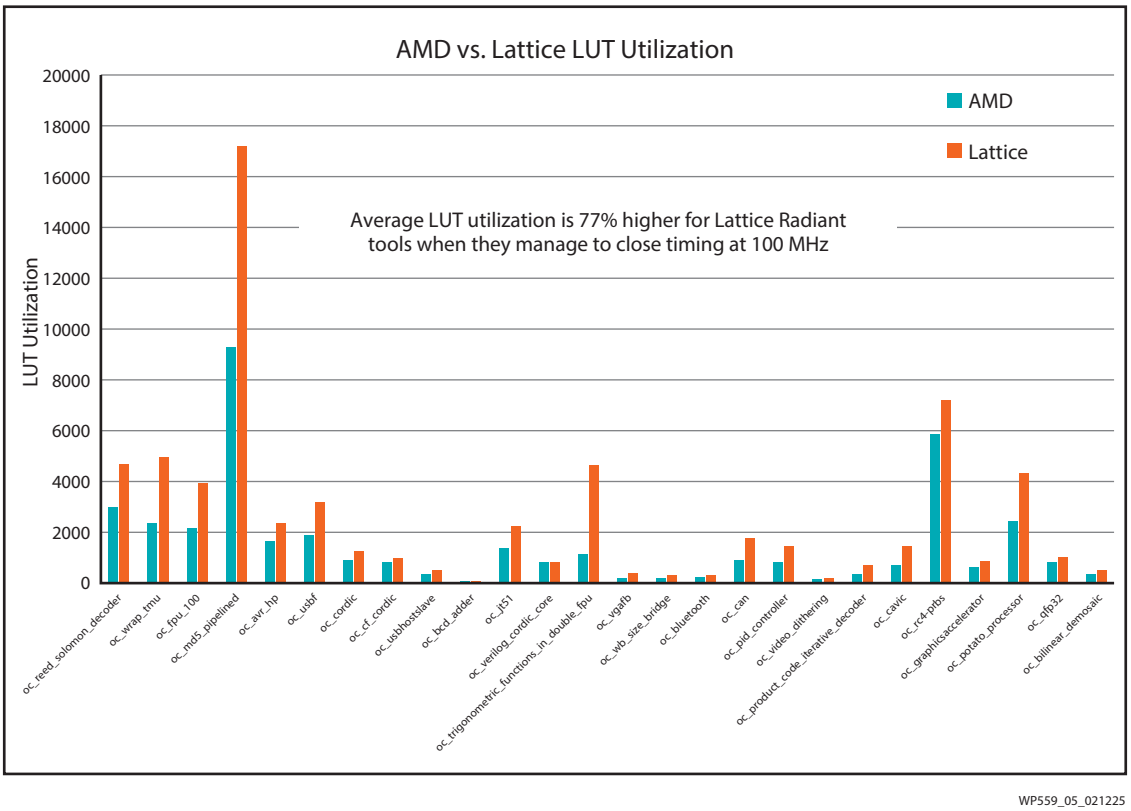

Figure 5: **Average LUT Utilization at 100 MHz for AMD Artix UltraScale+ AU10P vs. Lattice MachXO5-NX (LFMXO5-100T)**

To investigate whether there were utilization improvements in the synthesis and mapping algorithms for newer Lattice Avant-E devices, we also compared the 16 nm Avant-E to the 16 nm AMD Kintex UltraScale+ family. The results with the associated OpenCore designs are shown in Figure 6:

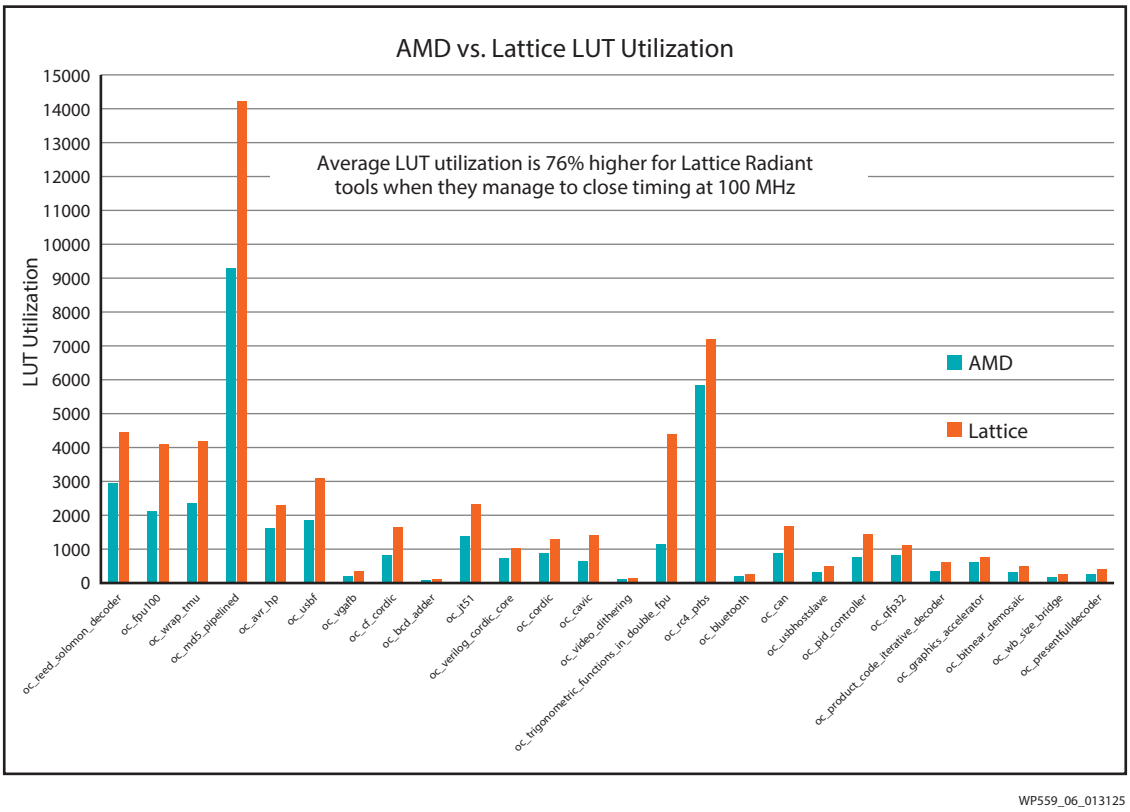

Figure 6: **Average LUT Utilization at 100 MHz for AMD Kintex UltraScale+ KU5P vs. Lattice Avant-E E70**

On average, the Lattice tools require 76% more LUTs to achieve the same target F_{MAX}, even for the newer Lattice Avant-E family. This suggests that the Lattice tools will need considerable improvement to achieve results comparable with the AMD Vivado Design Suite. The task of co-optimizing synthesis, mapping, and place-and-route algorithms to achieve optimum results is a multi-year journey.

Optimizing QOR might require hand tweaking. However, it is surprising that such optimization measures would be required for such small, simple designs in Lattice's case. For larger designs—exceeding 100k LUTs or more—it seems likely that the utilization with Lattice would be significantly higher than with AMD FPGAs. In some cases, this might necessitate the selection of a larger, more expensive Lattice FPGA for a given design, resulting in higher static power consumption and potentially more PCB real estate, and most probably a less cost-effective design.

Another consideration is that it is both possible and desirable to increase the design clock rate while applying techniques such as time division multiplexing to reduce fabric utilization. If you can double the clock rate of a datapath or accelerator, you may simultaneously reduce the logic utilization by as much as 50%. These techniques can be applied to a variety of circuit topologies, and potentially impact not just logic utilization, but also reduce internal RAM and DSP utilization. Thus, a push-button flow that achieves a higher F_{MAX} can have advantages on product size, weight, power, and cost. In simple terms, leveraging fewer FPGA resources clocked at a higher frequency can reduce system-level power consumption and, in

some cases, will facilitate the selection of a smaller, lower-cost target FPGA. The AMD Vivado toolchain offers clear advantages as enumerated above.

Key Takeaways:

- Lattice design tools are not capable of consistently achieving timing closure at an F_{MAX} of 150 MHz or above, across a variety of OpenCore designs.
- If your design has a lower performance requirement, e.g., 100 MHz or below, the Lattice tools will require 70% more LUTs than an AMD FPGA to meet that target.
- These results suggest that for larger designs, Lattice development tools will struggle to meet moderate F_{MAX} targets. And even when by chance these targets are met, it will be at the expense of significant LUT usage.

FPGA Tools Feature Comparison

While the AMD Vivado and Lattice Radiant toolchains superficially appear to enable a similar compilation flow, the Vivado Design Suite supports a much broader set of features that accelerate design, timing closure, and debug. AMD designed synthesis and simulation flows are co-developed alongside our implementation workflow, enabling seamless integration. Due to this level of integration, we can drive development of advanced features such as out-of-context synthesis, advanced floor planning, timing closure, team design (collaborative tool), Dynamic Function eXchange (partial reconfiguration), and much more.

The Vivado Design Suite provides advanced synthesis timing closure features such as machine learning and rules-based predictive guidance for design closure. Also, the Vivado tool's GUI enables powerful time-saving features, e.g., the ability to customize the interface with multiple views. Users can simultaneously inspect elements including RTL source code, report files, floorplans, routing paths, and package views. In addition, extensive cross-probing, enabled by the Vivado tools' underlying unified database, allows the user to highlight and inspect specific components of the design across all views. And if you are not a GUI user, then the entire workflow, from synthesis through simulation, implementation, and debug, is fully scriptable, leveraging TCL.

In addition to conventional RTL workflows, the Vivado toolchain, in combination with the AMD Vitis™ unified software platform, provides advanced workflows for embedded design, signal processing, and application development, including:

- Vitis Embedded: For developing C/C++ application code running on integrated Arm® processors and AMD MicroBlaze™ soft processors
- Vitis HLS: For developing C/C++ based IP blocks that target FPGA fabric
- Vitis Model Composer: A model-based design tool that enables rapid design exploration within the MathWorks Simulink® environment
- A set of open-source, performance-optimized library functions, such as DSP, Vision, Solver, Ultrasound, BLAS, and many more, that can be implemented in FPGA fabric

Overall, AMD tool solutions for FPGA and SoC design offer the designer with a wide range of approaches, allowing the customer to tailor and customize their workflow to cater to a variety of developers who might have specialized expertise, such as application development or DSP.

See the tables below for a detailed side-by-side feature comparison between AMD Vivado and Lattice Radiant design tools:

Table 2: **Simulation**

Feature / Function	Radiant	Vivado	Vivado Design Suite Advantage
Integrated simulator	-	✓	A fully integrated AMD simulator is provided at no additional cost to the user.
Third-party simulator	✓	(Optional)	
Logic analyzer for debugging	✓	✓	

Table 3: **Synthesis**

Feature / Function	Radiant	Vivado	Vivado Design Suite Advantage
Opt design	-	✓	Post synthesis opt_design optimizes a design netlist for the target part. Optimization provides improvements to synthesized netlists from third-party tools or for netlists that are not optimized during synthesis. Run this command after synthesis but prior to implementation to optimize the design and simplify the netlist before placing and routing the design.
Power opt design	✓	✓	
Synth Timing Analysis			Retiming provides an option improve circuit performance for intra-clock sequential paths by automatically moving registers (register balancing) across combinatorial gates or LUTs. It maintains the original behavior and latency of the circuit and does not require changes to the RTL sources. Logic replication can be used to replicate drivers with high fan out. This allows the driver to be placed closer to the downstream loads and eases timing closure. Auto pipelining enables the user to optionally insert additional pipeline registers during placement to address timing closure challenges on specific and custom buses and interfaces.
• Timing closure	✓	✓	
• Retiming	-	✓	
• Pipelining	-	✓	
• Logic replication	-	✓	
Out-of-context	-	✓	Any HDL object can be synthesized out-of-context. This enables various advantages including modular or bottom-up design flows. OOC synthesis results can be re-used run to run, which results in a significant runtime improvement for the top-level design.

Table 4: **Implementation**

Feature / Function	Radiant	Vivado	Vivado Advantage
Mapping			
• Map design	✓	✓	
• Map timing analysis	✓	✓	
Place			Power optimization is an optional step that optimizes dynamic power using clock gating. It can be used in both Project Mode and Non-Project Mode and can be run after logic optimization or after placement to reduce power demand in the design. Power optimization includes AMD intelligent clock gating solutions that can reduce dynamic power in your design, without altering functionality.
• Place design	✓	✓	
• Power opt	-	✓	
• Phys opt	✓	✓	
Route			Physical optimization performs timing-driven optimization on the negative-slack paths of a design. Physical optimization has two modes of operation: post-place and post-route. In post-place mode, optimization occurs on timing estimates based on cell placement. Physical optimization automatically incorporates netlist changes due to logic optimizations and places cells as needed. In post-route mode, optimization occurs based on actual routing delays. In addition to automatically updating the netlist on logic changes and placing cells, physical optimization also automatically updates routing as needed.
• Route design	✓	✓	
• Post-route Phys opt	-	✓	

Table 5: **Additional Features**

Feature / Function	Radiant	Vivado	Vivado Advantage
Timing analysis			
• STA	✓	✓	
Bitstream generation	✓	✓	
Programming	✓	✓	
Hardware Verification	✓	✓	
IP	✓	✓	
HLS	-	✓	Vitis High-Level Synthesis accelerates design implementation by enabling C, C++, and System C specifications to be directly targeted into AMD devices without the need to manually create RTL.
Cross-probing from reports	-	✓	Cross-probing provides the user with dynamic hyperlinks that can be used to directly view synthesized netlist paths, RTL, or physical placement. Within the Vivado environment, it is possible to cross-probe directly from user reports (e.g., a timing report) to the specific elements of the design pointed to by these reports. For instance, users can cross-probe between RTL, netlist, and physical implementation to better understand the Vivado tools' handling of various design elements.
Vectorless estimation	-	✓	Vectorless power estimation enables early power estimation without the need to create a top-level testbench for the design.
Design analysis and QoR assessment	-	✓	Timing closure can be a significant challenge for even the most experienced FPGA developer. Vivado provides advanced features that leverage a combination of rule-based and machine learning predictions to guide the user on the most appropriate steps and strategies to close timing.

The AMD power estimation tools, XPE/PDM, offer enhanced accuracy by providing users with greater control over routing parameters compared to Lattice Radiant. This includes aspects such as routing complexity, GT protocols, and additional factors that influence power estimates. Design engineers want an accurate estimate, and more control helps to ensure better accuracy. Real-world designs will likely need the option of tuning default parameters to get a more accurate power estimation for real-life applications. You can see the full feature-to-feature breakdown for AMD vs. Lattice power estimations tools in Table 6.

Table 6: **Power Estimation Tool Comparison**

	Lattice PCF Features/Defaults	AMD XPE Features/Defaults
Summary	Junction Temperature is controlled by changing ambient temperature and increases with higher utilization. There are cases where the max safe ambient temperature goes to negative scale.	Junction temperature can be set manually.
Logic	Default Activity Factor - 10% for NX, 12.5% for Avant. No control on routing complexity of a logic cell but has an enabling factor.	Default Toggle Rate - 12.5% Additional settings like routing complexity are available to get more precise power numbers.
Block RAM	Similar features	
DSP	Fewer modes/configurations available for DSP. User must have additional know-how to correctly populate data. Ex: To use a MULT18, the user must specify 1 MULT18 and 2x MULT9s. This was confirmed using simple multiplication Verilog code.	Similar features. It has more DSP configurations. Resource entry is straight forward.
GT/DDR	Cannot configure GT protocols from the tool. DDR PHY/delay blocks and I/Os used should be manually filled or exported from the Radiant tool.	Instantiate various protocols using IP manager.
Post Place & Route Power tools	All factors are set to default in power tools. The tool offers no user control, and the routing factors are unknown.	Vectorless estimation values are set automatically in XPE - this includes toggle rate, clock, and logic routing factors.
Power Delivery Design	N/A	PDM 2024.2 will have power delivery design support and consolidate all rails where possible as well as decoupling requirements based on the user design.

A Broad IP Offering Enables Faster Time to Market

Depending on your application, you might require additional steps and IP to complete your design. AMD understands the importance of the whole solution, emphasized by our continued investment in soft IP to help our customers get to market quickly. Whether you are looking for basic infrastructure IP or market-specific functions, like motor control or vision libraries, AMD offers over 250 free AMD Vivado IP cores, over 700 Vitis library functions, and free soft-core processors MicroBlaze and MicroBlaze V. Everything you need to get your project done quickly is provided in a single, unified download.

With Lattice, you are required to select from two distinct tool flow options, third-party synthesis, and simulation tools, along with IP delivery that involves downloading six separate software solution stacks designed for various end applications. This results in numerous downloads and comes with a notable lack of cross-functionality between different use cases. For instance, if you are targeting a camera-related application, you might need to:

1. Pay for a Lattice Radiant license, download the program
2. Pay for Synplify Pro & ModelSim, separate downloads
3. Download mVision, the Lattice vision-related software solution stack, to access ONLY vision-related IP libraries
4. Pay a per-device royalty to instantiate the Lattice MIPI IP in the design

If you chose an AMD Artix UltraScale+ device, you need to:

1. Download the free AMD Vivado Standard Edition

Clearly, the AMD approach is elegantly simple by comparison.

Conclusion

The AMD Vivado Design Suite is a significantly more sophisticated solution than the Lattice Radiant tool. This ensures maximum productivity and minimal design iterations for customers. The Vivado Design Suite delivers a higher success rate, meeting F_{MAX} targets with 40–60% fewer resources, minimizing design iterations and costly delays. The Vivado Design Suite provides a fully integrated solution from simulation to debug. Eliminating the need for third-party tools helps to reduce development friction while also minimizing tool expenditures. In addition, all IP libraries are available in a single download. This contrasts with the fragmented IP offering from Lattice—across six different software solution stacks—which requires the purchase of baseline IP such as MIPI. Finally, the AMD advanced power estimation tools enable the developer to tune design parameters, thereby ensuring an accurate power estimation. In summary, the selection of AMD for your next design will help to pave the way to a high-quality, proven path to design success.

Finding Additional Documentation

Technical Information Portal

The AMD Technical Information Portal is an online tool that provides robust search and navigation for documentation using your web browser. To access the Technical Information Portal, go to https://docs.amd.com.

Documentation Navigator

Documentation Navigator (DocNav) is an installed tool that provides access to AMD Adaptive Computing documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the AMD Vivado™ IDE, select **Help > Documentation and Tutorials**.
- On Windows, click the **Start** button and select **Xilinx Design Tools > DocNav**.
- At the Linux command prompt, enter `docnav`.

Note: For more information on DocNav, refer to the *Documentation Navigator User Guide* (UG968).

Design Hubs

AMD Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- Go to the Design Hubs web page.

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
02/12/2025 Version 1.1.1	
Figure 5	Typographical edit.
02/11/2025 Version 1.1	
Abstract	Updated.
Background of the Experiment	Text and Table 1.
Comparison of Achieved Timing Closure for All 26 Designs	Text and Figure 3, Figure 4.
11/21/2024 Version 1.0.1	
Typographical edit.	
11/20/2024 Version 1.0	
Initial AMD release.	

Please Read: Important Legal Notices