

Cynthesizer™
The Industry Standard for SystemC Synthesis

The two wildly different reasons we bought Arithmatica

Brett Cline
Vice President, Marketing & Sales
Forte Design Systems

April 2010

Last year we bought Arithmatica. For those that don't know, Arithmatica was a provider of optimization tools, IP, and services for datapath design. The team was mostly made up of hardcore mathematicians with datapath design expertise. The company was best known for the use of their CellMath Designer product by customers like Imagination Technologies, a leading multimedia technologies company catering primarily to the mobile device market, and for the licensing of their CellMath IP to companies like Xilinx for use in the DSP-48 slice of their popular FPGA devices.

Now find out why we did it.

The Not-so-obvious Reason - "RTL datapath optimization? Isn't that like Module Compiler™?"

Hey, wow, there's a datapath design tool here for Verilog designers! CellMath Designer is an RTL datapath synthesis and optimization tool. Every year we talk to hardware designers around the world that are using RTL and their job is to make better hardware by getting more performance, area savings, and power savings out of the existing RTL designs.

CellMath Designer fits easily into the Verilog flow right before logic synthesis. It optimizes datapath elements in the RTL code by applying patented mathematical techniques and by taking advantage of built-in CellMath IP. The CellMath IP works especially well for designs that need to do floating point computations. Common applications are 3D graphics (floating point), high-speed communications/networking, and any other datapath design that can benefit from better performance, smaller area, and lower power.

CellMath Designer takes advantage of the existing ASIC flow by reading the same Verilog code and the same technology ".lib" file as your current logic synthesis tool. You control it using a design constraints script to optimize and trade off area, power, and performance using existing RTL code prior to running your existing logic synthesis tool.

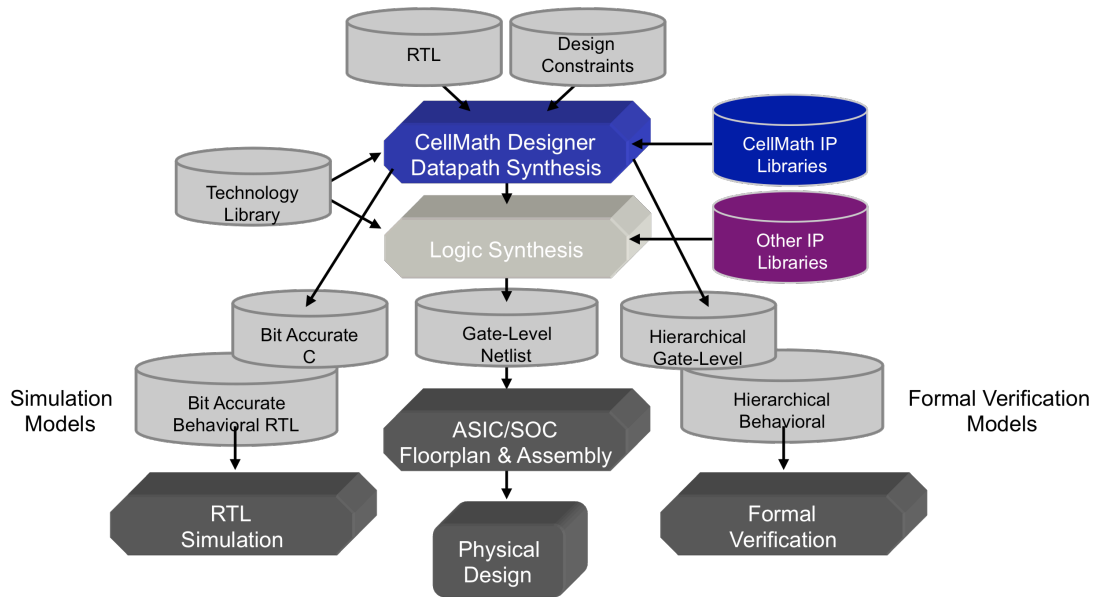


Figure 1 - CellMath Designer as an optimizer in the design flow

Formal Verification

One of the great things about CellMath Designer is that it is “formal verification aware”. Formal equivalence checking of complex datapath components is typically quite difficult. In order to make it possible for the current generation of formal equivalence checking tools to successfully prove equivalence between RTL and gates for these circuits, CellMath Designer produces intermediate hierarchical models after each optimization step. This allows an incremental approach to formal verification, making an easier task of this essential step that is difficult or impossible if you are implementing datapath designs using logic synthesis alone. It even works with common IP libraries.

Results

Obviously the main point of putting a datapath optimization tool in your design flow is to improve the quality of the circuits you produce! On one recent design just using CellMath Designer in the flow reduced area by 11%. At the same time, it reduced power consumption by 8%. Other sample results on industrial examples are shown below.

Application: Wireless LAN		Area is in (um2)				
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Slack	Area	Slack	Area Reduction	% Area Reduction
Block 1	1,443,572	Met	1,334,502	Met	109,070	7.56%

Application: 10GBASE-T PHY		Area is in (um2), Power in (fW)				
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Power	Area	Power	%Area Reduction	% Power Reduction
FFT (for area)	447,013	42,878,912	415,145	42,878,912	7%	3%
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Power	Area	Power	%Area Reduction	% Power Reduction
FFT (for power)	447,013	42,878,912	416,036	25,720,125	7%	40%

Application: Broadband		Area is in (um2)				
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Slack	Area	Slack	Area Reduction	% Area Reduction
Design 1	22,606	Met	17,550	Met	5,056	22.3%
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Slack	Area	Slack	Area Reduction	% Area Reduction
Design 2	28,837	Met	27,493	Met	1,344	4.7%
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Slack	Area	Slack	Area Reduction	% Area Reduction
Design 3	31,349	Met	28,081	Met	3,268	10.5%
	Logic synthesis alone		CellMath Designer Enhanced QoR			
	Area	Slack	Area	Slack	Area Reduction	% Area Reduction
Design 4	13,615	Met	12,347	Met	1,268	9.3%

Figure 2 - Sample QoR improvements with CellMath Designer

As you can see, CellMath Designer can optimize existing RTL designs improving area, power, and speed. And it works without requiring changes to your existing RTL code. As an added bonus, design teams that deploy CellMath Designer for optimizing existing RTL designs will be able to take advantage of these optimized parts in a high-level synthesis flow when they're ready to take that step.

The Obvious Reason – “This stuff fits right into our high-level synthesis product and produces more awesome results than ever!”

For years the conventional wisdom has been to use high-level synthesis on datapath (or algorithmic) designs. After all, algorithms are often written in C and they are very datapath oriented. This meant, in the early days, that HLS was used mostly by systems companies building consumer electronics devices such as HDTVs, DVD players, CD players, digital still cameras, digital video cameras, digital printers and copiers, etc.

As you can imagine, these designs are filled with adders, multipliers and more complex datapath-oriented blocks so producing good, optimized datapath parts has been a focus for us for a long time. We first introduced this idea of embedded datapath optimization as a part of HLS in 2002 and

received a patent for “data path synthesis apparatus and method for optimizing a behavioral design description being processed by a behavioral synthesis tool” in 2007.

Cynthesizer finds (and/or the user specifies) sections of the design that can benefit by implementation as a specialized part using a datapath optimizer. It uses the datapath optimizer “under the hood” to create customized datapath parts for these sections in gate-level and RTL form. Cynthesizer uses the detailed area, propagation delay, and latency of these parts (targeted for a specific technology library and clock speed) to schedule the design, construct the datapath for the design and to build the finite state machine. Now, by incorporating the CellMath Designer technology we acquired from Arithmatica, Cynthesizer users get the advantage of the advanced mathematical techniques that CellMath Designer implements to create better datapath parts than ever. While this improves all kinds of designs, it really makes a big difference on some of the big datapath-dominated designs where Cynthesizer has been used.

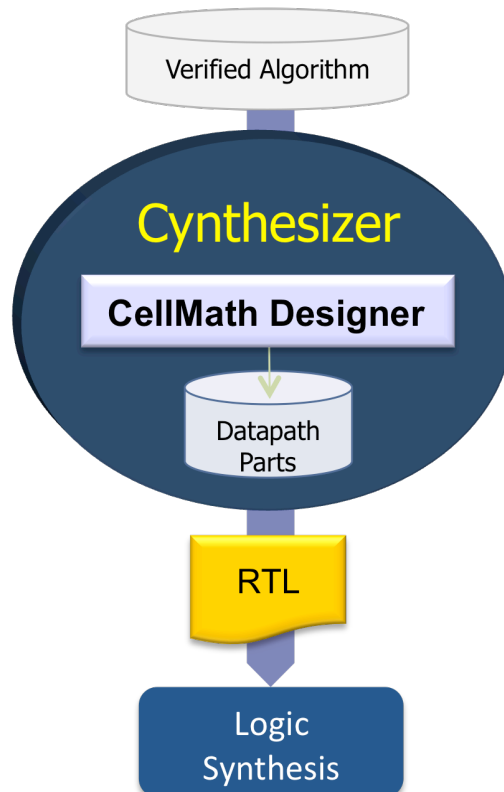


Figure 3 - Cynthesizer with CellMath Designer Datapath Synthesis and Optimization

Using HLS with embedded datapath optimization provides significant benefits to design teams including:

1) Quality of results

The main concern that designers have at first is whether they will be able to meet their tough power, performance, and area targets using high-level synthesis. They recognize that no amount of productivity and verification performance gains will really help them if they can't achieve the needed quality of results (QoR). Basically they're skeptical that they can get better results with an HLS tool than by hand coding RTL. Also, I keep seeing reports that other HLS tools give the productivity benefits but force users to sacrifice QoR. This is simply not true for Cynthesizer. Designers should expect their results using a leading high-level synthesis tool (like Cynthesizer) to be better than their hand coded RTL results. Of course, achieving these results requires you to apply your engineering skill to drive the tool to the results that are optimal for your design goals. You can get a first version of working RTL out of the tool in a few days and using HLS allows you to get through the optimization process far more quickly than using RTL.

In HLS, the high-level view of the methodology is:

- Create a working high-level model
- Verify the high-level model
- Synthesize to RTL
- Optimize the high-level model
- Validate the RTL
- Repeat until satisfied with the results

In reality, most of the time is spent in the optimization phase where you are changing options to the tool, coding styles for architecture, etc. This is the heart of hardware design. And this is where datapath optimization (and CellMath Designer) comes in to give you the best building blocks to do the job.

Embedding a synthesis tool inside of Cynthesizer allows it to make better decisions by improving the accuracy of its estimates of what the hardware will look like in terms of area, performance, and power -- ultimately leading to better designs.

Given enough time, expert designers can find near optimal implementations to meet their targets, especially for small designs where they can keep all the details in their heads. However, as designs grow beyond 50 million gates and delivery schedules get shorter, it becomes impossible for RTL design teams to keep up without adding a prohibitive number of new engineers to the team. Massive productivity improvements are required to break this bottleneck. With larger designs the state-of-the-art HLS tools can manage complexity far better than an individual designer. Optimization techniques like register and functional unit sharing are performed across a much larger portion of the design. This finds optimization opportunities that would never even be apparent to the RTL coder.

2) Design and debug productivity

As everyone would guess, if you write less code, you get fewer errors and it takes a lot less time to implement. This has always been the most widely understood argument for high-level synthesis. The good news is that it is true. Generally speaking, Cynthesizer users get a 2-10x productivity improvement over RTL design with an average around 6x. This means that designers are achieving between 300k and 2M verified gates per engineer year with Cynthesizer with multimillion gate designs.

This productivity benefit largely comes through abstraction. By eliminating many of the mundane tasks in the hardware design process, engineers can focus on the fun (and value-added) parts of design. Overall, Cynthesizer makes design fun again.

3) Verification performance

High-level design models run faster than RTL, it's that simple. The main reason is that there are fewer details in the high-level model than there are in the RTL model. Simulation speeds of 10-100x are not unheard of and can increase further using transaction-level modeling (TLM). Also, high-level design models are generally available much earlier in the design process. With code reductions of 10-25x, the model is simply available earlier in the design process for the verification team to work with.

Forte also uniquely provides multiple ways to test the high-level model. First, using a TLM interface, the designer can test the block functionality without slowing the simulation by executing every pin on the interface. This means a lot of traffic can flow through the model and thoroughly test the basic functions.

Second, the designer can add fully pin-cycle accurate interfaces to the model and test the interaction between blocks with their correct protocol timing. And, since the RTL is guaranteed to have the same interface after synthesis, the designer can now test the interfaces with the behavioral functions at a reasonably high speed (much faster than RTL).

If it were not for this second step, designers would find themselves with a huge verification headache once reaching RTL. *Specifically, by relying on the synthesis tool to add interfaces during the synthesis process the designer has no way to accurately test those interfaces before reaching RTL.* Now the RTL verification task involves testing the interfaces and block interaction while running slow RTL simulations. The result: a decrease in verification productivity. Amazingly many ANSI-C-based and even newer SystemC-based tools on the market have this flaw.

Summary

We acquired Arithmatica primarily because we could see that embedding CellMath Designer within Cynthesizer would let us further improve the quality of results users could get using Cynthesizer. We have seen an average area improvement (decrease) of 8% along with lower power across the thousands of designs we maintain in our test suite, many of which are representative designs provided by our users over the last 10 years.

We also knew that CellMath Designer would help RTL design teams get better quality circuits but we didn't realize exactly how popular it would be. In the short amount of time since we bought the products, we've been able to work with companies around the world and quickly use CellMath Designer to give them better results.

Maybe these are wildly different reasons but to me they fit together quite well. Designers want to build great designs – whether it is in Verilog or SystemC. With CellMath Designer, Forte can now address a much wider selection of designs with the next generation of design synthesis.

Forte Design Systems
100 Century Center Court, Suite 302
San Jose, CA 95112
info@ForteDS.com
www.ForteDS.com

For more information, please feel free to contact me at:
brett <at> ForteDS . com

*All trademarks and registered trademarks are the property of their
respective owners*