Benefits of Smart High-Level Synthesis for FPGA Design

Introduction

As FPGA designs continue to get larger and more complex, engineers need to improve their productivity to meet tight design schedules. This white paper explains how engineers can speed up their FPGA development by using the SmartHLS tool to generate their hardware blocks from the C++ software.

Smart High-Level Synthesis (SmartHLS™) provides an integrated development environment tool that enables engineers to compile the C/C++ software into Verilog targeting a Microchip® FPGA device, improving productivity, and time-to-market. For more information, visit the SmartHLS webpage.

High-Level Synthesis Design Flow for FPGA

1. In the high-level synthesis design flow, the engineer implements the design in C++ software and verifies the functionality with software tests.
2. Next, they specify a top-level C++ function, which SmartHLS will compile into an equivalent Verilog hardware module. SmartHLS can run co-simulation to verify whether the hardware module behaviour matches the software. SmartHLS uses Libero SoC to generate the post-layout timing and resource reports for the Verilog module.
3. Finally, SmartHLS generates a SmartDesign IP component that the engineer can instantiate into their SmartDesign system in Libero SoC.

The following figure shows the high-level synthesis FPGA design flow for targeting a Microchip PolarFire® FPGA.

Figure 1. Smart High-Level Synthesis FPGA Design Flow Targeting a PolarFire FPGA

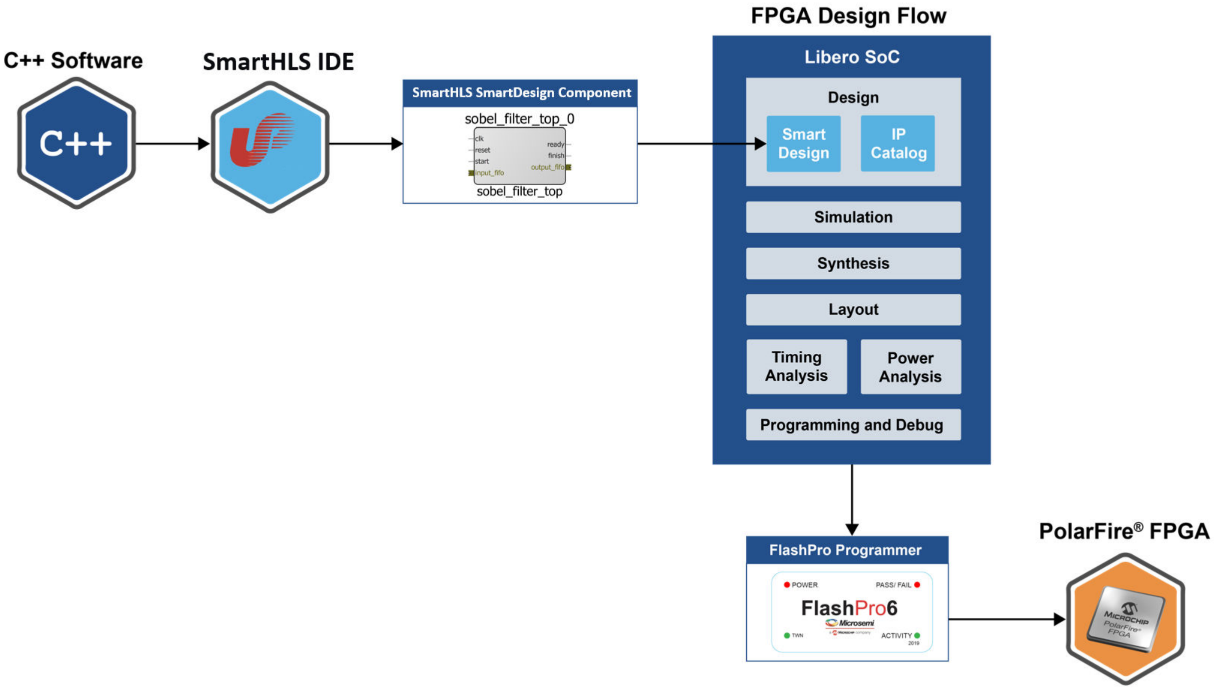


Table of Contents

1. FPGA Design: RTL vs. HLS

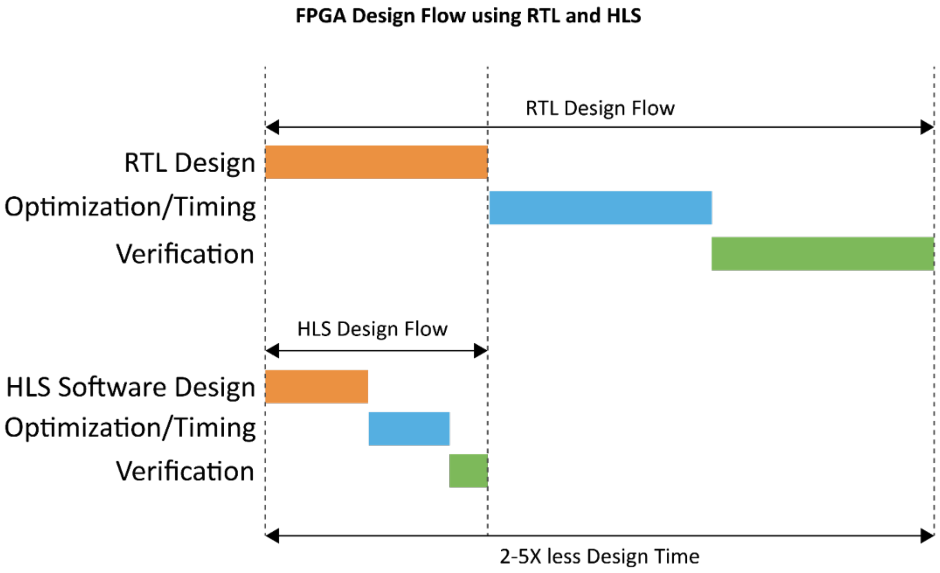
For readers unfamiliar to high-level synthesis, it compares the traditional FPGA design flow using a register transfer level (RTL) language like Verilog/VHDL to the newer high-level synthesis design approach using the C++ software.

Traditional RTL Design Flow for FPGAs:

1. RTL design: Hardware engineer decides the hardware microarchitecture and manually writes the hardware block in RTL language like Verilog/VHDL.
2. Optimization & Timing closure: Hardware engineer iterates on design to meet the clock period and area constraints by adding pipeline registers and restructuring the RTL.
3. Verification: Hardware engineer writes a test bench to verify the RTL block in simulation. Acceptance testing compares functionality to the original software model.

The following figure displays using high-level synthesis to design FPGA hardware blocks with the C++ software code. This can save engineers 2-5x design time compared to traditional RTL design.

Figure 1-1. FPGA Design Flow Using RTL vs. HLS



High-Level Synthesis Design Flow for FPGAs

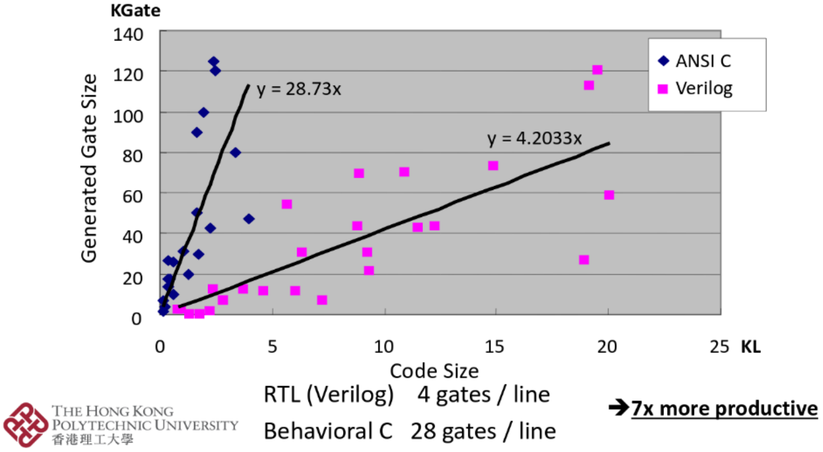
1. HLS Software Design: Engineer writes a software model of the design in the C++ software. Software is tested to verify the desired functionality. Engineer generates a hardware block from the C++ software using high-level synthesis. Engineer rewrites the C++ in HLS style, as necessary. For example, using FIFO types for dataflow computation to get the desired throughput.
2. Optimization & timing closure: Engineer adds HLS constraints for the desired clock period and tunes the HLS settings. For example, to reduce area by sharing hardware operators. Engineer runs the Libero SoC design suite to verify area/timing.
3. Verification: The generated circuit will be corrected by construction. Engineer can use co-simulation to verify the generated RTL block in Mentor Graphics ModelSim® simulation. Co-simulation creates a test bench for the tests from the original software model.

2. Benefits of High-Level Synthesis

2.1 HLS Productivity Gain

Engineers developing an FPGA hardware in C++ using high-level synthesis design tools reduce their design effort significantly compared to engineers writing hardware in RTL. Writing software code is much easier for engineers than writing hardware in RTL because software code is more concise, with 5-10x less lines of C++ required than RTL, as shown in the following figure[1].

Figure 2-1. Lines of C++ Code Versus Lines of RTL Code



Software code is also much easier to read and understand for future improvements or maintenance compared to RTL. Software conciseness and readability mean less bugs in your FPGA design. Engineers will see a 2-5x reduction in the FPGA design time by using HLS as shown in Figure 1-1. Better designer productivity leads to faster time-to-market. The C++ software code describes an algorithm, which is at a much higher level of abstraction than hardware descriptive languages. For example, in C++ code the programmer does not need to specify any hardware timing constructs like clock cycles. In contrast, a hardware engineer writing RTL must specify cycle-by-cycle hardware behavior using finite state machines (FSM)s, and other control logic. If any control logic is off by one cycle, then circuit functionality will be interrupted. High-level synthesis also supports C++ data types to represent floating or fixed-point computation. These data types are not natively supported in Verilog/VHDL languages, therefore the engineer must handle these operations manually.

High-level synthesis also offers much more flexible hardware module parameterization compared to Verilog/VHDL. In a software description of the design, the C++ template parameters offer more fine-grained control versus simple RTL parameters. Furthermore, HLS user-provided constraints can fine-tune the generated hardware microarchitecture for different latency, timing, and area design constraints. Meanwhile, for an RTL design the engineer must manually add pipeline registers to the data path to meet different timing constraints. Last-minute pipeline changes are painful and difficult to parameterize in RTL.

2.2 Port Existing Software to FPGA

SmartHLS allows engineers to reuse a pre-existing software implementation of their algorithm and port this software to an FPGA. In many cases, this existing software implementation has been verified and tested extensively. For an engineer to manually re-implement the software design in Verilog/VHDL could introduce subtle errors that can be tedious to catch and time consuming to verify. High-level synthesis greatly simplifies this process and avoids manual reimplementation by saving design time and engineering resources.

For example, a real-time control application might include software or firmware running on a microcontroller that can no longer meet the required response time. The solution is to migrate the software to an FPGA, which offers deterministic latency. High-level synthesis allows the engineer to automatically generate Verilog/VHDL from the

[1] Innovative Technology Series (ITS)- High Level Synthesis

existing C++ software. See the customer case-study white paper: *Migrating Motor Controller C++ Software from a Microcontroller to a PolarFire FPGA with Smart High-Level Synthesis*.

2.3 Faster Verification and Less Bugs

For engineers, verifying the functional correctness of a hardware design written in RTL is usually the most time-consuming part of designing for an FPGA. The engineer must manually write test benches in RTL to simulate their design under various input/output test stimulus. In contrast, an engineer designing hardware in C++ with high-level synthesis can significantly reduce their time spent on verification. While using high-level synthesis, the engineer can write tests in software to verify their design, since the circuit generated from HLS will be correct by construction. Writing tests in the C++ software is simpler than writing hardware test benches in RTL, which usually leads to writing more comprehensive tests increasing test coverage.

SmartHLS supports automatic co-simulation of the generated hardware with ModelSim, by re-using the C++ software tests. During co-simulation, the SmartHLS tool runs the software program with instrumentation added around the top-level C++ function to collect golden inputs/outputs for each function argument. Then SmartHLS automatically generates a hardware test bench that reads the golden inputs/outputs from test vector files. The test bench verifies that the generated hardware module behaviour matches the software model. Engineers can also write their own custom test bench with a more restricted set of tests to be run on the HLS-generated RTL as a sanity check.
Much less RTL simulation is necessary with HLS design. RTL simulation times can become long and impractical for larger FPGA designs as the number of tests increase. High-level synthesis allows for software-based testing and verification, which has 100-1000X faster runtime than RTL simulation. Faster software-based verification runtimes mean that engineers using HLS can still verify hardware after last-minute design changes if requirements change late in the design process.

2.4 Design Space Exploration

Design space exploration is the process of making hardware microarchitecture design trade-offs within a set of constraints on FPGA clock frequency, throughput, latency, and area. An RTL engineer will usually decide on a hardware microarchitecture early in the design process. After the RTL is written, an engineer will typically avoid making microarchitecture changes, such as adding pipeline stages, because RTL redesign is a time-consuming process. In contrast, high-level synthesis simplifies design space exploration and allows continuous refinement of the hardware microarchitecture throughout the design process. For example, the HLS tool automatically inserts pipeline registers based on the HLS target clock period constraint. The engineer can easily modify the HLS target clock period to achieve different performance/area targets without manual redesign. HLS typically runs in a few minutes, so the designer can quickly get feedback on resource and throughput estimates without waiting for a time-consuming FPGA synthesis run.

SmartHLS offers a rich set of user-constraints and pragmas for the engineer to specify their desired hardware microarchitecture based on the software description. For example, loops can be pipelined to improve performance, or inner loops can be unrolled. Functions with FIFO dataflow streaming inputs/outputs can be pipelined. C++ arrays can be partitioned into registers or RAMs to achieve better memory bandwidth and performance. High-level synthesis also supports sharing larger hardware operations, such as floating-point cores. HLS constraints give the designer the ability to easily perform more design space exploration, which can lead to better trade-offs between performance and area for their FPGA designs.

2.5 FPGA Device Portability

A verified hardware IP block can be reused for many years. Therefore, future proofing is important if the hardware block could later target another FPGA device family. RTL designs have a specific microarchitecture targeting one FPGA family. Therefore, RTL redesign will be required if the timing constraints are not met while porting. RTL designs may also use primitive blocks specific to the FPGA family, which will need to be rewritten.

High-level synthesis automatically adds pipelining stages to your hardware depending on the target FPGA device and clock period constraints. HLS settings can easily be changed to re-generate a hardware block targeting a new FPGA family while still meeting the timing constraints. Portability is simplified because there are no FPGA family-specific C++ design constructs.

3. Conclusion

In conclusion, FPGA design with C++ using high-level synthesis can offer 2-5x better design productivity compared to writing in RTL. The resulting C++ code is 5-10v shorter than an equivalent RTL design and C++ is expressed at a higher level of abstraction. This results in less bugs and easier readability. HLS software-based verification and testing saves significant design effort compared to RTL verification. HLS also enables easier design space exploration and FPGA device portability.

3.1 Key Takeaways

- SmartHLS simplifies FPGA design by allowing you to program the FPGA using C/C++ software
- 2-5X better design productivity
- Higher abstraction level means less code and less bugs
- Faster verification and testing
- Design space exploration
- FPGA device portability

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user’s guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip’s product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods being used in attempts to breach the code protection features of the Microchip devices. We believe that these methods require using the Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Attempts to breach these code protection features, most likely, cannot be accomplished without violating Microchip’s intellectual property rights.
- Microchip is willing to work with any customer who is concerned about the integrity of its code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable.” Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-8676-3

Quality Management System

For information regarding Microchip’s Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4485-5910
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-72884388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820