High Level Synthesis in VLSI Design

Arsal Abbasi
Hochschule Hamm-Lippstadt
VLSI Design
Hamm, Germany
arsal.abbasi@stud.hshl.de

***Abstract*—In this paper, I present the research about High Level Sythesis in VLSI Design, tools that exist for High Level Synthesis along with their functionality and comparision. This paper provides a good overview of High Level Synthesis and also how it is diferent from Logic Syntheis.**

***Index Terms*—synthesis, VLSI Design, Logic Synthesis.**

I. Introduction

Very Large Scale Integration which is also called VLSI is a technology that is used in many digital systems. VLSI technology currently allows hundreds of thousands of transistors in one Application-Specific Integrated Circuit (ASIC), and the level of integration is increasing at a rapid rate [1] VLSI Design is a design of a single Integrated Circuit (IC) which can perform several complex functions. This includes integrated circuits made for a specific design as well as generic components (like microprocessors and memories) that are created once and produced in large quantities (i.e., ASICs). An integrated circuit (IC) is a very small semiconductor chip. In this semiconductor chip there are electronic components and their interconnections are fabricated. ICs started to become known after in the early 1960s. The design and method evolved enormously since then. How ICs have evolved depends upon the number of components integrated on the chip. There are several kinds of integration, they are called as Small-scale integration (SSI), Medium-scale integration (MSI), Large-scale integration (LSI) and Very Large-Scale integration (VLSI). Figure 1 represents a chart to show how many transistors are available in each of these integrations.

Integration Scale	Number of components	Examples
SSI	<10 transistors	Logic Gates
MSI	10 - 1000 transistors	Adders, Counters
LSI	1000 - 10,000 transistors	Multipliers
VLSI	>10,000 transistors	Microprocessors

Fig.1 Chart with different integration scales

Prior to VLSI, there was an attempt to identify and calibrate different levels of large-scale integration. There were words like ultra-large-scale integration (ULSI). However, such minute variations are now irrelevant due to the enormous quantity of gates and transistors found in typical devices. Terms implying integration levels higher than VLSI are no longer commonly used.

Billion-transistor processors started to be sold commercially in 2008. With the development of semiconductor manufacture beyond the then-current generation of 65 nm techniques, this became more typical. Contrary to early devices, modern designs lay out the transistors using considerable design automation and automated logic synthesis, resulting in increased levels of complexity in the logic functionality.To maintain the utmost efficiency, some high-performance logic blocks, such as the SRAM (static random-access memory) cell, are still created by hand.

The process to develop VLSI chip is complex and involves many steps. Figure 2 represents the order of these steps.

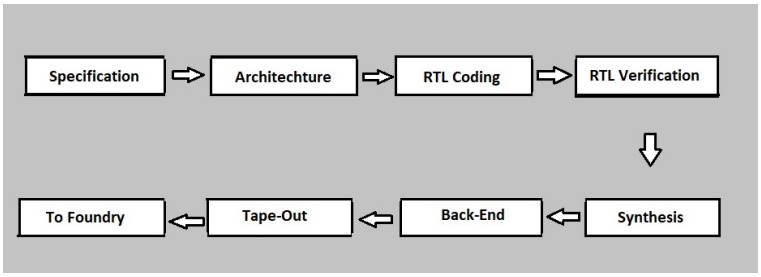

Fig.2 Chip development

- Specification phase: First step of any design process is specification phase, high level representation of the system. The factors include performance, functionality and physical dimensions.
- Architecture Phase: The basic architecture is designed in this phase.
- RTL Coding phase: RTL is Register Transfer level description. It is expressed in a HDL such has VHDL and Verilog. The deisgn is simulated and corrected in this phase. In some special cases logic design can be automated using High level sythsis tools as well.
- RTL verification phase: It is very important phase, in this phase testbench is simulated to verify it´s correctness.
- Synthesis phase: In this phase the abstract design is converted to accurately implemented chip in regard to logic gates.
- Back end phase: In this phase circuit representation is converted to geometric representation
- Tape out phase: After layout and back end phase the design is ready to be fabricated. Since the data is sent on a Tape, the phase is called Tape Out phase.
- Foundry phase: Finally in this the design is fabricated. Each chip chip is packaged and tested to ensure it meets all the design specifications.

II. High Level Sythesis in VLSI Design

In VLSI Designing, synthesis is a very important step during initial stages of designing. A method known as high-level synthesis (HLS) aids in the conversion of a behavioral description of hardware into an RTL model. High Level Synthesis can be defined as the automated designing process that transforms the behavioral or functional description of the design into a digital hardware implementation [2].

As the design complexity increased in VLSI, the hardware designing process involved the usage of Hardware Description Languages also known as HDLs. The HDLs used were majorly Verilog and VHDL. As machine learning algorithms become more sophisticated, particularly when dealing with huge and complicated ones, HDL modelling methodolgy appeared to be impossible to meet the designer's requirements. The following were the limtations of HDLs:

- Difficult to design complex algorithms (e.g. Image Processing).
- Faster time to market with high-quality results is very difficult to achieve.
- Very high verification cost and debug time
- With the evolution in technology library, design has to be updated, e.g. FPGA to ASIC.

The shortcomings listed above emphasize the need for methodology that focuses on functionality and allows the freedom to build complicated algorithms without worrying about the requirements of the architecture.

High-level synthesis offers an error-free route from abstract specifications to RTL, addressing the underlying cause of this issue. Design teams use High Level Synthesis to significantly shorten design time while simultaneously lessening the overall verification burden. High Level Synthesis started the use of C, C++ and SystemC as designing languages which made writing complex algorithms easier for the designers. Figure 3 demonstrates the process.

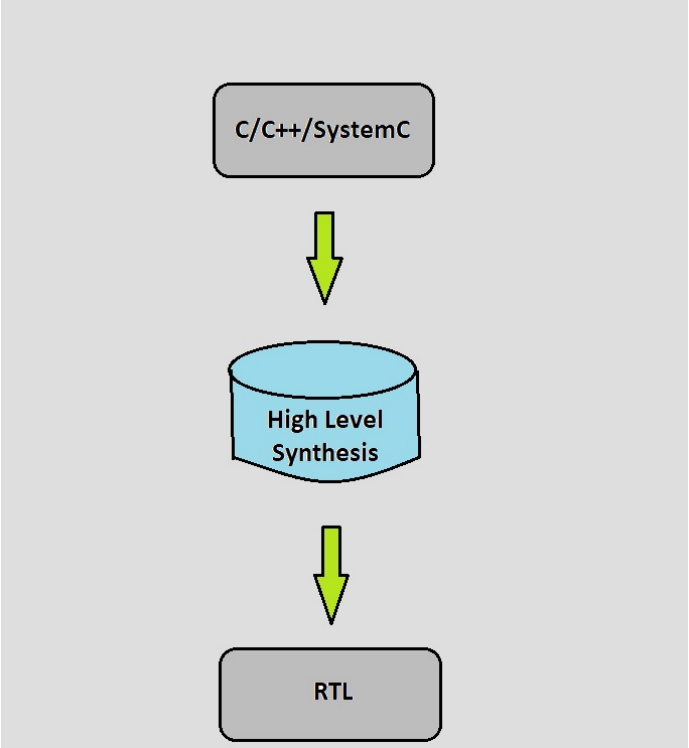

Fig.3 High Level Synthsis process

After creating and verifying the high-level model, High-Level Synthesis automates the RTL implementation procedure. High-Level Synthesis tools do not, however, remove engineering intervention even while they do eliminate manual interventions and errors. That is to say, choices must still be made. High-Level Synthesis allows the designers to continue to be in charge; the High-Level Synthesis tools merely executes their judgments. They are more effective and efficient at their work. As an illustration, the designer determines the appropriate level of parallelism for an ideal architecture and limits the High-Level Synthesis tool accordingly. The tool then handles scheduling and assigning the required hardware resources, creating the datapath, and developing the control structures to create an optimal implementation. With High-Level Synthesis, accurate RTL can be obtained more quickly, cutting the creation phase in half. In turn, the verification workload is lightened and the debug overhead [3]..

III. Toolchains

Many tools exist for High Level Synthesis. In this section some of the important toolchains for High-Level Synthesis will be discussed in detail.

A. Xilinx AccelDSP

Xilinx AccelDSP is a High-Level Synthesis tool by Xilinx. This tools is based on DSP technology. The Xilinx AccelDSP tool is an advanced ESL design tool. It transforms a MATLAB floating point design into hardware module that can be implemented in Xilinx FPGA [4]. The user-friendly Graphical User Interface of the AccelDSP Synthesis Tool provides an environment that is integrated with other design tools like

MATLAB, Xilinx ISE tools, and other widely used HDL simulators and logic synthesizers [5]. Accel DSP only works on streaming function which the designer has to code explicitly. By scanning the Matlab code and instantiating fundamental building blocks, automatic HDL code synthesis is carried out, this is comparable to how HDL code is generated by Xilinx System Generator.

In this, the designer has to start by writing a floating-point golden reference Matlab model. Automatic floating point to fixed point conversion is done by dynamically predicting the required bit width, the fixed width is limited to 53 bits. In case more bits are needed, the user has to address the fixed-point toolbox.

After all fine-tuning is done, three different results are generated; RTL code, Xilinx System Generator and Design Simulation are generated for the user to select.

B. Agility Compiler

Agility compiler is a tool Agility Design Solutions. This tool is used to synthesize SystemC. It adds extra Agility hardware features in addition to full support for the OSCI SystemC synthesizable subset. Actel, Altera, and Xilinx FPGAs are all supported for automatic code generation.

With the use Agility Compiler, designer can do several things [5].:

- Explore complex algorithm and architectures in the design flow
- Reduce overall design time by generating RTL or EDIF netlists from systemC models.
- Tun around very complex system designs much closer to the go-to-narket window

Figure 4 represents the workflow diagram.

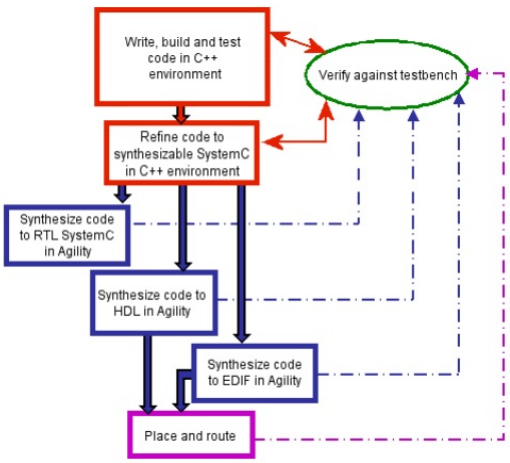

Fig.4 Workflow diagram [6].

In Agility Compiler, designer needs to write the hardware processes manually in SystemC, define sensitivity losts etc. It also supports the automatic generation of control and data flow charts. Black boxes that can import existing HDL IP, global asynchronous reset signals, and signals with preset values are some of the extra Agility hardware capabilities.The main advantage of Agility Compiler is the use of C++ which makes it easy for some designers to design using this tool.

C. Auto Pilot

AutoPilot was developed by AutoESL, which is acquired by Xilinx.It generates HDL from a number of C-based input languages, including SystemC, ANSI C, and C++. To generate an RTL design, only a few small changes to the C code are required. Area, latency, and battery consumption can all be optimized by autopilot. Xilinx stated that this tool's new name is Vivado ESL.

AutoPilot's arbitrary-precision datatypes, starting from C generic code. In the next step, algorithm and interface synthesis are performed on the design. Datapath behavior is extracted by AutoPilot, and control behavior is found in loops and conditions that are mapped to an FSM. By unrolling the loop and evaluating the conditions, datapath evaluation is accomplished. Scheduling and binding are decided upon in accordance with user restrictions, such as those for latency and resource utilization.AutoPilot generates a wrapper that facilitates communication between the generated RTL and the C testbench automatically, allowing the reuse of the C-code testbench for simulation and verification.

D. Blue Spec

BlueSpec is a tool by design company BlueSpec Inc.. It uses a different approach for abstraction, the design is entered in BlueSec using system Verilog. Bluespec offers high-level synthesis with RTL for ASIC and FPGA hardware designers and architects.

All design elements are implemented as individual modules, which leads to long portmaps and connection lists. Compilation is controlled using BlueSpec's GUI. BlueSpec examines the source code line by line before producing RTL, this benefits the designer as the readability and traceability in enhanced.

E. Catapult C

CatapultC is a commercial electronic design automation product of Mentor Graohics, it is used for High-Level synthesis. Catapult C takes ANSI C/C++ and SystemC inputs and generates RTL code targeted to FPGAs and ASICs [7].. Catapult C supports SystemC model generation intended for virtual platforms, and a SystemC test environment to verify the generated RTL against the original C++ using the original C++ testbench [7].

Catapult C has an excellent interface that halps the designer through out the High-Level Synthesis process. Steps include the selection of source files (design and testbench), design setup (target technology, clock rate etc.), design constraints (area and latency, I/O and array mapping, loop optimizations), scheduling and RTL generation.The different design elements, such as I/Os, arrays, loop nests, and the generated schedule, are all very clearly visible to the designer. The program can also be launched via scripts in addition to the GUI.

The GUI is very helpful and gives access to many different features to play with, these include the selection of the target technology and clock rate and the selection of interfaces.

Cataopult C generates testbenches for ModelSim, for cycle accurate, RTL and gate level implementations of the design,

for the purpose of verification. Additionally, a verification testbench can be created by combining the produced design with the original C++ design and testbench. This verification testbench compares the output of both designs using the identical input data applied to both designs. This greatly expedites the verification process.

F. C-to-Silicon

C-to-Silicon is a High-Level Synthesis tool by Cadence. Cadence Design Systems is a global leader in electronic design innovation. C-to-Silicon Compiler raises the design abstraction level so that verification can be more productive and the design can be more readily re-used across varying configurations in end devices. The unique incremental synthesis capabilities that C-to-Silicon Compiler offers integrate tightly with Silicon Realization technologies, including Encounter RTL Compiler, Encounter Conformal® ECO and Conformal LEC, to deliver a predictable TLM-to-GDS flow with full ECO capabilities [8].

C-to-Silicon uses SystemC langauge for designing. SystemC data types support fixed-point data types and arbitrary bit widths.Unlike other tools, CtoS creates the technology library while the design is being built, relying on RTL libraries and also utilizing an RTL synthesis.

G. Synphony C Compiler

Synphony C Compiler is High Level Synthesis tool by Synopsys'. Synopsys specializes in software security and quality, silicon design and verification, and silicon intellectual property. The designing languages used in Synphony C compiler are ANSI C and C++. A variety of C and C++ constructs are supported by the Synphony C Compiler. Optimisation options are set by adding pragmas in the source code. In Synphony C Compiler automated verification occurs at several stages, including on the generated design, in the source code, after preprocessing, scheduling, and synthesis.

IV. Comparision

Many tools exist for High-Level Synthesis. All tools have their pros and cons, mostly it depends on the designer how he finds the tool and what he feels comfortable in. In Xilinx's AccelDSP is a powerful tool for High-Level Synthesis.In AccelDSP The code needs to be written manually which is a major time constraint while typing the code. The ease of implementation is good in this tool, it has a very easy to use Graphical User Interface GUI which helps the designer. Agility Compiler is considered a good tool for High-Level Synthesis. With Agility Compiler designer can explore complex algortihm and architectures design flow. It maintains the systemC coding standard throughout. Agility Compiler outputs RTL SystemC as SystemC netlist, the clock cycle timing of the output matches with clock cycle of the input, this feature reduces the verification overhead and therefore helps the designer. AutoESL's tool Autopilot is a great High-Level synthsis that uses SystemC, ANSIC and C++. To help the designer AutoPilot generates a wrapper that facilitates communication between the generated RTL and the C testbench automatically, allowing the reuse of the C-code testbench for simulation and verification. AutoPilot accepts as input a high-level C, C++, or SystemC description of functionality. A register-transfer-level (RTL) description of a hardware implementation aimed at an FPGA (made, for instance, by Altera or Xilinx) or ASIC is subsequently generated, device-specific in Verilog or VHDL. By doing this, the labor-intensive and error-prone process of manually building the RTL implementation is eliminated. For the synthesis findings, AutoPilot additionally creates a cycle-accurate SystemC simulation model to help the designer. Just like other, BlueSpec is also a very appreciable tool for High Level Synthesis. It's GUI is not complicated and user friendly. The design elements are implemented as individual modules. Blue Spec's property of examining the source code line by line before producing RTL benefits the designer in readilibility and traceability. Catapult C is a quite powerful tool in the eyes of many designer. It takes varitey of languages for High Level Synthesis ANSI C/C++ and SystemC. It provides an excellent interface, that makes it stand out in the eyyes of many designers. The GUI is very helpful and gives access to many different helpful features. Catapult C is an allround tool, it proves to be the best in every department, be it verification, abstraction or ease of implementation. In CatapultC a verification testbench can be created by combining the produced design with the original C++ design and testbench. This verification testbench compares the output of both designs using the identical input data applied to both designs. C-to-Silicon is also a sucessful High-level Synthesis tool by Cadence. It uses C language for designing. It supports Xilinx and Altera FPGA devices and synthesis flows. It synthesizes datapath and control logic together, supporting all type of designs in the mixed datapath-control spectrum [9]. C-to-Silicon produces RTL and constraints compatible with third-party RTL synthesis and logic equivalence checking as well as Cadence. The C-to-Silicon Compiler incorporates production logic synthesis technologies to offer precise analysis and optimization recommendations. In order to more correctly simulate wire timing effects, it characterizes the timing, power, and area of logic components in the context in which they are employed in the design. These features of C-to-Silicon makes this tool very reliable and easy to use for High-Level Synthesis. Synohony C compiler is too a great tool for High-Level Synthesis. It can multiple designing languages, which include ANSI C and C++. By generating application accelerators from high level C/C++ code and automating the verification process all the way through implementation, Synphony C Compiler increases productivity significantly. By adopting a special parallelizing compiler, multi-level hierarchical abstraction, and IP reuse, Synphony C Compiler achieves outstanding QoR. Amongst all the mentioned tools for High Level Synthesis, Autopilot proves to be the one that provides many features with the ease of use to the designer.

V. Analysis

The difference between High-Level Synthesis and Logic Synthesis is quite board. Logic Synthesis has been there much longer High-Level Synthesis. In Logic Synthsis, the user

provides RTL to the synthesizer which then converts it to gate level representation. In Logic Synthesise the paremeters can be easily manipulated by the the designer according to the needs. Meanwhile, High-Level Synthesis converts C/C++ program to RTL.

Since High-Level Synthesis is new, there is still a lot of work going on for the improvement. Due to the reason that it is new, many designers unaware of the functionalities of the High-Level Synthesis tools and they find it difficult to use. Logic Synthesis is old and easy, but it has it's limitations. It is very time consuming and complex for designers to write RTL meanwhile in High Level Synthesis many tools use C/C++ for design description, algorithm and logics which is easier and less time consuming as compared to Logic Synthesis. In addition to that High-Level Synthesis tools provide C based simulation which helps the designer in easy verification of the results.

VI. Conclusion

In VLSI designing, synthesis is a very important step during the initial stages of designing. High Level Synthesis aids in the conversion of a behavioral description of hardware into an RTL model. To conclude, several tools exist for High Level Synthesis, and they help the designer in different ways. Since High Level Synthesis is the technology of toda, there is still work going on trying to make it better. As compared to Logic Synthesis which has been there for a while, it needs improvements. In Logic Synthesis designer works by writing RTL which then converts it to gate level representation, because of it's age, many designers are aware of the tools and their functionality which is not the case with High Level Synthesis.

References

[1] https://www.proquest.com/openview/832a0aa8432a949eba4770b37beeda39/1?pq-origsite=gscholarcbl=18750diss=y

[2] http://www.vlsi-expert.com/2021/05/high-level-synthesis-intro.html: :text=High

[3] http://www.vlsi-expert.com/2021/05/high-level-synthesis-intro.html: :text=High

[4] AccelDSP Synthesis Tool User Guide, Vol. UG634 (v11.4), www.Xilinx.com

[5] https://www.yumpu.com/en/document/read/31206105/agility-compiler-for-systemc-europractice

[6] http://rdsl.csit-sun.pub.ro/docs/Uni$Brown/ic.engin.brown.edu/classes/EN2911XF07/agility_manual.pdfUniversityofOuluRapidSchedulingofEfficie$
$GenerationHSDPAWirelessSystemUsingCSynthesis$

[7] http://www.scientificlib.com/en/Technology/Electronics/CatapultC.html

[8] https://www.cadence.com/ko$_K R/home/company/newsroom/press -$
$releases/pr/2010/cadencectosiliconcompilersupportedinfujitsusemiconductorsasicflowforsystemrealization.htmlhttp:$
$//pdf2.solecsy.com/564/5c0644f6 - 808c - 4d18 - 9b31 -$
$0eec054873e5.pdf$