Abstract

Nowadays, because of the higher hardware complexity and revolutionary evolution in microprocessor architectures, more and more embedded software components are being integrated into modern electronic systems. This means that architects, designers, and verification engineers need to deliver correct functional designs in shortest possible time.

From the architecture and design points of view, high-level synthesis (HLS) flows are getting popular in recent times. By employing languages such as C, C++, MATLAB, and Python, HLS flows allow designers and architects to describe the design behavior at a much higher level of abstraction. They also share the connection to downstream tools for implementation. Studies have shown that HLS flows help to reduce the design cycle time. This is because they allow faster iterations between multiple implementation choices from a common functional specification and assure a continued verification throughout the design flow.

Field programmable gate arrays (FPGAs) are additionally gaining popularity in VLSI design flows. They provide acceleration to the verification process owing to their faster speed of operation as compared to conventional hardware description language (HDL) simulation tools. Due to the same reason, FPGAs are also the platforms of choice for accelerated verification of hardware designs. FPGAs are also commonly used to run real-world customer application scenarios in a pre-silicon environment to have better confidence in functionality and performance. In fact, some of the design and firmware verification tests which involve interaction with real physical debuggers and peripherals cannot be run in a simulation environment and hence are required to be run on reconfigurable hardware like FPGAs. Hence, for any VLSI design application, area, speed and power-efficient implementations on FPGAs is the need of the hour. Conventional HDL synthesis and FPGA design flows have matured over the last few decades and allow multiple hooks and options or constraints for optimal hardware implementations, but these are not valid for HLS flows directly. In the recent past, even though multiple researchers and electronic design automation (EDA) tool vendors have invested heavily in optimization techniques for HLS-based design flows. However, such techniques are very “design,” “application,” and “tool” specific and not generic at all. This work proposes a methodology for the area, speed, and power optimization, which are independent of the used

design application and HLS flow. The first objective of this work is to study about different HLS tools and use them to create different application designs. The second objective is to study the effect of HLS optimization directives offered by these tools on different designs. These directives help in the area, speed, and power optimization of the designs depending on how they are used. We demonstrated the use of these directives for multiple designs from different application areas like signal, image processing etc. The third objective of this thesis is to propose a novel HLS-based design methodology known as "application-specific bit width for intermediate data nodes." The efficacy of the proposed design methodology is proved on designs from diverse application areas such as image processing, signal processing, and computing algorithms. This work provides a comparison of the results with benchmark implementations available in the literature. The methodology can be termed as generic as designs from vast application areas have been chosen for this thesis. The designs have been created using MATLAB HDL coder and Vivado HLS for these applications as part of this study because of their wide acceptance in the industry.

The final objective of this work is to apply a combination of HLS directives along with the proposed novel HLS-based design methodology to design, verify and optimize a radio detection and ranging (RADAR) signal processor for automotive applications and compare the FPGA functional simulation results against a golden taped-out system-on-chip (SoC) model.