Approximate Adder Structures on FPGAs

Andreas Becher, Jorge Echavarria, Daniel Ziener, and Jürgen Teich
Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany
Email: {andreas.becher, jorge.a.echavarria, daniel.ziener, juergen.teich}@fau.de

***Abstract*—In this talk, we propose novel approximate adder structures for FPGA-based implementations. These adder structures take advantage of the available FPGA resources. Compared with a full featured accurate adder, the longest path is significantly shortened which enables the clocking with an increased clock frequency. By using the proposed adder structures, the throughput of an FPGA-based implementation can be significantly increased. On the other hand, the resulting average error can be reduced compared to similar approaches for ASIC implementations.**

I. Introduction

The processing of multimedia content allows the decrease of accuracy due to many reasons. One is the limitation of human sense organs which are unable to recognize very subtle variations. On the other hand, if the input data gathered, for example, from an image sensor has a decreased quality, e.g., due to not optimal light conditions, the following digital signal processing chain must not evaluate the input data at the highest possible accuracy. The usage of approximate computing permits performing multimedia processing in an energy- and resource-efficient way [1], [2]. This emerging paradigm is drawing a lot of interest from researchers since several years. The application and exploitation in programmable and adaptive systems based on reconfigurable FPGAs is rather new and subject of current research. In order to build efficient FPGA-based circuits for approximate computing, we have to look first at basic structures, like adders.

In contrast to ASIC approaches [3], FPGA implementation of approximate adders must deal with available FPGA resources. For adders, LUTs, the corresponding carry-logic inside FPGA slices and dedicated carry routing channels between slices are used. At the first look, the freedom of design choices is massively restricted compared to ASIC approaches which can easily use additional logic gates. Furthermore, current FPGA-based implementations of adders are very efficient due to the usage of special carry resources. On the other hand, an accurate adder uses many FPGA resources which are not fully utilized. For example, modern FPGAs consist of LUTs with six inputs and can implement a five input LUT with two outputs. A standard adder uses only two inputs for each bit and left the others unused. These inputs cannot or only partly be used for the mapping of other logic due to the occupied LUT output. By improving the utilization of available resources, an approximate adder can be designed with the same amount of LUTs and a small resulting error compared to an accurate adder. The advantage is to significantly shorter the longest path and, therefore, a remarkable increased throughput.

II. Proposed Example Adder

In [3], the authors propose to split the inputs of an n bit adder at the position $m : 0 < m < n - 1$. In the *most significant part* (MSP) an accurate addition is performed from the *least significant bit* (LSB) m to the *most significant bit* (MSB) $n - 1$. In the *least significant part* (LSP), they propose to perform the approximate addition from the MSB (bit $m - 1$) to the LSB (bit 0). Standard one bit additions are performed without carry propagation unless both input bits a_i and b_i at bit position i are 1. The remaining output bits s_i to s_0 are set to 1. This technique allows saving area and power of an ASIC implementation with a maximum error of $2^m - 1$. However, implemented with available logic on an FPGA, n Lookup Tables (LUTs) are used thus leading to no resource saving in terms of number of LUTs compared to an accurate adder implementation as our experiment have shown. The main benefit achieved with their proposed architecture of an approximate adder is the drastic reduction of the longest path. This holds true for both, the FPGA and the ASIC implementation.

As it can be seen in Figure 1, LUTs on modern FPGAs have more than three inputs as it would be necessary to implement full adders. We now propose to make use of these additional, not needed but available, inputs to reduce the overall error introduced by approximate adders. Our proposed design performs an accurate addition on both sides of the adder and from the LSB to the MSB. If the carry from the MSB of the LSP c_{m-1} is one, all bits of the LSP are set to one, except the case that a_{m-1} and b_{m-1} are one. In this case, the MSP result is incremented by also feeding the inputs a_{m-1} and b_{m-1} to the adder in the MSP in order to reduce the error. By sharing the inputs $a_{m-1}, b_{m-1}, a_{m-2}, b_{m-2}$ to the LUTs in stage $m-1$ and $m-2$, the overall latency can be reduced. We achieved a five times lower mean error compared to the implementation of [3] and the maximum error is $2^{m-1} - 1$.

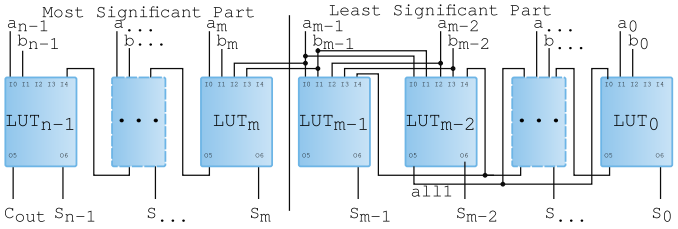

Fig. 1. Our example approximate adder implemented with LUTs as available on FPGAs. It makes use of the not connected inputs of the *5x2* LUTs provided in Xilinx 7-series devices in order to reduce the error generated by disconnecting the carry chain. This can be done on occasions where a carry would be produced at $m-1$ but is not forwarded to LUT_m. By simply connecting the bits at $m-1$ to LUT_{m-2}, an $all1$ signal can be generated and forwarded to the lower bits, setting their s output to 1. As already both outputs of LUT_{m-2} are used, the bits $m-2$ and the carry from $m-3$ are connected to LUT_{m-1} to reconstruct the carry signal from LUT_{m-1}. Furthermore, bits $m-1$ are connected to LUT_m in order to allow correct carry propagation if bits $m-1$ are both 1 reducing the error drastically.

References

[1] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Test Symposium (ETS), 2013 18th IEEE European*. IEEE, 2013, pp. 1–6.

[2] K. Nepal, Y. Li, R. Bahar, and S. Reda, "Abacus: A technique for automated behavioral synthesis of approximate computing circuits," in *Proceedings of the conference on Design, Automation & Test in Europe*. European Design and Automation Association, 2014, p. 361.

[3] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 18, no. 8, pp. 1225–1229, 2010.