ISSN:1306-3111
e-Journal of New World Sciences Academy
2011, Volume: 6, Number: 4, Article Number: 2A0072

İbrahim Şahin
Süleyman Çakıcı
Pakize Erdoğmuş
Duzce University
ibrahimsahin@duzce.edu.tr
suleymancakici@duzce.edu.tr
pakizeerdogmus@duzce.edu.tr
Duzce-Turkey

TECHNOLOGICAL APPLIED SCIENCES
Received: September 2011
Accepted: October 2011
Series : 2A
ISSN : 1308-7231
© 2010 www.newwsa.com

PERFORMANCE AND COST EVALUATIONS OF ADDERS USED IN FPGA-BASED SYSTEMS

ABSTRACT

One important component of the most digital designs is binary adders which greatly affects the total performance of the designs. In the literature several different types of adders were proposed. In this study, performance and cost evaluations of five selected adders, two of which were generated using IP Core Generator and three of which were designed adders, were done on two selected FPGA chips. The results show that, the adders generated using the IP Core Generator with DSP48Es block are the best in most cases. Among the three non-generated adders, the carry select adder showed slightly better performance on average on both chips than others. On the other hand, in contrary to the expectations, it costs about the same amount of hardware with the other two. Another outcome of this study is that using larger Look-up Tables did not improve the costs of the designed adders as much as expected.

Keywords: Binary Adders, Cost, Performance, FPGA, VHDL

FPGA TABANLI SİSTEMLERDE KULLANILAN TOPLAYICILARIN MALİYET VE BAŞARIM DEĞERLENDİRMESİ

ÖZET

Sayısal tasarımların başarımını büyük ölçüde etkileyen önemli bileşenlerden birisi de ikili toplayıcılardır. Literatürde önerilen birçok toplayıcı türü bulunmaktadır. Bu çalışmada, ikisi IP Core Generator tarafından oluşturulmuş üçü ise tasarlanmış toplam beş farklı toplayıcının iki farklı FPGA yongası üzerinde başarım ve maliyet değerlendirmesi yapılmıştır. En iyi başarımı DSP48Es blokları kullanılarak IP Core Generator tarafından oluşturulan toplayıcıların sunduğu tespit edilmiştir. Tasarlanan toplayıcılar arasında ise "carry select adder" her iki yonga üzerinde de diğerlerine oranla daha yüksek başarım göstermiştir. Diğer taraftan, beklenilenin aksine, donanım gereksiniminin diğer iki tasarlanan toplayıcı ile aynı miktarda olduğu görülmüştür. Bu çalışmadan elde edilen diğer bir önemli sonuç ise, toplayıcıların gerçeklenmesinde büyük arama tablolarının (Look-Up Tables) kullanılmasının toplayıcı maliyetinde beklenen ölçüde iyileşme sağlamamasıdır.

Anahtar Kelimeler: İkili Toplayıcılar, Maliyet, Başarım, FPGA, VHDL

1. INTRODUCTION (GİRİŞ)

Nowadays, the usage of Field Programmable Gate Arrays (FPGAs) has been increasing for the applications requiring real time and high performance signal processing such as image processing, various network systems and digital communications. FPGAs were originally developed to serve as a test device for testing digital designs, since any digital circuit design can easily be converted into an FPGA configuration and tested [1]. Recent advancements in FPGA technology provide us with a high degree of parallelism [2] and enable us map more complex applications to FPGAs such as Fast Fourier Transform (FFT) and digital filtering.

Most digital designs done for mapping applications to FPGAs require design units for performing basic arithmetic operations [3 and 4]. These arithmetic operations such as addition, subtraction, multiplication and division can all be achieved using a suitable combination of different types of binary adders.

Definition: "Critical Path" in a digital system is a path between two registers, through which the propagation delay is the biggest. The path can only include combinational logic elements and can not include any memory elements such as latches and flip-flops.

The critical path of a design is the key factor which determines the performance of the whole design. The shorter the path is, the higher the design can be clocked. In most cases, the critical path of a digital design goes through an adder or a design unit which includes adders. So, designing an adder with a shorter critical path greatly affects the performance of the whole design. On the other hand, adders with shorter path expected to cost more hardware resources.

So far, many different adder types and combinations have been proposed and their cost and performance analysis were done. In [5], a unified framework for optimal sizing of parallel prefix adders in the energy-delay space is presented. To determine which adder performs best in a specific system, a circuit sizing tool is used to optimize the performance of carry look-ahead adders in the energy-delay space.

In [6], the costs and the operational delays of fixed-point adders are discussed on Xilinx 4000 series devices, and timing models and optimization schemes for carry skip and carry select adders are proposed. Cost is measured as the number of Configurable Logic Blocks (CLBs) used.

In another study [7], Bečvář et al. made a comparison among fixed-point arithmetic units implemented with *Virtex-II* FPGA. Their results showed that the best adder structure in terms of both time and area is the Generated Adder that uses dedicated carry chains generated by a synthesis tool. In contrast to RCA, the synthesis tool utilizes the dedicated carry chains of Xilinx *Virtex-II* FPGAs. *Virtex-II* FPGA chip is relatively old chip. Bečvář et al. conducted their study only on this chip [7]. In our study, we made a comparison between an old and a new chip [8 and 9]. Besides these studies, some existing works built on the adders are also available. For instance, in [10], some methods used to construct high performance floating-point components are described.

The remaining of the paper consists of five sections. Research significance of the study is justified in the second section. An overview of the selected adders is given in section three. Section four explains the adders generated on DSP48E and FPGA fabric. Performance and cost results of the adders on FGPA and the comment on the results are presented in section five. In the last section, a brief comment and conclusions about the results are given.

2. RESEARCH SIGNIFICANCE (ÇALIŞMANIN ÖNEMİ)

FPGA chips and synthesis tools for these chips are continually improved. Every day, a new FPGA chip comes to market with new features, improved timing and cost properties, and the synthesis tools are updated to cover these new chips.

Recently, as far as we know, not many works analyzing cost and performance of the adders on new FPGA chips have been done. Therefore, there is no definitive answer on which adder implementation leads to the fastest possible adder, which one has the smallest delay [5], or which one has the minimum cost. Existing comparisons of delay deal with the impact of wires with fixed sizing [11], impact of carry tree topology on logic depth [12] and optimal transistor sizing using logical effort [13].

So, as an alternative to the aforementioned studies, in this paper, we presented cost and performance evaluations for five different adders which were implemented in VHDL and mapped on one relatively new and one relatively old chip. Three of them were user designed adders which are Ripple Carry Adder (RCA), Carry Select Adder (CSA) and Carry Look-ahead Adder (CLA). The other two adders were generated adders which were generated using Xilinx's IP Core Generator.

The aim of this study is to present a comparison that helps the designers to decide which adder is optimal for their application on different chips with different Configurable Logic Block (CLB) structures.

The adders were mapped the two different FPGA chips which are Xilinx's *Virtex-5* and *Spartan-3*, and performance and costs of them were observed in terms of propagation delay and in terms of FPGA slices and Look-up Tables (LUT) or Digital Signal Processing (DSP) block usage, respectively.

The results showed that in terms of both performance and cost, the adders generated using IP Core Generator on DSP48E blocks showed the best performance in most cases. If enough DSP48E blocks are available, then adders should be built using these blocks. Otherwise, if design tools such as IP Core Generator available, the adders should be formed using these tools. If none of the above is available, then CSA can be used. CSA shows slightly better performance on average compared to the other adders and costs about the same amount of hardware resource.

3. AN OVERVIEW OF THE SELECTED ADDERS (SEÇİLEN TOPLAYICILARA BİR BAKIŞ)

Adders differ in the way that the carry signals propagate [6]. As mentioned previously, many different adder types and combinations of them were proposed. We selected three well known adders which are suitable for FPGAs and known as fast in the literature. We also selected two different types of adders generated by Xilinx's IP Core Generator. In the subsequent sections, we'll briefly introduce these adders, their specifications, advantages, and disadvantages.

3.1. Ripple Carry Adder (Ripple Carry Toplayıcı)

The Ripple Carry Adder (RCA) is a basic and commonly used adder type based on full adders connected into a chain. The logic for bit-sequential addition of two-bit numbers is described in Equation (1) and it can be implemented as a combinational circuit using full adders connected in series [14] as shown in Figure 1.

It is called ripple carry adder since the carry signals "ripple" from one full adder to another through the adder chain. Despite of its basic structure and being easier to implement, RCA has a basic handicap, which is linearly increasing carry-propagation delay.

Logic Equation:

$$s_i = a_i \oplus b_i \oplus c_i$$
$$c_{i+1} = a_i\, b_i + a_i\, c_i + b_i\, c_i \quad (1)$$

where $i = 1 , 2 ,\dots, n-1$, $c_{in} = c_{n-1}$ and $c_{out} = c_n$

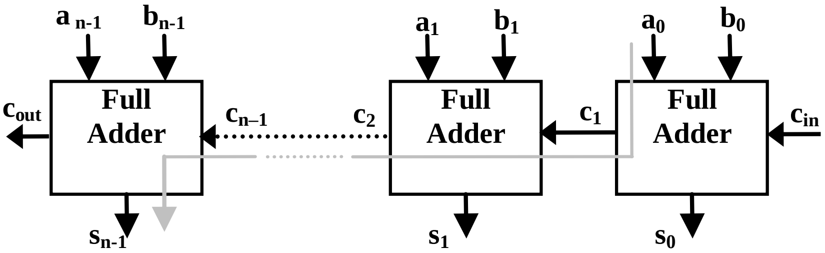


Figure 1. n-Bit generic ripple carry adder
(Şekil 1. *n*-bitlik jenerik ripple carry toplayıcı)

The computation time of this adder grows linearly with addend length, *n*, due to the serial carry-propagation [14]. So, the RCA is relatively slow, since each full adder must wait for the carry bit coming from the previous full adder to be calculated. Being easier to implement makes this adder preferable among the others; therefore, we included it to the scope of this study for comparison with the other adders.

3.2. Carry Look-Ahead Adder (Carry Look-Ahead Toplayıcı)

The Carry Look-ahead Adder (CLA) consists of partial full adders and a carry look-ahead unit as shown in Figure 2. CLA is a complex binary adder compared to the RCA. It uses the same carry look-ahead circuits to construct the higher-bit CLA recursively. It is widely used due to its superior performance over RCA [15].

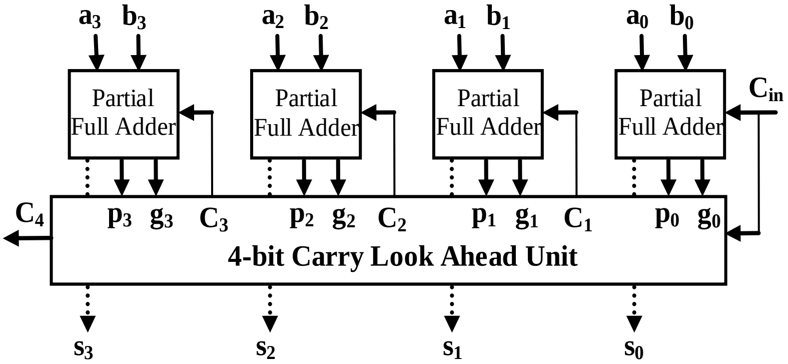


Figure 2. 4-bit CLA adder block diagram
(Şekil 2. 4-bitlik CLA toplayıcı blok diyagramı)

The most prominent feature of CLA is the carry calculation process. At each bit position, the carry-out signal is derived regardless of the previous bit position. This feature is only valid for Application Specific Integrated Circuits (ASIC) implementation of the adders, not for the FPGA implementation.

The carry-out bit of the most significant adder and the sum bit will be available only two gate delays or a total of two gate delays after the input signals a_i and b_i have been applied. Calculations of sum (s_i) generate (g_i), propagate (p_i), carry signals (c_i) are given in Equation 2.

Logic equation:

$$\begin{aligned} g_i &= a_i\, b_i \\ p_i &= a_i \oplus b_i \\ s_i &= a_i \oplus b_i \oplus c_i = p_i \oplus c_i \\ c_{i+1} &= g_i + p_i\, c_i \quad (2) \end{aligned}$$

where $i = 1\ ,\ 2\ ,\ldots,\ n-1$ and $c_0 = c_{in}$

The advantage of CLA is that the carry delays of each bit position are the same regardless of the number of previous bits in the adder, while the carry delay increases linearly in RCA as the bit count increases. On the other hand, one disadvantage of CLA is that the complexity of the adder increases in accordance with the number of bits in the adder.

3.3. Carry Select Adder (Carry Select Toplayıcı)

The Carry Select Adders (CSAs) are generally formed using 4-bit adder blocks. Each block includes two 4-bit RCAs and 5 multiplexers to select correct result among two possible results as shown in Figure 3.

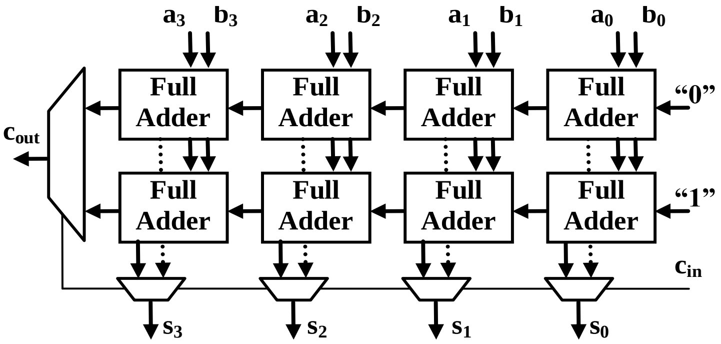


Figure 3. 4-bit carry select adder design
(Şekil 3. 4-bitlik carry select toplayıcı tasarımı)

Addition of two binary numbers is done with two separate RCAs and two separate results are produced. The reason for generating two results is that one adder assumes that the carry-in from the previous adder is '0' and the other assumes that the carry-in is '1'. By doing so, the adders at the current stage calculate two results without waiting for the carry-in signal from the previous adder. As soon as the carry signal is determined at the previous stage, it is used to select correct result at the current stage and this process goes through the adders. The selection is done with the multiplexer.

The CSA can calculate addition of two binary numbers faster than the RCA since it has a modified carry-propagation design. Moreover, inside CSA design, 4-bit RCAs can be replaced with CLA for fast addition [16].

The CSA is especially beneficial for the calculation of large binary numbers. However, as the number of bits to be added increases, it is expected that the cost of a CSA grows faster compared to the other types of adders.

4. THE ADDERS GENERATED ON FPGA FABRIC AND ON DSP48E BLOCKS USING IP CORE GENERATOR (DSP48E BLOĞU VE FPGA DOKUSU ÜZERİNDE IP CORE GENERATOR TARAFINDAN OLUŞTURULMUŞ TOPLAYICILAR)

IP Core Generator is one of Xilinx's design tools which automatically generate HDL code for several logic design units. The core generator is also able to generate adders on both FPGA fabric and digital signal processing blocks (DSP48E) if available on the target FPGA chip.

The CLBs in some FPGA chips include special logic units called carry chain for faster calculation of the carry information in the adders. Since these carry chains are specially designed, they are able to calculate carry information much faster than any carry logic mapped to look-up tables, and since IP Core Generator utilizes these carry chains when generating adders, the resulting adders becomes much faster than traditional adders [8 and 9]. DSP48E blocks are specially designed blocks for Digital Signal Processing (DSP) operations. Each DSP48E contains a 25×18 bit two's complement multiplier, one up to 48 bit configurable adder and some more logic for some extra functionality. The adder inside a DSP48E block is a dedicated adder and therefore, works much faster than the other adder implementations [17].

The adders inside multiple DSP48E blocks can be cascaded to form larger adders. Up to 48 bit adders, only one DSP48E block is used, and for larger adders more than one DSP48E blocks are used. The major drawback of using DSP48E is that not all FPGA chips include these blocks. Moreover, DSP48E resources in FPGA chips are very limited, and one or more of the DSP48E blocks must be utilized to build an adder.

5. PERFORMANCE AND COST RESULTS OF THE ADDERS WHEN MAPPED TO FPGAs (TOPLAYICILARIN FPGA ÜZERİNDE BAŞARIM VE MALİYET ANALİZİ)

Here, experimental test results and discussions about the results are given. We focused on the performance and cost of the selected adders when they are implemented on different FPGAs. Performance parameter of this study is propagation delay of the critical path of the adders when all inputs and outputs are registered, and the cost parameters are the number of slices, LUTs and DSP48E blocks used to implement the adders. Selected three adders, RCA, CLA and CSA, and the adders generated on fabric and DSP48E were implemented in several standard bit widths. 16, 32, 64 and 128 bit adders are commonly used for integer addition operations. 24, 53 and 113 bit adders are used in single, double and quad precision floating-point addition operations described in IEEE-754 standard. The other bit width, 96 bits, is selected to see how cost and performance change as the adder width changes. Two different FPGA chips, *Spartan-3* and *Virtex-5*, were selected for test/simulation purposes. *Spartan-3* is an older and relatively simpler chip compared to *Virtex-5* chip. It includes only CLBs but not DSP48E blocks. Moreover, *Spartan-3* LUTs have four inputs and *Virtex-5* LUTs have six inputs. The reason for selecting one older and newer chip is to show how advances in FPGA chip technology affect the cost and performance of the adders. All adders designed for this study were mapped to both chips using Xilinx's ISE 12.1 Electronic Design Automation (EDA) tool. During the mapping process (-3) was selected for *Virtex-5* and (-5) was selected for *Spartan-3* as the speed grade.

Critical path delay results of the adders on FPGA chips are presented in Figure 4. The results showed that RCA, CLA and CSA all have similar delay values for all bit widths on both chips and these delay values increase almost linearly. When the numbers are analyzed more precisely, it is seen that CLA performs slight better on *Virtex-5* and CSA performs slightly better on *Spartan-3* compared to the other two non-generated adders. Since there is no DSP48E block in *Spartan-3*, it was not tested. As expected, the generated adders outperformed the other adders on both chips. This is due to the fact that the synthesis tool knows it is synthesizing an adder and it uses built-in carry chains in the slices for carry calculations while mapping the adders to FPGA fabric. Using these dedicated carry chains yields better adders compared to non-generated adders. When two generated adders are compared on *Virtex-5*, it is observed that adders formed with DSP48E outperformed all adders on most cases. This is due to the fact that when the adders are formed using DSP48E blocks, the synthesis tool just lets you use the adders in the block. Since these adders are dedicated and highly optimized for speed, they yielded better adders for bit widths 32 and more. All adder delays except the ones on DSP48E increase almost linearly as the bit widths of the adders increase. The delay values on DSP48E decreases up to 32 bits, but then, increase slightly. This happens due to the fact that the adders in the DSP48E blocks are dedicated adders.

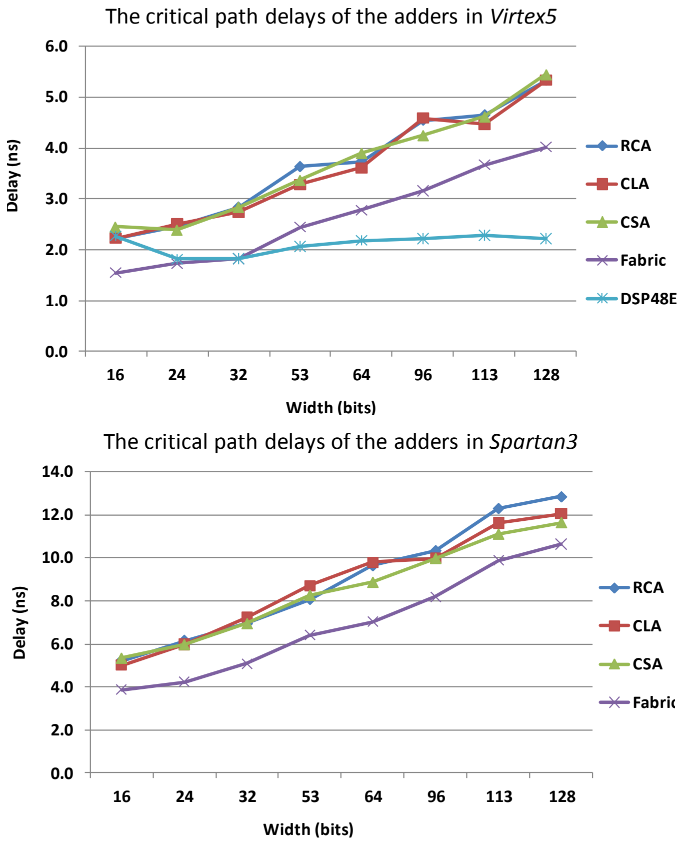


Figure 4. The comparison charts for critical path delays of the adders on Virtex-5 and Spartan-3, respectively.
(Şekil 4. Virtex-5 ve Spartan-3 üzerinde toplayıcıların kritik yol gecikmesi karşılaştırma grafiği)

When we consider these delay results, we saw that the delay values for CLA and CSA on FPGAs did not change as they were expected to be in ASIC implementation. This is due to the fact that the adders were mapped to configurable logic block (CLB) inside FPGA chips using a synthesis tool. The synthesis tool partitions the adders into small logic blocks and each block is then mapped to a CLB. The tool tries to keep block size as big as it can so that it can fit as much logic as to a single CLB. Because of this strategy, adder logic is not linearly mapped to CLBs. Sometimes logic for calculating one bit addition is mapped to a CLB while some other times logic for calculating more than one bits is mapped a CLB when the whole adder is mapped to an FPGA.

Although the propagation delay of each CLB is the same, due to this mapping variation, total delays of the adders did not change similar to ASIC implementations as the adders bit width increase. Some other factors also affect the adder delays on FPGAs such as optimization done during placement and routing processes. Since these processes determine final placement and routing configurations by performing optimization techniques on initial random configurations, total delay of the final configuration of the same adder can be slightly different each time the adder is mapped to same FPGA chip.

Figure 5 and Figure 6 show how cost of all adders changed as the bit widths increased when the adders were mapped to *Virtex-5* and *Spartan-3*. The cost information is given in terms of both the number of occupied slices and the number of occupied LUTs.

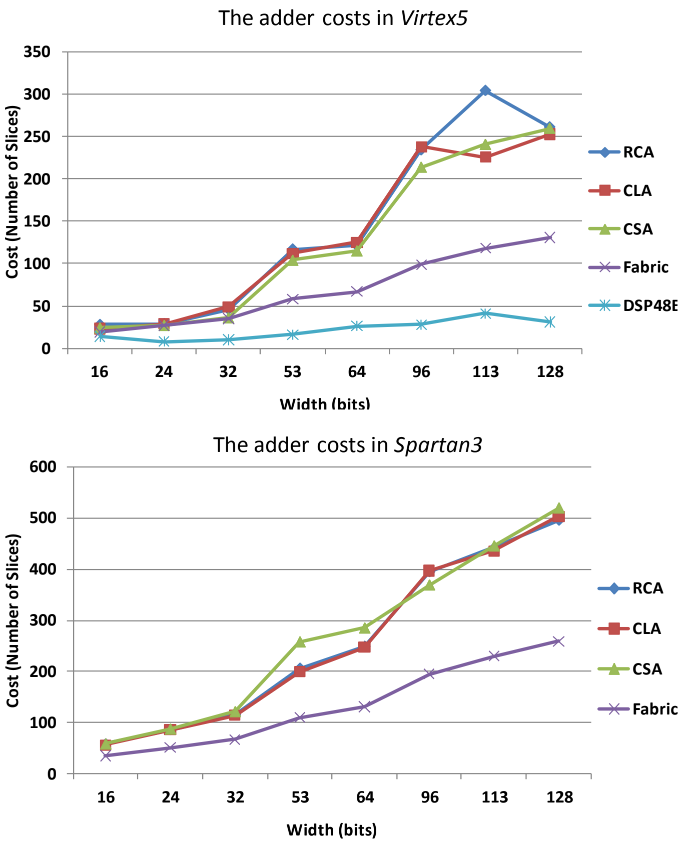
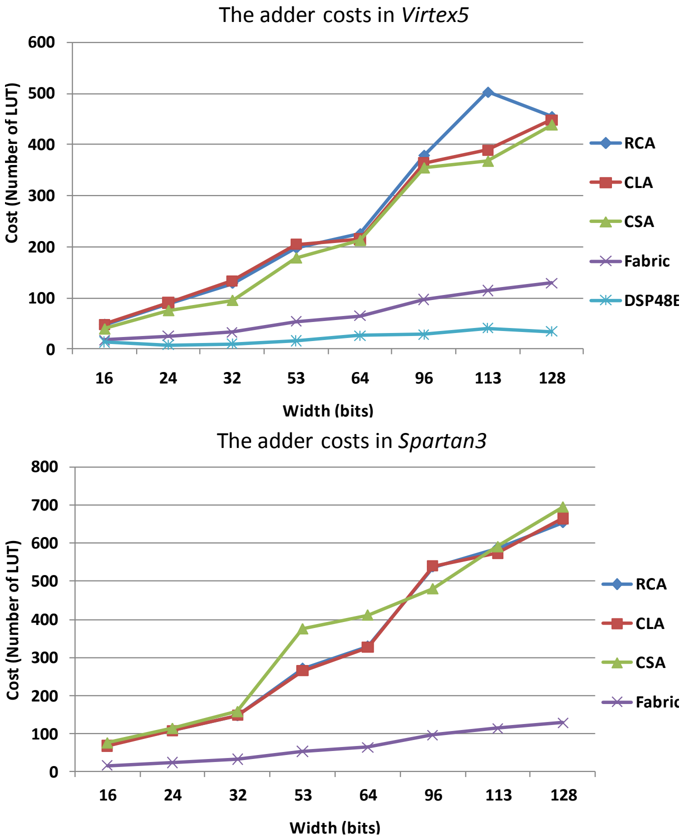


Figure 5. The comparison charts for costs of the adders in terms of slices on Virtex-5 and Spartan-3, respectively
(Şekil 5. Virtex-5 ve Spartan-3 üzerinde toplayıcıların Slice türünden maliyet karşılaştırma grafiği)

A CSA contains two RCA or CLA and some multiplexers. A CLA contains additional logic for calculating the carry values compared to a RCA. When we consider these factors, CSAs are expected to require the most amount of hardware and CLAs are expected to require more hardware than RCAs. Contrary to expectations, all three adders for each bit widths required almost the same amount of hardware. Moreover, when the numbers are closely analyzed, CSA requires slightly less amount of slices and LUTs for most adders on *Virtex-5*.


The adder costs in *Virtex5*

The adder costs in *Spartan3*

Figure 6. The comparison charts for costs of the adders in terms of LUTs on Virtex-5 and Spartan-3, respectively
(Şekil 6. *Virtex-5* ve *Spartan-3* üzerinde toplayıcıların LUT türünden maliyet karşılaştırma grafiği)

When we look at the adders generated with IP Core Generator, we see that these adders are always superior to non-generated adders. The adders generated on fabric use the built-in carry chains in the slices. Utilizing carry chains in constructing adders lets the synthesis tool save logic resources. When adders are generated using DPS48E, the dedicated adders in these blocks are used and the hardware requirements become minimum in terms of slices. On the other hand, since the number of DPS48E blocks in an FPGA chip is limited, and since sometimes more than one DPS48Es are need to build one adder, using these block is not always a feasible solution. One important conclusion from these results is that the synthesis tools are not mature enough to identify carry logic in the designed adders and map them to the built-in carry chains inside the CLBs. As a result, carry logic is also mapped to the LUTs and slices are wasted. We expected that at least the carry definition in RCA can be identified by the synthesis tool and mapped to the carry chains but it was not the case. In logic designs, one way to increase performance is to sacrifice cost. Generally, circuits are designed in parallel or pipelined fashion. In parallel designs, some redundant hardware is used but operations are run in parallel. In pipelined designs, circuits are portioned and some additional registers are inserted between these partitions. In either way, some more hardware is used and performance is increased. This general tradeoff rule between cost and performance is valid for ASIC implementation of the adders, but not for the FPGA implementations. For example, CSA designs required more than double the hardware required by RCA. On the other hand, when both adders are mapped to same FPGA chip, they cost about the same amount of hardware and their performance is about the same. This happens because of mapping strategy of synthesis tools and CLB structure of the FPGA chips.

When we compare costs of the adders on two chips, we find out interesting results. *Virtex-5* and *Spartan-3* CLBs contain 2 and 4 slices, respectively. Each *Virtex-5* slices contains 4 6-input LUTs, 4 flip-flops, and 1 carry chain and each *Spartan-3* slice contains 2 4-input LUTs, 2 flip-flops, and 1 carry chain. It is possible to form more complex logic functions and more functionality in *Virtex-5* slices than *Spartan-3* slices. But this feature comes with one disadvantage. If only one LUT is used in a slice for a simple logic function, the rest of the slice is left unused.

LUT to slice ratio for designed adders on *Virtex-5* is 2.00 on average. Since each *Virtex-5* slice contains 4 LUTs, this ratio means that only a half of the LUTs in the slices were utilized and the other half was wasted. The same ratio for the same adders on *Spartan-3* is 1.3 on average. Since each slice of *Spartan-3* contains 2 LUTs, more than half of the LUTs were utilized.

LUT to slice ratios for generated adders on *Virtex-5* and *Spartan-3* are 0.95 and 0.49, respectively. We know that each *Virtex-5* slice contains 4 LUTs and each *Spartan-3* slice contains 2 LUTs. These numbers indicate that about 75% of the LUTs on both chips were wasted.

Spartan-3 LUTs have 4 inputs and *Virtex-5* LUTs have 6 inputs. These LUTs are constructed from multiplexers [12]. A (*n+1*)-input multiplexer contains more than double the transistor that an *n*-input multiplexer contains. With this information in mind, when the number of LUTs spent to build the same adders on both chips is considered, it is seen that same amount of LUTs were used for generated adders. The ratio between LUT utilizations on *Spartan-3* and *Virtex-5* for designed adders is 1.48 on average. The same ratio for generated adders is 1.00. These results showed that using 6-input LUTs instead of 4-input LUTs did not changed the cost of the generated adders in terms of LUTs usage. On the other hand, an average of 30% improvements in LUTs usage was provided for designed adders. This improvement does not meet the increase in transistor count from 4-input to 6-input LUTs.

The adders built using DSP48Es were left outside the cost analysis due to the fact that these adders were formed using dedicated adders in DSP48E blocks not using the LUTs in the slices. On the other hand, from these results, it is concluded that if there are DSP48E blocks available in the chip then, they must be utilized to build faster adders with bit width 32 or more. Otherwise, generated adders on fabric must be selected.

Another purpose of this study is to show how many adders of each type can fit in to the selected chips.

Table 1 shows the theoretical upper limits of the adders. These numbers were calculated using the available slice count of the chips and the adders’ costs in terms of occupied slices. These numbers may not be reached due to optimizations done during placement and routing stages, but the table gives us an idea about how many adders may fit to the target chip.

Table 1. The number of adders that can fit in the target FPGA devices
(Tablo 1. Hedef FPGA çipine yerleştirilebilecek toplayıcı adetleri)

Width (Bits)	*Virtex-5*					*Spartan-3*			
	RCA	CLA	CSA	Fabric	DSP48E	RCA	CLA	CSA	Fabric
16	171	209	200	253	32	132	137	130	219
24	166	166	178	178	32	89	89	87	151
32	104	98	133	137	32	67	67	63	115
53	41	43	46	83	16	37	39	30	70
64	40	38	42	72	16	31	31	27	59
96	21	20	22	48	11	19	19	21	39
113	16	21	20	41	8	17	18	17	33
128	18	19	18	37	8	15	15	15	30

6. CONCLUSIONS (SONUÇLAR)

The critical path of a design is the key factor which determines the performance of the whole design. The shorter the path is, the higher the design can be clocked. In most cases, the critical path of a digital design goes through an adder or a design unit which includes adders; therefore, in this study, performance and cost evaluations for five different adders were done. Three of adders were very well known user designed adders which were ripple carry adder, carry select adder and carry look-ahead adder. The other two adders were generated adders which were generated using Xilinx’s IP Core Generator. The adders were mapped to two different FPGA chips which were Xilinx’s *Virtex-5* and *Spartan-3*, and their performances and costs were observed. In terms of both performance and cost, the best results were obtained from the adders generated using IP Core Generator on DSP48E blocks. If enough DSP48E blocks are available, then adders should be built using these blocks. Otherwise, adders generated on FGPA fabric using design tools such as IP Core Generator should be selected. Among the other three adders, CSA showed slightly better performance on average. Moreover, CSA was expected to cost more but contrary to the expectation, it costs about the same amount of hardware with the other two. If none of the aforementioned options are available, then CSA can be used. As a future work, this study can be extended to cover some new FPGA chips with new features.

REFERENCES (KAYNAKLAR)

1. Dehon, A., (2000). The Density Advantage of Reconfigurable Computing, IEEE Computer, vol. 33, pp.41-49.
2. Qasim, S.M., Abbasi S.A., and Almashary, B., (2009). An Overview of Advanced FPGA Architectures for Optimized Hardware Realization of Computation Intensive Algorithms, IMPACT’09, pp. 300-303.
3. El-Atfy, R., Dessouky, M.A., and El-Ghitani, H., (2007). Accelerating Matrix Multiplication on FPGAs, 2nd International Design and Test Workshop, pp. 203-204.
4. Sahin, I., (2010). A 32-bit floating-point module design for 3D graphic transformations, Scientific Research and Essays, vol.5, issue.20, pp. 3070-3081.
5. Zlatanovici, R. and Nikolic, B., (2003). Power – Performance Optimal 64-Bit Carry Look Ahead Adders, 29th European Solid-State Circuits Conference (ESSCIRC), pp. 321-324.
6. Xing, S., Yu, W.W.H., (1998). FPGA Adders Performance Evaluation and Optimal Design, IEEE Design & Test Computers, pp. 24-29.

7. Bečvář, M. and Štukjunger, P., (2005). Fixed-Point Arithmetic in FPGA, Acta Polytechnica, vol.45, pp.67-72.

8. The Virtex®-5 FPGA User Guide, http://www.xilinx.com/support/documentation/user_guides/ug190.pdf.

9. The Spartan®-3 Generation FPGA User Guide, http://www.xilinx.com/support/documentation/user_guides/ug331.pdf

10. Karlstrom, P., Ehliar, A., and Liu, D., (2008). High-performance, low-latency field-programmable gate array-based floating-point adder and multiplier units in a Virtex-4, Journal of IEEE Computers & Digital Tecniques, vol.2, issue.4, pp.305-313.

11. Huang, Z. and Ercegovac, M.D., (2000), Effect of Wire Delay on the Design of Prefix Adders in Deep-Submicron Technology, 34th Asilomar Conference on Signals, Systems and Computers, vol.2, pp. 1713-1717.

12. Beaumont-Smith, A. and Lim, C.C., (2001). Parallel Prefix Adder Design, 15th IEEE Symposium on Computer Arithmetic, pp. 218-225.

13. Dao, H.Q. and Oklobdzija, V., (2001). Application of Logical Effort Techniques for Speed Optimisation and Analysis of Representative Adders, IEEE Computer Society Press, vol.2, pp.1666-1669.

14. Zimmermann, R., (1996). Binary Adder Architectures for Cell-Based VLSI and their Synthesis, PhD dissertation, Swiss Federal Institute of Technology, Zurich.

15. Pai, Y. and Chen, Y., (2004). The fastest carry lookahead adder, 2nd IEEE International Workshop on Electronic Design, Test and Applications (DELTA'04), pp. 434-436.

16. Corsonello, P., Perri, S., and Cocorullo, G., (1999). Hybrid carry-select statistical carry look-ahead adder, IEEE Electronic Letters, vol.35, issue.7, pp.549-551.

17. The Xilinx Virtex-5 FPGA XtremeDSP Design Considerations User Guide, http://www.xilinx.com/support/documentation/user_guides/ug193.pdf.