A PROPOSED METHOD OF IMPLEMENTATION OF AN OPTIMAL FAST AND VERY LOW COST ADDER BY USING XILINX FPGA

Dr. W.A.Mahmoud Dhafer R.Zaghar Dr. Mohannad K. Sabir
Department of Electrical Engineering - College of Engineering
University of Baghdad -Iraq

ABSTRACT
There are two main methods of design the digital adder circuits. First is the serial adder that implement using full adders, this method has a low cost and a low speed. Second is the fast parallel adder or the so-called fast lookahead adder that has twice speed of the serial adder and very high cost.
Xilinx FPGA is a modern IC used to implement high-speed digital circuits. The optimization of speed and cost in design is achieved using Xilinx FPGA, which requires several rules that differ from that of optimization of Boolean algebra.
This paper shows the principle of the Xilinx FPGA architecture as well the set of conditions of optimal design, and proposes a new method of implementing a fast adder with very low cost using Xilinx FPGA.
A 16-bit adder was used to show the applications of this method and comparison with two classical methods. The advantages of the proposed method are demonstrated through a numeric example.

الخلاصة

إن هناك طريقتين أساسيتين لبناء الجامع الرقمي الأولى هي الجامع المتوالي (serial adder) والثانية هي الجامع المتوازي السريع (fast parallel adder) الذي يملك ضعف سرعة الجامع المتوالي ولكنه ذو كلفه عالية جدا. إن دائرة الجامع الرقمي يمكن أن تبنى باستخدام مصفوفة البوابات المبرمجة بواسطة المجال (FPGA) نوع < Xilinx > والتي تساعد على الحصول على دوائر رقمية ذات سرعة عالية إذا ما احسن استخدامها في التصميم.

إن هذا البحث سيقوم بشرح المبادئ الأساسية لهذه الدوائر ثم استنباط مجموعة من القواعد التي تساعد على بناء أنظمة رقمية ذات سرعة عالية و كلفة واطئه في ذات الوقت. إن تطبيق هذه القواعد سوف يساعد على اقترح طريقة تصميم جديدة لدائرة الجامع الرقمي. ثم سيتم اختبار كفاءة هذه الطريقة من خلال بناء جامع الرقمي ذو 16 خط دخل باستخدام الطرق الثلاثة ومقارنة الكلفة و السرعة لكل منها.

KEY WORDS
DSP, synchronous, asynchronous, short processing time, FPGA, Xilinx, Virtex, CLB, digital adder, serial adder, fast parallel adder.

INTRODUCTION

A large digital system such as Programmable Digital Down Converter (DDC) that processes data in speeds less than 100 Mega Sample Per Second (MSPS) can be implemented by using three methods.

First by using special programmable devices such as multipliers and programmable Finite Impulse Response (FIR) and Direct Digital Synthesizers (DDS) or one chip DDC. This approach is easy and fast in design but it has three disadvantages:

1- High cost.

2- It is restructured by the characteristics of the available of the standard ICs and components.

3- Need to interfacing with other components because the different sources of ICs mean different speeds and technologies (TTL, CMOS, ECL ....)[Balasubramanian, 1997] [Pratt, 1991].

Second is building large digital system using processors such microprocessors (μP), microcomputers (μC) or the digital signal processor (TMS) with suitable software. This technique is suitable for real time applications in speeds less than 20 MSPS with the modern technologies of TMS. This approach is used generally for simulations.

Third is to use general programmable devices such as ROM, this approach is complex in design and needs to build all components in a form suitable with the architecture of the base (general programmable) devices. This approach is commercial and very flexible compared with the first approach, it is suitable for real time applications for speeds less than 200 MSPS which is an advantages over the second approach.

The choice of the third approach is more suitable for large digital systems for speeds up to 70 MSPS. The additional advantages this choice gives low cost and universal chips system as [Mintzer, 1987] [Frerking, 1994].

CLASSIFICATION OF PROGRAMMABLE DEVICES

There are a large number from programmable devices (general programmable devices) that can be classified from the view of user to build a DSP system to two main types depending on the function of there cell as shown in **Fig.(1)** that show the most famous programmable devices [http] [Xilinx toolset] [Altera, 1996].

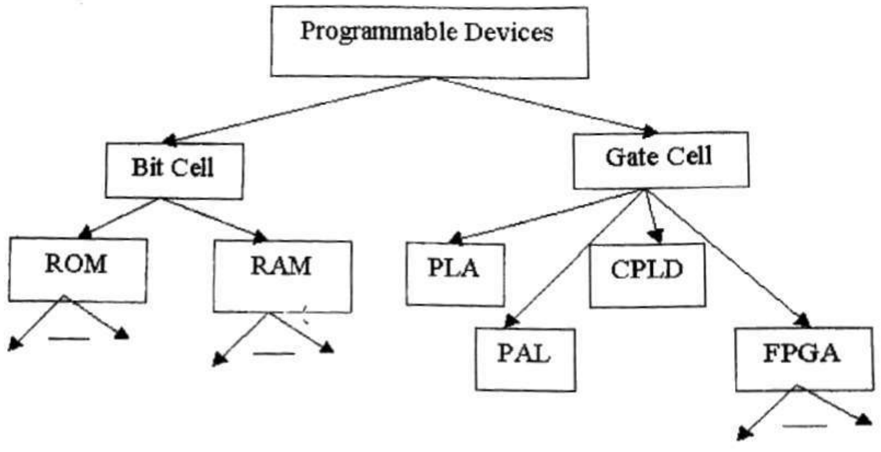


Fig.(1) Basic classification of famous programmable devices.

Most DSP systems used the gate cell types because that has a possibility to reduction the Boolean expressions of DSP circuits.

Fore example a 16-bit adder need to about 73,000,000,000 bits when it built by using bit cell devices such ROM but it can implemented by use FPGA by 20 cells.
Programmable logic array (PLA) and programmable array logic (PAL) have a low flexibility and low capacity (maximum 32 inputs and 32 outputs) therefore they use to small DSP circuits.
Complex programmable logic devices (CPLD) is the first generation of FPGA has high flexibility but less than FPGA and midum capacity and that use to built small circuits and midum systems such small control circuits.
The invention of the field-programmable gate array (FPGA) as a programmable device has given rise to an alternative method of computing. The FPGA provides the means for achieving hardware performance and software versatility. It can be optimized for a specific_circuit, but can be reused and re-optimized for multiple circuits by reprogramming the device [Ray] [Sjstrm, 1996].
The FPGA consists of reconfigurable logic that can be programmed to compute applications ranging from very fine-grained, highly repetitive to coarse-grained, general-purpose computational tasks.
FPGA is the new type of the programmable devices that has the higher flexibility and has a different internally architecture depend on the company design such Altera, Xilinx…
But the common of these types is that FPGA is a set from free connection matrix gates, with possibility to programming the gate it self. Xilinx FPGA is the most FPGA flexibility and it is very suitable to built the large inputs component such adders and multipliers…
It has a different series these have a small differences in there internally architecture such XC3000, XC4000, Virtex-E, Virtex-II…[http] [Wanhammar, 1999].
This paper will select the series Virtex-II as example to Xilinx FPGA and show the basic architecture, then select XCV600 chip from this series to give a numeric view for advantages of proposed design method.

ARCHITECTURAL DESCRIPTION OF VIRTEX-II ARRAY

The Virtex-II user-programmable gate array, shown in **Fig.(2)**, comprises two major configurable elements: configurable logic blocks (CLBs) and input/output blocks (IOBs).

- CLBs provide the functional elements for constructing logic.
- IOBs provide the interface between the package pins and the CLBs.
- CLBs interconnect through a general routing matrix (GRM).

The GRM comprises an array of routing switches located at the intersections of horizontal and vertical routing channels. Each CLB nests into a Versa Block that also provides local routing resources to connect the CLB to the GRM.
The Versa Ring I/O interface provides additional routing resources around the periphery of the device. This routing improves I/O routability and facilitates pin locking.
The Virtex-II architecture also includes the following circuits that connect to the GRM.

- Dedicated block memories (BRAMs) of 4096 bits for each block.
- • Clock DLLs for clock-distribution delay compensation and clock domain control.
- • 3-State buffers (BUFTs) associated with each CLB that drive dedicated segmentable horizontal routing resources.

Values stored in static memory cells control the configurable logic elements and interconnect resources. These values load into the memory cells on power-up, and can reload if necessary to change the function of the device.
Input/Output block in the Virtex-II IOB select I/O inputs and outputs that support a wide variety of I/O signalling standards [Wanhammar, 1999] [DS022, 1999].
There are three IOB storage elements function either as edge-triggered D-type flip-flops or as level-sensitive latches. Each IOB has a clock signal (CLK) shared by the three flip-flops and independent clock enable signals for each flip-flop.

In addition to the CLK and CE control signals, the three flip-flops share a Set/Reset (SR). For each flip-flop, this signal can be independently configured as a synchronous Set, a synchronous Reset, an asynchronous Preset, or an asynchronous Clear. The output buffer and all of the IOB control signals have independent polarity controls.
Input path in the Virtex-II IOB input path routes the input signal directly to internal logic and/ or through an optional input flip-flop.
An optional delay element at the D-input of this flip-flop eliminates pad-to-pad hold time. The delay is matched to the internal clock-distribution delay of the FPGA, and when used, assures that the pad-to-pad hold time is zero.
Output path includes a 3-state output buffer that drives the output signal onto the pad. The output signal can be routed to the buffer directly from the internal logic or through an optional IOB output flip-flop.
The 3-state control of the output can also be routed directly from the internal logic or through a flip-flop that provides synchronous enable and disable [Wanhammar, 1999] [DS022, 1999].

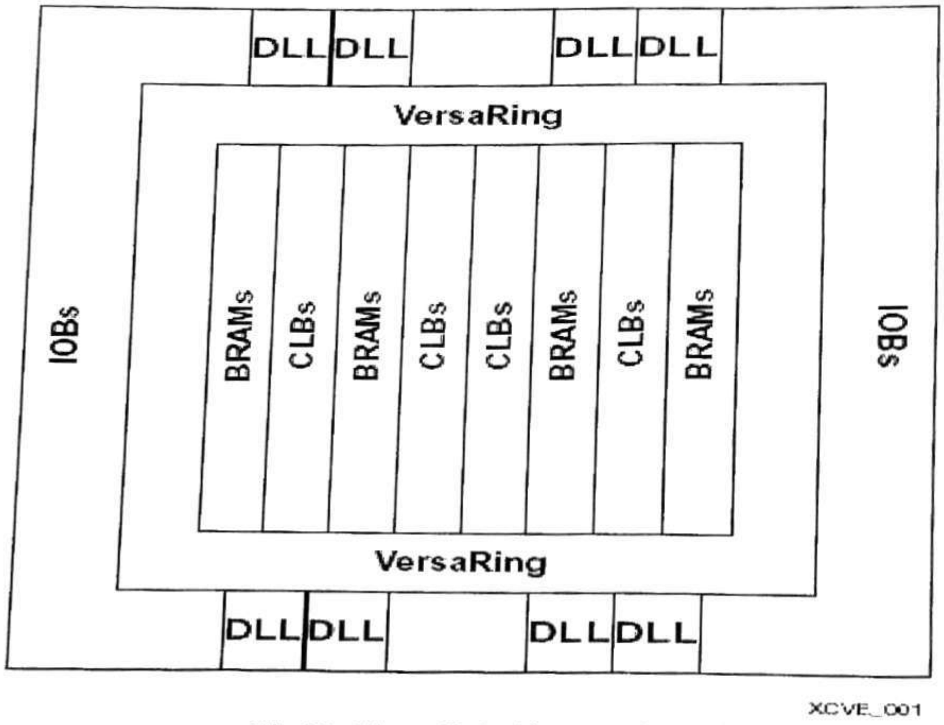


Fig.(2) Virtex-II Architecture Overview.

XILINX FPGA CLB ARCHITECTURE

The basic building block of the Virtex-II Configurable Logic Block (CLB) is the logic cell (LC). An LC includes a 4-input function generator, carry logic, and a storage element. The output from the function generator in each LC drives both the CLB output and the D input of the flip-flop as shown in **Fig.(3)**. Each Virtex-II CLB contains four LCs, organized in two similar slices, as shown in **Fig.(4)**.
In addition to the four basic LCs, the Virtex-II CLB contains logic that combines function generators to provide functions of five or six inputs. Consequently, when estimating the number of system gates provided by a given device, each CLB counts as 4 LCs.

Virtex-II function generators are implemented as 4-input look-up tables (LUTs). In addition to operating as a function generator, each LUT can provide a 16 x 1-bit synchronous RAM.
Furthermore, the two LUTs within a slice can be combined to create a 16 x 2-bit or 32 x 1-bit synchronous RAM, or a 16x1-bit dual-port synchronous RAM [Petersen, 1995].
F5 in **Fig.(4)** multiplexer in each slice combines the function generator outputs. This combination provides either a function generator that can implement any 5-input function, a 4:1 multiplexer, or selected functions of up to nine inputs.
Similarly, F6 multiplexer combines the outputs of all four-function generators in the CLB by selecting one of the F5-multiplexer outputs [Wanhammar, 1999] [DS022, 1999].

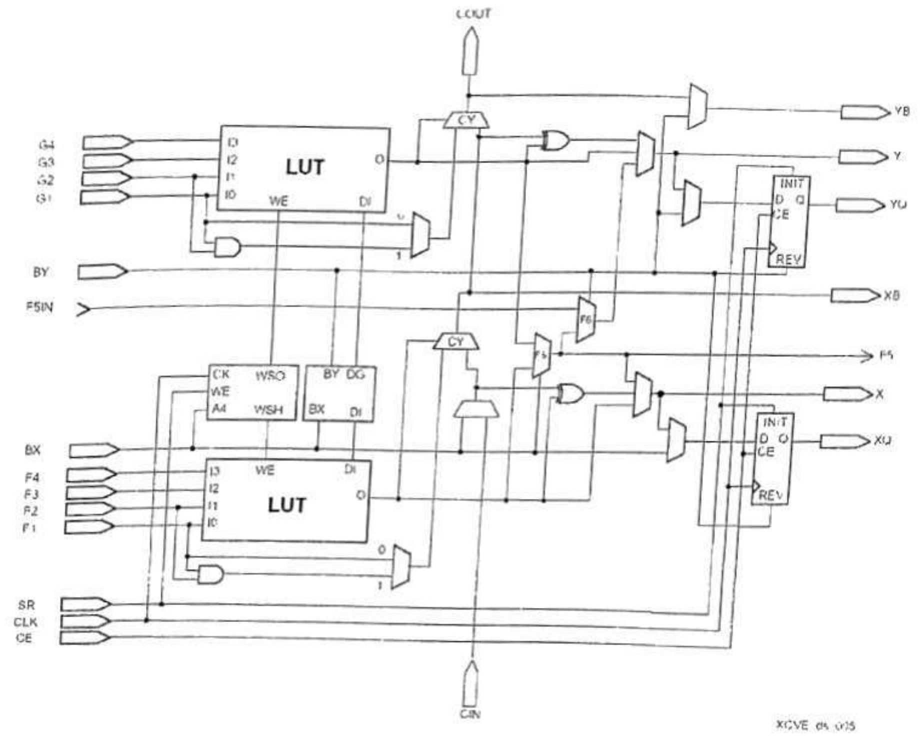


Fig.(3) Detailed View of Virtex-II Slice.

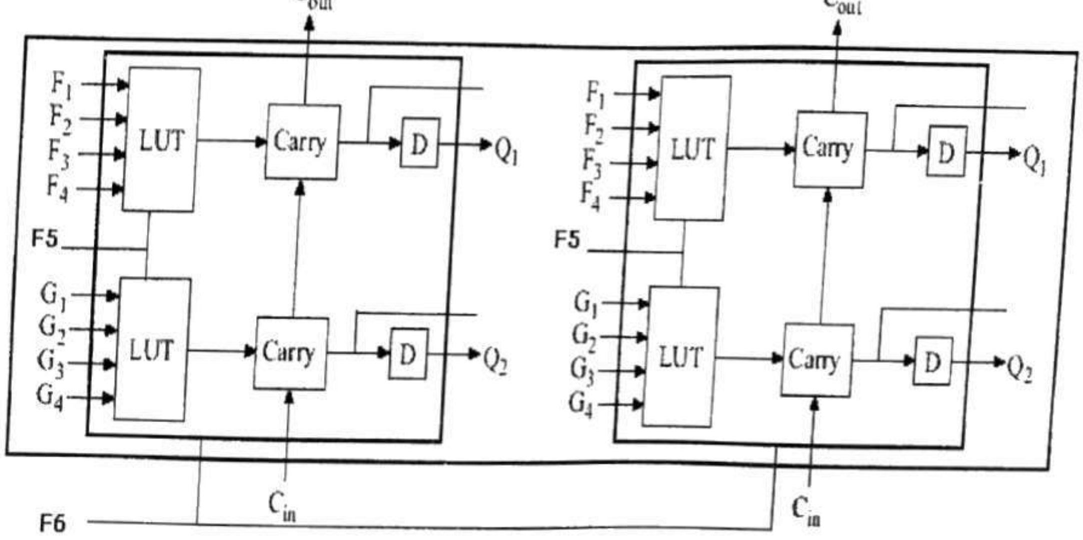


Fig.(4) A simplified structure of a CLB of Virtex-II

This permits the implementation of any 6-input function, an 8:1 multiplexer, or selected functions of up to 19 inputs.
Each CLB has four direct feedthrough paths, two per slice. These paths provide extra data input lines or additional local routing that does not consume logic resources.

CONVENTIONAL CIRCUIT DESIGN USING XILINX FPGA

The characteristics of FPGA give a set of rules to reduce the cost and increase the speed of DSP systems differ from the rules that used in Boolean design.
These rules varied from FPGA to other depending on the type of it. Such example the rules to built an optimal cost and speeds in Xilinx FPGA are that:
I-Divided the large circuit to small sub circuits with input pins less than 7-inputs, because the large inputs function is uncontrollable space and delay design.
II-Each sub circuit has 4-inputs for minimum cost, because each 4-inputs function need to one cell and one delay.
III-Each sub circuit has 6-inputs for maximum speed, because each 6-inputs function need to four cells (one CLB) and one delay.
IV-Each sub circuit has 5-inputs for optimal cost and speed, because each 5-inputs function need to two cells and one delay.
V-When the above rules are not satisfied try to use 3-inputs and not increase the input number over 6 inputs, because each 7-inputs function need to eight cells and two delay, when each 1,2,3-inputs function need to one cell and one delay.

SHORT PROCESSING TIME

The digital systems classified depending on the synchronous timing to synchronous system and a synchronous system.
A synchronous system execute their jobs or transfer functions between the input and the output without any internally storage and they take a long time to reach to stable case. These systems use for low speed and small controls circuits [Brown, 2000] [Melham, 1993].
Synchronous systems built as a multi-layers process/storage systems as shown in **Fig.(5)**.

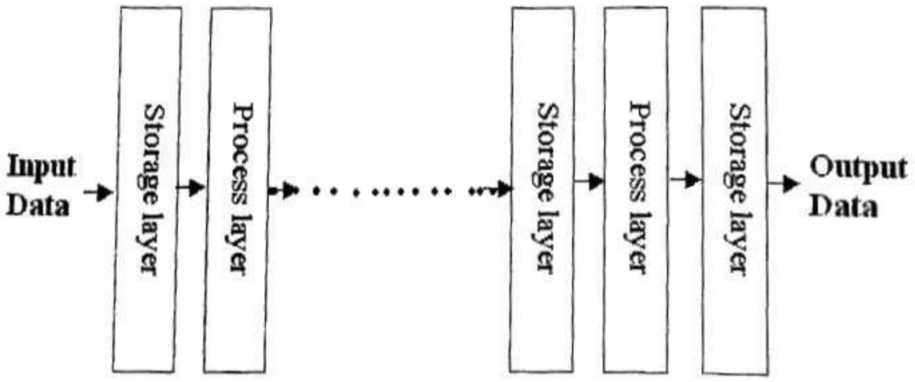


Fig.(5) Synchronous Systems Architecture.

The storage layers have a common clock controlled on the time storage and input/output the data in process layers dissipated a short constant time comparing with the processing layer time.

The maximum speed of the clock system depends on the minimum speed in the processing layers of the system. This is because the time of the clock must be larger than the maximum processing time layer plus the storage time.
Thus the full adder circuit as a processing layers in large synchronous system must possess a short delay to give a high-speed adder and thus result in high-speed digital system.

SERIAL ADDER

This method depend on using a serial full adders (FA) to built n-bits adder as shown in **Fig.(6)** that represent a 16-bit adder.
Each full adder (FA) need to 2 cells cost one to sum and one to carry and one delay or 2.5 nsec delay in XCV600 to give stable output, that mean the 16-bit adder need to 32 cells cost and 40 nsec as minimum time to give output result [Brown, 2000] [Melham, 1993].

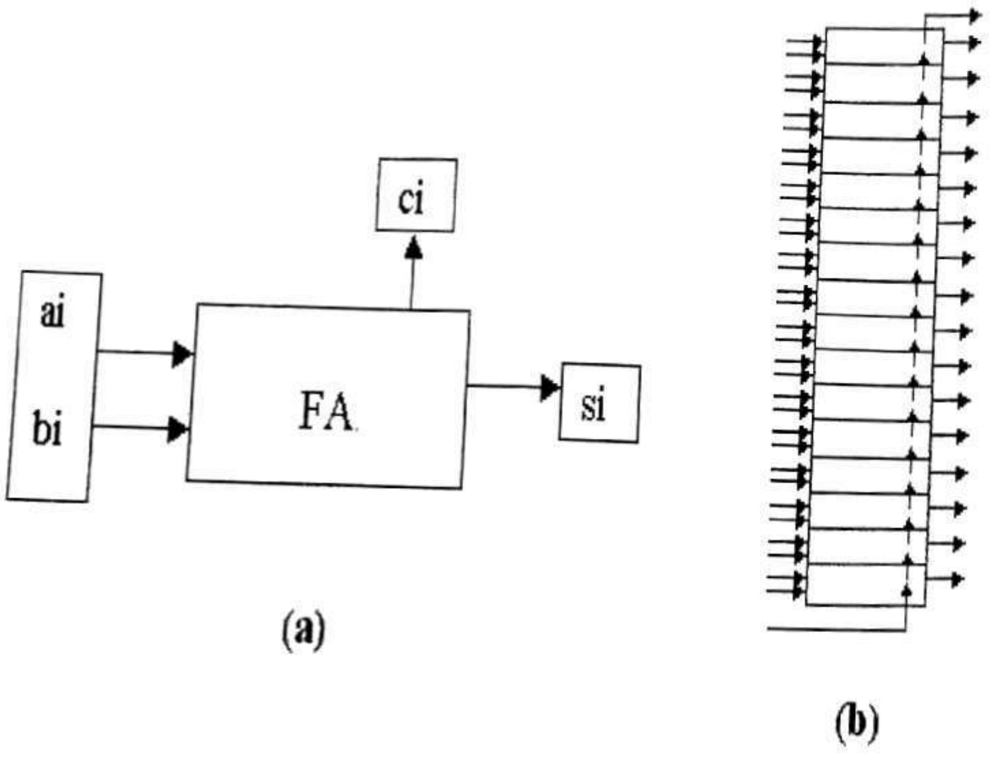


Fig.(6)a Full Adder unit, b) 16-bit adder by using FAs.

FAST PARALLEL ADDER

This approach depend on generating the carry directly from inputs as in the following equations:

$g_i = a_i b_i$

$c_{i+1} = g_i + p_i g_{i-1} + p_i p_{i-1} g_{i-2} + \dots\dots + p_i p_{i-1} \dots p_1 p_0 c_0$
$s_i = x_i \oplus y_i \oplus c_i$

Then generate all outputs parallel from these Boolean equations and as shown in **Fig.(7)**.

The analysis of these equations show the 16-bit fast adder need to about 500 cells that can reduce to 227 cells, and the maximum delay is 6 delay or 15 nsec by using XCV600 [Brown, 2000] [Melham, 1993].

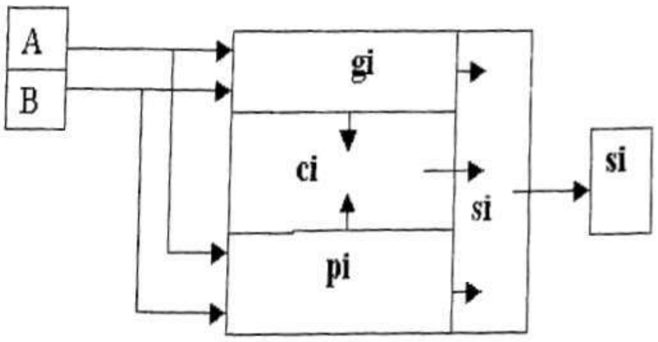


Fig.(7) parallel fast adder.

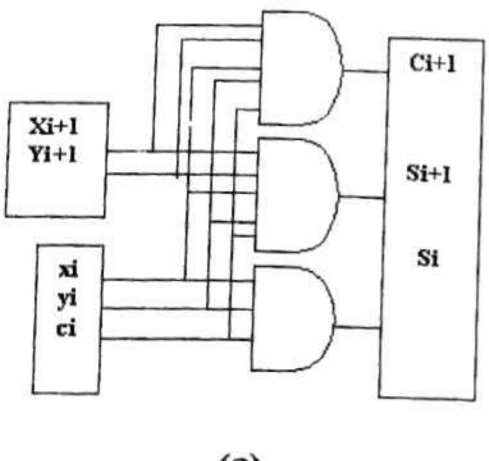
THE PROPOSED METHOD

The basic idea of this method depend on divided the adder to group of 2-bit adders as shown:

$s_i = x_i \oplus y_i \oplus c_i$

$s_{i+1} = x_{i+1} \oplus y_{i+1} \oplus x_i y_i \oplus x_i c_i \oplus y_i c_i$

$c_{i+1} = x_{i+1} y_{i+1} \oplus x_{i+1} x_i y_i \oplus x_{i+1} x_i c_i \oplus x_{i+1} y_i c_i$
$\oplus\ y_{i+1} x_i y_i \oplus y_{i+1} x_i c_i \oplus y_{i+1} y_i c \oplus x_i y_i c_i$



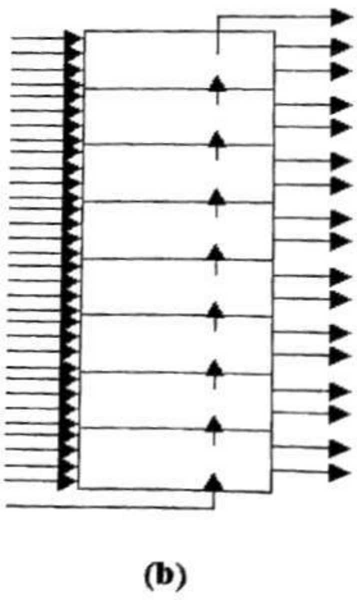


Fig.(8) a) 2-bit Adder unit, b) 16-bit adder by using 2-bit adders.

From the above equations that represent the Boolean expression for outputs of 2-bit adder circuit we can note these equations have 3 and 5 inputs that satisfy the optimal space and speed as in rules of FPGA design, **Fig.(8)** show the total circuit of 16-bit adder.
The analysis of these equations show each 2-bit adder need to 5 cells cost and one delay or 2.5 nsec delay by using XCV600 to give stable output, that mean the 16-bits adder need to 40 cells cost and 20 nsec as minimum time to give output result.

CONCLUSIONS

Each unit for serial method (full adder) contain a two Boolean expressions one for sum and the other for carry that has 3-variables, this case is the nearest case to minimum cost (rule II) and maximum delay.
So this method has a floating (not used) input in each cell or that mean it escape about 25% from the total inputs of cells that will reduce the total efficiency or in other word that will increase the cost of this method.
The parallel method has very long intermediate Boolean expressions in *ci* equation reach to 153 in *c17* and the bad designer may be need to a few thousands cells to implement this equation and very high delay reach about to 26 delay.
The good division to the terms of this equation using conventional circuit design rules for Xilinx FPGA the designer can be built a 16-bit adder by using 227 cells and 6 delay with about 15% floating inputs.
The unit for proposed method (2-bits adder) contains three Boolean expressions two for the sum and the other for carry first has 3-variables and the two other have 5- variables, this case is the nearest case to optimal cost/speed (rule IV).
So this method has one floating (not used) input in each cell or that mean it escape about 5% from the total inputs of cells that will reduce the total efficiency or in other word that will increase the cost of this method in limited value.
The compression of the three methods in **Table (1)** shows the proposed method has a small additional cost over the serial method but it give a twice speed, then the parallel has triple speed of serial but with 7 times of cost.

Table (1) Comprussion of the three methods for adder design.
*Approximation

Method	Serial	Parallel	Proposed
Rational Speed	1	3	2
Rational Cost	1	7	1.25
Ratio of Inputs Used	75%	85% *	95%

The additional advantage in proposed method comes from the good divisions for adder functions that make most equations have 5 inputs, so that give optimal speed/cost in Xilinx FPGA and use the maximum ratio from the inputs that reduce the total cost.
Note the serial method has 25% floating (not used) from the total inputs of LCs, when the proposed method has enough 5% floating from the total inputs of LCs.
The above rules and the proposed method can use with other types of FPGAs by modification them to become suitable with architecture of selected FPGA.
Also they can use to design other optimal DSP components such comparetor and multipliers by using same approach.

REFERENCES

Balasubramanian N. B, (1997), Optimal design of digital filters using CSD coefficients,, University of California.

Pratt W. K., (1991), Digital Signal Processing, Second Edition, Wiley-Interscience Publication.

Altera, 1996, Corporation. Data Book.

Mintzer L., (1987), Mechanization of Digital Signal Processors, Handbook of Digital Signal Processing, pp. 941-973.

Frerking M. E., (1994), Digital Signal Processing in Communications Systems, Van Nostrand Reinhold.

http://www.xilinx.com/legal.htm.

Xilinx toolset: Foundation BAS 1.5i, Core generator.

Ray Andraka's website: http://users.ids.net/~randraka.

Sjstrm U., (1996), Karlsson M., and Hrlin M :.A Digital Down Converter Chip, in Proc. of European Signal Processing Conference EUSIPCO'96, Vol.1, Trieste, Italy, pp. 284–287.

Wanhammar L., (1999), DSP Integrated Circuits, Academic Press.

DS022, (1999), (v1.0) December 7: Advance Product Specification (Virtex-E 1.8 V Field Programmable Gate Arrays).

Petersen R. J. and Hutchings B. L., (1995), An assessment of the suitability of FPGA-based systems for use in digital signal processing,. In W. Moore and W. Luk, editors, Field-Programmable Logic and Applications, Oxford, England, August.

Brown S. and Vranesic Z., (2000), Fundamental of Digital Logic with VHDL Design, McGraw-Hill.

Melham T. F., (1993), Higher Order Logic and Hardware Verification. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

LIST OF SYMBOLS

μC: microcomputer.
μP: microprocessor.
CLB: Configurable Logic Block.
CPLD: Complex programmable logic devices.
DDC: Digital Down Converter.
DDS: Direct Digital Synthesizer.
DSP: digital signal processing.
FA: full adder.
FIR: finite impulse response.
FPGA: field-programmable gate array.
GRM: general routing matrix.
LC: logic cell.
LUT: look-up table.
MSPS: magi sample per second.
PAL: programmable array logic.
PLA: programmable logic array.